CVSI, Inc.
4G Crosby Drive
Bedford, MA 01730

Operator's Guide to File System Maintenance

Revision 23.3

DOC9300-6LA

Operator's Guide to File System Maintenance

Sixth Edition

George W. Gove

This manual documents the software operation of the PRIMOS operating system on 50 Series computers and their supporting systems and utilities as implemented at Master Disk Revision Level 23.3 (Rev. 23.3).

Prime Computer, Inc., Prime Park, Framingham, Massachusetts 01701

Printing History

First Edition (DOC9300-1LA) December 1984 for Revision 19.4
Update 1 (UPD9300-11A) December 1985 for Revision 20.0
Second Edition (DOC9300-2LA) August 1986 for Revision 20.2
Third Edition (DOC9300-3LA) July 1987 for Revision 121.0
Fourth Edition (DOC9300-4LA) August 1988 for Revision 22.0
Fifth Ediiton (DOC9300-5LA) July 1989 for Revision 22.1
Update 1 (UPD9300-51A) August 1990 for Revision 23.0
Sixth Edition (DOC9300-6LA) May 1992 for Revision 23.3

Credits

Project Support: Andy Griffith, Robert Love
Editorial: Judy Gordon, Irene Rubin
Illustration: Elizabeth Wahle
Production: Judy Gordon

Contents

5 Formatting Disks – MAKE

6 Repairing Partitions – FIX_DISK

Part III: Special Topics

7 Robust Partitions

D Disk Errors

E MAKE_ROBUST Messages

F Summary of Command Syntax for MAKE and FIX_DISK

G Glossary

Index

About This Book

About This Series

The Operator's Guide series is designed to help you, as a System Operator or a System Administrator of a Prime computer, do your job. This preface describes the eight Operator Guides and other Prime documentation that are particularly useful for System Operators and System Administrators. To display an online list of all Prime documentation, use the HELP DOCUMENTS command.

For the System Operator

Before reading this book, you should have some familiarity with Prime systems. A good way to begin is to read the *PRIMOS User's Guide* (DOC4130-5LA), which explains the PRIMOS file management system and provides introductory and tutorial information about essential commands and utilities. When you read any Prime documentation, be sure to consult the section entitled Prime Documentation Conventions, which follows the preface; this section is essential to understanding how information is presented.

After you are familiar with Prime systems, read the *Operator's System Overview*, which outlines the material in the Operator's Guide series. Then select the other books in the series as they apply to the tasks you must perform.

As you learn more about system operations, you will use the *Operator's Guide to System Commands* as a reference for many of the special system commands and arguments that you, as an Operator, will need to perform your job. The *Operator's Guide to System Commands* documents most of the commands described in the Operator's Guide series.

The Operator's Guide Series

The following books contain detailed information for the System Operator.

- *Operator's System Overview* (DOC9298-3LA) introduces the series and describes computer-room operation of Prime systems.
- *Operator's Guide to System Monitoring* (DOC9299-3LA) describes how to monitor system usage, activity, and messages.
- *Operator's Guide to Data Backup and Recovery* (DOC10324-1LA) and its update documents (UPD10324-11A and UPD10324-21A) describe how to save information on disk or tape and how to restore that information later.
- *Operator's Guide to the Batch Subsystem* (DOC9302-3LA) describes how to set up, monitor, and control the Batch subsystem.
- *Operator's Guide to the Spooler Subsystem* (DOC9303-5LA) describes how to set up, monitor, and control the Spooler subsystem.
- *Operator's Guide to System Commands* (DOC9304-6LA) serves as a reference guide for most of the commands described in the other books in the series.
- *Operator's Guide to Prime Networks* (DOC10114-1LA) and its update package (UPD10114-11A) provide reference information about running network-related programs and monitoring network events.

Other Books for the Operator

- *Operator's Master Index* (DOC10110-5LA) indexes all the Operator and System Administrator Guides. Consulting this index is often the quickest way to find which manual has the information you need.
- The computer handbook for your particular CPU explains such topics as booting the system, shutting down PRIMOS, handling halts and hangs (including warm starts), performing tape dumps, and using the Virtual Control Panel (VCP).
- The *Using Your CPU* guide (available only for office CPUs) is intended for nontechnical users who are acting as System Operators, and covers system startup and shutdown, system backups, troubleshooting, and other day-to-day system management issues.
- *MAGNET User's Guide* (DOC10156-1LA) and its update package (UPD10156-11A) describe the MAGNET utility, used to transfer data by magnetic tape from other operating systems to PRIMOS and vice versa.

For the System Administrator

In addition to the documentation in the Operator's Guide series, be sure to read the System Administrator Guide series, which describes how to set up, configure, and maintain PRIMOS:

- *System Administrator's Guide, Volume I: System Configuration* (DOC10131-3LA) explains how to set up a system and allocate resources.
- *System Administrator's Guide, Volume II: Communication Lines and Controllers* (DOC10132-2LA and RLN10132-21A) explains how to configure communication lines.
- *System Administrator's Guide, Volume III: System Access and Security* (DOC10133-3LA) explains PRIMOS security features and how to prevent unauthorized use of your system.
- *DSM User's Guide* (DOC10061-3LA) explains how to use the Distributed Systems Management (DSM) subsystem, including how to configure and operate DSM.

The System Administrator Guides also provide information about most of the commands necessary to operate your Prime system.

Other Recommended Reading

In addition to the books listed above, you may find the following books useful:

- *New User's Guide to EDITOR and RUNOFF* (FDR3104-101B) is a basic reference for any user of a Prime system and provides information about the Prime text editor and formatter.
- *PRIMOS Commands Reference Guide* (DOC3108-8LA) provides detailed information about user commands.
- *Magnetic Tape User's Guide* (DOC5027-2LA) and the update documents for Rev. 20.1 (UPD5027-21A) and Rev. 20.2 (UPD5027-22A), which describe the magnetic tape utility programs for users.
- *PRIMENET Planning and Configuration Guide* (DOC7532-4LA) and its update (UPD7532-41A) describe how to plan, configure, and maintain PRIMENET software for a system.
- *NTS User's Guide* (DOC10117-3LA) explains the Network Terminal Service (NTS).
- *50 Series Technical Summary* (DOC6904-2LA) describes the features of the 50 Series systems, including advanced architecture concepts and the software and hardware products the concepts support.

- The *System Architecture Reference Guide* (DOC9473-3LA) describes internal functioning of all 50 Series computers.
- The *Subroutines Reference II: File System* (DOC10081-2LA) describes subroutines that deal with access to and management of the file system.

About This Book

The *Operator's Guide to File System Maintenance* is one of a series of books designed to help you, as an Operator or a System Administrator of Prime computers, to do your job. It discusses the tasks necessary to create and maintain the file system.

The purpose of this book is to provide the information you need to know in order to maintain your file system. The operations you must become familiar with include how to

- Determine physical device numbers for partitions
- Format partitions
- Repair partitions
- Convert the format of a partition from one revision to another
- Decode disk error messages

If you are a System Operator, this book is intended to help you become familiar with the tasks required to keep the file system operating efficiently.

If you are a System Administrator, this book is intended to help you gain insight into some of the tasks required of the Operator to maintain the file system and includes discussions of these topics:

- The file system
- Using and assigning disks
- Robust partitions
- Dynamic badspot handling
- Mirroring
- File system record allocation
- New file system features at Rev. 23.3

Organization of This Book

This book is divided into four parts:

- Part I introduces you to the file system and discusses disks, the storage media used to keep files.
- Part II tells you what physical device numbers are and how to construct them. It also explains how to assign and use disks. It describes in detail how to create and format disk partitions and how to repair file structures.
- Part III describes in detail the concepts of robust partitions, dynamic badspot handling, mirroring of partitions, and file record allocation.
- The appendices explain the meaning of error messages related to disk operations, summarize the commands and options used for creating and repairing partitions, and include a glossary of terms used in this book. A comprehensive index lists all subjects discussed in this book.

You are expected to have some familiarity with Prime systems before reading this book. If you are not familiar with the PRIMOS operating system, you should read the *PRIMOS User's Guide* (DOC4130-5LA), which explains Prime's file management system and provides introductory and tutorial information about essential commands and utilities.

How This Book Is Structured

This book contains 10 chapters and 7 appendices:

- Chapter 1 explains terminology, the preparation and maintenance of disks used by PRIMOS, and concepts related to the file system.
- Chapter 2 describes varieties of disks used on Prime systems, terminology related to these disks, badspots, and equivalence blocks.
- Chapter 3 explains physical device numbers and how to construct them and describes partitioning of disk types supported by PRIMOS.
- Chapter 4 explains the procedures to assign disks to yourself in order to perform maintenance on them and the procedures to make disks available to users.
- Chapter 5 explains how to use the MAKE utility, which formats and partitions disks for use on your system.
- Chapter 6 explains how to use the FIX_DISK utility, which repairs file partitions.

- Chapter 7 describes robust partitions, the file system objects on them, allocation of CAM files, fragmentation, and the use of utilities in conjunction with robust partitions. This chapter also discusses the MAKE_ROBUST utility for creating robust partitions. In addition, this chapter describes the concept of logical file type.
- Chapter 8 explains the concept of dynamic badspot handling introduced at Rev. 21.0, which makes use of intelligent disk controllers and allows partitions to be mirrored.
- Chapter 9 explains the concept of partition mirroring introduced at Rev. 21.0, which allows you to dynamically make duplicate copies of partitions and which depends on dynamic badspot handling by intelligent disk controllers.
- Chapter 10 explains the concepts of reverse and forward sectoring and interleave factors introduced at Rev. 21.0, which provide a method to allow efficient record allocation and use of intelligent disk controllers.
- Appendix A explains error messages and other messages from MAKE.
- Appendix B explains FIX_DISK messages.
- Appendix C explains messages related to the mirroring process.
- Appendix D explains error messages related to disk operations.
- Appendix E explains error messages related to the MAKE_ROBUST utility.
- Appendix F summarizes the command syntax for the MAKE and FIX_DISK utilities.
- Appendix G provides summary explanations of terms introduced in the main text.

Prime Documentation Conventions

The following conventions are used throughout this document. The examples in the table illustrate the uses of these conventions.

Convention	*Explanation*	*Example*
Uppercase	In command formats, words in uppercase bold indicate the names of commands, options, statements, and keywords. Enter them in either uppercase or lowercase.	**MAKE**
Italic	Variables in command formats, text, or messages are indicated by lowercase italic.	**MAKE –DISK** *pdev*
Abbreviations	If a command or option has an abbreviation, the abbreviation is placed immediately below the full form.	**–DISK_TYPE** **–DT**
Brackets	Brackets enclose a list of one or more optional items. Choose none, one, or several of these items.	$\textbf{LD} \begin{bmatrix} \textbf{-BRIEF} \\ \textbf{-SIZE} \end{bmatrix}$
Braces	Braces enclose a list of items. Choose one and only one of these items.	$\textbf{CLOSE} \begin{Bmatrix} \textit{filename} \\ \textbf{-ALL} \end{Bmatrix}$
Braces within brackets	Braces within brackets enclose a list of items. Choose either none or only one of these items; do not choose more than one.	$\textbf{BIND} \left[\begin{Bmatrix} \textit{pathname} \\ \textit{options} \end{Bmatrix} \right]$
Monospace	Identifies system output, prompts, messages, and examples.	`address connected`
Underscore	In examples, user input is underscored but system prompts and output are not.	`OK, ASSIGN DISK`
Hyphen	Wherever a hyphen appears as the first character of an option, it is a required part of that option.	**FIX_DISK –FIX**
Ellipsis	An ellipsis indicates that you have the option of entering several items of the same kind on the command line.	*pdev-1* [. . .*pdev-n*]
Subscript	A subscript after a number indicates that the number is not in base 10. For example, the subscript 8 is used for octal numbers.	200_8

The File System

1

This chapter provides introductory information about the PRIMOS file system. (For a detailed discussion of files, see the *Advanced Programmer's Guide, Volume II: File System.*)

The sections that follow offer an overview of

- Online storage
- Disk preparation with the MAKE utility
- Disk maintenance and repair with the FIX_DISK utility
- The structure of the PRIMOS file system

Online Storage

All Prime systems use **online storage**, or disks, to provide a storage area for files, or data, and to allow large volumes of data storage. Disks store much larger quantities of data than main memory can and they retain the information whether power is on or off.

A disk drive includes a collection of physical disks (surfaces, platters) and data can be stored on the disk surfaces. The disks are generally divided into a number of sections, or **logical disks**, which use variable numbers of surfaces, or the entire physical disk can be one logical disk. These logical disks are generally referred to as **partitions**, but may also be referred to as **disks, devices, volumes**, and **MFDs**. Generally, the term **partition** is used throughout this book to refer to the logical division of physical disks, or spindles, into file system partitions.

Disk Preparation

Unlike storage provided by main memory, disk storage must be prepared to conform to the addressing method used by PRIMOS and must be given a name before it is used. This preparation is called **formatting, partitioning,** or creating a disk, or partition. It is generally performed by an Operator using a PRIMOS utility called MAKE.

The MAKE Utility

This section presents a short, introductory description of how to use MAKE; more detailed information is presented in Chapter 5, Formatting Disks – MAKE.

To prepare a disk, you must first determine how large the partition is to be and where it is within the physical disk, or spindle, so that you can determine the partition's physical device number. (Physical device numbers are discussed in detail in Chapter 3.) You use the DISKS command to add the partition to the **Assignable Disks Table.** This table, which is kept by PRIMOS, contains a list of physical device numbers of partitions that may be specifically assigned to a single user. Then you use the ASSIGN command to assign the partition to yourself and you use the MAKE utility to create the partition. As part of the MAKE process, you also name the partition. Once you have prepared a user disk, or partition, that partition conforms to system requirements for use by PRIMOS. In addition, MAKE creates these file system objects on the partition:

- The files BOOT, DSKRAT (which gets as its filename the name of the partition) and, conditionally, BADSPT or DYNBSP, if there are any defects, or **badspots**, on the partition
- The top-level directory MFD
- The empty top-level directories CMDNC0 and DOS

You may also have to create a partition that will be used for paging. PRIMOS uses **virtual memory** to run programs. This technique allows the computer to run a program that is larger than main memory. Portions of the program that are not being used may either reside in main memory or be moved out of main memory onto disk storage. This operation of moving parts of programs to and from main memory is performed automatically by PRIMOS and is called **paging**. Even a program that may fit into main memory when run by itself is paged by PRIMOS when other users are also running programs. Paging thus makes the available memory appear much larger than it actually is. PRIMOS must have at least one paging partition.

You may also want to create **crash dump disks.** These are partitions similar to paging partitions that are used to store a dump of main memory in the event that the system crashes. The crash dump disk helps you to recover from a crash automatically and efficiently.

Chapter 5 tells you how to create paging partition and crash dump disks. For more detailed information on virtual memory and paging, see the *System Architecture Reference Guide.* For an introductory discussion of this subject, see the *Operator's System Overview.*

Disk Maintenance

Disk maintenance consists of checking the quality of data on disks and using utilities designed to make any necessary restorations to the file structure.

The FIX_DISK Utility

To provide an efficient and thorough way to check the integrity of data on a PRIMOS disk, PRIMOS provides a file maintenance program, FIX_DISK. The purpose of FIX_DISK is to repair file structures that may have been damaged by a power loss or by an unexpected system halt or hang. In addition, FIX_DISK is also used to repair file structures that have been damaged by using PRIMOS II to make changes to files or directories on Rev. 19.0 partitions. Using PRIMOS II for these purposes results in incorrect Access Control Lists (ACLs) or quota information that require fixing. If PRIMOS finds that the file structure has been damaged, it asks you to run FIX_DISK.

Note PRIMOS II (DOS) will *not* write on partitions formatted with Rev. 20.0 or subsequent revisions of MAKE. It will read these partitions and will continue to both read and write Rev. 19.0 and earlier partitions. You should not use PRIMOS II on Rev. 20.0 and later partitions.

FIX_DISK reads every physical record in every file, top-level directory, and segment directory (SEGDIR) and checks to see that the information is consistent. If discrepancies are found, FIX_DISK generates error messages. You can request FIX_DISK to repair damaged files and quota information. You can also request FIX_DISK to convert a partition made as an older revision partition to a more recent revision partition. For details and examples, refer to Chapter 6.

File System States

The file systems on your disks can be in one of six states: clean/shutdown, clean/added, corrupt/shutdown, corrupt/added, corrupt/past, and corrupt/corrupt. These states are a result of when and if the system halts and when you run FIX_DISK with the –FIX option. Figure 1-1 shows these states and how the file system arrives at a particular state. Note that in the two states shown on the left of the figure, the file system is available to users, whereas in the four states on the right of the figure, the file system is unavailable to users.

Bits in the DSKRAT indicate the state (clean or corrupt) of the file system to PRIMOS and are set appropriately whenever the partition is shut down (either normally or by a system halt) or is added to the system.

As shown in Figure 1-1, when you first create a partition, the file system is clean and shutdown. It remains clean when you add it for file system access. On a subsequent normal shutdown, the partition returns to the clean/shutdown state.

If the system halts while the clean partition is added, the partition is now shutdown and possibly corrupt (corrupt/shutdown). If you run FIX_DISK with the –FIX option after the halt, the partition returns to the clean/shutdown state and you can add it to the system and it will be in the clean/added state.

If you add the partition to the system without running FIX_DISK after a halt or other abnormal shutdown, the partition is in a corrupt/added state. If you now shut down the partition in a normal manner, the partition is not newly corrupt but it is indicated as being corrupt at some time in the past (corrupt/past). If you now add the partition to the system again, it returns to the corrupt/added state. A halt in this state causes the partition to appear to PRIMOS as corrupt at present and corrupt in the past (corrupt/corrupt).

Running FIX_DISK with the –FIX option on the partition whenever it is in any of the corrupt states returns the partition to the clean state. Prime recommends that you run FIX_DISK on a partition whenever it is shut down abnormally or whenever a message from PRIMOS indicates that you should run FIX_DISK.

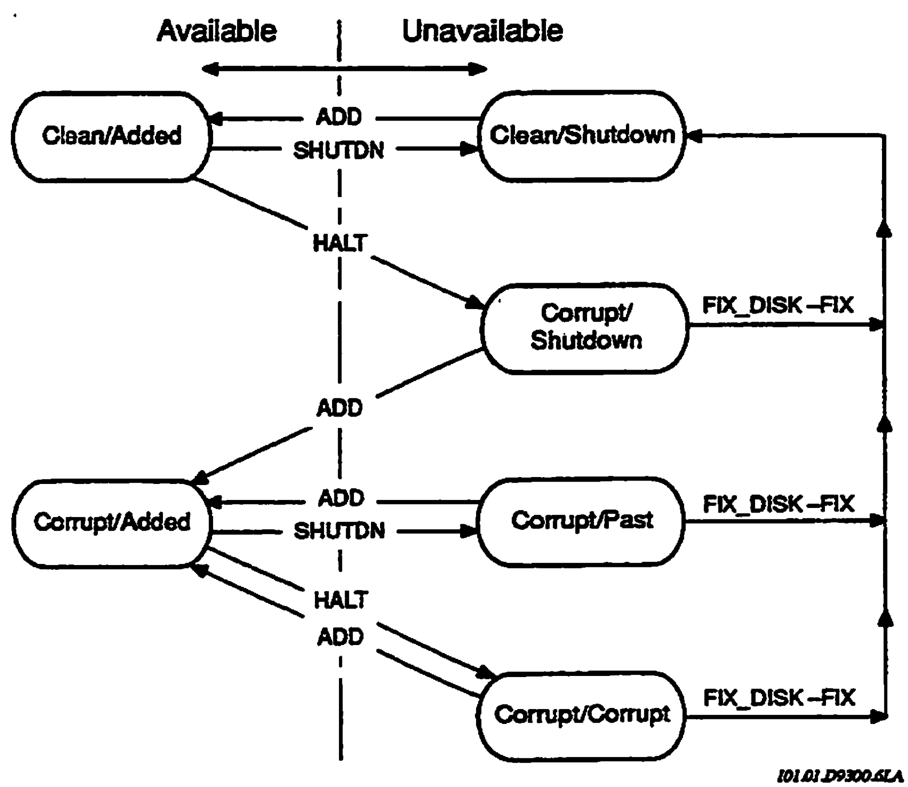


Figure 1-1. File System States

FIXRAT (Pre-Rev. 19.0 Disks)

FIXRAT is an older file maintenance program. FIXRAT checks file integrity on pre-Rev. 19.0 disks. However, FIX_DISK can be used on pre-Rev. 19.0 disks and it is recommended that you use FIX_DISK to repair partitions of any revision.

WARNING FIXRAT *must not* be used on any partition that is made as a Rev. 19.0 or later partition. Use of this command on Rev. 19.0 or subsequent revision partitions may result in the loss of data. To repair partitions of any revision, use FIX_DISK.

Disk-error Messages

When disk errors are detected by a disk controller because of read errors or write errors, messages are sent to the supervisor terminal and/or to a log file, as configured by DSM. The format and meaning of these messages are discussed in Appendix D.

File System Concepts

The remainder of this chapter gives you a brief overview of the structure of the PRIMOS file system. Although this information is not necessary for you to perform your duties as an Operator, it will help you to understand how the file system works.

Purpose of the File System

The purpose of the file system is to simplify the manipulation of large quantities of data. The file system

- Automatically allocates disk storage space for files
- Efficiently stores and retrieves data on behalf of the user
- Provides a file system object management service to allow users to cluster related information together and to refer to the information in files by name

Allocating Disk Storage Space – the DSKRAT: In order to allocate disk storage space automatically, PRIMOS maintains a file on each disk, the **Disk Record Availability Table (DSKRAT)**, whose name is the partition name. This file contains a header, which describes the partition, and a bit map, which indicates the status (used or unused) of every physical record on the disk and, thus, is a record of available space on that disk.

The DSKRAT file gets the partition name when you create the disk with the MAKE command. For example, if the name of a partition is ZEPHYR, the name of the DSKRAT file for this partition is also ZEPHYR. Each record on the partition is represented in the file by a single binary bit; a one (1) means the record is available, and a zero (0) means it is in use. PRIMOS uses this information to allocate disk space automatically.

Referring to Files by Name: In order to reference files by name, you must select the desired file by giving PRIMOS or its utility programs a string of alphanumeric characters that constitute the filename. The file system reserves one special file as a directory, the **Master File Directory (MFD)**. The MFD contains the names of other files and their locations on the partition. PRIMOS can find the MFD readily, because both its name (MFD) and its location on the partition are always the same.

Clustering Related Information: Clustering of related information is achieved in two ways. The first way is to have many file directories; this approach allows similar files to have their names and locations saved in one file directory. The second way is to allow nested file directories so that a file directory may contain names, not only of files, but also of other file directories. Thus, each user may divide files into appropriate groups and subgroups. File directories also provide some degree of access protection to the files contained within them. To examine the files in a directory, the user must have appropriate access to that directory. File directories are discussed in the next section.

File Directories

Directories are specialized files containing entries that point to files or to other directories. PRIMOS uses a hierarchical file structure, also known as a **tree structure**, that allows directories to be nested as the user wishes. See Figure 1-2.

Directories are the nodes in the file system tree structure; files are the branches. Each partition has one special directory called a Master File Directory (MFD), which contains an entry for each top-level directory in the MFD. Directories are either top-level directories, subdirectories, or segment directories. (See the following discussion.) In turn, each top-level directory contains an entry for every file and subdirectory in that top-level directory. Top-level directories and MFDs are accessed by PRIMOS in much the same way as other files.

Master File Directory (MFD): MFD entries include an identification for some special files that have unique use in the file system and are not normally accessed by a user. These files are BOOT, the DSKRAT, BADSPT, DYNBSP, and MFD, and are discussed in Chapter 5.

Top-level Directory: A file that contains a list of subordinate directories and files by name is a **top-level directory**. The top-level directory is the point from which the user accesses those subordinate directories and files.

Each user is assigned a directory, and that directory is the initial attach point (IAP) to the system when the user logs in. The directory contains an entry for every file and subdirectory created by the user. Each entry includes a filename and the address of the first physical record of the file. This address is the **beginning record address (bra)** of the entry.

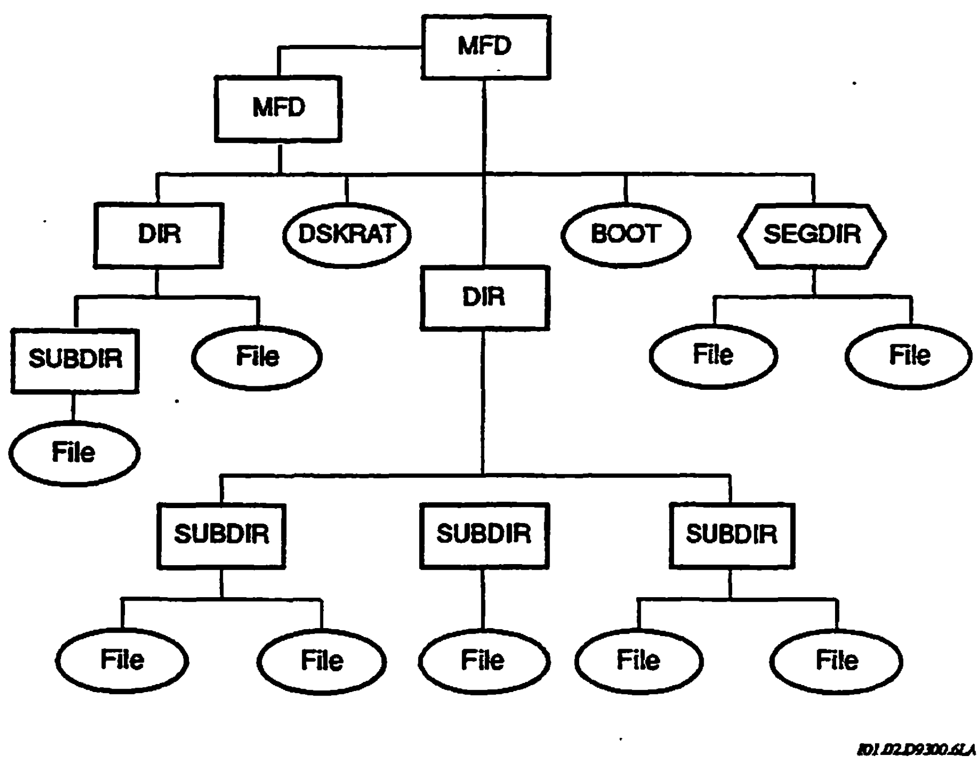


Figure 1-2. Prime File System Tree Structure

Segment Directories: Segment directories (SEGDIR) are similar to file directories except that entries in segment directories are identified by single entry numbers ranging from 0 through 65535. Segment directories are therefore limited to 65536 (64K) entries. These directories are generally referenced by programs rather than by users.

Segment directories differ from top-level directories in one fundamental respect: they contain file numbers rather than file names. In terms of the file system, the files in a segment directory have no symbolic names. However, the user may refer to files within a segment directory by their entry number, which can range from 0 to 65535, such as

1
2
99
185

The preceding four numbers are names of files in segment directories. Segment directory entry numbers are decimal numbers.

Hashed Directories: Since Rev. 20.0, the MAKE utility creates disk partitions that support **hashed directories**. The term **hashed** refers to the way in which PRIMOS gains access to data controlled by that directory. Hashing speeds up processing because the system does not have to perform a sequential search to locate a file system object.

The process of hashing involves converting the name of a file or directory to a number, known as its **hash index**. The hash index in turn is used to look up the list of names that have this index. Then, the names are compared. The system can then locate the file system object using this hash index. All hashed directories are Direct Access Method (DAM) files with an index block for the records in the directory.

Only ACL directories can be hashed. Password directories can still be created but they will not be hashed.

Note Rev. 22.1 and later PRIMOS continue to support pre-Rev. 22.1 format partitions, but a Rev. 22.1 format partition must not be locally connected to a pre-Rev. 22.1 system because of differences in the DSKRAT headers. A Rev. 22.1-format partition, however, may be accessed remotely by a pre-Rev. 22.1 system. The format of partitions has not changed since Rev. 22.1 so all partitions since Rev. 22.1 are Rev. 22.1 format partitions.

File Structures

File storage on a disk is composed of many separate blocks of data-recording space called disk records, or sectors. How these records are put together to make a file can greatly affect the efficiency of data retrieval from a file. To allow for different uses of files, the file system has three different ways of linking disk records to form a file:

- The **Sequential Access Method** (SAM) is very efficient for sequential access to a file. However, this method is much slower for repeated random positioning over a file because each record must be accessed sequentially.
- The **Direct Access Method** (DAM) is efficient for random file access to medium size files (less than 512 records) but uses additional disk records for file index information to allow direct access to each record.
- The **Contiguous Access Method** (CAM) is efficient for random file access to large size files (more than 512 records). A CAM file requires one more disk record than a SAM file requires for file index information.The primary benefit of CAM files, however, is that records for them are pre-allocated and fragmentation of files on disk is reduced considerably. (See Chapter 7 for a discussion of fragmentation.)

Thus, SAM, DAM, and CAM files differ in performance and storage efficiency but are functionally equivalent in all other respects. The structural differences among these file types are transparent to the user.

SAM Files: SAM file organization is the basic way of structuring disk records into a sequentially ordered set. A SAM file is a threaded list of physical disk records in which each record points to the preceding record, points to the next sequential record, and points to the first record in the file, as shown in Figure 1-3.

Thus, a SAM file is a collection of disk records chained together by forward and backward pointers. Each record in a SAM file (or in any other type of file) contains a pointer to the Beginning Record Address (BRA) of the file. The first record has a pointer (root, or parent, pointer) to the directory in which this file is an entry. The file system maintains the record headers and is responsible for the structure of the records on the disk.

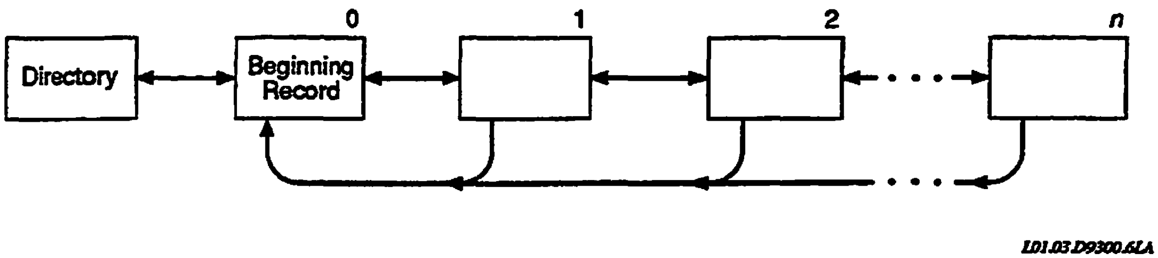


Figure 1-3. SAM File Structure

DAM Files: DAM file organization uses the SAM file method of making a sequentially ordered set but uses a special technique to rapidly access any data record directly:

- The first record of a DAM file is reserved for use by the system. No user data is ever written in this record, which is always the top-level index. An index is a list of pointers either to subindexes or to the records in a DAM file. See Figure 1-4 for a graphic representation.
- The top-level index is always exactly one record long. If the file is short (less than 512 records), the record address pointers point to records containing user data. If there are more than 512 data records, a multilevel index is maintained so that any record in the file can be directly accessed.

CAM Files: Whereas each SAM and DAM file contains records that are allocated wherever space is available (and therefore may not be contiguous), a CAM file contains groups of contiguous records only. These groups of contiguous records are called extents and are indexed by the CAM file's extent map. The extents themselves are not necessarily contiguous; only the records within each extent are contiguous. An extent is a variable number of contiguous 2048-byte records.

Figure 1-5 illustrates the structure of a CAM file. For each extent, the extent map records the address of the first record in the extent and the length of the extent, that is, the number of records in the extent. The extent map occupies the first record in the file.

Within each extent, the records exist in sequential order. Because of this order, CAM files require less indexing information than DAM files unless the CAM file is fragmented into many small extents. A DAM file index contains one disk address for each record. The CAM file's extent map contains one disk address for each extent, which means there is one pointer for many records.

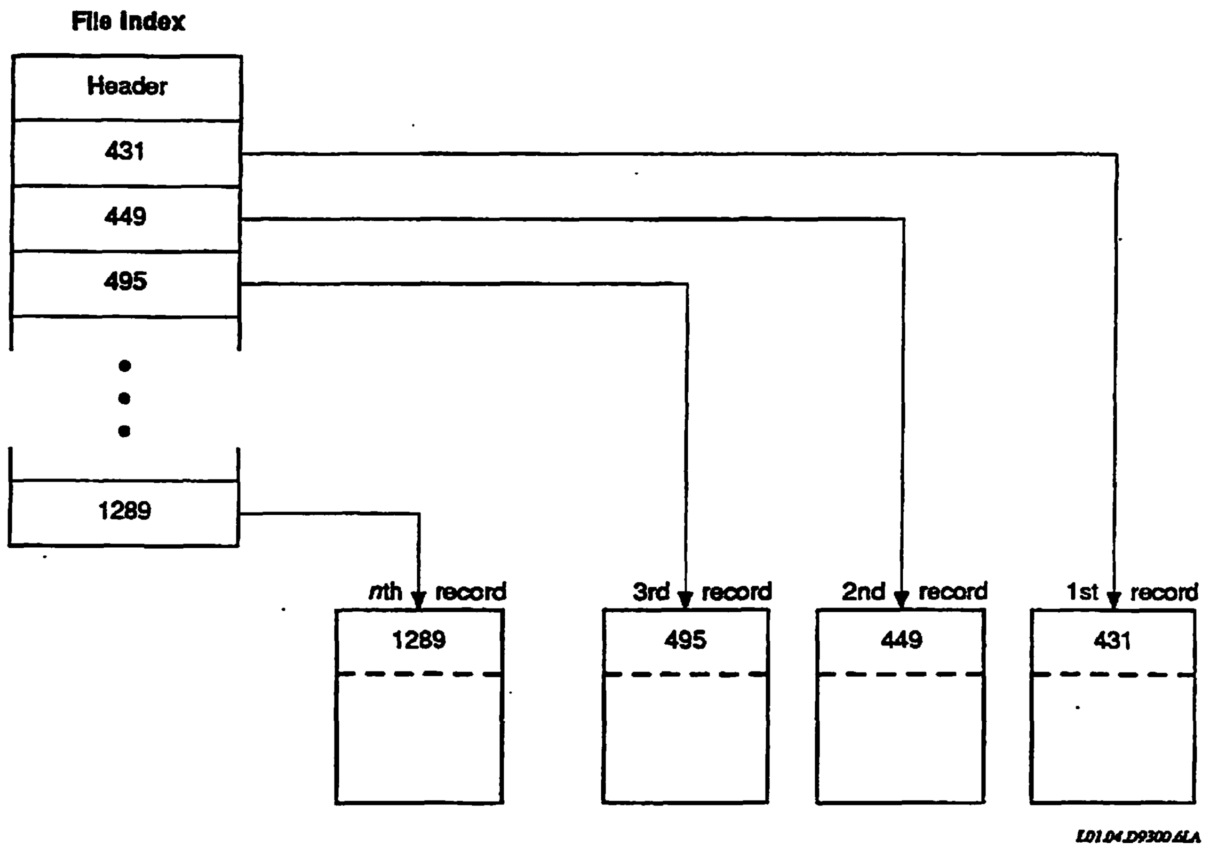


Figure 1-4. DAM File Structure

By using fewer pointers, PRIMOS can access a record in a CAM file more efficiently than in a DAM file that contains more than 512 randomly placed records. For example, if a program needs data in record 974 of Extent Map 1, as shown in Figure 1-5, PRIMOS reads the extent map into memory when the file is opened and then the effect is as if PRIMOS reads only two entries in the extent map and then moves to the second extent. Because extents contain only sequentially ordered records, PRIMOS moves automatically to the 51st record in the extent to find record 974. If this file were a DAM file, PRIMOS would have to read two index blocks and then find the pointer to record 974 in the second index block.

The real benefit of CAM files is that, since records are allocated in contiguous blocks, there is less fragmentation of CAM files and, thus, less seeking is required of the disk drive.

CAM files have various numeric parameters. Although all records are equal in size, the number of records per extent can vary from 1 to 32767. The maximum of 32767 records per extent is a software limitation. In actual use, however, the maximum number of records is also limited by the number of consecutive records that are available on the disk.

The number of extents per file can range from 0 to a maximum of 340, inclusive, on systems running Rev. 20.0 through Rev. 22.0 PRIMOS. At Rev. 22.1, the number of extents per file is unlimited for both standard and robust partitions if the partition is created by using Rev. 22.1 or later MAKE or if you convert a Rev. 22.0 partition to Rev. 22.1 with Rev. 22.1 or later FIX_DISK. (The actual maximum number of extents at Rev. 22.1 is 16381.) Each extent map for a CAM file can index a maximum of 340 extents; thus, a CAM file can have multiple extent maps. As shown in Figure 1-5 file with the maximum number of extents with 340 extents indexed in each extent map would have 49 extent maps.

It is not possible to create or access CAM files by using pre-Rev. 20.0 versions of PRIMOS.

Figure 1-6 schematically shows the relationship among the various file system object types discussed in the preceding sections.

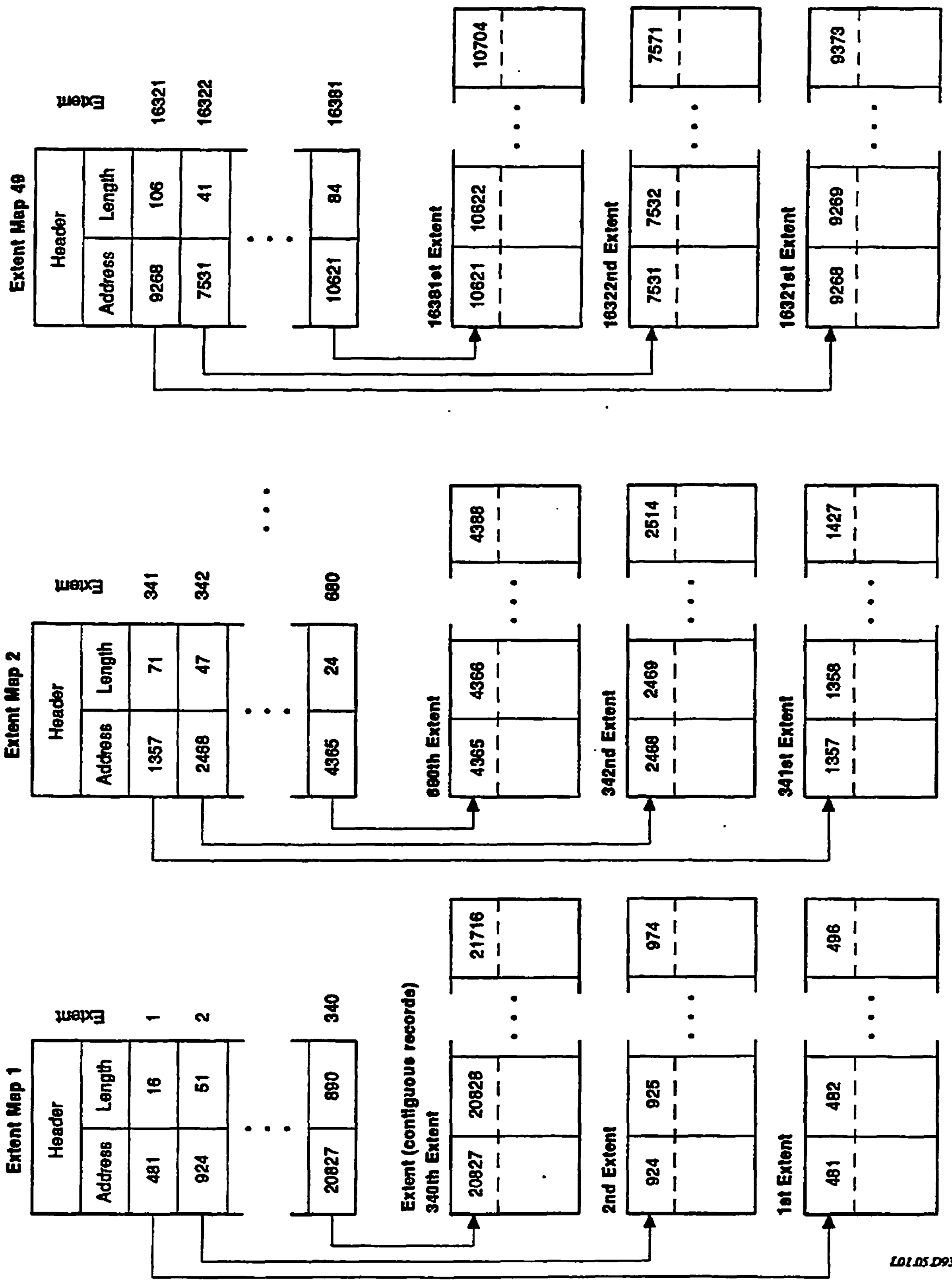


Figure 1-5. CAM File Structure

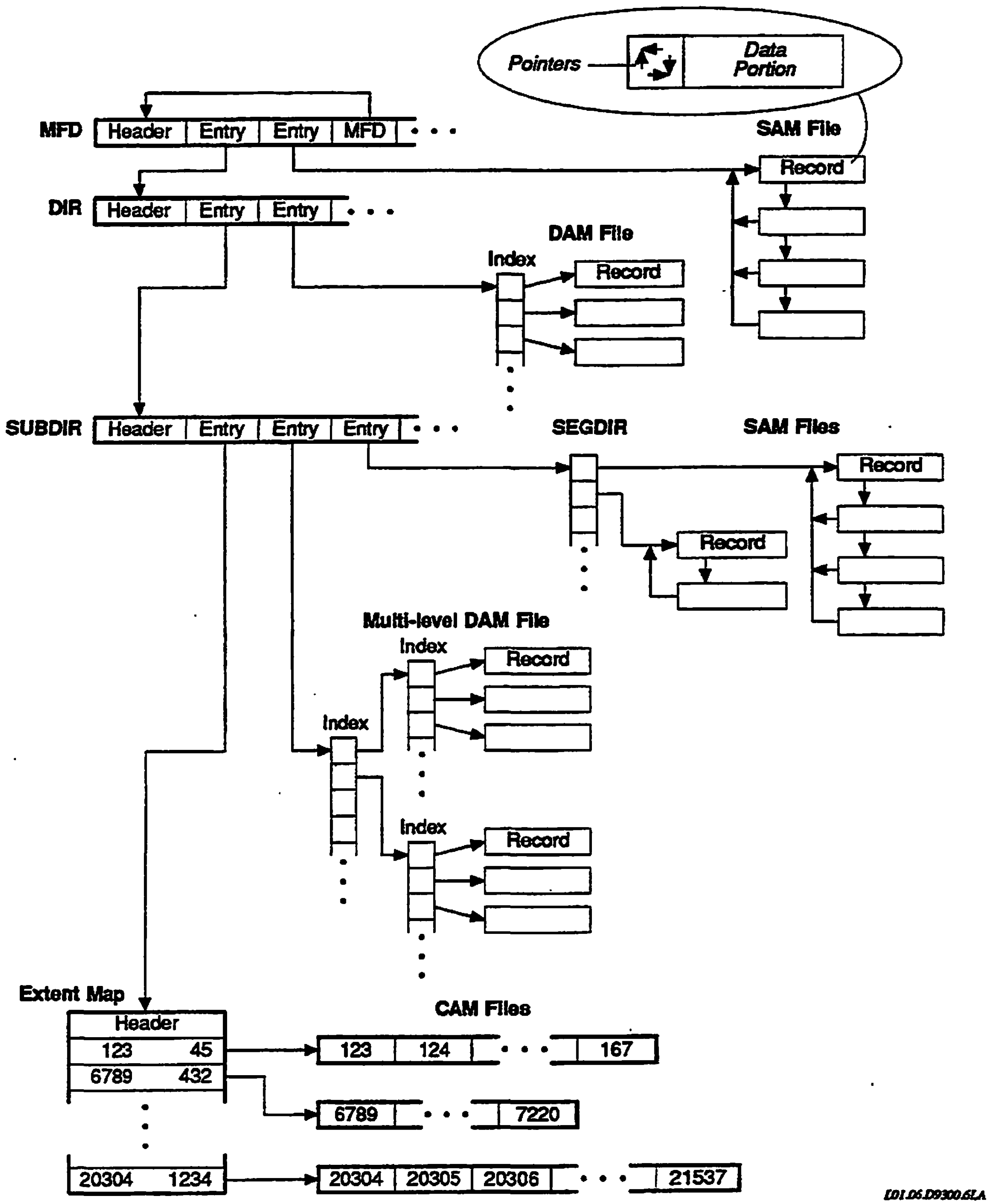


Figure 1-6. Relation Among File Structures

Physical Disks

2

This chapter briefly describes the types of disks that Prime systems support and introduces some of the terminology associated with the use of these disks. It then goes on to describe badspots and the utilities that use badspot information.

What Are Physical Disks?

The term **physical disk**, or **spindle**, refers to an entire multi-surface disk pack or disk drive. **Disk geometry** refers to the physical attributes of a physical disk such as the number of tracks, or cylinders, per surface, the number of records, or sectors, per track, and the number of surfaces, or heads, making up the disk. The physical disk has a medium on which data is written. The medium may have badspots, or physical defects, on it that prevent data from being written to or read from those areas of the disk.

Several of the MAKE and FIX_DISK options involve handling badspots. The following sections explain what badspots are and how MAKE and FIX_DISK handle them. For specific information on each of MAKE's badspot handling options, see Chapter 5. Similarly, for specific information on each of FIX_DISK's badspot handling options, see Chapter 6.

Types of Disks

Prime systems support three basic types of physical disks, or spindles:

- Storage Module Disks (SMDs)
- Fixed-Media Disks (FMDs)
- Cartridge Module Devices (CMDs)

Storage Module Disks (SMDs)

SMDs consist of two units: the **disk drive** and the **disk pack**. The disk pack is a covered stack of disks, or platters, that contain the actual data on their surfaces. This disk pack is mounted in the disk drive so that the data can be accessed and changed (read and written). The disk drive contains all of the circuitry and equipment to manipulate data on the disk pack. Because the disk pack may be removed from the disk drive, vast amounts of data can be stored on many disk packs, which can then be alternated among a few drives. In addition, because of this removal capability, disk-to-disk backups are possible.

Fixed-Media Disks (FMDs)

Fixed-Media Disks (FMDs), also called **Winchester disks**, consist of essentially one unit: the disk drive. The platters within these disk drives are not removable. Permanent sealing of the disk platters into the disk drive enables FMDs to store larger amounts of data than SMDs or CMDs can store in the same amount of space. This increased storage density is possible because tolerances can be made much smaller on these sealed drives.

Cartridge Module Devices (CMDs)

CMDs include a removable pack of one usable surface. However, the disk drive for a CMD also includes one or more nonremovable platters. CMDs generally store less data than either SMDs or FMDs, because they have fewer surfaces.

Badspots

Vast quantities of data can be stored on a disk. Portions of a disk, however, are sometimes unable to store data reliably. These portions of a disk are called **badspots or flaws**. PRIMOS does not store data onto badspots.

While creating a disk, MAKE searches it for badspots by writing test data onto every record on the disk and reading the test data back. In addition, if you (or your System Administrator) already know of badspots on the disk and their location, MAKE allows you to enter the locations of the badspots. Some disks also contain a **flaw map** written to a reserved area of the disk; MAKE can read the flaw map and then create a **badspot file** with the information from the flaw map.

MAKE uses the test results, user input, and the disk flaw map, if it exists, to build a list of badspots on the disk. MAKE writes this list on the newly created partition in a file named **BADSPT** (the badspot file). On partitions that have no badspots, MAKE does not write a badspot file.

MAKE also creates a file called the **DSKRAT** (Disk Record Availability Table), which contains a list of all records on the partition. The information in this file indicates whether a record is in use by the file system. A single bit in this file corresponds to each record in the partition and is set (=1) to indicate that the record is available or reset (=0) to indicate that the record is in use. PRIMOS uses the DSKRAT either to determine the status of a record or to change the status when users create, delete, extend, or shorten files.

When MAKE creates the DSKRAT, most of the records on the disk are free for use. To prevent PRIMOS from using badspots on the disk, MAKE initializes the DSKRAT so that all badspots are marked as being in use. When PRIMOS needs to find an unused record, it skips over badspots.

Dynamic Badspot Handling

Generally, all physical disks have badspots. Historically, **static badspots** (badspots that exist on a disk when it is first created) have been handled in these ways:

- Found by MAKE when you partition a disk
- Entered by you when running MAKE or FIX_DISK
- Read by MAKE from a vendor flaw map

Dynamic badspots (DBS) are those static badspots that are marginally defective and are missed by MAKE or those badspots that are actually developing due to disk media degradation. FIX_DISK handles these badspots only when a partition is shut down for repair, not while the partition is in operation.

It is now possible for a disk controller to handle all badspots and to do so while the disk is in operation. This is referred to as **Dynamic Badspot Handling**. Dynamic Badspot Handling allows PRIMOS to access an apparently error-free partition. Dynamic Badspot Handling can be done only by the Model 6580 (IDC1) **intelligent disk controller**, that is, disk controllers that are microprocessor-based.

Note Model 6580 (IDC1) intelligent disk controllers are programmable and are downline loaded at cold start with Intelligent Channel Order Protocol (ICOP) software. This software enables the intelligent disk controller to operate either in Dynamic Badspot Handling (–DBS ON or –IC) mode or in Nondynamic Badspot Handling (–DBS OFF or –AC) mode. In either mode, the controller acts as an intelligent disk controller. If the controller is not downline loaded with ICOP software, it functions as a nonintelligent disk controller in 4005 mode.

The IDC1 also handles original, or static, badspots (those supplied by a vendor flaw map on the disk, those supplied by the badspot file BADSPT, or those that have been manually entered with MAKE or FIX_DISK). The controller remaps these badspots using the same technique as for dynamically occurring badspots. The controller gets these original badspots from the dynamic badspot, or DBS, file on the disk. The DBS file contains the addresses of all known badspots on the entire physical disk. It also contains the addresses of the records in the **remapped area**, or RMA, that are available for remapping badspots.

The RMA is a reserved area of the spindle designed to contain file records that are originally addressed to a badspot. The RMA also contains records that are available for new, or dynamically occurring, badspots.

Note The discussion of Dynamic Badspot Handling applies only to disks that support this method of badspot handling and are connected to a Model 6580 intelligent disk controller (IDC1), as described in Chapter 5. SCSI disks connected to Model 7210 and Model 2382 disk controllers handle badspots within the disk drive itself and this discussion of badspots does not apply to them.

For a further discussion of dynamic badspot handling by Model 6580 (IDC1) disk controllers, see Chapter 8.

Caution The need to remap a badspot on a Model 4719 disk in a 2450™ system is extremely rare. If, however, the controller must remap a badspot on a Model 4719 disk while the system is in operation, the system may appear to be hung. The reason is that PRIMOS allows the controller as much time to do the remapping operation as the controller may need in the worst case. Allow up to sixty seconds to pass before you attempt to do anything. When the controller has finished remapping a badspot, a message is displayed indicating that a record has been remapped. If no message is displayed and more than a minute has passed, the system is hung for some other reason and you will have to take action. See the section on halts and hangs in your CPU handbook.

Utilities That Use Badspot Information

When a copy of a partition is made for backup purposes, the copy process may or may not utilize the file system management of PRIMOS. A **logical save** (for example, using MAGSAV) utilizes this management; a **physical save** (for example, using PSR) does not. A physical save may be faster than a logical save if the partition is nearly full.

Physical Saves: Unlike a logical save, which copies files and directories individually, a physical save copies the entire partition, record by record, as illustrated in Figure 2-1. It does not involve the DSKRAT on the disk to which the data are being copied (the *target* partition), because all information on that partition is overwritten with the data on the partition from which the data are being copied (the *source* partition). However, the BADSPT file must be preserved on the target partition, because the lists of badspots on the source and target partitions will differ.

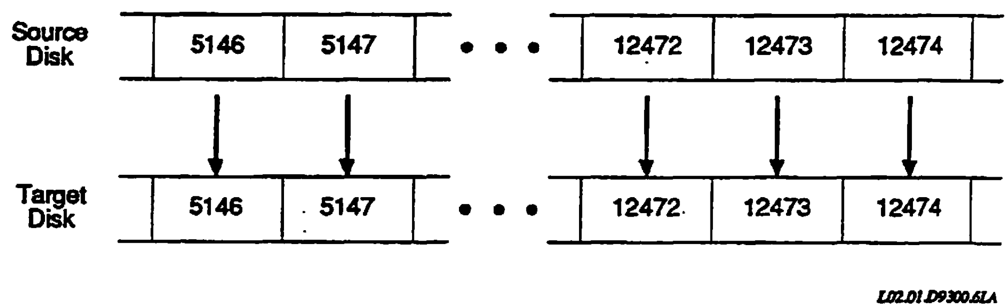


Figure 2-1. Ideal Physical Copy

If there are badspots on the target partition, the program performing the copy (PSR) cannot copy records from the source partition to the corresponding records on the target partition when those records correspond to badspots on the target partition. Therefore, another location for these records must be found on the target partition.

To find this location, when PSR needs to copy a record whose corresponding record on the target partition is a badspot, it temporarily adopts the characteristics of a logical save operation. It uses the DSKRAT of the source partition to find a record that is not being used on the source partition and that therefore does not need to be copied. Then it uses the corresponding record on the target partition as a home for the record that it could not copy before, as illustrated in Figure 2-2. If this utility cannot find a free record, it displays an error message.

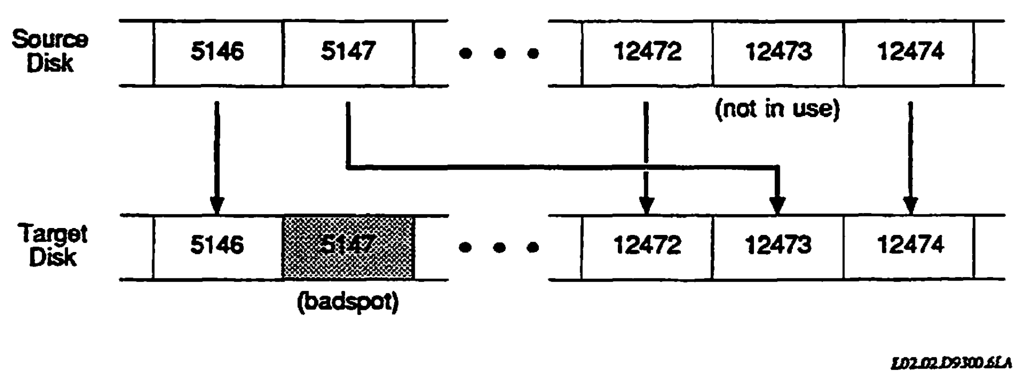


Figure 2-2. Physical Copy With Badspot Handling

Note that once a record has been redirected in this manner, the target partition must not be used under PRIMOS. As shown in Figure 2-3, this limitation exists because there are other records on the partition that point to, for example, record number 5147, when in fact the data stored in record number 5147 on the source partition was copied to record number 12473 on the target partition. Before the target partition can be used, all references to remapped records (such as record number 5147) must be changed to point to the actual records (such as record number 12473). FIX_DISK performs the task of changing these pointers as illustrated in Figure 2-3.

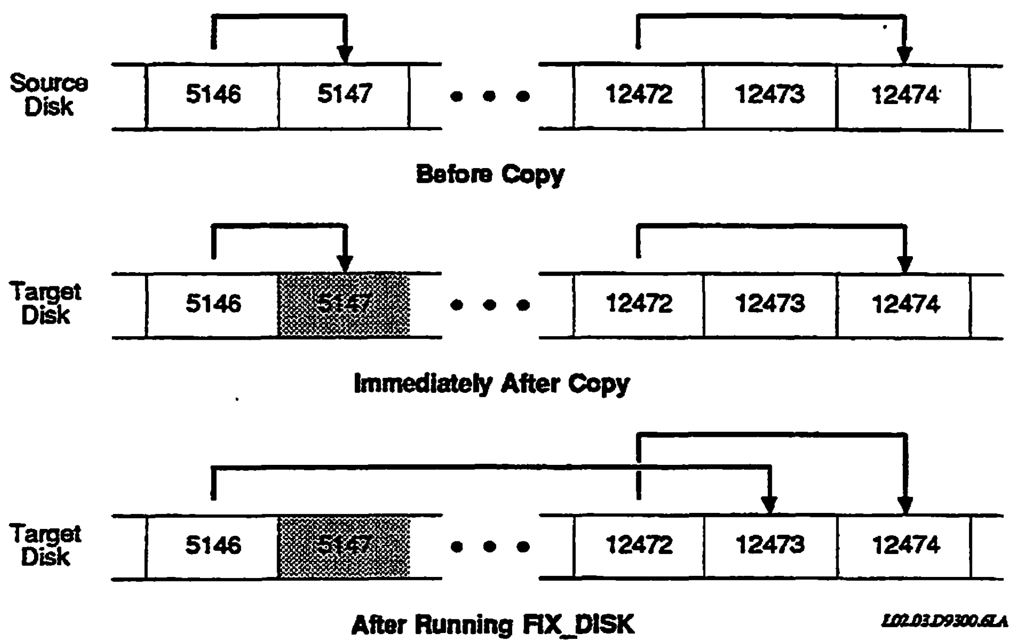


Figure 2-3. Badspot Handling and File Pointers

For FIX_DISK to know where remapped records (such as record number 5147) were copied to (such as record number 12473), PSR creates **equivalence blocks**. These equivalence blocks tell FIX_DISK which records were copied and to where the records were copied.

Equivalence Blocks

An **equivalence block** describes where one record is actually stored on the partition. There is one equivalence block for every record remapped by PSR on a partition. These equivalence blocks are stored in the BADSPT file of the target partition. They are created by PSR in order to indicate that badspot handling has taken place for the partition to which data were copied (the target partition). Until the equivalence blocks are deleted by FIX_DISK, the partition must not be used for any purpose; you must run FIX_DISK on it before using it.

Need for Creating Equivalence Blocks: When PSR is copying to a disk, it copies record number 1 from the source partition to record number 1 on the target partition, record number 2 to record number 2, and so on. (Even though PSR is restoring data from tape, the tape contains a record-by-record image of the source partition; therefore this explanation still applies.) If a record is a badspot on the *source* partition, the record is not copied. However, a record that is to be copied to a badspot on the *target* partition presents two basic problems:

- The data in the record must be saved. The record is not a badspot on the source partition; therefore, it contains data if it is in use (as indicated by the source partition DSKRAT). However, it cannot be written to the same place on the target partition, because that spot is bad and will not hold data.
- If the record is in use, other records on the source partition (and their copies on the target partition) contain pointers to the record. These pointers are record numbers.

To solve the first problem, PSR tries to find a different place in which to store the record on the target partition – an alternate record. This alternate record must satisfy the following requirements:

- It must not be a badspot on the target partition (although it can correspond to a badspot on the source partition).
- It must not correspond to a record that is in use by the source partition (as indicated in the DSKRAT of the source partition).
- It cannot already have been used as an alternate record for some other badspot on the target partition.

If an alternate record that satisfies these requirements is not found, PSR issues an error message. If an alternate record is found, the data in the original record is copied there (remapped).

Thus, when PSR performs this operation, it creates an equivalence block, indicating that the data intended for the original record has been stored in the alternate record instead. This brings us to the second problem: although the data are stored in the alternate record, all of the other records on the target partition identify the alternate record by its original record number (a badspot). There are no records that point to the alternate record number. FIX_DISK solves the problem by changing all file system pointers to point to this new record.

Results of the Creation of Equivalence Blocks: When PSR has finished copying to a partition on which it created one or more equivalence blocks, it displays the message

```
Badspots handled on partition pdev, please run FIX_DISK
```

This message tells you that before you try to use the ADDISK command on *pdev*, and before you try to copy *pdev* to another disk by using PSR, you must run FIX_DISK on *pdev*, using the –FIX option.

When you run FIX_DISK with the –FIX option on a disk that contains equivalence blocks in the BADSPT file, FIX_DISK updates all records on the disk that point to the original (bad) records so that they point to the corresponding alternate (good) records instead. FIX_DISK then deletes the equivalence blocks. When FIX_DISK is finished, the disk can be used.

3 Physical Device Numbers

Each disk or disk partition has a **physical device number (pdev)** that identifies the type of storage device, the drive unit on which it is mounted, the controller to which the drive unit is attached, the size of the partition, and the location of the partition on the disk. These physical device numbers are used in the following commands:

ADDISK	FIX_DISK	RECORD_TO_PATH
ASSIGN DISK	MAKE	SHUTDN
CDD	PSR	UNASSIGN DISK
DISKS		

Physical device numbers are also used in configuration directives such as COMDEV, PAGING, and others.

A physical device number is an octal number representing the setting of bits in a 16-bit halfword. The halfword is the address of a disk partition and the address allows PRIMOS to access the partition. The physical device number makes each disk partition physically and logically unique.

This chapter describes how to determine a physical device number for a given disk.

Types of Disks

As explained in Chapter 2, PRIMOS supports three varieties of disks: Storage Module Disks (SMDs), Fixed-Media Devices (FMDs), and Cartridge Module Devices (CMDs). PRIMOS can use these disks for file storage and for paging or crash dump area. Each variety is available in two or more sizes.

Size of a Disk Partition

The size of a disk partition is measured in records. The number of records on a disk (and the amount of usable storage per surface) depends on the type of disk.

Table 3-1 summarizes these values for all disks that are fully supported at Rev. 23.3.

Table 3-1. Disk Size Data

Disk Type	*Model Number*	*Number of Surfaces*	*Records per Surface*	*Total Number of Records*	*Usable MB*	*Removable*
80MB SMD		5	7407	37035	75.85	yes
300MB SMD		19	7407	140733	288.22	yes
60MB FMD	4711	4	7140	28560	58.49	no
68MB FMD		3	10071	30213	61.88	no
84MB FMD	4714	5	8120	40600	83.15	no
120MB FMD	4715	8	7140	57120	116.98	no
158MB FMD		7	10071	70497	144.38	no
160MB FMD		10	7389	73890	151.33	no
213MB FMD[4]	4730	31	3335	103378	211.72	no
258MB FMD	4719	17	7320	124440	254.85	no
315MB FMD[1]	4475	19	7407	140733	288.22	no
328MB FMD	4721	12	13128	157536	322.63	no
328MB FMD[3]	4721	31	5080	157480	322.52	no
421MB FMD[4]	4731	31	6530	202438	414.59	no
496MB FMD	4735	24	9954	238896	489.26	no
673MB FMD[4]	4729	31	10447	323850	663.25	no
675MB FMD[2]		40	7569	302760	620.05	no
770MB FMD	4845	23	16112	370576	758.94	no
817MB FMD	4860	15	26201	393015	804.89	no
1.34GB FMD[4]	4732	31	20746	634128	1317.13	no
32MB CMD		2	7407	14814	30.34	1 surface
64MB CMD		4	7407	29628	60.68	1 surface
96MB CMD		6	7407	44442	91.02	1 surface

Notes [1]Sometimes referred to as a 300MB FMD.

[2]Sometimes referred to as a 600MB disk or as a 630MB disk. The official name is 675MB disk.

[3]The 328MB FMD is partitioned with 31 logical surfaces when connected to the Model 7210 SCSI disk/tape controller.

[4]The 213MB, 421MB, 673MB, and 1.34GB FMDs operate with the Model 7210 SCSI disk/tape controller and are partitioned with 31 logical surfaces. The number shown in the table above is an average number of records per surface. Table 3-2 shows the actual records per surface for these disks.

Table 3-2. Records per Surface for SCSI Disks

Disk	*Surfaces*	*Records per surface*	*Surfaces*	*Records per Surface*
213MB Model 4730	0–3	3556	4–30	3302
421MB Model 4731	0–21	6604	22–30	6350
673MB Model 4729	0–3	10668	4–30	10414
1.34GB Model 4732	0–20	20828	21–30	20574

For all of the disks in Table 3-1, a record holds 2048 bytes of user data plus 32 bytes for housekeeping information. The amount of usable data in megabytes (*Usable MB*) is based on 2048 bytes per record and is always less than the rated size of the disk. Approximately 5% to 8% of a disk's capacity is used for housekeeping information.

The *Removable* column in Table 3-1 specifies whether the disk drive supports removable disk packs. SMDs allow the entire disk pack to be removed. FMDs allow no such removal. CMDs allow the removal of one platter, which has one surface of usable information. The remaining surfaces (1, 3, or 5 surfaces) are not removable.

The size of a disk partition is specified as an integral number of surfaces. A disk partition may contain 1 through 31 surfaces. Because a partition cannot contain more than 31 surfaces, the 40 surfaces available on a 675MB disk cannot be contained by one partition. You must define at least two partitions in order to utilize the disk fully.

The surface offset, or the starting surface number, for a partition is always an *even* number, ranging from 0 through 30, inclusive.

Disk Drive Unit Numbers

Individual disk drive units are identified by drive unit numbers. A **drive unit number** is selected on the disk drive unit. During disk-related activities, the Operator supplies this number to PRIMOS as part of the physical device number. PRIMOS uses this drive unit number to identify the particular drive unit being referenced.

Disk drives are connected to disk controllers. The **disk controller** is a circuit board containing hardware and microcode that allow communication between the CPU and the drive unit, or disk storage device. A maximum of four or eight disk drives may be connected to each disk controller, depending on the particular controller, and one system may have a maximum of eight disk controllers. It is therefore possible for a system to have a total of 64 disk drives.

No two disk drive units connected to the same disk controller may have the same drive unit number. In addition, the disk controller has a specific address that becomes part of the identification PRIMOS uses to reference the disk media.

It is important to keep a record in the system logbook of disk controller addresses and numbers, drive unit numbers, and the physical device numbers for disk partitions mounted on these drives so that you have a ready reference to a permanent record of them.

Disk Drive Unit Numbers for SMDs and CMDs

Drive unit numbers for SMDs and CMDs are indicated and set by the removable buttons, or logical address plugs, on the front of the disk drive. These buttons, which have the drive unit number printed on them, are numbered 0 through 7. These removable buttons can be interchanged among drives, thus changing the logical addresses of the drives.

Disk Drive Unit Numbers for FMDs

A drive unit number for most FMDs can range from 0 through 7, inclusive. For FMDs associated with a Model 7210 SCSI disk/tape controller, drive unit numbers can range from 0 through 5 if the controller is in ICOP+ mode operating as a disk controller only and can range from 0 through 6 if the controller is in 4005 mode or is operating as a disk/tape controller.

The drive unit numbers for the Model 4475 FMD and the Model 4735 FMD are set by a removable button on the front of the drive. The Model 4845 FMD has a slotted-screw rotary switch behind the front removable access panel.

The Model 4860 has a button marked ADDRESS on the drive control panel. You hold this button in to cycle through the hexadecimal numbers displayed in the four LEDs marked 8 4 2 1 above the button. Since drive unit numbers can range only from 0 through 7, only numbers indicated by the LEDs marked

4 2 1 are valid. The drive unit numbers are indicated by addition of the numbers above lit LEDs; thus if LEDs 1 and 4 are lit, this indicates drive unit 5; if all three are lit, this indicates drive unit 7; and if none of the LEDs is lit, this indicates drive unit 0.

The SCSI disk drives in the Model 75500-6PK device module have push button switches that allow you to change the drive unit number. The switches are next to the drive unit number. To increase the number, simply press the switch labeled with a plus (+) sign; to decrease the number, simply press the switch labeled with a minus (–) sign.

Drive unit numbers for other FMDs cannot be set by the Operator. They are set internally by a PrimeService representative.

Construction of Physical Device Numbers

The remainder of this chapter discusses the construction of physical device numbers (pdevs). Tables 3-3 and 3-7 present pdevs for the disks supported by PRIMOS, and the text presents examples of their construction and use. This information is presented in the following tables and figures:

Table 3-3	Basic Physical Device Numbers for Storage Modules and Fixed-Media Devices
Table 3-4	Additions to pdev for Disk Drive Unit Number and Disk Controller Address
Table 3-5	Example Combinations of Controller Addresses and Disk Drive Unit Numbers for Basic pdev 103020
Table 3-6	Example of Basic and Adjusted pdevs
Table 3-7	Basic Physical Device Numbers for Cartridge Module Devices
Figure 3-1	Sample pdev Worksheet
Figure 3-2	Example of SMD Partitions
Figure 3-3	Construction of a Physical Device Number

Information Needed for Physical Device Numbers

Generally, the System Administrator supplies you with the following information necessary to construct a physical device number. Write this information in the system logbook.

- Size or model and type of disk (SMD and FMD, or CMD)
- Starting surface number

- Number of surfaces in the partition
- Drive unit number
- Controller address or number

From this information, you can construct a physical device number. The method used to construct a physical device number depends on the type of disk. Physical device numbers are specified in octal.

Physical Device Numbers for SMDs and FMDs

Table 3-3 summarizes all of the valid physical device numbers for SMDs and FMDs. It does not list numbers that, although valid, will result in an unused surface. These numbers result from using an odd number of surfaces in a partition that is not the last partition of a physical disk, or spindle. If there is a partition with an odd number of surfaces in the middle of a physical disk, the next surface is not used because the next partition must have an even numbered starting surface.

Note The numbers in Table 3-3 appear different from versions of this table prior to Rev. 21.0 because the basic pdev is now computed for the controller at address 24_8, whereas versions prior to Rev. 21.0 used the controller at address 26_8 to compute the basic pdev. Any basic pdevs you may already be using based on controller 1 at address 26_8 are still valid. You do not have to change basic pdevs.

Use of PRIMOS External Command PDEV

You can form a physical device number (pdev) by using the PRIMOS command PDEV. You invoke PDEV with this command format:

PDEV *options*

The PDEV command does the following:

- Lists the controller address, disk drive unit number, starting surface number, and number of surfaces of a given *pdev*. Use the –DECODE *pdev* option for this.
- Converts a given controller address, disk drive unit number, starting surface number, and number of surfaces into a *pdev*. Use –ENCODE –CONTROLLER *cc* –UNIT *unit* –STARTING_HEAD *s* –NUM_HEADS *n* for this.

  The abbreviation for each option is underlined in the above description.

- Return the logical device number (ldev), pdev, controller address and number, disk drive unit number, starting surface number and number of surfaces, or heads, of a named disk. Use –DISK for this. You can use wildcards with the name.
- As a command function, returns only the pdev. Enclose the function in square brackets and use the –ENCODE or –DISK option for this but you cannot use wildcards in the command function.
- Display the command usage. Use the –HELP option for this.

The argument to –CONTROLLER, *cc*, is the controller address (see Table 3-4), the argument to –UNIT, *unit*, is the disk drive unit number (0–7), the argument to –START, *s*, is the starting surface number, and the argument to –HEADS, *n*, is the number of surfaces in the partition. The following are examples of using PDEV both as a command and as a command function. You can use the command function in a command.

```
OK, PDEV -DECODE 65420
For PDEV 65420, controller = '24 (0), unit = 0, start head = 12, heads = 22

OK, PDEV -DECODE 103270
For PDEV 103270, controller = '27 (5), unit = 4, start head = 16, heads = 12

OK, PDEV -ENCODE -CTRL 45 -UNIT 6 -START 16 -HEADS 12
For controller = '45 (4), unit = 6, start head = 16, heads = 12, PDEV = 103234

OK, TYPE [PDEV -ENCODE -CTRL 23 -UNIT 5 -START 4 -HEADS 27]
126773

OK, PDEV -DISK QPUB@
Disk      LDEV   PDEV    Controller  Unit  Start  Heads
-------   ----  ------   ----------  ----  -----  -----
QPUBS7      6     2264    '27 (5)     2      0      8
QPUBS8      7    42264    '27 (5)     2      8      8
QPUB11     10   101665    '27 (5)     2     16      7

OK, TYPE [PDEV -DISK QPUB11]
101665
```

You should check all pdevs that you calculate using the PDEV command against the pdevs in Table 3-3. The PDEV command will allow you to calculate inappropriate pdevs. For example, you should not have an odd number of surfaces in a partition in the middle of a disk. In addition, PDEV will allow you to calculate pdevs that are beyond the range of the table. All valid pdevs are listed in Table 3-3. See the *Operator's Guide to System Commands* for details on the PDEV command.

Tabular Values for Physical Device Numbers

Table 3-3 shows all the valid physical device numbers for the following disks.

60MB (Model 4711)	213MB (Model 4730)	496MB (Model 4735)
68MB	258MB (Model 4719)	673MB (Model 4729)
80MB SMD	300MB SMD	675MB
84MB (Model 4714)	315MB (Model 4475)	770MB (Model 4845)
120MB (Model 4715)	328MB (Model 4721)	817MB (Model 4860)
158MB	421MB (Model 4731)	1.34GB (Model 4732)

Numbers in Table 3-3 ending with 1 result in specification of an odd number of surfaces in a partition. These numbers should be used only as the *last* partition on the following disks:

- 80MB and 300MB SMDs
- Models 4475, 4714, 4719, 4845, and 4860 FMDs
- Models 4721, 4729, 4730, 4731, and 4732 FMDs when they are connected to the Model 7210 SCSI disk/tape controller
- 68MB and 158MB FMDs

Using these numbers for partitions on other disks reduces the storage capacity of the disk by 7MB to 43MB per unused surface. Note also that no partitions can have a starting surface number greater than 30 and no partition can be composed of more than 31 surfaces.

To obtain the values you need to construct a pdev for SMDs and FMDs, use the following steps. Also consult Table 3-4 and Figure 3-1.

The steps for constructing a pdev are

1. Determine the number of surfaces in the partition.
2. Determine the surface number of the first surface in the partition – the starting surface number.
3. Look up the basic physical device number in Table 3-3.
4. For the disk drive unit, add the value shown in Table 3-4.
5. For the disk controller address or number, add the value shown in Table 3-4.

Table 3-3. Basic Physical Device Numbers for Storage Module Disks and Fixed–Media Devices

	Starting Surface Number																
Number of Surfaces in Partition	0	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	
1	--------	010021	020021	030021	--------	--------	--------	070021	100021	110021	--------	130021	--------	--------	--------	170021	1
2	000420	010420	020420	030420	040420	050420	060420	070420	100420	110420	120420	130420	140420	150420	160420	--------	2
3	000421	010421	020421	--------	--------	--------	060421	070421	100421	--------	120421	--------	--------	--------	160421	--------	3
4	001020	011020	021020	031020	041020	051020	061020	071020	101020	111020	121020	131020	141020	151020	--------	--------	4
5	001021	011021	--------	--------	--------	051021	061021	071021	--------	111021	--------	--------	--------	151021	--------	--------	5
6	001420	011420	021420	031420	041420	051420	061420	071420	101420	111420	121420	131420	141420	--------	--------	--------	6
7	001421	--------	--------	--------	041421	051421	061421	--------	101421	--------	--------	--------	141421	--------	--------	--------	7
8	002020	012020	022020	032020	042020	052020	062020	072020	102020	112020	122020	132020	--------	--------	--------	--------	8
9	--------	--------	--------	032021	042021	052021	--------	072021	--------	--------	--------	132021	--------	--------	--------	--------	9
10	002420	012420	022420	032420	042420	052420	062420	072420	102420	112420	122420	--------	--------	--------	--------	172420	10
11	--------	--------	022421	032421	042421	--------	062421	--------	--------	--------	122421	--------	--------	--------	--------	--------	11
12	003020	013020	023020	033020	043020	053020	063020	073020	103020	113020	--------	--------	--------	--------	163020	--------	12
13	--------	013021	023021	033021	--------	053021	--------	--------	--------	113021	--------	--------	--------	--------	--------	--------	13
14	003420	013420	023420	033420	043420	053420	063420	073420	103420	--------	--------	--------	--------	153420	--------	--------	14
15	003421	013421	023421	--------	043421	--------	--------	--------	103421	--------	--------	--------	--------	--------	--------	--------	15
16	004020	014020	024020	034020	044020	054020	064020	074020	--------	--------	--------	--------	144020	--------	--------	--------	16
17	004021	014021	--------	034021	--------	--------	--------	074021	--------	--------	--------	--------	--------	--------	--------	--------	17
18	004420	014420	024420	034420	044420	054420	064420	--------	--------	--------	--------	134420	--------	--------	--------	--------	18
19	004421	--------	024421	--------	--------	--------	064421	--------	--------	--------	--------	--------	--------	--------	--------	--------	19
20	005020	015020	025020	035020	045020	055020	--------	--------	--------	--------	125020	--------	--------	--------	--------	--------	20
21	--------	015021	--------	--------	--------	055021	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	21
22	005420	015420	025420	035420	045420	--------	--------	--------	--------	115420	--------	--------	--------	--------	--------	--------	22
23	005421	--------	--------	--------	045421	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	23
24	006020	016020	026020	036020	--------	--------	--------	--------	106020	--------	--------	--------	--------	--------	--------	--------	24
25	--------	--------	--------	036021	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	25
26	006420	016420	026420	--------	--------	--------	--------	076420	--------	--------	--------	--------	--------	--------	--------	--------	26
27	--------	--------	026421	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	27
28	007020	017020	--------	--------	--------	--------	067020	--------	--------	--------	--------	--------	--------	--------	--------	--------	28
29	--------	017021	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	29
30	007420	--------	--------	--------	--------	057420	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	30
31	007421	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	31

All partitions must begin on an even starting surface number. Use an odd number of surfaces only when the partition is the last partition on a physical disk having an odd number of surfaces; otherwise disk space will be wasted.

100.00.D9300.6LA

Table 3-4. Additions to pdev for Disk Drive Unit Number and Disk Controller Address

Number to Add for Disk Drive Unit Number — Disk Drive Unit		Number to Add for Disk Controller Address — Disk Controller		
Number	Add	Address	Number	Add
0	0	24_8	0	0
1	2	26_8	1	40
2	4	25_8	2	100
3	6	22_8	3	140
4	10	45_8	4	200
5	12	27_8	5	240
6	14	46_8	6	300
7	16	23_8	7	340

Example of Determining a pdev

Suppose that you are creating a partition with these characteristics:

Starting surface number	16
Number of surfaces	12

From Table 3-3, the basic pdev is 103020. By adding the appropriate values from Table 3-4, you obtain all the possible pdevs for this partition and for all the controller and disk drive unit number combinations, as shown in Table 3-5. For example, the pdev for this partition on disk drive unit number 2 connected to the disk controller at address 27_8 (controller 5) is 103264.

Table 3-5. Example Combinations of Controller Addresses and Disk Drive Unit Numbers for Basic pdev 103020

		Disk Drive Unit Number							
		0	*1*	*2*	*3*	*4*	*5*	*6*	*7*
A	24_8	103020	103022	103024	103026	103030	103032	103034	103036
d	26_8	103060	103062	103064	103066	103070	103072	103074	103076
d	25_8	103120	103122	103124	103126	103130	103132	103134	103136
r	22_8	103160	103162	103164	103166	103170	103172	103174	103176
e	45_8	103220	103222	103224	103226	103230	103232	103234	103236
s	27_8	103260	103262	103264	103266	103270	103272	103274	103276
s	46_8	103320	103322	103324	103326	103330	103332	103334	103336
	23_8	103360	103362	103364	103366	103370	103372	103374	103376

Figure 3-1 is a worksheet that you can fill out to arrive at a pdev. It includes an example.

Example

1. Starting surface number: 16
2. Number of surfaces in partition: 12
3. Basic pdev from Table 3-3: 103020
   - Controller address: 22
   - From Table 3-4, add: 140
   - Drive Unit Number: 3
   - From Table 3-4, add: 6
4. pdev to use: 103166

Worksheet

1. Starting surface number: ________
2. Number of surfaces in partition: ________
3. Basic pdev from Table 3-3: ________
   - Controller address: ________
   - From Table 3-4, add: ________
   - Drive Unit Number: ________
   - From Table 3-4, add: ________
4. pdev to use: ________

Figure 3-1. Sample pdev Worksheet

Example of Partitioning SMDs

Figure 3-2 shows an example of using Table 3-3. The system in this example contains three drive units connected to the disk controller with address 26_8 (controller 1); drives 0 and 1 have 300MB SMDs, and drive 2 has an 80MB SMD.

As shown in Figure 3-2, the spindles are to be partitioned as follows:

Drive 0 — Partitions of 2, 2, 6, 2, 2, 2, and 3 surfaces
Starting surface numbers 0, 2, 4, 10, 12, 14, and 16

Drive 1 — Partitions of 14 and 5 surfaces
Starting surface numbers 0 and 14

Drive 2 — Partitions of 2 and 3 surfaces
Starting surface numbers 0 and 2

Since each of these spindles has an odd number of surfaces, the last partition in each spindle has a 1 in the basic pdev. Since there are three disk drives involved, a number must be added to each of the basic pdevs to designate the disk drive units. From Table 3-4, these numbers are 0 for drive unit 0, 2 for drive unit 1, and 4 for drive unit 2. Also, from Table 3-4, 40_8 must be added for disk controller 1 at address 26_8.

The basic physical device numbers for this example, as taken from Table 3-3, are shown on the left side of Table 3-6. The appropriate numbers to designate the disk controller and the disk drives are added and the adjusted pdevs, as they are to be used, are shown on the right side of the table.

Table 3-6. Example of Basic and Adjusted pdevs

Basic pdev			*Adjusted pdev*		
Drive 0	*Drive 1*	*Drive 2*	*Drive 0*	*Drive 1*	*Drive 2*
000420	003420	000420	000460	003462	000464
010420	071021	010420	010460	071063	010465
021420			021460		
050420			050460		
060420			060460		
070420			070460		
100421			100461		

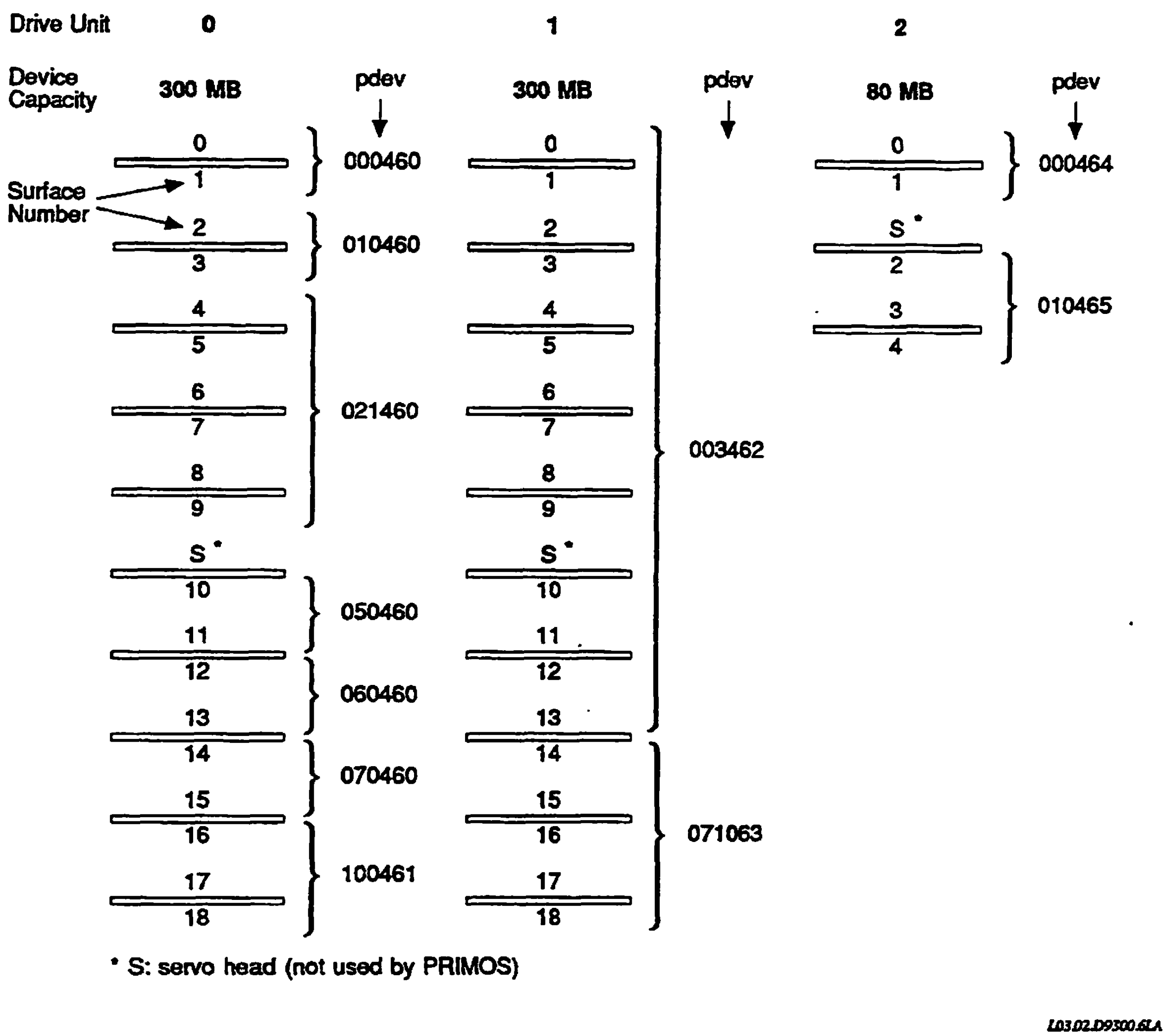


Figure 3-2. Example of SMD Partitions

Binary Breakdown of Physical Device Numbers

Figure 3-3 shows the binary breakdown of a physical device number. Physical device numbers (pdevs) tell the system the type of storage device being used, the drive unit on which the device is mounted, the controller to which the drive unit is connected, and the size of the partition and its location on the disk pack. The meaning of the bits in the halfword making up the physical device number are shown in the figure and discussed here:

- Bits 1 through 4 represent the starting surface number divided by two; thus the highest starting surface number can be no more than 30 since the highest number represented by four bits is 15.
- Bits 5 through 8 plus bit 16 (the odd surface bit) represent the value for the number of surfaces in a partition; thus no more than 31 surfaces can be in a partition since the highest number represented by five bits is 31.
- Bits 9, 10, and 11 represent the controller number associated with the address of the disk controller; thus there can be eight disk controllers (0-7) represented by three bits.
- Bit 12 is reserved and is set (=1).
- Bits 13, 14, and 15 represent the disk drive unit number; thus there can be eight disk drive units (0-7) represented by three bits.

Figure 3-3 also shows the meanings of two of the pdevs from the previous example. These are the two pdevs for the partitions on Drive 1. For pdev 003462, the octal-to-binary conversion indicates a surface offset (starting surface number) of 0, 14 surfaces in the partition, controller 1 at address of 26_8, and a drive unit number of 1.

For pdev 071063, the conversion indicates the same drive unit number and controller address as above. These portions of the pdevs for the two partitions are the same because the partitions are part of the same spindle. The remaining bits indicate that there are five surfaces in this partition with a starting surface number of 14.

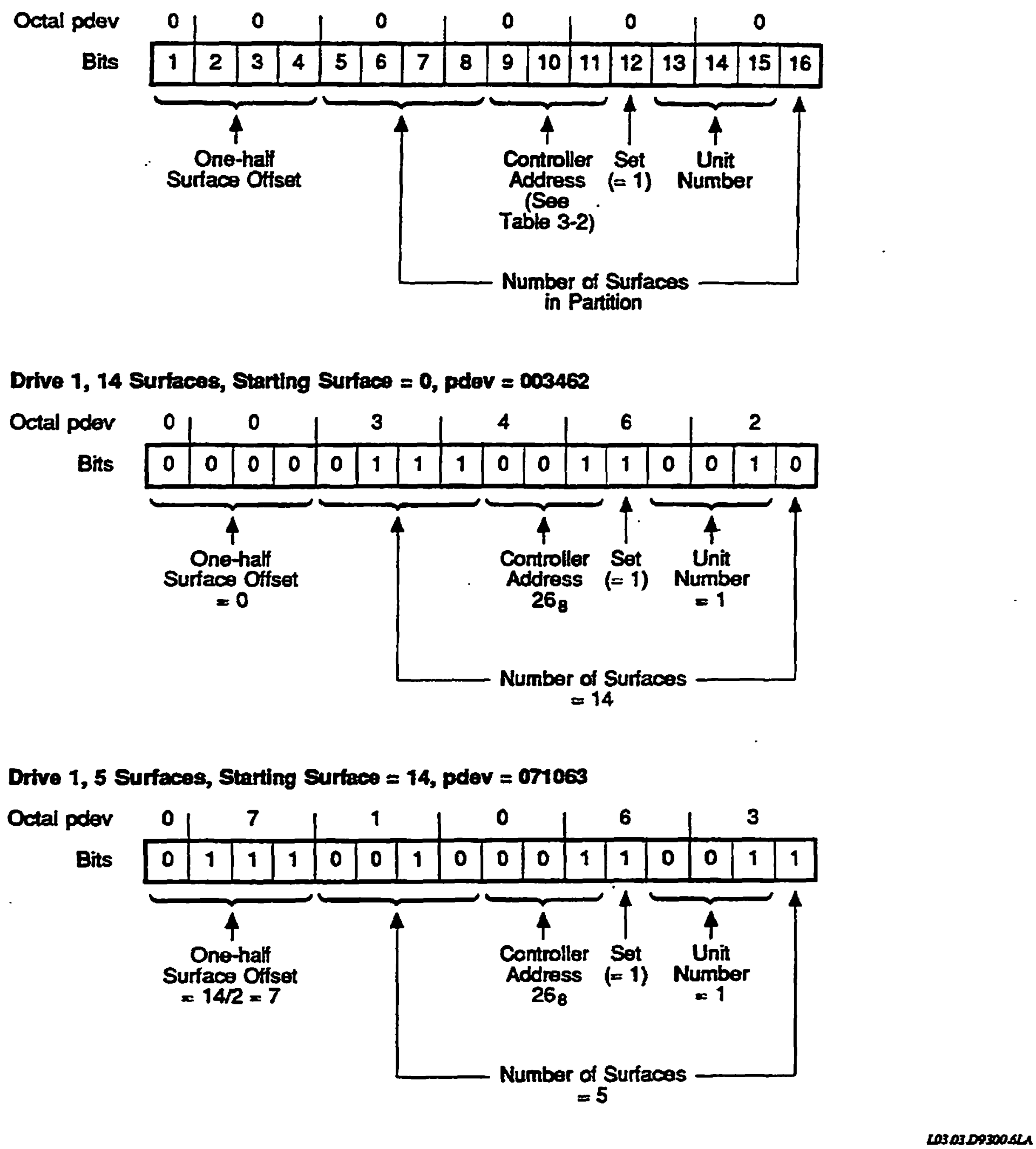


Figure 3-3. Construction of a Physical Device Number

Physical Device Numbers for CMDs

Physical device numbers for CMDs are similar to those for SMDs and FMDs. However, a peculiarity exists in the assignment of surface offsets: the removable portion of the disk is surface offset 0, but the nonremovable portion starts at surface offset 16; thus pdevs for the nonremovable portions of CMDs always start with a 1.

This peculiarity renders the specification of surfaces 1 through 15 ineffective because no actual surfaces correspond to the specification. Therefore, a CMD must contain at least two partitions. The removable portion of the disk is always one partition, and the nonremovable portion is one or more partitions.

Valid pdevs for CMDs are summarized at the end of the next section, Partitioning Specific Disk Types, and are shown in Table 3-7 at the end of this chapter.

Partitioning Specific Disk Types

This section describes each disk type and size that may be partitioned and tells how it may be partitioned. All sizes of SMDs, CMDs, and FMDs may be partitioned. Partitioning a disk allows you to use it as several logical disks, each with its own name, rather than as one physical disk. Table 3-1, earlier in this chapter, summarizes the disk characteristics.

Note Prime recommends that you create Models 4721, 4729, 4730, 4731, and 4732 SCSI disks as single partitions; that is, that you use the entire disk as one partition except in the case of creating a paging partition or a crash dump partition.

The following sections discuss the methods for each type of disk.

80MB Storage Module Disk

The 80MB SMD has five surfaces. Table 3-3 presents the valid basic physical device numbers for 80MB SMDs. The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 5 or less. Because the disk has an odd number of surfaces, the last partition must have an odd number of surfaces, or else some disk space is not utilized. Therefore, use one of the valid entries for 80MB disks ending in 1 in Table 3-3 as the last partition, to fully utilize the disk. However, use only the entries ending in 1 for which adding the number of surfaces to the starting surface number results in 5. Do not use those entries resulting in 3, because they are for 68MB FMDs only and will result in unused space on 80MB SMDs. An 80MB disk can be set up as a single partition.

300MB Storage Module Disk

The 300MB SMD has 19 surfaces. Table 3-3 presents the valid basic physical device numbers for 300MB SMDs. The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 19 or less. Because the disk has an odd number of surfaces, the last partition must have an odd number of surfaces, or else some disk space is not utilized. Therefore, use one of the valid entries for 300MB SMDs ending in 1 in Table 3-3 as the last partition, to fully utilize the disk. However, use only the entries ending in 1 for which adding the number of surfaces to the starting surface number results in 19. Do not use 1 entries resulting in 3, 5, 7, 15, and 17 because they are for 68MB FMDs, 80MB SMDs, 84MB FMDs (Model 4714), 158MB FMDs, 817MB FMDs (Model 4860), and 258MB FMDs (Model 4719) only and will result in unused space on 300MB SMDs. A 300MB disk can be set up as a single partition.

60MB Fixed-Media Disk (Model 4711)

The 60MB FMD (Model 4711) has four surfaces. Table 3-3 presents the valid basic physical device numbers for 60MB FMDs. The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 4 or less. Because the disk has an even number of surfaces, no partition should have an odd number of surfaces, or else some disk space is not utilized. Therefore, do not use any of the entries ending in 1 in Table 3-3. A 60MB disk can be set up as a single partition.

68MB Fixed-Media Disk

The 68MB FMD has three surfaces. Table 3-3 presents the valid basic physical device numbers for this disk. The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 3 or less. Because the disk has an odd number of surfaces, the last partition must have an odd number of surfaces, or else some disk space is not utilized. Therefore, Table 3-3 shows only two valid entries for this disk ending in 1 (000421 and 010021), and you must use one of these entries as the last partition, to fully utilize the disk. A 68MB disk can be set up as a single partition.

84MB Fixed-Media Disk (Model 4714)

The 84MB FMD (Model 4714) has five surfaces. Table 3-3 presents the valid basic physical device numbers for 84MB FMDs. The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 5 or less. Because the disk has an odd number of surfaces, the last partition must have an odd number of surfaces, or else some disk space is not utilized. Therefore, use one of the valid entries for 84MB

FMDs ending in 1 in Table 3-3 as the last partition, to fully utilize the disk. However, use only the entries ending in 1 for which adding the number of surfaces and the starting surface number results in 5. Do not use those resulting in 3, because they are for 68MB FMDs only.

120MB Fixed-Media Disk (Model 4715)

The 120MB FMD (Model 4715) has eight surfaces. Table 3-3 presents the valid basic physical device numbers for 120MB FMDs. The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 8 or less. Because the disk has an even number of surfaces, no partition should have an odd number of surfaces, or else some disk space is not utilized. Therefore, do not use any of the entries ending in 1 in Table 3-3. A 120MB disk can be set up as a single partition.

158MB Fixed-Media Disk

The 158MB FMD has seven surfaces. Table 3-3 presents the valid basic physical device numbers for 158MB FMDs. The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 7 or less. Because the disk has an odd number of surfaces, the last partition must have an odd number of surfaces, or else some disk space is not utilized. Therefore, use one of the valid entries for 158MB FMDs ending in 1 in Table 3-3 as the last partition in order to fully utilize the disk. However, use only the entries ending in 1 for which adding the number of surfaces and the starting surface number results in 7. Do not use those resulting in 3 or 5, because they are for 68MB FMDs and 80MB SMDs only.

160MB Fixed-Media Disk

The 160MB FMD has 10 surfaces. Table 3-3 presents the valid basic physical device numbers for 160MB FMDs. The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 10 or less. Because the disk has an even number of surfaces, no partition should have an odd number of surfaces, or else some disk space is not utilized. Therefore, do not use any of the entries ending in 1 in Table 3-3. A 160MB disk can be set up as a single partition.

213MB Fixed-Media Disk (Model 4730)

The 213MB FMD, available to operate with the Model 7210 SCSI disk/tape controller, has 31 logical surfaces. Because of the pseudo-geometry used with this disk, surfaces 0–3 contain 3556 records per surface and surfaces 4–31 contain 3302 records per surface. Table 3-3 presents the valid basic physical device numbers for 213MB FMDs (Model 4730). The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 31 or less. Because the disk has an odd number of logical surfaces, the last partition must have an odd number of surfaces, or else some disk space is not utilized. Therefore, use one of the valid entries for Model 4730 FMDs ending in 1 in Table 3-3 as the last partition to fully utilize the disk. A Model 4730 disk can be set up as a single partition.

258MB Fixed-Media Disk (Model 4719)

The 258MB FMD has 17 surfaces. Table 3-3 presents the valid basic physical device numbers for 258MB FMDs (Model 4719). The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 17 or less. Because the disk has an odd number of surfaces, the last partition must have an odd number of surfaces, or else some disk space is not utilized. Therefore, use one of the valid entries for Model 4719 FMDs ending in 1 in Table 3-3 as the last partition, to fully utilize the disk. However, use only the entries ending in 1 for which adding the number of surfaces and the starting surface number results in 17. Do not use those resulting in 3, 5, 7, and 15 because they are for 68MB FMDs, 80MB SMDs, 84MB FMDs (Model 4714), 158MB FMDs, and 817MB FMDs (Model 4860) only. A Model 4719 disk can be set up as a single partition.

315MB Fixed-Media Disk (Model 4475)

The 315MB FMD has 19 surfaces. Table 3-3 presents the valid basic physical device numbers for 315MB FMDs (Model 4475). The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 19 or less. Because the disk has an odd number of surfaces, the last partition must have an odd number of surfaces, or else some disk space is not utilized. Therefore, use one of the valid entries for Model 4475 FMDs ending in 1 in Table 3-3 as the last partition, to fully utilize the disk. However, use only the entries ending in 1 for which adding the number of surfaces and the starting surface number results in 19. Do not use those resulting in 3, 5, 7, 15, and 17 because they are for 68MB FMDs, 80MB SMDs, 84MB FMDs (Model 4714), 158MB FMDs, 817MB FMDs (Model 4860), and 258MB FMDs (Model 4719) only. A Model 4475 disk can be set up as a single partition.

328MB Fixed-Media Disk (Model 4721)

There are two methods of partitioning the 328MB FMD. The one you use depends on the controller to which the disk is connected. Table 3-3 presents the valid basic physical device numbers for 328MB FMDs (Model 4721).

If the disk is connected to the Model 7210 SCSI disk/tape controller, the 328MB FMD has 31 logical surfaces. The only valid entries for this geometry are those entries from Table 3-3 for which adding the number of surfaces to the starting surface number produces a total of 31 or less. Because the disk has an odd number of logical surfaces, the last partition must have an odd number of surfaces or else some disk space is not utilized. Therefore, when the 328MB FMD (Model 4721) is connected to the Model 7210 SCSI disk/tape controller, use one of the valid entries for the disk ending in 1 in Table 3-3 as the last partition to fully utilize the disk. In addition, a 328MB disk can be set up as a single partition.

If the 328MB disk is not connected to the Model 7210 SCSI disk/tape controller, the 328MB FMD has 12 surfaces. With this geometry, the only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 12 or less. Because the disk has an even number of surfaces, no partition should have an odd number of surfaces or else some disk space is not utilized. Therefore, do not use any of the entries ending in 1 in Table 3-3. A 328MB disk can be set up as a single partition.

421MB Fixed-Media Disk (Model 4731)

The 421MB FMD operates with the Model 7210 SCSI disk/tape controller and has 31 logical surfaces. Because of the pseudo-geometry used with this disk, surfaces 0–21 contain 6604 records per surface and surfaces 22–31 contain 6350 records per surface. Table 3-3 presents the valid basic physical device numbers for 421MB FMDs (Model 4731). The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 31 or less. Because the disk has an odd number of logical surfaces, the last partition must have an odd number of surfaces or else some disk space is not utilized. Therefore, use one of the valid entries for Model 4731 FMDs ending in 1 in Table 3-3 as the last partition to fully utilize the disk. A Model 4731 disk can be set up as a single partition.

496MB Fixed-Media Disk (Model 4735)

The 496MB FMD has 24 surfaces. Table 3-3 presents the valid basic physical device numbers for 496MB FMDs (Model 4735). The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 24 or less. Because the disk has an even number of surfaces, no partition should have an odd number of surfaces, or else some disk space is not utilized. Therefore, do not use any of the entries ending in 1 in Table 3-3. A 496MB disk can be set up as a single partition.

673MB Fixed-Media Disk (Model 4729)

The 673MB FMD operates with the Model 7210 SCSI disk/tape controller and has 31 logical surfaces. Because of the pseudo-geometry used with this disk, surfaces 0–3 contain 10668 records per surface and surfaces 4–31 contain 10414 records per surface. Table 3-3 presents the valid basic physical device numbers for 673MB FMDs (Model 4729). The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 31 or less. Because the disk has an odd number of logical surfaces, the last partition must have an odd number of surfaces or else some disk space is not utilized. Therefore, use one of the valid entries for Model 4729 FMDs ending in 1 in Table 3-3 as the last partition. A Model 4729 disk can be set up as a single partition.

675MB Fixed-Media Disk

The 675MB FMD has 40 surfaces. Table 3-3 presents the valid basic physical device numbers for 675MB disks. Because the disk has an even number of surfaces, no partition should have an odd number of surfaces, or else some disk space is not utilized. Therefore, do not use any of the entries ending in 1 in Table 3-3. A 675MB disk cannot be set up as a single partition because a partition cannot contain more than 31 surfaces. Therefore, you must create at least two partitions in order to fully utilize a 675MB disk. In addition, the last partition on a 675MB disk must use at least 10 surfaces; thus it is desirable to use the last partition for a file system that is very large.

770MB Fixed-Media Disk (Model 4845)

The 770MB FMD has 23 surfaces. Table 3-3 presents the valid basic physical device numbers for 770MB FMDs (Model 4845). The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 23 or less. Because the disk has an odd number of surfaces, the last partition must have an odd number of surfaces, or else some disk space is not utilized. Therefore, use one of the valid entries for Model 4845 FMDs ending in 1 in Table 3-3 as the last partition to fully utilize the disk. However, use only the entries ending in 1 for which adding the number of surfaces and the starting surface number results in 23. Do not use those resulting in 3, 5, 7, 15, 17, and 19 because they are for 68MB FMDs, 80MB SMDs, 84MB FMDs (Model 4714), 158MB FMDs, 817MB FMDs (Model 4860), 258MB FMDs (Model 4719), 315MB FMDs (Model 4475), and 300MB SMDs only. A Model 4845 disk can be set up as a single partition.

817MB Fixed-Media Disk (Model 4860)

The 817MB FMD has 15 surfaces. Table 3-3 presents the valid basic physical device numbers for 817MB FMDs (Model 4860). The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 15 or less. Because the disk has an odd number of surfaces, the last partition must have an odd number of surfaces, or else some disk space is not utilized. Therefore, use one of the valid entries for Model 4860 FMDs ending in 1 in Table 3-3 as the last partition, to fully utilize the disk. However, use only the entries ending in 1 for which adding the number of surfaces and the starting surface number results in 15. Do not use those resulting in 3, 5, and 7 because they are for 68MB FMDs, 80MB SMDs, 84MB FMDs, and 158MB FMDs only. A Model 4860 disk can be set up as a single partition with 15 surfaces.

1.34GB Fixed-Media Disk (Model 4732)

The 1.34GB FMD operates with the Model 7210 SCSI disk/tape controller and has 31 logical surfaces. Because of the pseudo-geometry used with this disk, surfaces 0–20 contain 20828 records per surface and surfaces 21–31 contain 20574 records per surface. Table 3-3 presents the valid basic physical device numbers for 1.34GB FMDs (Model 4732). The only valid entries are those entries for which adding the number of surfaces to the starting surface number produces a total of 31 or less. Because the disk has an odd number of logical surfaces, the last partition must have an odd number of surfaces or else some disk space is not utilized. Therefore, use one of the valid entries for Model 4732 FMDs ending in 1 in Table 3-3 as the last partition to fully utilize the disk. A Model 4732 disk can be set up as a single partition.

Partitioning Cartridge Module Devices (CMDs)

CMDs exist in three sizes: 32, 64, and 96 megabytes. They may be partitioned as indicated in Table 3-7 following.

Partitioning of CMDs is done in a manner similar to that of SMDs and FMDs. However, the removable portion of a CMD is always a separate partition. The nonremovable portion is partitioned separately from the removable portion. The nonremovable surfaces are treated as if they start at a surface offset, or a starting surface number, of 16.

Table 3-7. Basic Physical Device Numbers for Cartridge Module Devices

CMD Type	*Surfaces*	*Type of Surfaces*	*pdev for Controller 0*
32MB	1	Removable	21 (16MB)
	1	Nonremovable	100021 (16MB)
64MB	1	Removable	21 (16MB)
	3	Nonremovable	100420 (32MB)
			110021 (16MB)
			or
			100421 (48MB)
96MB	1	Removable	21 (16MB)
	5	Nonremovable	100420 (32MB)
			110420 (32MB)
			120021 (16MB)
			or
			101020 (64MB)
			120021 (16MB)
			or
			101021 (80MB)
			or
			100420 (32MB)
			110421 (48MB)

Notes

See Table 3-4 for numbers to add to the basic pdev if the disk is mounted in a disk drive unit other than unit 0, that is, drive units 1 through 3.

See Table 3-4 for numbers to add to the basic pdev if the disk drive is connected to a disk controller other than controller 0 at address 24_8.

The removable surface of all CMDs is organized as one partition.

The nonremovable surface of the 32MB CMD is also organized as one partition.

The nonremovable surfaces of the 64MB CMD can be organized as one or two partitions.

The nonremovable surfaces of the 96MB CMD can be organized as one, two, or three partitions.

4 Using and Assigning Disks

This chapter explains the procedure by which you

- Make disks available for general use
- Assign disks to yourself (or to other users), so that you can perform maintenance tasks on them

File System Disks and Assignable Disks

Think of disks (that is, partitions, or logical disks) as being in one of the two categories shown in Figure 4-1: File System Disks and Assignable Disks. **File system disks** are disks that are available to system users for shared access. They are turned on and made available for use when you give the ADDISK command followed by their physical device numbers (pdev). While the disks are on (available), users may perform all normal operations on directories and files, such as creating and listing directories and editing, copying, deleting, and renaming files.

When it becomes necessary for you to perform some operation on a logical disk, such as copying it or repairing it, use the SHUTDN command to shut the disk down. This action makes the disk unavailable to users.

So that you can assign the disk to yourself for exclusive, direct access, you must place the disk in the Assignable Disks Table. **Assignable disks** are partitions that a single user can assign for exclusive access.

Note Assignable disks can be either formatted or unformatted. File system disks must be formatted so that they contain at least the basic elements of the PRIMOS file system.

Physical Device Number

The common link between file system disks and assignable disks is the **physical device number (pdev)**. The pdev is the partition's address; that is, the pdev tells PRIMOS what disk drive and disk controller the partition is connected to and where on the physical disk, or spindle, the partition is located.

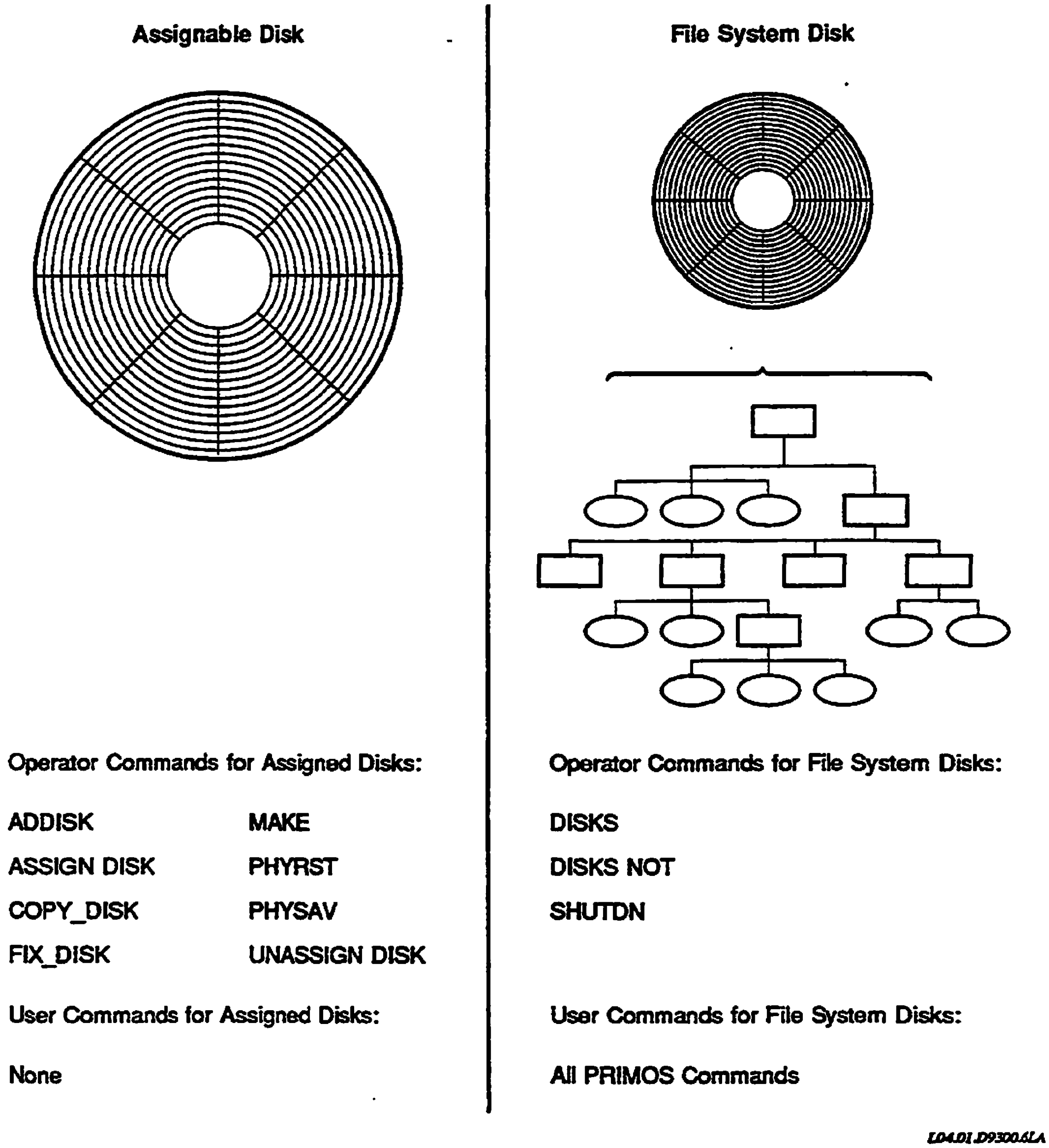


Figure 4-1. File System Disks and Assignable Disks

If there are any changes to the partition's address, its pdev must change. For example, if a partition has been associated with a particular disk drive, and the disk pack containing that partition is then placed in a different drive or the original drive's designation is changed, the partition's address changes; therefore, its pdev must be changed.

To make the disk available or unavailable, to repair it, or to otherwise operate on it, you must use the pdev to specify which logical disk you mean. Although users identify disks by their partition names, you often perform operations before the relationship between a partition and its disk drive has been defined to the system. To identify the partition, you must use the pdev. See the descriptions and examples in Chapter 3 to determine physical device numbers.

Assigning and Unassigning Disks

Before you run MAKE or FIX_DISK to create or repair a partition, you must shut down the partition to make it unavailable to users. When you have finished creating or repairing a partition, you should reverse the shutdown process and make the partition again available to users.

Shutting Down a Partition

Before you shut down a partition, a disk drive, or a system, give users sufficient time to finish their work and stop working on that partition or on that system. Use the MESSAGE command to give adequate warnings. You may need to use the -FORCE option. For example:

```
OK, MESSAGE -ALL -NOW -FORCE
Disk PTUSER will be shutting down in 5 minutes Ctrl-G
OK,
```

Press Ctrl-G to ring the bell at each terminal.

If you are going to run FIX_DISK on the command device, do the following:

1. Use the MAXUSR 0 command to prevent any users from logging in.
2. Use the STOP_LSR command to stop the Login server.
3. Use the STOP_DSM command to stop DSM.
4. Stop any other system servers that depend on the command device.
5. Log out out any other phantoms and any users remaining on the system after the appropriate warning time has expired.

For example:

```
OK, MAXUSR 0
OK, STOP_LSR
Really? YES
OK, STOP_DSM
OK, LOGOUT ALL
OK,
```

Messages appear indicating that the Login server, the DSM phantoms, and any other phantoms are logging out. It is necessary to log out users and system servers because, when a partition is shut down, all files on the partition, including runfiles for EPFs, are closed. Thus, if a process attempts to access memory, it fails because the EPF is unmapped.

Procedure for Assigning and Unassigning Disks

To create new partitions or to perform maintenance on an existing partition, use the following procedure, which is summarized in Figure 4-2. Note that most of this procedure can be carried out only from the supervisor terminal. Remember to warn users before shutting down a partition.

1. Use the SHUTDN command to shut down the partition to be repaired.

   ```
   OK, SHUTDN 1060
   OK,
   ```

Note If you are running FIX_DISK on the command device, stop the Login server with the STOP_LSR command and stop DSM with the STOP_DSM command. Log out other system phantoms. Use the -COMDEV option to shut the partition down; do not use the SHUTDN command. Remember to start the Login server (START_LSR), DSM (START_DSM), and any other system phantoms when you finish running FIX_DISK on the command device.

2. Use the DISKS command followed by the logical disk's physical device number (pdev) to place the disk in the Assignable Disks Table.

   ```
   OK, DISKS 1060
   OK,
   ```

3. After placing the disk in the Assignable Disks Table, use the ASSIGN DISK *pdev* command to assign it to yourself.

   ```
   OK, ASSIGN DISK 1060
   OK,
   ```

4. You can now perform these operations:

    - Use MAKE to format a partition.
    - Use FIX_DISK to repair a partition.
    - Use PSR to physically copy a partition, to physically save a partition, or to physically restore a partition.

5. After you have completed the operations in step 4, use the UNASSIGN DISK *pdev* command to unassign the disk.

```
OK, UNASSIGN DISK 1060
OK,
```

6. Use the DISKS NOT *pdev* command to remove the disk from the Assignable Disks Table.

```
OK, DISKS NOT 1060
OK,
```

7. Use the ADDISK command to make the disk available to users again.

```
OK, ADDISK 1060
Starting up revision 22.1 partition "RASCAL".
OK,
```

Once a partition has been placed in the Assignable Disks Table, any user can assign the partition and use many of the commands shown in Figure 4-2 at any user terminal unless your system uses device ACLs. However, it is assumed for the purposes of this document that only an Operator at the supervisor terminal would carry out the processes described here.

The STATUS DEVICE command shows what devices are in the Assignable Disks Table and what devices are assigned. The following example shows one tape drive and one disk assigned to user SYSTEM and one disk in the Assignable Disks Table.

```
OK, STATUS DEVICE

Device User name                                Usrnum Ldevice
MT0    SYSTEM                                      1    MT0
100461 SYSTEM                                      3

Available assignable disks:

           20660

OK,
```

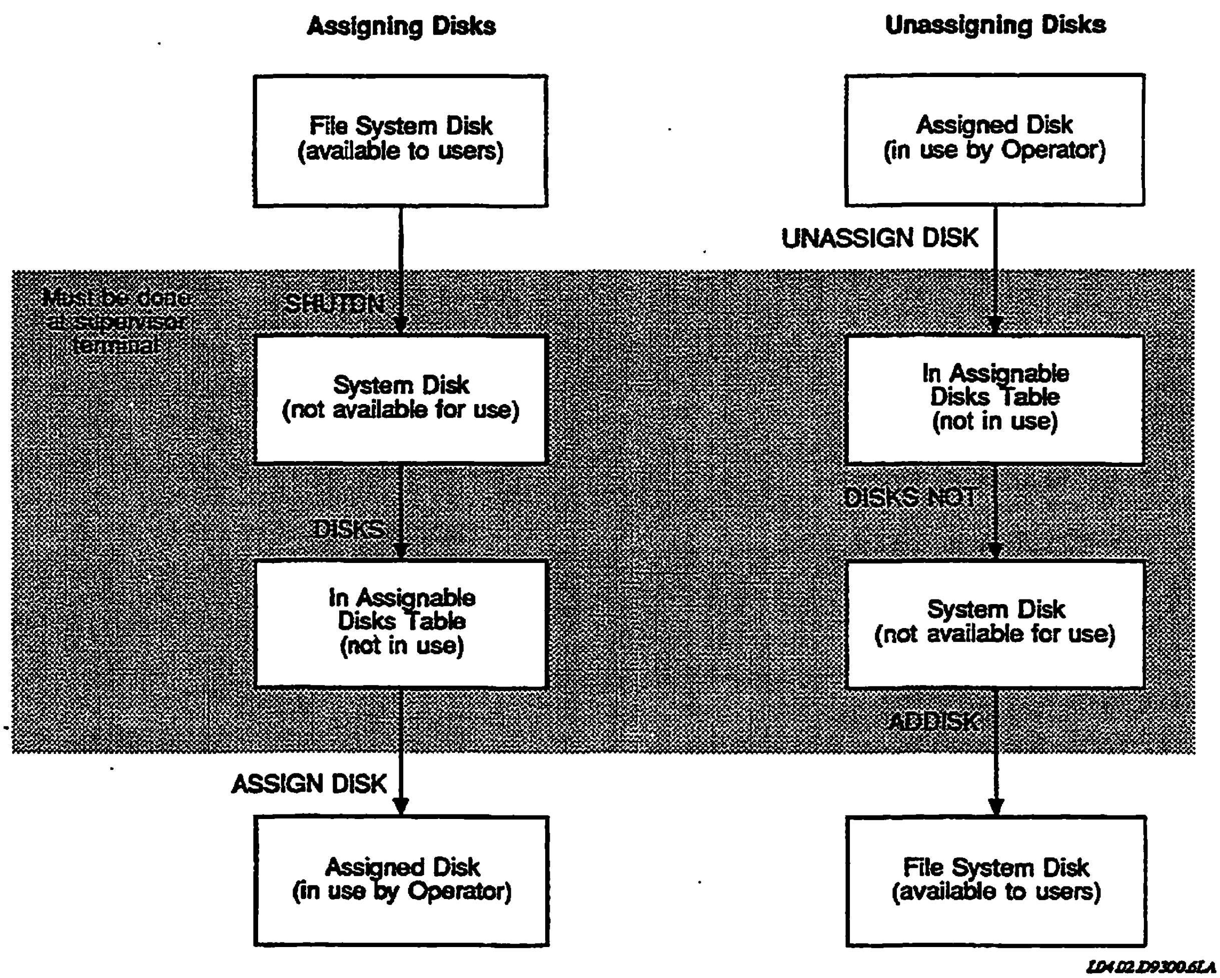


Figure 4-2. Assigning and Unassigning Disks

5 Formatting Disks – MAKE

This chapter describes the MAKE utility, used for formatting and partitioning physical disks, or **spindles**. The first part tells why you use MAKE, explains how to choose the options you want for particular jobs, and discusses handling of badspots by PRIMOS. The second part takes you step-by-step through the actions that may precede, accompany, and follow the normal use of MAKE, and provides examples of typical sessions with MAKE. The final part discusses how to use MAKE when PRIMOS is not running.

Why You Need to Run MAKE

Before a disk can be used by your computer system, it must conform to the PRIMOS method of organizing data on the disk. Bringing a disk into conformity with your system's requirements is called **making** or **creating** a disk. Your System Administrator may require you to subdivide a disk's memory area into two or more partitions; this action is referred to as **partitioning** a disk. You may also format a disk. **Formatting** involves organizing the medium in each track into sectors where PRIMOS stores each data record. You may also split a disk; **splitting** divides a partition into a file system area and an area for paging or for crash dump use.

MAKE is the system utility for creating and partitioning disks. MAKE can create and partition user disks, paging disks, and crash dump disks. **User disks** provide the actual storage for users' files. **Paging disks** are areas of disks that PRIMOS uses when managing virtual memory. **Crash dump disks** are areas of disks that are set aside to receive critical system information in the event that the system crashes and this information must be saved for later analysis.

Note MAKE does not format the SCSI disks connected to the Model 7210 or Model 2382 SCSI disk/tape controllers but does create file systems on these disks.

After a user partition is created, PRIMOS writes the Master File Directory (MFD) on it. The MFD is the top-level directory of the file system and contains the directories and files at the top level of the partition. When the partition is first created, the following two or three files and two directories are in the MFD.

- The Disk Record Availability Table (DSKRAT), which contains information about the physical structure of the partition. The DSKRAT file gets the same name as the partition when the partition is created.
- An empty directory, CMDNC0, which can be deleted if this is not the command device.
- An empty directory, DOS, which can be deleted.
- The BOOT file, which you use to start, or bootstrap, the system from disk.
- Either a badspot file (BADSPT) or a dynamic badspot (DBS) file, which records the locations of any physical defects on disks. (The BADSPT file is created only if badspots exist on the partition and the partition is in –DBS OFF (–AC) mode.) The DBS file is not visible but there is a file named DYNBSP that controls access to the DBS file. The DBS file is created on the first, or head zero, partition of a spindle, only on those disks listed in Table 5-3 (page 5-17), which support Dynamic Badspot Handling and are connected to a Model 6580 (IDC1) disk controller.

The MAKE command starts the MAKE utility, which is stored as CMDNC0>MAKE.SAVE. You normally invoke MAKE directly with the MAKE command. You can also run MAKE from a command input (COMINPUT) or CPL file (see page 5-57). In addition, you can boot MAKE.SAVE as a standalone program either from disk or from tape; this is discussed at the end of this chapter on page 5-91.

Caution To avoid mistakenly invoking a pre-Rev. 23.3 version of MAKE, delete any older versions of MAKE. When you invoke MAKE, a copyright line with a revision stamp is displayed. If you invoke an earlier version than Rev. 23.3, you may not get the results that you expect.

No partition conversion has been required for any PRIMOS revision since Rev. 22.1. MAKE at Rev. 23.3 thus creates Rev. 22.1-format partitions by default. When using FIX_DISK, you cannot further convert partitions if they are already at Rev. 22.1.

SCSI Disk Support

PRIMOS Rev. 23.2 supports several new SCSI disk drives and also supports the ICOP+ protocol for enhanced performance on SCSI disks. You should be aware of the following considerations. Except where noted, these statements apply to these SCSI disk drives, regardless of whether ICOP+ is in use:

Model 4721	328MB
Model 4729	673MB
Model 4730	213MB
Model 4731	421MB
Model 4732	1.34GB

- SCSI disk drives perform automatic badspot checking and remapping within the disk drive itself. MAKE, therefore, does not check for badspots on these disks. SCSI badspot handling is completely automatic and invisible to the user of the disk. Do not use any MAKE options related to badspot handling with SCSI disk drives.

  During MAKE processing with a Model 7210 disk controller, the Models 4721, 4729, 4730, 4731, and 4732 default to a badspot checking level of 0.

- MAKE and FIX_DISK ignore the –DBS ON (–IC) and –DBS OFF (–AC) options on SCSI disk drives on a Model 7210 controller and you should not use them. Commands that display the disk controller mode, such as MAKE and FIX_DISK, do not display a line indicating Dynamic Badspot Handling mode for SCSI disks.

- MAKE, by default, sets SCSI disk drives on a Model 7210 controller to forward sectoring with an interleave factor of 1. MAKE and FIX_DISK ignore the –SECTOR (–RDI and –ODI) option used to change sectoring and the interleave factor. You should not use –SECTOR.

  When reading records, SCSI disks cache, or buffer, records starting with the desired record. This is why PRIMOS allocates records on these disks in a forward order with an interleave factor of 1. You cannot change this method of record allocation. (See Chapter 10 for a discussion of sectoring.)

- You may partition SCSI disks but, for performance reasons, it is strongly recommended that you specify a SCSI disk as a single partition. (There is one exception, as described in the following section.)

The only MAKE options that you should use on these SCSI disks connected to a Model 7210 disk controller are the following:

–DISK	–PARTITION
–DISK_TYPE	–SPLIT
–BAUD_RATE	–DISK_REVISION
–MAX_EXTENT_SIZE	–MIN_EXTENT_SIZE
–REPORT	

Paging Partitions on SCSI Disks

Partitions on SCSI disks may be split (using MAKE –SPLIT) to create either a paging partition or a crash dump disk. Splitting a disk creates two portions: a file system portion and a non-filesystem portion that can later be activated either for paging or for crash dump to disk. The following recommendations apply to the use of SCSI disks for paging partitions.

Prime has always recommended that, for performance reasons, you do not use a split disk for both paging and file system I/O. When you split a SCSI disk, allocate a minimal number of records for the file system and the rest for paging by using the MAXIMUM argument to the –SPLIT option, as discussed later. Do not use the file system portion of the disk. To prevent use of the file system portion, do not add the disk by using its pdev with the ADDISK command.

Paging performance can be improved by establishing multiple paging partitions (up to 8). However, allocating several entire SCSI disks as paging partitions is likely to result in more paging space than you need and more disk space than you can afford to lose. The following recommendations may help you to strike a balance between disk space and paging performance.

- If possible, use SMD (non-SCSI) disks for paging partitions. Create multiple paging partitions on different SMD disks.
- If you use a SCSI disk for a paging partition, it is recommended, for performance reasons, that you do not partition the disk but use the entire disk as a single paging partition.

  If you must use a SCSI disk for both paging and file system I/O, you can minimize the performance impact by locating files on the file system portion that are rarely accessed.
- If you use SCSI disks for paging partitions, use several small SCSI disks, rather than one large SCSI disk, to maximize paging performance. However, to maximize disk space utilization, a single large SCSI paging disk may be preferable to several smaller SCSI paging disks.
- If a SCSI disk is larger than you need for paging, you can use part of the disk for paging and part for a crash dump disk. To do this, you partition the disk as two split partitions, allocating a minimal file system portion to each partition by using the MAXIMUM argument of the –SPLIT option. Use the non-filesystem portion of one partition for paging and the non-filesystem portion of the other partition as a crash dump disk. Because these two partitions are not accessed concurrently, there should be no significant performance impact.
- If possible, avoid using disks in a 75500-6PK device module as paging partitions. Disk drives containing paging partitions cannot be swapped. (See *Disk Replacement Procedure for Model 75500-6PK Device Module* for details.)

The Head Zero Partition of Disks Connected to a Model 6580 (IDC1) Disk Controller

The first, or head zero, partition (the partition containing surface 0) of a Rev. 21.0 or later spindle connected to a Model 6580 (IDC1) disk controller has special features. The head zero partition defines Dynamic Badspot Handling for the spindle as a whole, that is, for all the partitions on the spindle. In particular, the head zero partition contains the DBS file which is used by the controller to handle badspots on all partitions on the spindle. (See also Chapter 8 for additional information on Dynamic Badspot Handling.)

Note Starting at Rev. 21.0, MAKE creates a DBS file and an RMA on the first, or head zero, partition of the disks listed below. MAKE uses the DBS file and the RMA in handling badspots on all partitions on that spindle when the partitions on the spindle are in Dynamic Badspot Handling (–DBS ON or –IC) mode.

Because all disk drives connected to the Model 7210 and the Model 2382 SCSI disk controllers handle all media defects during a MAKE session, neither the DBS file nor the badspot file are created on partitions of spindles connected to those controllers.

The following disk types are capable of having Dynamic Badspot Handling occurring on them when they are connected to a Model 6580 (IDC1) disk controller and MAKE creates a DBS file and the RMA on these disks.

SMD (300MB and 80MB)

68MB	MODEL_4475
158MB	MODEL_4735
160MB	MODEL_4845
600MB	MODEL_4860

When you enable Dynamic Badspot Handling on these disks connected to an IDC1, Dynamic Badspot Handling is enabled for the entire spindle. When you use MAKE to create partitions on these disks, you must do the following:

- Use MAKE to create the first, or head zero, partition first.
- Assign the head zero partition when you run MAKE on other partitions on the spindle.
- Create all partitions on the spindle with the same –DBS argument, either –DBS ON or –DBS OFF.
- If you run MAKE on the first partition and it already has a DBS file and RMA on it, you may have to run MAKE with the –FORMAT option on all partitions on that spindle or you may risk losing data. MAKE warns you if this is the case.

All partitions on a spindle must be in the same mode: either Dynamic Badspot Handling (–DBS ON or –IC) mode or Nondynamic Badspot Handling (–DBS OFF or –AC) mode. All partitions on a spindle should also be of the same revision.

WARNING Do not use a pre-Rev. 23.3 version of MAKE on any partition of a Rev. 21.0 or later disk that has Dynamic Badspot Handling (–DBS ON) occurring on it. Do not use any version of MAKE with the –FORMAT option on any partition of a DBS spindle unless you remake all other partitions on the spindle, starting with the head zero partition. If you do either, the entire spindle (all partitions on the spindle) may be corrupted because MAKE overwrites the DBS file in these cases. Consequently, the data for records that initially were to be written to badspots on any partition of the spindle, and that were thus written to the RMA, are lost. Be sure all the data on all partitions on a spindle is backed up before using MAKE.

In addition, do not change the number of surfaces on an existing Rev. 21.0 or later head zero partition. If you do, the basic pdev and geometry of the first partition are changed and all other partitions on that spindle lose any data that was remapped to the RMA. You must then run MAKE with the –FORMAT option on all partitions on the spindle.

The MAKE Utility and Hashed Directories

Starting at Rev. 20.0 PRIMOS, the MAKE utility creates ACL-protected partitions that support hashed directories. The term **hashed directory** refers to the way PRIMOS gains access to data controlled by a directory. Hashing speeds up processing because the system does not have to perform a sequential search to locate a file system object.

Only ACL directories are hashed. If, for some reason, you create password directories on Rev. 20.0 and later disks, they will not be hashed. In addition, if you create robust partitions with Rev. 22.1 MAKE_ROBUST, the directories on these partitions are also not hashed but they are ACL-protected. (See Chapter 7 for a discussion of robust partitions.)

The protection placed on the MFD when a partition is created is SYSTEM:ALL and $REST:LUR. All directories created on these partitions are ACL-protected by default.

In order to use hashed directories on existing pre-Rev. 20.0 partitions, you must first convert those partitions to Rev. 20.0 or later format. Conversion of partitions is discussed in the next section.

Note Because Rev. 22.1 format partitions have file system attributes that pre-Rev. 22.1 format partitions do not, a Rev. 22.1 format partition *cannot* be added locally to a pre-Rev. 22.1 system. A Rev. 22.1 format partition may be accessed remotely by pre-Rev. 22.1 systems, however, through PRIMENET.

Since a partition created by Rev. 23.3 MAKE has the same attributes and revision stamp as Rev. 22.1 partitions, it is possible to add a Rev. 22.1 format partition to a Rev. 23.3 system.

Converting Partitions

Because Rev. 22.1-format partitions are new format PRIMOS partitions and Rev. 23.3 partitions are Rev. 22.1 format, it is necessary for you to convert existing pre-Rev. 22.1 partitions to Rev. 22.1 format so that your system can use all of the features of Rev. 23.3. You must use MAKE to convert pre-Rev. 22.0 partitions to Rev. 22.1-format partitions; you cannot convert pre-Rev. 22.0 partitions to Rev. 22.1-format with FIX_DISK. You can convert Rev. 22.0 partitions to Rev. 22.1-format with FIX_DISK or you can create Rev. 22.1-format partitions with MAKE. If you want to use robust partitions, you must use Rev. 22.1 or later MAKE to create the partitions prior to using MAKE_ROBUST to create the robust partition. (For a discussion of robust partitions, see Chapter 7.)

Note Rev. 23.3 PRIMOS partitions have the same format as Rev. 22.1 partitions. Thus, any reference to Rev. 22.1 partitions includes Rev. 23.3 PRIMOS partitions.

Conversion Procedure

The procedure below explains how to convert existing partitions to Rev. 22.1 format by using MAKE.

1. Save all of the data on the existing partition to magnetic tape by using one of the logical backup utilities, such as MAGSAV. (See the *Operator's Guide to Data Backup and Recovery*.)
2. Run Rev. 23.3 MAKE on the partition, specifying all the necessary options to create the particular Rev. 22.1-format partition that you want. (See the description of the options in the next section.)
3. Restore the data saved in step 1 by using a compatible logical backup utility, such as MAGRST if you used MAGSAV in step 1.

When converting existing partitions that are connected to a Model 6580 (IDC1) disk controller to Rev. 22.1-format, you must create the head zero partition of the disk types listed previously first since MAKE must write the DBS and RMA files on the head zero partition of those disks. The DBS and RMA handle badspots for all partitions on the spindle when the partition is in Dynamic Badspot Handling (–DBS ON) mode on a Model 6580 controller (IDC1).

File System Features

The file system features available on the various revision partitions are summarized in Table 5-1. The features are additive from left to right; that is, Rev. 22.1-format partitions support all features of the previous revisions plus robust partitions and unlimited numbers of extents for CAM files. At PRIMOS Rev. 23.3, partitions are Rev. 22.1 format.

Table 5-1. Partition Features Available at Rev. 20.0 Through the Current Disk Revision

Rev. 20.0	*Rev. 21.0*	*Rev. 22.0*	*Rev. 22.1*
Hashed directories (Not available on robust partitions)	Mirroring	Setting minimum and maximum extent sizes for CAM files	Robust partitions
ACL-protected directories CAM files	Dynamic Badspot Handling with IDC1 disk controller		Unlimited CAM file extents
Reverse/forward sectoring			

Options Available for Running MAKE

MAKE resides in CMDNC0 as MAKE.SAVE. You invoke MAKE with one of these command formats:

MAKE -DISK ***pdev*** **-PARTITION** ***disk_name*** **-DISK_TYPE** ***disk_type*** **[*options*]**
MAKE
MAKE -USAGE
MAKE -HELP

In the first format, *pdev* is the physical device number (pdev) of the disk you are creating. See Chapter 3 for directions on constructing physical device numbers. *name* is a six or less character name that you assign to the partition. *type* is the type of spindle that you are running MAKE on. If you omit any of the arguments for these three options or omit the options themselves, MAKE prompts you for the information in the order pdev, partition name, and disk type.

In the second format, you can enter the command name only and MAKE will prompt for the essential information, that is, the pdev, the partition name, and the disk type in that order. MAKE must have this information at a minimum.

The third format returns a brief description of MAKE usage and a list of options. The fourth format returns a brief description of MAKE usage and a series of screens giving brief descriptions of all the options. You can also type HELP MAKE to get help with MAKE.

The following list briefly summarizes the MAKE command line options in the order in which they are discussed in the next section of this chapter. See the detailed discussions of these options for information on how they should be used.

Each option has a three letter abbreviation as shown in the detailed discussions that follow. For every option that requires an argument, MAKE will prompt if you do not supply an argument or if you specify an invalid argument. You can enter Q or QUIT as a response to such a prompt and MAKE aborts.

Note Do not use any of the options having to do with badspots on SCSI disks that are connected to the Model 7210 or the Model 2382 disk controller. Badspots on these disks are handled by the disk drive.

–DISK *pdev*

Specify the pdev of the partition that you are creating. MAKE requires this information. See page 5-12.

–PARTITION *diskname*

Specify the name of the partition. MAKE requires this information. See page 5-15.

–DISK_TYPE *disk_type*

Specify the type of physical disk, or spindle, that the partition is on. MAKE requires this information. MAKE will also implicitly use vendor flaw maps on disks that have them when you specify the relevant disk model numbers with the –DISK_TYPE option. See page 5-14.

–SPLIT *recs*, –SPLIT MAXIMUM

Specify the number of records to be used for paging or for storing crash dumps, if the partition is to be a paging partition or a crash dump partition. See page 5-15.

–DISK_REVISION *rev*

Create the partition as a specific revision disk other than Rev. 22.1. See page 5-21.

–BAUD_RATE *baud*

Specify the initial baud rate of the supervisor terminal, if the disk will be used for booting a system. See page 5-22.

–FORMAT

Initialize or reinitialize all the tracks on the partition, organizing the medium into sectors to hold data records. Also reinitializes or destroys any dynamic badspot remappings on the spindle. See page 5-23.

–NEW_DISK

Tells MAKE to expect to find no valid data on the partition either because it is a new disk or the disk is corrupt. See page 5-25.

–INIT, –NO_INIT

Initialize or do not initialize file system records. –NO_INIT is the default. See page 5-27.

–REPORT

Report the percent completeness of formatting, badspot checking, or file system initialization at approximately 5% intervals. See page 5-27.

–DBS ON (–IC)

Create the partition as a Rev. 21.0 or later disk, taking into consideration the capabilities of the disk and of a Model 6580 (IDC1) disk controller to support Dynamic Badspot Handling and disk mirroring. You cannot use this option when running MAKE standalone. See page 5-29.

–DBS OFF (–AC)

Create the partition on a Rev. 21.0 or later disk type that supports Dynamic Badspot Handling so that it can be used on a non-intelligent disk controller or an IDC1 in non-intelligent mode. This is the only –DBS mode that you can use when using MAKE standalone. See page 5-31.

–NO_QUERY

Do not pause for confirmation in certain situations when running MAKE from a CPL or a COMI file. See page 5-32.

–FORMAT_OK

Enable the –FORMAT option if necessary when running MAKE as a phantom. See page 5-33.

–BADSPOT_LEVEL *level*

Specify the level of verification (0 - 4) to be used in checking for badspots. See page 5-34.

–NO_FLAW_MAP

Do not use vendor flaw maps on disks that have them but use the default level of badspot checking instead (not recommended). See page 5-35.

–QUERY_BADSPOTS

Allow user to enter a list of known badspots. See page 5-38.

–COPY_BADSPOTS (–CPYDEV, –CPYNAM)

Add badspots from an existing badspot file (BADSPT) on another partition. See page 5-38.

–LIST_BADSPOTS

List all known badspots collected by MAKE from various sources. This list represents the contents of the DBS file or the BADSPT file created by MAKE. See page 5-39.

–MAP_UNCORR

Do not consider records having correctable errors to be badspots (not recommended). See page 5-40.

–SECTOR FORWARD, –SECTOR REVERSE (–ODI, –RDI)

Use the forward sectoring method or the reverse sectoring method of file record allocation for Rev. 20.0 or later partitions. See page 5-41.

–MAX_EXTENT_SIZE, –MIN_EXTENT_SIZE

Specify maximum and minimum extent sizes for CAM files for Rev. 22.0 and later partitions. See page 5-42.

–USAGE, –HELP

Display the command line syntax and a complete list of options or a description of the options. See page 5-44.

The following options are obsolete but are supported. Their replacements are shown in the right column.

Obsolete Options	*Replacement*
–INTELLIGENT_CONTROLLER (–IC)	–DBS ON
–ALL_CONTROLLER (–AC)	–DBS OFF
–COPY_BADSPOTS_BY_DEVICE (–CPYDEV)	–COPY_BADSPOTS *pdev*
–COPY_BADSPOTS_BY_NAME (–CPYNAM)	–COPY_BADSPOTS *name*
–OVERRIDE_DEFAULT_INTERLEAVE (–ODI)	–SECTOR FORWARD
–RESTORE_DEFAULT_INTERLEAVE (–RDI	–SECTOR REVERSE

The only options you should use on SCSI disks are the following:

–DISK	–PARTITION
–DISK_TYPE	–SPLIT
–BAUD_RATE	–DISK_REVISION
–MAX_EXTENT_SIZE	–MIN_EXTENT_SIZE
–REPORT	

Deciding Which Options to Use

This section explains the options listed above in more detail. Use these options to create partitions under various circumstances. List the options that you need to achieve the desired results. All of the MAKE options are also summarized in Appendix F for your convenience.

Before invoking MAKE, you must put the partition in the Assignable Disks Table with the DISKS command and assign it with the ASSIGN DISK command. The complete procedural steps for running MAKE are discussed and diagrammed after the options are described. See Figure 5-1 on page 5-53.

Specify the Physical Device Number: You must specify the spindle and the logical disk, or partition, on which MAKE is to operate. You do this by specifying the physical device number (pdev) of the partition either by using the –DISK option (abbreviation –DSK) or by entering the pdev at the `Physical device?` prompt. (See Chapter 3 for information on determining physical device numbers.) For example:

```
OK, MAKE -DISK 21220
```

Caution The –DISK option *must precede* the physical device number on the command line. If you omit the –DISK option but include the pdev or enter –DISK after the physical device number, the program may halt or unpredictable behavior may occur.

If you do omit the –DISK option before the pdev or enter it after the pdev, do the following:

- Press the Control and P keys simultaneously to stop the program. If you have assigned only one disk (the one to be formatted) to your terminal, it is unlikely that any harm has occurred.
- Use the RELEASE_LEVEL –ALL (RLS –ALL) command, then invoke MAKE again. Include the –DISK option on the command line in the correct place.

If you did not previously assign the partition, MAKE informs you and waits for you to assign it:

```
Device 21220 is not assigned.
MAKE paused to allow device assignment.  Type START to continue.
OK,
```

You can assign the disk at this point and enter START to resume execution of MAKE. MAKE will prompt for the pdev.

If you do not specify the physical device number (pdev) following the –DISK option or do not specify the –DISK option, MAKE prompts you for the pdev. If the pdev that you enter is invalid, MAKE tells you that it is invalid and again prompts you for the pdev:

```
Physical device? 21228
Invalid physical device number "21228"
Please enter octal number.
Physical device? 212200
Number too big, please reenter.
Physical device? 210
Pdev 210 is invalid, number of heads cannot be zero.
Physical device?
```

If you attempt to recreate the head zero partition of a disk type that is listed in Table 5-3 (page 5-17) and that is a Rev. 21.0 or later disk by using a pdev that specifies a different number of heads and a DBS file exists on the spindle, MAKE warns you with a message similar to the following and you are queried to be sure you want to continue:

```
Partition pdev appears to have been previously made with a different size.
This MAKE will destroy the existing DBS file.

MAKE will enable -FORMAT and create a new DBS file.

WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

OK to continue with MAKE?
```

If you answer YES, MAKE continues; if you answer NO, MAKE aborts.

WARNING If you change the number of surfaces in a head zero partition of these disks (that is, you change the basic pdev of the head zero partition), you force MAKE to rebuild the DBS file and the RMA of the spindle and all other partitions on that spindle may lose any data that was remapped to the RMA. This is why MAKE displays warnings similar to that in the preceding example.

Name the Partition: Use the –PARTITION option (abbreviations –PART, –PAR) to specify the partition name. The partition name may contain a maximum of six characters. The first character must not be a digit or a dash (–); the name can contain only alphabetic characters, digits, and the special characters _ # $ & * – . /. For example:

```
OK, MAKE -DISK 21220 -PARTITION SYS_24
```

The partition name serves as the name of both the partition and the file containing the DSKRAT. Make sure that each partition has its own unique partition name.

You may change the partition name later by using the –RENAME option of either the ADDISK or of the SHUTDN command.

If you do not specify the –PARTITION option on the command line or do not specify a name with the option or if you specify an invalid partition name, MAKE prompts you for a partition name:

```
Invalid partition name "diskname".
Partition name?
```

Specify the Disk Type: You must specify the type of disk either using the –DISK_TYPE option (abbreviations –DTP, –DT) or when prompted if you do not use the option. If you do not know the disk type, MAKE displays a list of disk types and prompts you for one, as in the example below. Follow the –DISK_TYPE option with one of the arguments shown in the following example and in Table 5-2. If you do not follow the –DISK_TYPE option with the disk type argument or if you specify an invalid disk type (one that is not known to MAKE), MAKE prompts you for a correct one and displays a list of valid disk types if you choose to see a list by pressing the Return key.

```
OK, MAKE -DISK 21260 -PARTITION FALCON -DISK_TYPE
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Disk type? (press <return> for a list) <Return>

Enter one of the disk types in the left column:

                                          Default    No.  of     Records
  -DISK_TYPE        Description           -BADLEV     heads     per head
  ----------  ----------------------      -------    -------    --------
  SMD          80MB or 300MB removable       1       5 or 19      7407
  CMD          Cartridge module device       1         21         7407
  68MB         68 megabyte fixed media       4          3        10071
  158MB       158 megabyte fixed media       4          7        10071
  160MB       160 megabyte fixed media       4         10         7389
  600MB       600 megabyte fixed media       4         40         7569
  MODEL_4475  300 megabyte fixed media       4         19         7407
  MODEL_4714   84 megabyte fixed media       2          5         8120
  MODEL_4711   60 megabyte fixed media       2          4         7140
  MODEL_4715  120 megabyte fixed media       2          8         7140
  MODEL_4735  496 megabyte fixed media       4         24         9954
  MODEL_4719  258 megabyte fixed media       2         17         7320
  MODEL_4845  770 megabyte fixed media       4         23        16112
  MODEL_4721  328 megabyte fixed media
                - on model 2382 controller   2         12        13128
                - on model 7210 controller   0         31         5080
  MODEL_4860  817 megabyte fixed media       4         15        26201
--More--
  MODEL_4729  673  MB SCSI fixed media       0         31        10414  [4]
  MODEL_4730  213  MB SCSI fixed media       0         31         3302  [4]
  MODEL_4731  421  MB SCSI fixed media       0         31         6350  [22]
  MODEL_4732  1.34 GB SCSI fixed media       0         31        20574  [21]

  Where "[n]" is shown, add 254 to Records/head for the first <n> heads.

Disk type? (press <return> for a list)
```

Caution Be sure that you specify the correct disk type when creating a Rev. 21.0 or later partition. MAKE uses the geometry information of the physical disk to construct the DBS file and the RMA on those spindles that are connected to an IDC1 controller. They will not be the correct size if MAKE has incorrect information.

Be sure that you specify the correct disk type in any case. If you do not, MAKE will attempt to create the type of disk you have specified and will encounter an error condition. You can recover by rerunning MAKE, using the correct disk type argument, but you will have wasted time. If MAKE ran on the disk previously, MAKE can detect that the type is different and displays this message:

```
WARNING:  Partition pdev appears to have been previously made with type
          MODEL_4732, but you have specified type MODEL_4729.

OK to continue with MAKE?
```

You should answer NO and enter the correct disk type.

Table 5-2 notes that disks that are connected to the Model 2382 and Model 7210 SCSI disk controllers are SCSI disks. Because media defects, or badspots, are handled by the drive on these disks, MAKE does not create a badspot file or a DBS file on these disks and references to these files do not apply to these disks.

The table also indicates, with *DBS spindle* in the comments column, which disks can be connected to an IDC1 disk controller and thus have Dynamic Badspot Handling occurring on them.

Split the Disk: If you are creating a partition that is to be used either for paging or for both paging and storage of user files, or if you want to create a crash dump disk, specify the –SPLIT option (abbreviation –SPL). If you omit an argument with –SPLIT or use an invalid argument, MAKE displays the total number of records available for paging and then prompts you for the number of records to reserve for paging. Both of these numbers are decimal. For example:

```
OK, MAKE -DISK 40535 -PARTITION PAGING -SPLIT
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Please specify how you want to -SPLIT this partition.
Maximum number of paging (or CDD) records available is 22213.
Number of paging/CDD records? 22213
```

Table 5-2. Disk Types and Arguments

Disk Type	*Argument*	*Comments*
Storage Module Disks: 80MB or 300MB	SMD	Removable disk packs DBS spindles
Cartridge Module Devices: 32MB, 64MB, or 96MB	CMD	Each has one removable surface
Fixed Media Disks:		
60MB FMD	MODEL_4711	2455™ system internal SCSI disk
68MB FMD	68MB	DBS spindle
84MB FMD	MODEL_4714	2455 system internal SCSI disk
120MB FMD	MODEL_4715	2455 system internal SCSI disk
158MB FMD	158MB	DBS spindle
160MB FMD	160MB	DBS spindle
258MB FMD	MODEL_4719	2455 system internal SCSI disk
315MB FMD	MODEL_4475	DBS spindle
328MB FMD	MODEL_4721	2455 system internal SCSI disk and SCSI disk on ICOP+
496MB FMD	MODEL_4735	Internal flaw maps, DBS spindle
675MB FMD	600MB	Same as 600MB FMD
770MB FMD	MODEL_4845	Internal flaw maps, DBS spindle
817MB FMD	MODEL_4860	Internal flaw maps, DBS spindle
673MB FMD	MODEL_4729	SCSI disk on ICOP+
213MB FMD	MODEL_4730	SCSI disk on ICOP+
421MB FMD	MODEL_4731	SCSI disk on ICOP+
1340MB FMD	MODEL_4732	SCSI disk on ICOP+

The effect of the –SPLIT option is to produce a partition split into two functional parts: a file system part (often minimal in size) and a part reserved for paging or for crash dumps. The argument after –SPLIT specifies the number of records to be reserved for paging or CDD.

If you enter a non-numeric character for the number of paging records or enter a return, MAKE simply reprompts for the number of paging records. If you specify an invalid number for the paging records – for example, if you specify more records than are in the partition – MAKE tells you and prompts again, telling you the maximum number of records available.

See the *System Administrator's Guide: Vol. I System Configuration* for a discussion of determining the amount of paging space needed. Prime recommends that you have 1000 paging records for each configured user (NTUSR+NPUSR+NRUSR+NSLUSR). If you are creating crash dump disks, see the discussion on page 5-18.

After you determine how much paging space you need and how you are going to distribute it among paging partitions, you must leave a minimum number of records for file system storage on each paging partition. The simplest method of specifying how many records to use for paging is to use the –SPLIT option with the MAXIMUM argument (abbreviation MAX). MAKE then determines the maximum space available for paging and leaves a minimum file system. If you do not specify any argument, MAKE displays the maximum available and you can enter that number as in the above example or enter MAX.

Table 5-3 shows the number of records to be used for the DBS file and the RMA in the file system portion of a split partition if it is the head zero partition of a disk type that supports Dynamic Badspot Handling and the spindle is connected to a Model 6580 (IDC1) disk controller. MAKE determines the correct number of records to use for the file system portion using information similar to that in Table 5-3 plus the amount of records for other file system entities, such as the DSKRAT. MAKE then subtracts those records from the total and displays the resulting maximum records available. You can then specify that number or specify MAXIMUM with the –SPLIT option.

Table 5-3. Records Needed for the DBS File and RMA on a Head Zero Partition of Spindles Supporting DBS

Disk Type	*DBS File*	*RMA*
80MB SMD	1	32
300MB SMD	1	32
68MB	1	100
158MB	2	200
160MB	2	200
600MB	6	1500
MODEL_4475	2	400
MODEL_4735	3	750
MODEL_4845	4	1000
MODEL_4860	4	1000

If you enter a non-numeric character for the number of paging records or enter a return, MAKE simply reprompts for the number of paging records. If you specify an invalid number for the paging records – for example, if you specify more records than are in the partition – MAKE tells you and prompts again, telling you the maximum number of records available.

If you intend to use a partition as a paging partition, use the –SPLIT option. If you do not specify the –SPLIT option on the command line and later use this partition for paging, the badspot file is overwritten. In addition, you will not be able to add the partition to the system with the ADDISK command, if this is desired, because only the file system portion of a split disk can be added. If you later specify this partition as a paging partition by including its pdev with the PAGING directive, PRIMOS queries you at cold start as follows:

```
PAGDEV pdev does not point to the beginning of a valid file system partition.
Are you SURE you want to page on PAGDEV pdev?
```

If you answer NO, PRIMOS prompts for another paging device. If you answer YES, PRIMOS splits the partition and does not prompt you again when the system is next cold started.

If you know how many records are to be used for paging and you do not want to be prompted for this information, specify the number of records to be used for paging or specify MAX following the –SPLIT option on the command line. For example:

```
OK, MAKE -DISK 20061 -PARTITION PAGING -SPLIT 22213
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Making 3 head partition:    PAGER               (Disk Revision 22.1)
Disk type:                  SMD
Number of tracks:           823
Sectors per track:          9
Sectoring mode:             FORWARD
Dynamic Badspot handling:   OFF  (-AC)

File system records:        8
Paging records:             22213
```

Note The file system portion of a split paging partition should be used only for storing information on badspots on the disk. The system suffers a performance penalty if you use the file system portion of a split partition for user file storage, particularly when the paging portion is used heavily.

This rule applies to all systems, with the possible exceptions of the 2350™, 2450™, 2455, and 2250™ systems. On these systems, split disks with heavy user and paging use are often necessary, particularly on those systems that have only one 60MB or 68MB disk.

Crash Dump Disks: The discussion concerning use of the –SPLIT option for creating a paging partition also applies to using –SPLIT to create a crash dump partition. The amount of space required for a crash dump (to disk or to tape) depends on the system configuration and the type of crash dump, full or partial. Prime recommends that you use partial crash dumps.

To determine the size of the crash dump and, thus, the value to use with the –SPLIT option when creating a crash dump disk, use the –INFO option of the CDD command. (For details on using the CDD command, see the *Operator's Guide to System Commands*.) Use of this option alone gives you the sizes for a full crash dump and for a partial crash dump.

You can use other options with –INFO to specify the pdev or name of the disk you will use for the crash dump and the type of this disk. You can also determine the dump sizes for other total memory sizes and CPU types, in order to plan crash disks for other machines in your network. Options for use in specialized circumstances allow you to override the dump sizes calculated by default and to display a table of all optimal –SPLIT values.

Normally you would use the following command format to determine the value to use with the –SPLIT option of MAKE. CDD determines the size of the memory for the system you are on and calculates the required dump sizes.

```
OK, CDD -INFO
[CDD Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.

------------------------------------------------------------------------------
| This system has 64 MB of core memory.  Expected total sizes for full and    |
| partial dumps are made up as follows:                                       |
|                                                                             |
|                                 FULL DUMP           PARTIAL DUMP            |
|                             ----------------      ---------------           |
|    CORE memory dump :          32767 records       16384 records (approx)  |
|    MAPS dump        :             42 records          42 records           |
|    PIOS dump        :             65 records          65 records           |
|    Safety margin    :            100 records         100 records           |
|                             ----------------      ---------------           |
|    TOTAL DUMP SIZE  :          32974 records       16591 records (approx)  |
|                             ================      ===============           |
|                                                                             |
------------------------------------------------------------------------------
For MAKE recommendations, please specify the disk you intend to use for CDD.
Enter "Q" to quit, or "H" for help.

Enter <pdev> or disk name:
```

For a partial dump, you can now see that you need approximately 16591 records of disk space. Assuming you have a Model 4729 disk, which has 10414 records per surface on the last 27 surfaces, you can dedicate the last three surfaces for the crash dump space (and some file system space) and the remaining surfaces for a file system. (One surface is too small and starting surface numbers must be even so you need three surfaces.) From Table 3-3, the basic pdev for the last three surfaces (starting surface 28) is 160421. Assuming this disk is on controller 26_8 and drive unit 0, you add 40 for a pdev of 160461. You can now enter this information to CDD and obtain the value to use with –SPLIT.

```
Enter <pdev> or disk name: 160461

Please specify a MAKE-compatible disk type for disk 160461.
Enter "H" for Help, or "Q" to quit.
```

```
Enter disk type (e.g.  "MODEL_4729"): MODEL_4729
--------------------------------------------------------------------------------
| The crash disk you have specified has the following characteristics:         |
|                                                                              |
|     Disk 160461       :     3 heads, starting head 28  (ctlr '26, unit 0)    |
|     Disk model        :     MODEL_4729                                       |
|     Total disk size   :     31242 records                                    |
|                                                                              |
| To MAKE this disk with the maximum possible crash dump capacity:            |
|                                                                              |
|     MAKE disk with    :     -SPLIT 30989          (see note 1 below)        |
|     Maximum dump size:      30988 records         (see note 2 below)        |
|                                                                              |

| ************* This disk is TOO SMALL for a full dump.  *************        |
| For this disk to accommodate a partial dump of the size predicted           |
| earlier, the smallest -SPLIT value you can specify to MAKE is:              |
|                                                                              |
|                               FULL DUMP           PARTIAL DUMP              |
|                            ---------------       ---------------            |
|     MAKE disk with    :    ** TOO SMALL **        -SPLIT 16764              |
|     Maximum dump size:                           (16763 records)            |
|                                                                              |
--------------------------------------------------------------------------------
Type <return> for explanatory notes, or "Q" to quit:
```

You now see that the three surfaces of this disk will accommodate a partial dump (but not a full dump). You then should use MAKE with the –SPLIT option with an argument of 16764. You can use the remaining records on this partition (31242 – 16764 = 14478) for a file system.

If you use only the –INFO option without specifying the pdev or the disk type, CDD prompts you for this additional information in order to recommend the values that MAKE needs to create the crash dump disk. To avoid being prompted for this information, you could use this command line:

```
OK, CDD 160461 -DT MODEL_4729 -INFO
```

If you want to depart from the –SPLIT value recommended by CDD –INFO, you should consult a table of optimal dump sizes for your particular system and disk type by using the CDD –DUMP_SIZE_TABLE option (abbreviation –DST). Be sure to use these optimal –SPLIT values. The table appears like this:

```
OK, CDD 160461 -DT MODEL_4729 -DST 14000 1000
[CDD Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
-------------------------------------------------------------------------------
| The crash disk you have specified has the following characteristics:       |
|                                                                             |
|     Disk 160461      :     3 heads, starting head 28  (ctlr '26, unit 0)    |
|     Disk model       :     MODEL_4729                                       |
|     Total disk size  :     31242 records                                    |
|                                                                             |
| To MAKE this disk with the maximum possible crash dump capacity:           |
|                                                                             |
|     MAKE disk with   :     -SPLIT 30989          (see note 1 below)         |
|     Maximum dump size:     30988 records         (see note 2 below)         |
|                                                                             |
| DUMP SIZE TABLE:                                                            |
|                                                                             |
| For this disk, optimal splits are those for which either the maximum dump  |
| size (MDS) or the -SPLIT value (S) is an exact multiple of 254 records,    |
| and S = MDS + 1.  Below is a table of optimal -SPLIT values, beginning     |
| from the dump size closest to 14000 records, and approx 1000 apart:        |
|                                                                             |
|     MAKE with -SPLIT  14224 for a maximum dump size of  14223 records      |
|     MAKE with -SPLIT  15240 for a maximum dump size of  15239 records      |
|     MAKE with -SPLIT  16002 for a maximum dump size of  16001 records      |
|     MAKE with -SPLIT  17018 for a maximum dump size of  17017 records      |
--More--
                    .                    .                    .
                    .                    .                    .
                    .                    .                    .

|     MAKE with -SPLIT  30226 for a maximum dump size of  30225 records      |
|     MAKE with -SPLIT  30989 for a maximum dump size of  30988 records      |
|     - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -      |
|     End of table - preceding line represents maximum capacity of disk      |
-------------------------------------------------------------------------------
Type <return> for explanatory notes, or "Q" to quit: Q
OK,
```

Create a Specific-Rev. Partition: Normally, Rev. 23.3 MAKE creates a Rev. 22.1-format partition. If the partition being created is to be run on a pre-Rev. 23 or a pre-Rev. 22.1 system, include the -DISK_REVISION option (abbreviations -DSKREV, -REV) on the command line. If you do not specify this option, MAKE creates a Rev. 22.1-format partition by default.

The valid arguments to this option are 18, 19, 20, 21, 22, and 22.1. (You can use a decimal point and zero combination with any of the major revisions, 21.0, for example.) If you do not specify one of these six arguments on the command line with the -DSKREV option, MAKE prompts you for one and checks the validity of the argument you enter. If the argument is out of the range of the arguments listed here, MAKE informs you that you entered a bad revision and reprompts until you enter a valid disk revision or enter QUIT.

```
OK, MAKE -DISK 41062 -DT SMD -PART JUPITR -DSKREV
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Disk revision? 24

Bad disk revision "24".  Must be 18, 19, 20, 21, 22, or 22.1.
Disk revision? 22.0

Making 4 head partition JUPITR
```

Set the Baud Rate: MAKE writes the boot program onto the disk when it creates a partition. The baud rate of the supervisor terminal is set in the boot program. If the partition being made will be used for booting PRIMOS, you can set or change the supervisor terminal baud rate when you create the partition.

If the baud rate setting of the partition disagrees with the actual baud rate set on the supervisor terminal and your machine does not have a VCP III or later Diagnostic or Maintenance Processor, you may be unable to use the supervisor terminal after booting from the newly created partition. Set the baud rate to the baud rate of your supervisor terminal with the –BAUD_RATE option (abbreviations –BAUD, –BAU) if the partition will be used for booting PRIMOS.

The default baud rate is 300 bits per second (bps). To set a different baud rate, include the –BAUD_RATE option on the command line, followed by the desired baud rate in decimal bits per second. Valid baud rates are: 110, 300, 1200, 9600. For example, to set the baud rate to 9600, use a command line like this:

```
OK, MAKE -DISK 1060 -BAUD_RATE 9600 -DT MODEL_4715
```

If you include the –BAUD_RATE option on the command line but do not specify the baud rate, MAKE prompts for a valid rate. If you specify an invalid baud rate, MAKE displays a list of valid baud rates and prompts you for a valid rate.

```
OK, MAKE -DISK 1060 -BAUD_RATE -DT MODEL_4715
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Partition name? PAYROL

Baud rate? 9600
```

If you use the –BAUD_RATE option, MAKE displays the baud rate you specify with the initial display of characteristics. If you do not specify –BAUD, MAKE does not display the baud rate and the rate is set to the default of 300.

Format a New Disk: The –FORMAT option (abbreviation –FMT) causes MAKE to format every track in the partition; that is, to organize the medium on the disk surfaces into sectors, or records, that the controller can recognize by placing special bit patterns before and after each record. This is necessary before the controller can read and write records on the disk.

–FORMAT is never needed when you use MAKE on disks connected to the Model 2382 or Model 7210 disk controllers.

–FORMAT is needed on other disk types in these situations:

- If the partition has never been used on a Prime system before
- To reinitialize a partition on a spindle on which Dynamic Badspot Handling is being enabled or disabled, or the DBS file is being rebuilt or removed

Reinitializing a partition is applicable only to disk types that support DBS. On these disk types, the purpose of formatting is to reinitialize the Dynamic Badspot Handling remapping pointers, as described in Chapter 8. On these spindles, you need to specify –FORMAT whenever you use MAKE to:

- Change the pdev or the –SPLIT value (the paging or crash dump area) of the head zero partition
- Convert a spindle from –DBS ON to –DBS OFF
- Convert a spindle from –DBS ON to pre-Rev. 21.0

You must make partitions on spindles that support Dynamic Badspot Handling in sequence, making the head zero partition first. You must also format (using the –FORMAT option) all partitions on such spindles in the same session. That is, if you format one partition, you must format all partitions on that spindle.

WARNING Do not use a pre-Rev. 23.3 version of MAKE on any partition of a Rev. 22.1-format disk that has Dynamic Badspot Handling (–DBS ON) occurring on it. Do not use any version of MAKE with the –FORMAT option on any partition of a DBS spindle unless you remake all other partitions on the spindle, starting with the head zero partition. If you do either, the entire spindle (all partitions on the spindle) may be corrupted because MAKE overwrites the DBS file in these cases. Consequently, the data for records that initially were to be written to badspots on any partition of the spindle, and that were thus written to the RMA, are lost. Be sure all the data on all partitions on a spindle is backed up before using MAKE.

You should not use the –FORMAT option on the head zero partition unless you are performing one of the operations in the list above on the spindle as a whole because of the risk to the other partitions. If you attempt one of these risky operations, MAKE will warn you like the following.

```
OK, MAKE -DISK 2062 -DT 4845 -PAR TEST -DBS OFF
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

An activated DBS file exists on partition 2062.  If you proceed with
-DBS OFF (-AC), MAKE will enable -FORMAT to deactivate the DBS file.

WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

Are you sure you want to make this partition with -DBS OFF?
```

With the –NO_QUERY option, the –FORMAT option has an extra and special connotation. In situations where the above warning is printed, if you use the –FORMAT or –FORMAT_OK option, –FORMAT and –FORMAT_OK are construed by MAKE as permission to proceed with the risky operation.

The option –FORMAT_OK (abbreviation –FOK) is a special form of the –FORMAT option intended for use in conjunction with the –NO_QUERY option. It is discussed on page 5-33 in conjunction with –NO_QUERY.

Note Since all disks connected to the Model 2382 and the Model 7210 disk controller are preformatted, you should not use the –FORMAT option on these disks. When you use the –FORMAT option on SCSI disks connected to the Model 7210 controller, MAKE displays the following message:

```
This is a pre-formatted disk: -FORMAT ignored.
```

MAKE then continues with normal operations. In addition, because badspots are handled by the disk drive, you should not use the –NEWDSK option on these disks.

The –FORMAT, –FORMAT_OK, and –NEW_DISK options are ineffective and you should not use them if the disk type is one of the following SCSI FMDs:

Model 4711 (60MB)	Model 4730 (213MB)
Model 4714 (84MB)	Model 4729 (673MB)
Model 4715 (120MB)	Model 4731 (421MB)
Model 4719 (258MB)	Model 4732 (1.34GB)
Model 4721 (328MB)	

These disks are preformatted when they are manufactured and badspots are handled by the disk drive. It is recommended that you do not use the –FORMAT option or the –NEW_DISK option with these disk types to save processing time.

Process a New or Corrupted Disk: The –NEW_DISK option (abbreviations –NEWDSK, –NEW) specifies that MAKE should not expect to find any valid data existing on the disk. The purpose of the –NEW_DISK option is to prevent MAKE from printing unwanted error messages if invalid structures (such as corrupted MFD, DSKRAT, or badspot files) are found on the disk.

The –NEW_DISK option is *never* essential but you can use it to suppress messages that MAKE displays if the disk is new or known to have a damaged and corrupted file system and you do not use the –NEW_DISK option.

Examples of some of these messages that are suppressed by the –NEW_DISK option are the following:

```
Cannot read DSKRAT on partition pdev.
Cannot determine if there is an active Dynamic BadSpot file on this spindle.

Invalid DSKRAT on partition pdev.
Cannot determine if there is an active Dynamic BadSpot file on this spindle.

Invalid DBS file on partition pdev.
Cannot determine if there is an active Dynamic BadSpot file on this spindle.
```

The following would occur if MAKE was looking for a BADSPT file or if the disk had been previously used on a Prime system but the file system was damaged in some way or you were rearranging the pdevs on the spindle. In all of these cases, MAKE continues.

```
BADSPT file on device pdev is not a SAM file!

BADSPT file on device pdev is invalid.

MFD on device pdev is invalid; no BADSPT file found.

Unable to read BADSPT file on device pdev.

No BADSPT file found on device pdev.
```

–NEW_DISK also suppresses the warnings displayed when MAKE detects that the partition has previously been made with a different disk type as specified with the –DISK_TYPE option (see examples on pages 5–14 and 5–52).

Because –NEW_DISK implies there is no valid data on the disk, if MAKE finds a valid BADSPT or DBS file on the partition, MAKE displays warnings to ensure that valid badspot information is not inadvertently discarded as in the following examples. In the first example, the partition has an existing BADSPT file and MAKE queries to see if the user really wants to ignore it. MAKE will also create a DBS file on this spindle but the DBS file will be inactive.

```
OK, MAKE -DISK 20462 -DT 4845 -PAR TEST3 -DBS OFF -NEW_DISK
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Making 2 head partition:     TEST3              (Disk Revision 22.1)
Disk type:                   MODEL_4845
Number of tracks:            848
Sectors per track:           19
Sectoring mode:              REVERSE
Dynamic Badspot handling:    OFF   (-AC)
Partition size in records:   32224

Processing flaw map on device 20462.
Processing of flaw map completed.

A BADSPT file exists on device 20462.  Really ignore and overwrite it? Y

Badspots on device 20462:
        1 known badspot (from badspot file/s, flaw map, or terminal input)
        0 new badspots found during badspot checking or file system I/O
Total of 1 badspot mapped out of device 20462.

Partition TEST3 created successfully.
OK,
```

The second example uses -DBS ON and -NEW_DISK. This would cause MAKE to ignore the existing DBS file and create a new one. In order to create a new one, MAKE must use -FORMAT to remove the existing DBS file. This will affect the RMA and all other partitions on the spindle.

```
OK, MAKE -DISK 462 -DT 4845 -PAR TEST3 -DBS ON -NEW_DISK
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

A valid DBS file exists on partition 462, containing 9 badspots.
-NEWDSK implies that you wish to ignore this DBS file and create a new one.

If you ignore this DBS file, MAKE will enable -FORMAT to remove it fully
before creating the new one.


WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT

Really ignore DBS file? [Y,N,Q]
```

This is a warning to let you know that a change in the DBS file results in the loss of any remapped records from other partitions on the disk. If you answer NO, MAKE aborts. Answer YES if your intent is to repartition the disk with new geometry and a new DBS file. If you do not specify the –FORMAT option, MAKE enables that option in order to remodify record headers.

When you specify the –NEW_DISK option, MAKE checks for the existence of a badspot file whether the partition is in Dynamic Badspot Handling (–DBS ON) mode or is in Nondynamic Badspot Handling (–DBS OFF) mode.

Note MAKE does not format or create badspot files on any disk drives connected to the Model 7210 and the Model 2382 SCSI disk/tape controllers.

Initialize File System Records: Formerly, it was necessary for MAKE to initialize the records in a partition. Initialization consists of writing the record's address into the CRA field of the record header. PRIMOS now initializes records when creating file system objects, so it is not necessary for MAKE to do it. At Rev. 23.3, MAKE does not initialize records by default, thus saving some processing time. Prior to Rev. 22.1, you had to use the –NO_INIT option (abbreviation –NIN) to prevent MAKE from initializing the records.

WARNING Although MAKE does not initialize file system records by default at Rev. 22.1 and previously by the use of the –NO_INIT option, MAKE does initialize the file system. All file system objects are effectively deleted when you run MAKE on a partition. Use of –NO_INIT, specifically or by default, merely reduces the amount of time needed to create a particular partition. Never use MAKE on a partition if it contains the only copy of any data that you want.

However, having MAKE initialize records may help if you are trying to locate file system errors, particularly errors resulting from hardware problems. If you want MAKE to initialize records, use the –INIT option. This causes MAKE to reinitialize the CRA field. You cannot use both the –INIT and the –NO_INIT options on the same command line. If you do, MAKE displays this error message and aborts:

```
Options -INIT and -NO_INIT cannot be used together.
```

Display the Percent Completeness of Formatting, Badspot Checking, and File System Initialization: If you use the –FORMAT option and if MAKE checks for badspots, you can use the –REPORT option (abbreviation –RPT) to cause MAKE to display how formatting and badspot checking are proceeding. The display looks like this:

```
OK, MAKE -DISK 5463 -PAR GOLD -DT MODEL_4845 -DBS ON -BADLEV 4 -INIT -RPT -FMT
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]
```

```
Making 23 head partition:    GOLD                (Disk Revision 22.1)
Disk type:                   MODEL_4845
Number of tracks:            848
Sectors per track:           19
Sectoring mode:              REVERSE
Dynamic Badspot handling:    ON  (-IC)
Partition size in records:   369572

Processing flaw map on device 5463.
9 badspots added from flaw map on device 5463.

Beginning format.
Format now  5% complete.
Format now 10% complete.
         .  .  .
         .  .  .  .  .
         .  .  .  .  .
Format now 90% complete.
Format now 95% complete.
Format completed.


Checking for badspots (level = 4).
Badspot checking now  5% complete.
New badspot detected writing record = '1160 (track = 4, head = 0, sector = 16)
Badspot checking now 10% complete.
Badspot checking now 15% complete.
         .  .  .
         .  .  .
         .  .  .
Badspot checking now 90% complete.
Badspot checking now 95% complete.
Done checking for badspots.  2 new badspots detected.

Badspots on device 5463:
          9 known badspots (from badspot file/s, flaw map, or terminal input)
          2 new badspots found during badspot checking or file system I/O
Total of 11 badspots mapped out of device 5463.

Partition GOLD created successfully.

OK,
```

In this example, the disk is one that has a vendor flaw map but the user specified the –BADLEV option causing MAKE to check for badspots. MAKE reports the percent completion of formatting and badspot checking.

If the badspot verification level is 0 and you use the –INIT option (–NO_INIT is the default), the badspot message is replaced by the message `Disk Initialization now XX% complete.` You do not see the initialization message if MAKE is doing badspot checking because the initialization time is part of the badspot checking time.

–REPORT also causes MAKE to report the number of badspots added from the vendor flaw map, the BADSPT file, and the DBS file, if they are present. See the section Reporting Badspots on page 5-44 for examples of reporting of badspots.

No reporting is done unless you specify the –REPORT option; –REPORT is off by default. In addition, because MAKE ignores the –FORMAT option and defaults badspot checking level to 0 for the SCSI disks connected to the Model 7210 disk controller, no reporting is done on these disks.

Create a Partition Capable of Dynamic Badspot Handling: If your system has a Model 6580 intelligent disk controller (IDC1) and the software to download the controller, this is indicated by the following message displayed at cold start.

```
DLL and init ICOP complete (30026) - (disk_init)
```

Use the –DBS ON option (–IC) so that Dynamic Badspot Handling can take place on the partitions on spindles listed below that are connected to IDC1 intelligent disk controllers. If you intend to mirror partitions on disks connected to the IDC1, you must use this option. Mirroring can be done only with Rev. 21.0 or later revision partitions on spindles connected to IDC1 disk controllers and in Dynamic Badspot Handling (–DBS ON) mode or on SCSI disk connected to a Model 7210 controller running ICOP+. If you intend to create a crash dump disk, you must also use the –SPLIT option as described on page 5-18. Otherwise the requirements for crash dump disks are the same as for mirrored disks. (See Chapter 8 for details on Dynamic Badspot Handling and Chapter 9 for details on mirroring partitions.)

The format for creating partitions on spindles capable of Dynamic Badspot Handling is

MAKE –DBS ON

The disk type, used with the –DISK_TYPE option or entered at the prompt for disk type, must be one of the types listed below and in Table 5-3 under item 4. Because SMDs can be moved from one disk drive to another and, thus, from one disk controller to another, it is recommended that you not use –DBS ON with SMDs. In addition, you cannot use –DBS ON with the CMD type, the disks (Models 4711, 4714, 4729, 4715, 4719, or 4721) connected to the Model 2382 SCSI disk/tape controllers on 2350, 2450, and 2455 systems, or the disks connected to the Model 7210 disk/tape controller. Thus, you can use –DBS ON only with the disk types listed here.

Disk types that support Dynamic Badspot Handling

SMD (80MB and 300MB)	
68MB	158MB
160MB	600MB
MODEL_4475 (315MB)	MODEL_4735 (496MB)
MODEL_4845 (770MB)	MODEL_4860 (817MB)

You must also use the –FORMAT option the first time you create the head zero partition of a disk with –DBS ON so that MAKE initializes record headers and creates the dynamic badspot file for the intelligent disk controller.

Note When running MAKE.SAVE standalone, MAKE does not support –DBS ON because MAKE cannot determine the type of controller before PRIMOS is running and because the controller cannot be downloaded at that time. You can later convert the partition to –DBS ON with FIX_DISK.

Whether you are running MAKE under PRIMOS or standalone, if you do not specify either –DBS ON or OFF, MAKE does the following:

- Ignores the –DBS option with disks on the Model 7210 and Model 2382 disk controllers because badspots are handled by the disk drive.
- For the DBS supporting disks listed on page 5-30 and the head zero partition, MAKE
  - Prompts for ON or OFF if PRIMOS is running and if the controller is an IDC1, otherwise requires OFF and prompts for confirmation.
  - Displays a message that says you must use –DBS OFF and prompts for confirmation if MAKE is run standalone.
- For the DBS supporting disks listed on page 5-30 and a non-head zero partition, MAKE
  - Defaults to the same mode as head zero partition if PRIMOS is running and informs you of this.
  - Aborts if the head zero partition is valid and assigned and MAKE is run standalone and the head zero partition has –DBS ON; otherwise, if the head zero partition has –DBS OFF, MAKE defaults to –DBS OFF.
  - Prompts for the mode of the head zero partition and uses that. Does not allow –DBS ON when running standalone if the head zero partition is not valid or assigned.

You cannot specify either the –DBS ON or –DBS OFF option with pre-Rev. 21.0 partitions.

Starting at Rev. 21.0, MAKE creates a DBS file and an RMA on the first, or head zero, partition of a spindle. (See Chapter 8.) MAKE uses the DBS file and the RMA to handle badspots on all partitions on that spindle. Thus, you must create the head zero partition of a –DBS ON mode spindle *before* creating other partitions on that spindle. The DBS file and the RMA are activated when the spindle is in Dynamic Badspot Handling (–DBS ON) mode.

When making a non-head zero partition on a disk that supports DBS and on an IDC1, MAKE attempts to enforce the rule that all partitions on the spindle must have the same mode. If the head zero partition is assigned, MAKE will not let you make the non-head zero partition with a different mode. If the head zero partition is invalid, MAKE recommends that you make it first. If the head zero partition is not assigned, MAKE recommends that you assign it. If the head zero partition cannot be assigned (for example, it is the COMDEV) or cannot be validly made, MAKE prompts you for the mode of the head zero partition and ignores any –DBS option if you specified one unless you used the –NO_QUERY option in which case MAKE interprets your –DBS option, which is required, as specifying the mode of the head zero partition.

Note When the head zero partition is not assigned, you must not continue with making a non-head zero partition unless you are certain of the –DBS mode of the head zero partition. If you are not certain of the head zero –DBS mode and get it wrong, undefined results may occur during subsequent use of the non-head zero partition under PRIMOS.

If there is any disturbance to the DBS file and the RMA, such as having to move them or changing the geometry of the head zero partition, you must remake all the partitions on the spindle starting with the head zero partition and using the –FORMAT option. MAKE warns you if such is the case.

See the section Examples of Running MAKE starting on page 5-60 for further information on the use of the –DBS option.

Create a Rev. 22.1 format Partition to Use on a Nonintelligent Disk Controller: If you wish to use a Rev. 21.0 or later disk type that is listed on page 5-30 on a drive that is connected to a nonintelligent disk controller, use –DBS OFF (–AC). This causes MAKE to create the file BADSPT (if there are badspots) for badspot handling. The DBS file and the RMA are created but are inactive. You can use FIX_DISK to switch between Nondynamic Badspot Handling (–DBS OFF) mode and Dynamic Badspot Handling (–DBS ON) mode.

Note Since all disk drives connected to the Model 7210 and the Model 2382 disk controllers handle all media defects during a MAKE session, neither the DBS nor the BADSPT file are created on a spindle connected to these controllers. In addition, –DBS OFF and –DBS ON are ignored by MAKE on these disks.

All partitions on a spindle should be in the same mode, either –DBS ON or –DBS OFF. You must create the head zero partition on the spindle before creating other partitions on that spindle.

The format for creating a partition in Nondynamic Badspot Handling mode is

MAKE –DBS OFF

You must specify the disk type, either with the –DISK_TYPE option or at the prompt, so MAKE can set aside the correct number of records for the DBS file and the RMA. You must use –DBS OFF when the disk type is CMD because you cannot use the CMD type with Dynamic Badspot Handling. You must also specify –DBS OFF when running MAKE standalone because intelligent disk controllers do not get downloaded with the necessary software until PRIMOS is booted. See the discussion on page 5-30 outlining what MAKE does if you do not specify either –DBS ON or –DBS OFF.

Note When running MAKE.SAVE standalone, MAKE does not support –DBS ON because MAKE cannot determine the type of controller before PRIMOS is running and because the controller cannot be downloaded at that time. You can later convert the partition to –DBS ON with FIX_DISK.

See the section Examples of Running MAKE starting on page 5-60 for further information on the use of the –DBS option.

Suppress Certain Queries when Running MAKE as a Phantom: Use –NO_QUERY (abbreviations –NQY, –NQ) when running MAKE as a phantom (from a CPL or COMI file), in order to prevent the phantom from aborting when it needs terminal input. Assuming other command-line options (particularly –FORMAT and –DBS, when these are needed) provide the information MAKE needs to proceed, –NO_QUERY prevents MAKE from prompting for the confirmation normally required in these situations:

- The partition has been previously made with a different disk type.
- When creating a non-head zero partition on a disk type that supports DBS, the head zero partition is not assigned or MAKE recommends that –FORMAT be enabled.
- When creating a head zero partition of a spindle that supports Dynamic Badspot Handling (DBS), MAKE will format the partition in order to deactivate, rebuild, or remove the DBS file in a way that requires –FORMAT to be enabled or in a way that risks loss of data on other partitions on the physical disk, or spindle.

With disk types that do not support DBS, only the first case is applicable. With the disk types that do support DBS (see page 5-30), all of the above are applicable. The additional command line options you must supply for –NO_QUERY to work properly are:

- Either –DBS ON or –DBS OFF, as appropriate.
- Either –FORMAT or –FORMAT_OK if MAKE determines that formatting is necessary. This is because MAKE will not enable –FORMAT without explicit permission because doing so can risk data loss on other partitions.

WARNING When you use –NO_QUERY, MAKE construes –FORMAT as permission to proceed in a way that risks loss of data on other partitions on the spindle. Use this combination with caution if there is any data of value on other partitions. It is recommended that you first run MAKE without –NO_QUERY to see any warnings. See the section Running MAKE as a Phantom on page 5-57 for further discussion.

When you use –NO_QUERY, if you do not supply the information needed, MAKE will abort with a message telling you that you supplied insufficient options and which options are needed, –FORMAT, –DBS ON, or –DBS OFF.

If you supply an invalid –DBS option with –NO_QUERY, MAKE will abort. In the following example, –NO_QUERY prevents MAKE from prompting the user to resolve the problem so MAKE has no option but to abort:

```
OK, MAKE -DISK 40760 -DT SMD -PAR TEST3 -DBS ON -LEV 0 -NQ
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition is not assigned;  cannot determine if there is an
active Dynamic BadSpot file on this spindle.

You should make this partition with the same -DBS option as was used to
make the head zero partition on this spindle.

MAKE is assuming the head zero partition to be made as you have specified,
with -DBS ON.  This partition will be made likewise.

Disk is connected to a controller that does not support Dynamic BadSpotting.
This partition cannot be made with -DBS ON (-IC).
ER!
```

Note –NO_QUERY does not suppress queries generated by –NEW_DISK. Do not use –NEW_DISK when running MAKE as a phantom.

Enable –FORMAT If Necessary: The option –FORMAT_OK (abbreviation –FOK) is useful when creating a head zero partition of a spindle that supports Dynamic Badspot Handling (DBS) and MAKE must format the partition in order to deactivate, rebuild, or remove the DBS file.

–FORMAT_OK tells MAKE to enable –FORMAT if MAKE determines that is necessary but do not format the partition otherwise. If you specify –FORMAT with –NQ, MAKE will always format the disk even if it was not strictly necessary and formatting is a time-consuming operation. –FORMAT_OK saves time by avoiding formatting whenever it is not needed.

When formatting is not necessary, –FORMAT_OK has no effect; MAKE always proceeds exactly as if –FORMAT_OK had not been specified. When MAKE determines formatting to be necessary, –FORMAT_OK is identical in effect to –FORMAT, and has the same power as –FORMAT to authorize a format that may risk data on other partitions. –FORMAT_OK has no effect whatever on any partitions except head zero partitions on disk types that support Dynamic Badspot Handling.

Specify the Level of Verification: MAKE can perform a maximum of four levels of media verification while creating a partition. Verification consists of writing a data pattern onto each record of the disk and reading the data back. This provides an automatic check for badspots on the disk.

Note As part of normal operation, MAKE may cause disk error messages to be generated and reported to the supervisor terminal. These messages are not true system disk errors. They are generated as a result of the detection of badspots. You should use CONFIG_UM to modify the DSM unsolicited message handler (UMH) and direct product LOG_DISK messages with INFORMATION severity to a log file and not to the supervisor terminal.

Normally, one (1) level of badspot checking is performed for SMDs and CMDs, two (2) levels of badspot checking are performed for Model 4711, Model 4714, Model 4715, Model 4719, Model 4721 (disks on the 2350, 2450, and 2455 systems), and four (4) levels of badspot checking are performed for all other FMDs except the Models 4729, 4730, 4731, and 4732 SCSI disks. MAKE, by default, does no verification on the latter disks because badspots are handled by the drive. If you have a Model 4721 (328MB) disk on a Model 7210 disk/tape controller, MAKE defaults to a level of 0 because badspots in that case are also handled by the drive and no badspot file is written to the disk. The –DISK_TYPE display shows the default level of badspot checking for all disks.

To override these defaults or to specify that no verification is to take place, include the **–BADSPOT_LEVEL** option (abbreviation **–BADLEV**, **–LEV**) on the command line, followed by an integer ranging from 0 through 4. This number represents the level of verification to be perfomed by MAKE. The 0 means no verification; 4 means four levels of verification (the most thorough and the most time-consuming). For example:

```
OK, MAKE -DISK 7625 -PARTITION IONIA -DISK_TYPE MODEL_4721 -BADSPOT_LEVEL 0
```

If you include the –BADSPOT_LEVEL option on the command line, but you do not include the number representing the level of verification, MAKE prompts you for it like this, in the case of an FMD:

```
Level of bad spot checking? (default = 4)
```

If you do not include the –BADSPOT_LEVEL option on the command line, MAKE defaults to the appropriate level of verification for the type of disk involved.

If you specify –BADLEV 0, MAKE does not do any badspot checking. If the partition has an existing badspot file, MAKE reads the existing badspot file. If the disk also has a flaw map, MAKE reads the flaw map. Specifying –BADLEV 0 may be useful if you are recreating an existing partition because MAKE uses less time if no badspot checking is done. However, do not use the –NEW_DISK option in this case. If you do, MAKE will write a new badspot file, overwriting the existing badspot file. Since you specified –BADLEV 0, the new badspot file will be empty.

Ignore the Flaw Map on Fixed-Media Disks That Have This Feature: Use the command line option –NO_FLAW_MAP (abbreviations –NOFLMP, –NFL) if you want to prevent MAKE from reading and processing the flaw map on the Model 4735, 4845, and 4860 FMDs. When you use the –NOFLMP option, MAKE does not attempt to find a flaw map on these disks, and thus no flaw map processing messages are displayed. You may want to use this option if it is known that the flaw map is bad or inaccessible. You ordinarily would *not* use this option.

MAKE also reverts to the default level of badspot checking for the disk involved if you use –NOFLMP. For the Model 4735, 4845, and 4860 disks, the default level of badspot checking is 4 (–BADLEV 4). To override the default level of badspot checking, use the –BADLEV option, specifying the level of badspot checking you want. For example, if you do not want MAKE to read the flaw map and you want two levels of badspot checking, use a command line like this:

```
OK, MAKE -DISK 1060 -DT MODEL_4735 -NO_FLAW_MAP -BADLEV 2
```

You may use the –BADLEV option regardless of which of these three disk types you have or the presence or absence of a flaw map on them. MAKE defaults to a badspot-checking level of 0 if a flaw map exists. If a flaw map does not exist, if it cannot be read, or if you use the –NOFLMP option, MAKE defaults to a badspot-checking level of 4 for these FMDs. If there is a flaw map on the disk and you do not specify the –NOFLMP option but you do specify the –BADLEV option, MAKE reads the flaw map and does the badspot checking specified.

Table 5-4 summarizes the following combinations.

- Whether or not flaw maps exist
- Whether or not you specify the –BADSPOT_LEVEL option
- Whether or not you specify the –NO_FLAW_MAP option

Table 5-4. Use of Options With Models 4735, 4845, and 4860 Disks

Flaw Map Exists	*Specify –BADLEV*	*Specify –NOFLMP*	*Result*
Yes	No	No	Read the flaw map No badspot checking performed
Yes	Yes	No	Read the flaw map Check badspots
Yes	No	Yes	Default level of badspot checking
Yes	Yes	Yes	Check badspots to level –BADLEV *n*
No	Yes	No	Check badspots to level –BADLEV *n*
No	No	Yes	Default level of badspot checking

Use the Flaw Map on Fixed-Media Disks That Have This Feature: The Model 4735 (496MB), the Model 4845 (770MB), and the Model 4860 (817MB) FMDs contain flaw maps, or lists of defects, written on a designated area of the disk medium during manufacturing. There is no option to specify that MAKE should use the flaw map on these disk types; MAKE reads the flaw map by default when you use the –DISK_TYPE option with an argument of MODEL_4735, MODEL_4845, or the MODEL_4860. To make a Model 4735 disk for the first time for example, use a command line like the following:

```
OK, MAKE -DISK 1060 -DISK_TYPE MODEL_4735 -FORMAT -NEW_DISK
```

This command line causes MAKE to read the flaw map because the argument to the –DISK_TYPE option identifies this disk type as having a flaw map. MAKE converts the information contained in the flaw map to badspot information, and writes the badspot information to the standard badspot file, MFD>BADSPT, or into the DBS file.

After you use any of these arguments (MODEL_4735, MODEL_4845, or MODEL_4860) to the –DISK_TYPE option and MAKE successfully reads and processes the flaw map, you see the following two messages related to flaw map processing.

```
OK, MAKE -DISK 1060 -DISK_TYPE MODEL_4735 -FORMAT
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Partition name? SYSCMD
        .
        .
        .

Processing flaw map on device 1060.
Processing of flaw map completed.
        .
        .
        .
```

If you use the –REPORT option on the command line, flaw map processing ends with this message:

nn `badspots added from flaw map on device` *pdev*.

nn is the number of badspots added from the vendor flaw map and *pdev* is the pdev of the partition on the spindle on which you are running MAKE.

If MAKE successfully processes the flaw map, it performs no further badspot checking. The system responds as if the –BADSPOT_LEVEL 0 (–BADLEV 0) option had been used on the MAKE command line. When a valid flaw map is found and read, the default to –BADLEV 0 considerably reduces the time necessary for MAKE to process a partition.

If MAKE cannot find a valid flaw map on the disk, cannot read the flaw map, or finds that the flaw map is corrupt, MAKE displays a message similar to one of the following and continues processing using the default level (4) of badspot checking for the disk type.

```
No flaw map found on device pdev. Continuing with MAKE.
Flaw map on device pdev is unreadable. Continuing with MAKE.
Processing flaw map on device pdev.
Flaw map is bad.  Continuing with MAKE.
```

You should use the MODEL_4735, the MODEL_4845, or the MODEL_4860 argument with the –DISK_TYPE option when you are partitioning any of these disk types. Using these arguments ensures that MAKE includes all known badspots from the vendor flaw map. If you specify the the wrong argument, MAKE may fail to read the flaw map perhaps causing you to lose badspot information and causing MAKE to do the badspot checking thus increasing the processing time. If MAKE is instructed to do the badspot checking on these disks or does the badspot checking by default, MAKE may miss marginally defective areas that are listed in the flaw map. See also the discussion under the –NO_FLAW_MAP option (page 5-35) and Table 5-4 to see what MAKE does when you use the –NO_FLAW_MAP or –BADLEV options on these disk types.

Note The Model 4735, the Model 4845, and the Model 4860 disks are the only disk types that support flaw map processing in this manner. Use the flaw maps for these disks.

Enter Known Badspots: If MAKE finds badspots as part of its verification procedure, it places them in either a BADSPT file, if you are creating a pre-Rev. 21.0 partition or you are creating a Rev. 21.0 or later partition in Nondynamic Badspot Handling (–DBS OFF) mode, or in the DBS file, if you are creating a Rev. 21.0 or later partition in Dynamic Badspot Handling (–DBS ON) mode. However, you may already be aware of badspots on the disk.

On some older disks, badspot locations are recorded in a printed list physically attached to the disk. Such a list is referred to as a **flaw map** (see Badspots in Chapter 2). Look for a list of pairs of numbers representing the track and the head, or surface, of those parts of the disk determined by the disk manufacturer to be probable badspots. Some other disks list track, head, and sector numbers of badspots. (Some disks – for example, the Model 4735 – have flaw maps recorded on the disk medium that are read by MAKE. See the option –NO_FLAW_MAP on page 5-35 and use of the flaw map on page 5-36.)

If you intend to enter known badspots, include the **–QUERY_BADSPOTS** option (abbreviations **–QRYBAD**, **–QBADS**, **–QRY**) on the command line. For example:

```
OK, MAKE -DISK 1060 -PART GEODE -DT 600MB -QUERY_BADSPOTS
```

When you invoke MAKE with the –QUERY_BADSPOTS option, you are asked to identify the locations of the badspots. A later section, Recording Badspots, shows you how.

Copy Badspot Information From Another Partition: If you have already created a partition in Nondynamic Badspot Handling (–DBS OFF) mode on the current physical disk, or spindle, and have entered all badspots known on the entire spindle, you may copy the badspot information from that partition to a new partition. (For a detailed example, see the section Keeping Redundant Badspot Files later in this chapter.)

To copy badspot information from a named and added partition on the current spindle, use the **–COPY_BADSPOTS** ***diskname*** option (abbreviations **–CPYBAD**, **–CPY**). The partition from which you are copying badspot information must have been started up with the ADDISK command. For example, to copy the badspots from partition SILVER to partition GOLDEN:

```
OK, MAKE -DISK 20063 -PART GOLDEN -COPY_BADSPOTS SILVER
```

To copy badspot information from an assigned partition (specified by a physical device number), use the **–COPY_BADSPOTS** option with ***pdev*** as the argument. The partition must have been assigned with the ASSIGN DISK command. For example:

```
OK, ASSIGN DISK 1062
OK, MAKE -DISK 20063 -PART GOLDEN -COPY_BADSPOTS 1062
```

Note You cannot use –CPY with the *diskname* argument when running MAKE standalone. You can only use the *pdev* argument and the pdev can include only disk drive units 0 through 3 and disk controller addresses 22_8, 23_8, 26_8, and 27_8.

If you do not include *pdev* or *diskname* with –CPYBAD, MAKE prompts you for one:

```
Specify pdev(s) or partition(s) from which badspots are to be copied.
You may specify up to 4 devices in all.  Enter "0" (zero) to finish.
Enter pdev or partition:
```

If you enter a partition name at this prompt, MAKE does not verify that the partition is added until later when it attempts to read the badspot file from the partition. If the partition is not added, MAKE displays this message and continues:

```
Partition "SILVER" not found.
Ignoring -COPY_BADSPOTS option.  Continuing with MAKE.
```

If you enter a *pdev* and the partition from which you want MAKE to copy the badspot file has not been assigned, MAKE informs you and pauses to allow you to assign the partition:

```
Enter pdev or partition: 1062
Device 1062 is not assigned.
MAKE paused to allow device assignment.  Type START to continue.
OK,
```

At this point, you can assign the device and then type START to continue MAKE.

You can enter up to four instances of either a *diskname* or a *pdev* or both with the –COPY_BADSPOTS option on a single command line.

List the Badspots on the Partition: You can list the badspots on a partition by using the –LIST_BADSPOTS option (abbreviations –LSTBAD, –LST, –LBS). All badspots from the BADSPT file, the DBS file, the vendor flaw map, and any you add with the –QUERY_BADSPOTS option are listed by Track, Head, and Sector and by their equivalent octal record number for those on the partition you are creating. MAKE lists the badspots at the completion of the

MAKE session and sorts them by head number so that all badspots for each partition are grouped together , as shown here:

```
OK, MAKE -DISK 100461 -BADLEV 0 -LIST_BADSPOTS -RPT
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Partition name? SYSABC
          .        .        .
          .        .        .
          .        .

List of badspots:

   Track        Head        Sector           Record
   -----        ----        ------         ----------
     432         14            5
     257         15            6
*    345         16            3              '22146
*    567         17            2              '35730
*    345         18            7              '22174

Note:  Badspots on device 100461 are marked above with a "*".

Badspots on device 100461:
         3 known badspots (from badspot file/s, flaw map, or terminal input)
         0 new badspots found during badspot checking or file system I/O
Total of 3 badspots mapped out of device 100461.

Partition SYSABC created successfully.
OK,
```

The badspots on the current partition (100461) are marked with an asterisk (*) and all badspots are sorted by head number. Badspots on other partitions on the same spindle are listed also but not by record number.

Save Records With Correctable Errors: When checking for badspots, MAKE writes data patterns in each record on the partition and reads those data patterns back. If the hardware detects that the medium is defective, MAKE gets a disk read or write error and marks the record as a badspot and it is not used in the file system. While doing badspot checking, the controller uses an error-correcting algorithm to attempt to correct the data pattern read. If you want to save records having correctable errors and want MAKE to mark only uncorrectable errors as badspots, include the -MAP_UNCORR option (abbreviation -UNCORR, -UNC). This will save marginally usable space on the partition.

Caution

The use of the –MAP_UNCORR option is not recommended unless you absolutely need the space and have no other choice, and unless you frequently back up your file system. The reason this option is not recommended is that, although the data pattern may be corrected at the time you use –MAP_UNCORR, it may not be corrected when PRIMOS reads and writes data to that record in the future. You may thus lose data.

Set the Method of File Record Allocation: Starting at Rev. 21.0, you can set either of two methods for PRIMOS to use for file record allocation: reverse sectoring with an interleave factor of 1 or forward sectoring with an interleave factor of 3. (See Chapter 10 for an explanation.) The default method of record allocation for Rev. 20.0, Rev. 21.0, and Rev. 22.0 partitions and Rev. 22.1 format standard (nonrobust) partitions depends on the combination of type of CPU and type of disk controller in your system. MAKE determines this combination and sets a bit in the DSKRAT to indicate the direction of record allocation and the interleave factor. It is recommended that you do not change the default method of record allocation.

Note

The method of file allocation on Rev. 20.0 partitions can be set and changed only if Rev. 21.0 or later utilities (MAKE and FIX_DISK) are used on them and only if they are run under Rev. 21.0 or later PRIMOS.

Table 10-2 in Chapter 10 presents the record allocation schemes based on the combination of the type of disk controller and the type of CPU in your system. Basically, the method is forward sectoring with an interleave factor of 3 for systems with a nonintelligent disk controller and a CPU in the 9950™ class. The method is reverse sectoring with an interleave factor of 1 for all other combinations of CPU and disk controller.

Note

CPUs in the 9950 class consist of the 4000, 5000, and 6000 series CPUs, the 2850™, the 2950™, the 9755™, and those with model numbers numerically equal to or larger than 9950, such as a 9955™. All other CPUs have model numbers numerically smaller than 9950.

You can change these recommended methods except in the case of SCSI disks connected to the Model 7210 disk controller; on these disks, the allocation direction is forward with an interleave factor of 1 as discussed earlier in this chapter under SCSI Disk Support.

On other disks, to set or change the allocation method, use the **–SECTOR** option (abbreviation –SEC). To set the allocation method to forward with an interleave factor of 3, use the **FORWARD** argument (abbreviation **FOR**).

The MAKE command format is

MAKE -SECTOR FORWARD

Use the **REVERSE** argument (abbreviation **REV**) to set the allocation direction to reverse with an interleave factor of 3. The MAKE command format is

MAKE -SECTOR REVERSE

The options and their associated methods of sectoring then are

Argument	*Method of Sectoring*	*Interleave Factor*	*Replaces*
FORWARD	Forward	3	-ODI
REVERSE	Reverse	1	-RDI

If you are running MAKE.SAVE standalone or if you use the -SECTOR option without including an argument, MAKE prompts you for the method of record allocation with the following query. If you are running MAKE.SAVE standalone, you should be prepared to answer this query by noting the type of CPU in your system as discussed above.

```
Which sectoring scheme would you like?
Enter "F" for Forward Sectoring, "R" for Reverse Sectoring:
```

Enter either F or R at the prompt. You can also enter Q for quit.

When you create robust partitions by using the MAKE_ROBUST utility, the method of file allocation is set by MAKE_ROBUST to forward with an interleave factor of 3. No reverse sectoring takes place on a robust partition.

Set the Maximum and Minimum Extent Sizes for CAM Files: Starting at Rev. 22.0, you can set the maximum and minimum extent sizes for all CAM files on a partition. An extent size is the number of contiguous 2048-byte records in one extent of the CAM file. (See the discussions of robust partitions and CAM files in Chapter 7 and the description of CAM files in Chapter 1.) You set the maximum and minimum extent sizes using the options -MAX_EXTENT_SIZE (abbreviations -MAXSIZ, -MAX) and -MIN_EXTENT_SIZE (abbreviations -MINSIZ, -MIN) in a command line like this:

MAKE -MAXSIZ *size* -MINSIZ *size*

size is a decimal number representing the maximum and minimum number of records per extent in a CAM file. *size* must be greater than zero and the minimum extent size must be less than or equal to the maximum. You can set the maximum extent size to any value up to 32767. You can set either or both of these values.

When you set the maximum and minimum extent sizes with the MAKE –MAXSIZ and –MINSIZ options, these values are written in the header information for the partition and remain for the life of the partition. However, you can later change these values with the FIX_DISK –MAXSIZ and –MINSIZ options.

If you do not specify the maximum and minimum extent sizes by using the –MAXSIZ and –MINSIZ options, MAKE uses the default maximum and minimum extent sizes, in number of records, for Rev. 22.0 and Rev. 22.1-format standard partitions as shown in Table 5-5.

If you convert a Rev. 22.1-format standard partition to a robust partition by using the MAKE_ROBUST utility, MAKE_ROBUST uses the defaults for robust partitions shown unless you use the MAKE_ROBUST –MAXSIZ and –MINSIZ options (see Chapter 7). If you intend to convert a Rev. 22.1-format standard partition to a robust partition by using the MAKE_ROBUST utility, as described in Chapter 7, do not use the MAKE –MAXSIZ and –MINSIZ options. To set the maximum and minimum extent sizes on a robust partition, you must use those options with MAKE_ROBUST. If you do not, MAKE_ROBUST sets the extent sizes to the default values.

Table 5-5. Default Maximum and Minimum Extent Sizes

	Robust Partition	*Standard Partition*
Maximum	256	32
Minimum	64	16

If you do use the –MAXSIZ and –MINSIZ options and you do not include the sizes, you are prompted for these values and they are checked for proper ranges of size. For example:

```
Minimum extent size? 70

Maximum extent size? 250
```

You enter the appropriate values at the prompts.

If the partition is a Rev. 20.0 or a Rev. 21.0 partition, the default minimum and maximum extent sizes for CAM files are set by PRIMOS to 16 records and 32 records, respectively.

If you attempt to use either the –MAXSIZ or –MINSIZ option but the partition you are creating is an earlier revision than Rev. 22.0, you see an appropriate error message and MAKE prompts for the disk revision, as follows.

```
Options -MINIMUM_EXTENT_SIZE and -MAXIMUM_EXTENT_SIZE are not available with
a Disk Revision earlier than Rev. 22.

Disk revision?
```

You should then reenter the disk revision with one that supports changing extent sizes or enter Q for quit.

Get Online Help With the MAKE Options: To see a list of the MAKE options, use the –USAGE option. To see a display of the MAKE options and a brief explanation of each option, use the –HELP option in this command line:

```
OK, MAKE -HELP
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

USAGE:    (1) MAKE
          (2) MAKE -DISK pdev -PARTITION name -DISK_TYPE type [ options ]
          (3) MAKE {-HELP | -USAGE}

OPTIONS:
```

A list of the valid disk types and a summary of the operation of each option are displayed in several screens. Note that you can use this command at any terminal. You can get more detailed help by using the PRIMOS HELP command with MAKE as the argument and you can get a list of the options with the –USAGE option.

Reporting Badspots

When MAKE finishes processing a partition, MAKE displays a badspot summary for the partition.

```
Badspots on device 2062:
          7 known badspots (from badspot file/s, flaw map, or terminal input)
          5 new badspots found during badspot checking or file system I/O
Total of 12 badspots mapped out of device 2062.
```

The first line enumerating badspots tells the total number of badspots that were collected from the various sources listed during the initial stages of MAKE processing in this order: from terminal input with the –QUERY_BADSPOTS option; from the vendor flaw map; from BADSPT and/or DBS files on the partition being made; and from devices specified with –COPY_BADSPOTS option.

The second line reports badspots found subsequently, either during badspot checking by MAKE, or as MAKE wrote to the disk while creating the file system. If badspot checking is not done, this line would usually report zero badspots.

The last line reports the total number of badspots affecting the partition being made. This may be fewer than the total number of badspots in the BADSPT or DBS file created by MAKE (and reported by –LIST_BADSPOTS), because the latter concern the whole spindle whereas the `Total` line concerns only the partition being made (only part of the spindle). The –LIST_BADSPOTS display makes this distinction clear by displaying head numbers, as shown in the example on page 5-46.

Unless you specify –REPORT, the entire badspot summary is reduced to one line if the total in the first line is zero:

```
Total of 4 badspots mapped out of device 660.
```

If you require more detailed information about which badspots reported in the first line came from which source, use the –REPORT option. With –REPORT, MAKE always prints the total number of new badspots added from any flaw map, BADSPT file, or DBS file. For example:

```
OK, MAKE -DISK 20462 -PAR TEST3 -DT MODEL 4845 -RPT -COPY BADSPOTS 10462
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]
          .        .        .
          .        .        .
          .        .        .
Processing flaw map on device 20462.
9 badspots added from flaw map on device 20462.

No BADSPT file found on device 20462.
Continuing with MAKE.

2 badspots added from BADSPT file on device 10462.

Badspots on device 20462:
         1 known badspot (from badspot file/s, flaw map, or terminal input)
         0 new badspots found during badspot checking or file system I/O
Total of 1 badspot mapped out of device 20462.

Partition TEST3 created successfully.
OK,
```

In the above example, the flaw map added nine badspots that MAKE had not known of previously. The BADSPT file on pdev 10462 added two more badspots. Since the flaw map was the first source processed, there are in fact nine entries in the flaw map. The line reporting badspots added from the BADSPT file on pdev 10462 counts only badspots not already known, so there could have been more than two entries in that file. The `Total` line reports the fact that only one of the known 11 badspots is on partition 20462. This becomes clear if you run the same MAKE with the -LIST_BADSPOTS option, as follows:

```
OK, MAKE -DISK 20462 -PAR TEST3 -DT MODEL_4845 -DBS OFF -CPY 10462 -RPT -LST
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Making 2 head partition:    TEST3            (Disk Revision 22.1)
Disk type:                  MODEL_4845
Number of tracks:           848
Sectors per track:          19
Sectoring mode:             REVERSE
Dynamic Badspot handling:   OFF  (-AC)
Partition size in records:  32224

Processing flaw map on device 20462.
9 badspots added from flaw map on device 20462.

No BADSPT file found on device 20462.
Continuing with MAKE.

2 badspots added from BADSPT file on device 10462.

List of badspots:

   Track        Head        Sector          Record
   -----        ----        ------        -----------
     416          0           18
     417          0           18
     848          0            9
      47          1            2
     112          3           17
     654          3            4
*    468          5           17             '42634
     141          6            2
     839         17            7
      91         19           12
     261         21           16

Note:  Badspots on device 20462 are marked above with a "*".

Badspots on device 20462:
         1 known badspot (from badspot file/s, flaw map, or terminal input)
         0 new badspots found during badspot checking or file system I/O
Total of 1 badspot mapped out of device 20462.

Partition TEST3 created successfully.
OK,
```

Recording Badspots

When you invoke MAKE with the –QUERY_BADSPOTS option, MAKE prompts you to enter the locations of badspots on the disk. You then select one of the following two formats for entering badspots.

Format of Badspot Information	*Number System*	*Offset From*
Track, head, sector	Decimal	Beginning of physical disk
Record number	Octal	Beginning of partition

In most cases you will probably prefer to use the track, head, and sector format. Here, head zero corresponds to head zero of the physical disk, not necessarily to the first head, or surface, of the partition being made. (See Figure 3-1 in Chapter 3.) This correspondence allows you to input known badspots anywhere on the physical disk, not just in the partition currently being created. Both formats are described below.

Note You do not enter badspots on disks connected to Model 2382 and Model 7210 disk controllers. Badspots are handled by the disk drive on these disks.

Entering Badspots by Record Number

If you choose to enter badspots by record number, answer YES when MAKE prompts

```
Use record number format?
```

MAKE then prompts:

```
Record numbers are OCTAL.  Enter a record number of 0 to finish.
Record =
```

In this case, you enter each record number in octal, as offset from the start of the partition. You cannot use record numbers to specify badspots outside the partition you are creating. MAKE continues prompting for record numbers with the `Record =` prompt until you enter a record number of zero (0) to indicate that you are done. MAKE then prompts `Verify data?` If you answer YES, MAKE displays a list of the corresponding badspot track, head, and sector numbers and the octal record numbers that you entered. See the example on page 5-86.

Entering Badspots by Track, Head, and Sector

Answer NO to the `Use record number format?` prompt. MAKE then repeatedly prompts for three values: Track, Head, and Sector. (If the -DISK_REVISION 18 option is included on the command line, MAKE does not prompt for Sector, and the entire track is considered a badspot.)

From DSM disk error messages, enter the CYL (cylinder) number when MAKE prompts for Track. If a hardcopy flaw map lists sector numbers as byte offsets, use Table 5-6 to convert the byte offset from the flaw map to the Sector number; for example, byte number 9987 is in Sector 4. If sector numbers are listed as decimal numbers, enter them as they are listed.

To terminate the dialog, enter 0 at the `Track =`, `Head =`, and `Sector =` prompts. MAKE then prompts

```
Verify data?
```

If you answer YES, MAKE displays a list of the badspot track, head, and sector numbers that you entered and their corresponding octal record numbers.

Table 5-6. Conversion of Byte Number to Sectors

Sector	*Byte Numbers*
0	0000 – 2239
1	2240 – 4479
2	4480 – 6719
3	6720 – 8959
4	8960 – 11199
5	11200 – 13439
6	13440 – 15679
7	15680 – 17919
8	17920 – 20159

The preceding conversion table is not applicable to the following listed FMDs. The Models 4735, 4845, and 4860 disks have flaw maps written on the disks' surfaces and badspots found by MAKE are written to the badspot file. You may enter any new badspots that are noted by PRIMOS or other utilities on these disks. Badspots on the remaining disks are handled by the disk subsystem.

Model 4711	Model 4729
Model 4714	Model 4730
Model 4715	Model 4735
Model 4719	Model 4845
Model 4721	Model 4860

After Known Badspots Have Been Entered and Verified

At this point, MAKE asks `List of badspots OK?` Check the list of badspot locations displayed by MAKE (assuming that you have requested the list by answering YES to the `Verify data?` prompt). If the list is correct, type YES. MAKE includes this information in the badspot file, which appears in the MFD with the filename BADSPT, or in the DBS file. MAKE also updates the DSKRAT file to ensure that badspot records are not available for file system use.

If the list is incorrect, a NO answer allows you to retain or delete those entered so far and then returns you to the `Use record number format?` prompt. Enter the badspots again or enter 0 to terminate the dialog.

Keeping Redundant Badspot Files

As a result of higher data storage densities, newer varieties of disks tend to have fewer badspots but some older varieties may be more susceptible to badspots. For example, a 675MB disk might have as many as 1200 badspots. To guard against loss of badspot information, MAKE allows you to maintain redundant badspot files for a physical disk with multiple partitions. MAKE supports redundant badspot files by allowing the badspot file on each partition to contain information about all the badspots on the disk, regardless of which partitions contain the badspots. For disk types that support Dynamic Badspot Handling, if the partitions on the spindle are made with –DBS ON, the badspots are in a DBS file, otherwise the badspots are in a BADSPT file. Use the following procedure in the latter case (that is, pre-Rev. 21.0 disks) or Rev. 21.0 and later disks in –DBS OFF mode.

Procedure for Copying Badspot Files: It is advisable to run MAKE on an entire physical disk, or spindle, before allowing users to access and store data on any partition of the disk. To do so, follow these steps.

1. Make a list of the physical device numbers for all partitions that you will create on the spindle.
2. Create the head zero partition, entering the track, head, and sector values of all badspots from the flaw map. (Typically, MAKE is run on a disk starting with the partition at surface offset zero (the head zero partition), but pre-Rev. 21.0 partitions can be made in any order. You must create the head zero partition first when making Rev. 21.0 and later partitions on spindles capable of Dynamic Badspot Handling.)

3. Create the second partition on your list. This time, you must include the –COPY_BADSPOTS option on the command line in order to copy the badspot information from the first partition.
4. Create the third partition, including –COPY_BADSPOTS in order to copy information from the second partition, and so on, always using the most recently created badspot file.

MAKE may find additional badspots as it creates the additional partitions. For example, if MAKE finds an additional badspot on the third partition of a five-partition spindle, the badspot files for partitions 3, 4, and 5 will then contain the entry for that new badspot.

The badspot files for partitions 1 and 2, however, will lack information on the new badspot. If MAKE subsequently finds a badspot on partition 5, only the badspot file on partition 5 will be complete.

Once all partitions on the disk have been made, but *before* any users have been allowed to access the disk, you can update the badspot files on the earlier partitions. To do this, remake the earlier partitions using the –COPY_BADSPOTS option. To minimize the time needed, use the –BADSPOT_LEVEL 0 option because badspots have already been found and recorded for the partitions you are remaking.

Example of Copying Badspot Files: To make a 675MB disk on drive unit 2 of the controller at address 26_8 as a four-partition disk with 10 surfaces per partition, construct a list of physical device numbers such as the following. (See Chapter 3 for details.)

pdev	*Starting Surface*	*Number of Surfaces*
2464	0	10
52464	10	10
122464	20	10
172464	30	10

Then assign these four partitions and use MAKE to create them using the following command lines. (Command lines using the option abbreviations are shown.)

```
MAKE -DSK 002464 -PART FALCON -NEWDSK -FMT -DT 600MB -QRY
MAKE -DSK 052464 -PART CORVID -NEWDSK -FMT -DT 600MB -CPYBAD 2464
MAKE -DSK 122464 -PART WILLET -NEWDSK -FMT -DT 600MB -CPYBAD 52464
MAKE -DSK 172464 -PART PLOVER -NEWDSK -FMT -DT 600MB -CPYBAD 122464
```

Now remake partition 2464, copying the badspot file from the last device, 172464. Use the –BADLEV 0 option in remaking these three partitions because MAKE has already found the badspots on these partitions. Using the –BADLEV option saves some MAKE processing time.

```
OK, MAKE -DSK 2464 -BADLEV 0 -CPYBAD 172464 -DT 600MB
Partition name? FALCON
```

Similarly, remake partition 52464 copying the badspot file from the last device, 172464.

```
OK, MAKE -DSK 52464 -BADLEV 0 -CPYBAD 172464 -DT 600MB
Partition name? CORVID
```

Finally, remake partition 122464, again copying the badspot file from the last device, 172464.

```
OK, MAKE -DSK 122464 -BADLEV 0 -CPYBAD 172464 -DT 600MB
Partition name? WILLET
```

What to Do Before Running MAKE

Take the following actions before you run MAKE:

1. Ensure that the new partition contains no important data that is not already backed up onto another disk or onto magnetic tape. When you create a partition, MAKE effectively deletes all existing data on that partition.
2. Determine the physical device number (pdev) of the logical disk, or partition, that is to be created. Chapter 3 explains how to determine this number. The physical device number tells the system the disk drive unit on which the disk is mounted, the address of the controller to which the drive unit is connected, and, for partitions, the size of the partition and its location on the spindle.
3. Use the following procedure, which is shown schematically in Figure 5-1:

   A. If you are creating a partition that replaces existing partitions on the disk, go to the supervisor terminal and shut down the partitions that are to be replaced. For example, assume two partitions are being replaced:

   ```
   OK, SHUTDN 460
   OK, SHUTDN 10460
   OK,
   ```

The above example shuts down the first two-surface partition with a surface offset of 0 and the second two-surface partition with a surface offset of 2, allowing a new four-surface partition, which has a pdev of 1060, to be created in their place.

B. Add the new, four-surface partition you are creating to the assignable disks table by using the DISKS command at the supervisor terminal:

```
OK, DISKS 1060
```

C. Assign the partition to be created to your terminal. Use the ASSIGN DISK command with the pdev:

```
OK, ASSIGN DISK 1060
```

The STATUS DEVICE command shows what partitions are in the assignable disks table and what partitions are assigned:

```
OK, STATUS DEVICE

Device User name                  Usrnum Ldevice
MT0    SYSTEM                       1      MT0
1060   SYSTEM                       2

Available assignable disks:

          121060
```

D. Now run MAKE with the appropriate options.

Running MAKE

You invoke MAKE with a number of options and arguments on the command line in order to specify the details of how the partition is to be created. MAKE *requires* that you specify three options and their arguments in order to obtain the following information or else MAKE prompts you for this information before proceeding:

- The physical device number of the partition to create (–DISK *pdev*)
- The name to give to the newly created partition (–PARTITION *diskname*)
- The type of spindle to partition (–DISK_TYPE *type*)

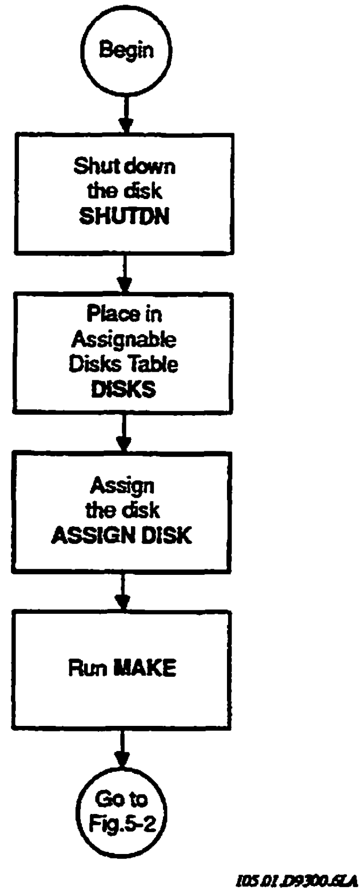


Figure 5-1. What to Do Before Running MAKE

Note MAKE cannot run under PRIMOS II (DOS). Attempting to use Rev. 23.3 MAKE under PRIMOS II results in the error message TOO BIG.

Invoking MAKE

After you determine the options you wish to specify, invoke MAKE as follows:

MAKE -DISK *pdev* -PARTITION *name* -DISK_TYPE *type* [*options*]

You can also invoke MAKE with no options. You will be prompted for the pdev, the partition name, and the disk type, in that order.

For example:

```
OK, MAKE
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Physical device? 660

Partition name? SYSCOM

Disk type? (press <return> for a list) MODEL_4735
```

In order to create a partition, you must specify the physical device number (pdev). You can determine the pdev from the information in Chapter 3.

If you do not include the –PARTITION option, MAKE prompts you for the name. *name* is the name that you assign to the partition and is limited to six alphanumeric characters.

If you do not include the –DISK_TYPE option, MAKE prompts you for one and displays a list of valid disk types if you press the Return key at the prompt. *type* is one of the valid types in Table 5-2 under item 3.

The physical disk, or spindle, that you specify with the –DISK_TYPE option is created as directed by the –DISK and –PARTITION options and by the other options that you specify on the command line. If you specify certain options more than once on the command line, the following message is displayed and MAKE aborts:

```
Option opt_name used more than once!
ER!
```

The following options, which take an argument, cannot be used more than once on the command line.

–DISK	–PARTITION
–DISK_TYPE	–SPLIT
–DISK_REVISION	–BAUD_RATE
–DBS	–SECTOR
–BADSPOT_LEVEL	–MAX_EXTENT_SIZE
–MIN_EXTENT_SIZE	

When MAKE Finishes

In the case of a disk type that does not handle badspots in the disk drive, MAKE signals its completion by displaying the following:

```
Done checking for badspots.

Badspots on device pdev:
         x known badspots (from badspot file/s, flaw map, or terminal input)
         y new badspots found during badspot checking or file system I/O
Total of z badspots mapped out of device pdev.

Partition diskname created successfully.
OK,
```

MAKE then returns you to the PRIMOS OK, prompt. *x* is the number of badspots from the sources listed, *y* is the number of new badspots found by MAKE, and *z* is the total number of badspots on this partition. If you are running MAKE on a SCSI disk, only the last message displays.

What to Do After Running MAKE

After running MAKE and creating all the partitions you need, you should add the partitions to the system so that they can be used. Then set any additional ACLs or the owner password for the partitions. Follow these steps, which are shown in Figure 5-2:

1. Unassign the partition with the UNASSIGN DISK command and remove it from the Assignable Disks Table with the DISKS NOT command:

   UNASSIGN DISK ***pdev***
   DISKS NOT ***pdev***

2. Add the partition to the file system with the ADDISK command

   ADDISK ***pdev***

   Use the same pdev in these commands as the one you constructed to create the partition.

3. Rev. 23.3 MAKE sets the ACLs SYSTEM:ALL, $REST:LUR on the MFD. If you use the –DSKREV 18 option, MAKE creates a password directory. If you wish to change the ACLs or the password, use either the SET_ACCESS command or the EDIT_ACCESS command to change ACLs, or use the PASSWD command to set an owner password other than XXXXXX. To do either, attach to the MFD of the newly created partition as follows:

   ATTACH *<name>***MFD [XXXXXX]**

where *name* is the name of the partition and the angle brackets (<>) are necessary in the command line. Use the password, XXXXXX, only if you created a password partition.

4. Issue either SET_ACCESS MFD or EDIT_ACCESS MFD to change ACLs or the PASSWD command to change the password. (If you use the PASSWD command, be sure to set the nonowner password to XXXXXX; otherwise AVAIL will not be able to size the partition.)

PRIMOS may be bootstrapped from this partition because MAKE writes the bootstrap program, BOOT, onto the MFD. If the partition is to be used only as a user partition, the empty directories CMDNC0 and DOS may be deleted from the newly created partition.

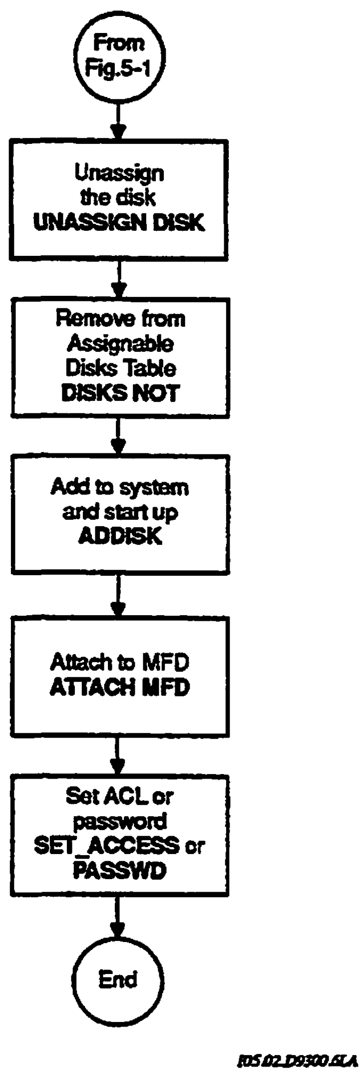


Figure 5-2. What to Do After Running MAKE

Running MAKE as a Phantom

When you run MAKE as a phantom, you use the –NO_QUERY option; see the discussion of –NO_QUERY on page 5-32.

Always provide all the information MAKE needs in the command line. It is poor practice to try to anticipate the order in which MAKE will prompt for missing options and to supply these in a COMI file or CPL &DATA block. The order of some prompts can be hard to predict reliably and may depart from what you expect in infrequently encountered circumstances.

The following exemplifies poor practice.

```
&DATA MAKE -DISK 40760 -DT SMD
          SYSXE3   /*-PART
          OFF      /*-DBS ON or OFF
&END
```

You should supply the information on the MAKE command line. For example:

```
OK, MAKE -DISK 40760 -PART SYSXE3 -DT SMD -DBS OFF -NQ
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Making 2 head partition:     SYSXE3     (Disk Revision 22.1)
Disk type:                   SMD
Number of tracks:            823
Sectors per track:           9
Sectoring mode:              FORWARD
Dynamic Badspot handling:    OFF  (-AC)
Partition size in records:   14814

Checking for badspots (level = 1).
          .         .         .
          .         .         .
          .         .         .
```

You will normally want to add the –NO_QUERY option to a MAKE command executed by a phantom. In the event that the command line does not contain all the information MAKE needs, –NO_QUERY ensures that MAKE displays a message indicating what further information you should supply. See the examples in the section Examples of Use of the –NO_QUERY Option on page 5-80.

Do not use the –NEW_DISK option with a phantom. It is never necessary and it may produce a prompt that will cause the phantom to abort regardless of whether –NO_QUERY was specified.

Disk types that support Dynamic Badspot Handling (listed on page 5-30) require the correct –DBS option to be specified and may also require –FORMAT. See the discussion on page 5-32 under the –NO_QUERY option.

When you use –NO_QUERY, MAKE construes –FORMAT (or –FORMAT_OK) as permission for MAKE to proceed in a way that risks loss of data on other partitions on the spindle. Use this combination with caution if there is any data of value on other partitions. It is recommended that you first run MAKE without –NO_QUERY to see any warnings.. There are two ways to do this:

- Run MAKE as a phantom without –NQ or –FMT. The phantom will abort if risk exists, otherwise it will proceed. Make sure your phantom creates a COMO file. If the phantom aborts, it will be clear from the COMO file whether risk exists and what options you should use to allow the phantom to proceed successfully.
- Run MAKE at a terminal without –NQ or –FMT and with –LEV 0. MAKE will ask permission to proceed if risk exists; you will have the opportunity to quit if the MAKE process will risk other partitions which you do not wish to lose. If there is no risk, MAKE will proceed but will finish quickly because –LEV 0 inhibits badspot checking. If badspot checking is wanted, rerun MAKE with the desired level of checking and as a phantom if you prefer to keep the terminal free for other uses.

If you do not supply the information needed with –NO_QUERY, MAKE will abort with a message telling you which options are needed, –FORMAT and –DBS ON or OFF.

Trial runs take very little time. If the phantom is going to abort, it will do so immediately. Following is an example of using a phantom in a trial run. The user creates M462.CPL to run MAKE as a phantom but forgets –NO_QUERY. The phantom immediately aborts because it needs terminal input to answer the query.

```
OK, SLIST M462.CPL
COMO M462.COMO
ASSIGN DISK 462
MAKE -DISK 462 -PAR TOMATO -DT 4845 -DBS OFF
COMO -END
OK, PH M462.CPL
PHANTOM is user 39
OK,
Phantom 39: Abnormal logout at 11:14
Time used: 00h 00m connect, 00m 00s CPU, 00m 00s I/O.
SLIST M462.COMO
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

An activated DBS file exists on partition 462.  If you proceed with
-DBS OFF (-AC), MAKE will enable -FORMAT to deactivate the DBS file.

WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.
```

```
Are you sure you want to make this partition with -DBS OFF?
ROBERT (user 39) logged out Thursday, 05 Mar 92 11:14:56.
Time used: 00h 00m connect, 00m 00s CPU, 00m 00s I/O.
OK,
```

The user corrects the CPL program by adding the –NO_QUERY option to the command line and tries again. The phantom aborts again because MAKE aborts when it determines that the –FORMAT option is necessary.

```
OK, SLIST M462.CPL
COMO M462.COMO
ASSIGN DISK 462
MAKE -DISK 462 -PAR TOMATO -DT 4845 -DBS OFF -NO_QUERY
COMO -END
OK, PH M462.CPL
PHANTOM is user 40
OK,
Phantom 40: Abnormal logout at 11:15
Time used: 00h 00m connect, 00m 00s CPU, 00m 00s I/O.
SLIST M462.COMO
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

An activated DBS file exists on partition 462.  To proceed with -DBS OFF
(-AC), you must specify -FORMAT to allow MAKE to deactivate the DBS file.

WARNING:  By disturbing the DBS file you will risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

Insufficient options with -NO_QUERY.  -FORMAT option needed.
MAKE aborted.
OK,
ROBERT (user 40) logged out Thursday, 05 Mar 92 11:15:36.
Time used: 00h 00m connect, 00m 00s CPU, 00m 00s I/O.
```

Finally the user includes the necessary –FORMAT option and MAKE successfully processes the partition. The user could have used the –FORMAT_OK option with the same results.

```
OK, SLIST M462.CPL
COMO M462.COMO
ASSIGN DISK 462
MAKE -DISK 462 -PAR TOMATO -DT 4845 -DBS OF -NO_QUERY -FORMAT
COMO -END
OK, PH M462.CPL
PHANTOM is user 41
OK,
Phantom 41: Normal logout at 11:18
Time used: 00h 00m connect, 00m 02s CPU, 00m 48s I/O.
```

```
SLIST M462.COMO
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

An activated DBS file exists on partition 462.  Because you have specified
-DBS OFF (-AC), MAKE will enable -FORMAT to deactivate the DBS file.

WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

Making 2 head partition:     TOMATO              (Disk Revision 22.1)
Disk type:                   MODEL_4845
Number of tracks:            848
Sectors per track:           19
Sectoring mode:              REVERSE
Dynamic Badspot handling:    OFF  (-AC)
Partition size in records:   32224

Processing flaw map on device 462.
Processing of flaw map completed.

Beginning format.
Format completed.

Badspots on device 462:
         4 known badspots (from badspot file/s, flaw map, or terminal input)
         0 new badspots found during badspot checking or file system I/O
Total of 4 badspots mapped out of device 462.

Partition TOMATO created successfully.
OK,
```

Alternatively, the user could have first run MAKE at a user terminal with the –BADLEV 0 option to see that the –FORMAT option is necessary, quit when MAKE prompts with a question, fixed the CPL to reflect the necessary conditions, and then run MAKE as a phantom.

Examples of Running MAKE

The following examples show various uses of MAKE when PRIMOS is running. Examples of booting MAKE.SAVE from disk and from magnetic tape are presented at the end of this chapter under Booting MAKE.SAVE.

Example of Normal MAKE Display

In the first example, MAKE creates a new head zero partition that has six surfaces, with starting surface 0, and is on drive unit 1. The Operator used the –DBS option but this SCSI disk type does not support Dynamic Badspot Handling. MAKE ignores the –DBS option and creates the partition displaying the disk type, the partition name, partition size-related information, and sectoring mode. Because this disk type does not support Dynamic Badspot Handling, MAKE displays no DBS mode information.

```
OK, DISKS 1062
OK, ASSIGN DISK 1062
OK, MAKE -DISK 1662 -PART GROUP3 -DISK_TYPE MODEL_4732 -DBS ON
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Option -DBS ON (-IC) is irrelevant for this disk type and will be ignored.

Making 6 head partition:     GROUP3            (Disk Revision 22.1)
Disk type:                   MODEL_4732
Number of tracks:            81
Sectors per track:           254
Sectoring mode:              FORWARD
Partition size in records:   124968

Total of 0 badspots mapped out of device 1662.
Partition GROUP3 created successfully.
OK, UNASSIGN DISK 1062
OK, DISKS NOT 1062
OK,
```

After MAKE has formatted the partition and checked for badspots, the partition is unassigned and removed from the Assignable Disks Table.

Specifying a Different Disk Type

MAKE can determine the disk type on a partition that has a valid DSKRAT by reading the field in the DSKRAT containing this information. If MAKE determines that you specify a disk type that is different from the information contained in the DSKRAT, MAKE warns you of the discrepancy and prompts to see if you want to continue.

You should not specify the wrong disk type because that can cause MAKE to encounter errors when attempting to create the partition and will probably cause severe operational problems under PRIMOS, in addition to wasting your time.

Specifying a different disk type can easily happen when creating partitions on SCSI disks because they have similar model numbers that specify the disk type.

Example of Remaking With a Different Disk Type: In this example, MAKE checks the DSKRAT and finds it to be valid and then checks the field in the DSKRAT where the disk type is stored and finds that it is different from the type specified by the user. MAKE prompts the user to continue. This could be a mistake on the user's part, particularly with the SCSI disks that have similar model numbers.

```
OK, MAKE -DISK 103461 -PAR BBOARD -DT MODEL_4729 -REV 20
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

WARNING:  Partition 103461 appears to have been previously made with type
          MODEL_4721, but you have specified type MODEL_4729.

OK to continue with MAKE? NO
```

In the second part of this example, the user runs MAKE on a non-head zero partition and MAKE checks the DSKRAT on the head zero partition and determines that the disk type specified is different from the field in the DSKRAT. Different disk types should not be specified on the same spindle and, of course, cannot exist on the same spindle. MAKE also notes the difference in Dynamic Badspot Handling mode and, because the mode is set for the entire spindle on the head zero partition, MAKE informs the user that this partition will be made the same as the head zero partition. MAKE prompts the user to continue.

```
OK, MAKE -DISK 10462 -PART BBOARD -DT MODEL_4860 -DBS OFF
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

WARNING:  Head zero partition (462) has been made with type MODEL_4845,
          but you have specified type MODEL_4860 for this partition (10462).
          Different types should never occur on the same spindle.

Head zero partition (device 462) has been made with -DBS ON (-IC);
Dynamic BadSpotting is enabled on this spindle.

To make this partition (device 10462) with -DBS OFF (-AC) you must first
remake the head zero partition with -DBS OFF.

This partition will be made with -DBS ON.
OK to continue with MAKE? QUIT
MAKE aborted.
OK,
```

Examples of Running MAKE on Disks That Support Dynamic Badspot Handling

The following examples are all concerned with running MAKE on disks that support Dynamic Badspot Handling when connected to a Model 6580 intelligent disk controller (IDC1) downline loaded with ICOP software. These disk types are the following with their capacities listed in parentheses.

SMD (300MB and 80MB)	
68MB	158MB
160MB	600MB
MODEL_4475 (315MB)	MODEL_4735 (496MB)
MODEL_4845 (770MB)	MODEL_4860 (817MB)

The first examples are of running MAKE on the head zero, or first, partition of Dynamic Badspot Handling spindles.

Normal Remaking of a Head Zero Partition: This disk had previously been created with –DBS ON so no problems exist and no messages are output. MAKE successfully creates the partition, displaying the disk type, the partition name, partition size-related information, sectoring mode, and Dynamic Badspot Handling mode. MAKE summarizes the badspot information at the end of the session.

```
OK, MAKE -DISK 462 -PART TEST -DT MODEL_4860 -DBS ON
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Making 2 head partition:     TEST              (Disk Revision 22.1)
Disk type:                   MODEL_4860
Number of tracks:            1379
Sectors per track:           19
Sectoring mode:              REVERSE
Dynamic Badspot handling:    ON  (-IC)
Partition size in records:   52402

Processing flaw map on device 462.
Processing of flaw map completed.

Badspots on device 462:
         128 known badspots (from badspot file/s, flaw map, or terminal input)
           0 new badspots found during badspot checking or file system I/O
Total of 128 badspots mapped out of device 462.

Partition TEST created successfully.
OK,
```

Creating a Head Zero Partition by Choosing DBS Mode: No –DBS option was specified so MAKE informs the user that Dynamic Badspot Handling may be enabled or disabled on this spindle (all partitions on the spindle) and describes the effects of enabling or disabling Dynamic Badspot Handling. The user specifies ON at the DBS prompt and MAKE creates the partition with Dynamic Badspot Handling enabled. Note that the user could quit by entering Q at the prompt. All prompts allow quitting out of MAKE.

```
OK, MAKE -DISK 462 -PART SYSONE -DT 4860
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

You can make this partition with Dynamic BadSpotting (-DBS) either ON or OFF.

A partition made with -DBS ON (-IC) ...
     - CAN be used on an IDC1 controller;  Dynamic BadSpotting WILL be done
     - CAN be mirrored, and CAN be used for CDD if split
     - Cannot be used on 4004 or 4005 controllers, which do not support
       Dynamic BadSpotting

A partition made with -DBS OFF (-AC) ...
     - Can be used on an IDC1 controller, but Dynamic BadSpotting is NOT done
     - Can be used on 4004 or 4005 controllers, which NEVER do
       Dynamic BadSpotting
     - CANNOT be mirrored, and CANNOT be used for CDD
Please specify "ON" or "OFF" ("Q" to quit): ON

Making 2 head partition:     SYSONE            (Disk Revision 22.1)
Disk type:                   MODEL_4860
Number of tracks:            1379
Sectors per track:           19
Sectoring mode:              REVERSE
Dynamic Badspot handling:    ON  (-IC)
Partition size in records:   52402

Processing flaw map on device 462.
Processing of flaw map completed.

Badspots on device 462:
         128 known badspots (from badspot file/s, flaw map, or terminal input)
           0 new badspots found during badspot checking or file system I/O
Total of 128 badspots mapped out of device 462.

Partition SYSONE created successfully.
OK,
```

Remaking a Head Zero Partition by Disturbing the DBS File: The partition was previously created with the –DBS ON option and has a valid DBS file on it. The user specifies the –FORMAT option which would cause the DBS file to be rebuilt. Since the DBS file contains badspot information for the entire spindle (all partitions on the spindle), remaking it with the –FORMAT option may cause data that was remapped from other partitions on the spindle to the RMA to be lost. In addition, if –FORMAT is used on the head zero partition, all other partitions on the spindle must be remade using the –FORMAT option because –FORMAT removes any old remappings in the record headers. The user should not continue with this session but should reenter the command line without the –FORMAT option unless the user backed up all partitions on the disk and is systematically making the entire spindle.

```
OK, MAKE -DISK 462 -PART GEORGE -DTP MODEL 4860 -DBS ON -FORMAT
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

A valid DBS file exists on partition 462.  You have specified -FORMAT,
which will cause the DBS file to be rebuilt.

WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

OK to continue with MAKE? NO
MAKE aborted.
OK,
```

Remaking a Head Zero Partition by Creating a New DBS File: In this example, the user runs MAKE on the head zero partition, which was previously created with –DBS ON, and specifies the –NEW_DISK option. Because a valid DBS file already exists on the partition, MAKE queries the user to be sure that the DBS file should be ignored and tells the user that MAKE will use –FORMAT to remove the existing DBS file. MAKE also warns the user that all other partitions on the spindle must be remade using the –FORMAT option.

If the user includes the –FORMAT option on the command line, MAKE does not display the message about enabling –FORMAT.

The user also included the –REPORT option causing the progress of the formatting operation to be reported in 5% intervals. Because this disk type has a vendor flaw map, MAKE displays messages about processing the flaw map and, because of the –REPORT option, displays the number of badspots entered from the flaw map and, at the end of the session, MAKE summarizes the badspot information. Because MAKE processed a vendor flaw map, MAKE did no other badspot checking.

```
OK, MAKE -DISK 462 -PART SYS 24 -DT MODEL 4860 -DBS ON -NEW_DISK -RPT
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

A valid DBS file exists on partition 462 containing 128 badspots.
-NEWDSK implies that you wish to ignore this DBS file and create a new one.

If you ignore this DBS file, MAKE will enable -FORMAT to remove it fully
before creating the new one.

WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.
```

```
Really ignore DBS file? [Y,N,Q] YES

Making 2 head partition:     SYS_24           (Disk Revision 22.1)
Disk type:                   MODEL_4860
Number of tracks:            1379
Sectors per track:           19
Sectoring mode:              REVERSE
Dynamic Badspot handling:    ON  (-IC)
Partition size in records:   52402

Processing flaw map on device 462.
128 badspots added from flaw map on device 462.

Beginning format.
Format now  5% complete.
     .    .    .
     .    .    .
     .    .    .
Format now 90% complete.
Format now 95% complete.
Format completed.

Badspots on device 462:
         128 known badspots (from badspot file/s, flaw map, or terminal input)
           0 new badspots found during badspot checking or file system I/O
Total of 128 badspots mapped out of device 462.

Partition SYS_24 created successfully.
OK,
```

Remaking a Head Zero Partition as a Split Partition: In this example, the user specifies the -SPLIT option without specifying the number of records to use for paging or crash dump space. MAKE displays the maximum number of records available for these purposes and prompts for a number. The user specifies a valid number.

Because a valid DBS file exists on the partition and the user did not specify a DBS option, MAKE prompts for one after outlining the consequences of specifying either ON or OFF. When the user specifies OFF, MAKE warns the user about the existence of the DBS file and the consequences to all partitions on the spindle of specifying -FORMAT to deactivate the DBS file. MAKE then gives the user the opportunity to reconsider and preserve the DBS file. The user then specifies ON.

MAKE creates the partition displaying the number of file system and paging records. Because this disk type has a vendor flaw map, MAKE displays messages about processing the flaw map.

```
OK, MAKE -DISK 462 -PAR ENGRNG -DT MODEL_4860 -SPLIT
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Please specify how you want to -SPLIT this partition.
Maximum number of paging (or CDD) records available is 51389.
Number of paging/CDD records? 12000

You can make this partition with Dynamic BadSpotting (-DBS) either ON or OFF.

A partition made with -DBS ON (-IC) ...
     - CAN be used on an IDC1 controller;  Dynamic BadSpotting WILL be done
     - CAN be mirrored, and CAN be used for CDD if split
     - Cannot be used on 4004 or 4005 controllers, which do not support
       Dynamic BadSpotting

A partition made with -DBS OFF (-AC) ...
     - Can be used on an IDC1 controller, but Dynamic BadSpotting is NOT done
     - Can be used on 4004 or 4005 controllers, which NEVER do
       Dynamic BadSpotting
     - CANNOT be mirrored, and CANNOT be used for CDD

Please specify "ON" or "OFF" ("Q" to quit): OFF

An activated DBS file exists on partition 462.  If you proceed with
-DBS OFF (-AC), MAKE will enable -FORMAT to deactivate the DBS file.

WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

Are you sure you want to make this partition with -DBS OFF? NO

You can make this partition with Dynamic BadSpotting (-DBS) either ON or OFF.

A partition made with -DBS ON (-IC) ...
     - CAN be used on an IDC1 controller;  Dynamic BadSpotting WILL be done
     - CAN be mirrored, and CAN be used for CDD if split
     - Cannot be used on 4004 or 4005 controllers, which do not support
       Dynamic BadSpotting

A partition made with -DBS OFF (-AC) ...
     - Can be used on an IDC1 controller, but Dynamic BadSpotting is NOT done
     - Can be used on 4004 or 4005 controllers, which NEVER do
       Dynamic BadSpotting
     - CANNOT be mirrored, and CANNOT be used for CDD

Please specify "ON" or "OFF" ("Q" to quit): ON

Making 2 head partition:     ENGRNG          (Disk Revision 22.1)
Disk type:                   MODEL_4860
Number of tracks:            1379
```

```
Sectors per track:          19
Sectoring mode:             REVERSE
Dynamic Badspot handling:   ON  (-IC)

File system records:        40402
Paging records:             12000

Processing flaw map on device 462.
Processing of flaw map completed.

Badspots on device 462:
         128 known badspots (from badspot file/s, flaw map, or terminal input)
           0 new badspots found during badspot checking or file system I/O
Total of 128 badspots mapped out of device 462.

Partition ENGRNG created successfully.
OK,
```

Remaking a Head Zero Partition as a Pre-Rev. 21 Partition: In this example, the user attempts to remake a head zero partition, which has an existing DBS file, as a Rev. 20 partition. Because Dynamic Badspot Handling was not available at Rev. 20, MAKE must convert the badspot information to a BADSPT file and remove the DBS file. To do this, MAKE will enable the –FORMAT option and warns the user that all other partitions on this spindle must be remade using the –FORMAT option. The user decides against this and MAKE aborts.

```
OK, MAKE -DSK 462 -PART OPRTNS -DT MODEL 4845 -REV 20
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

A valid DBS file exists on partition 462.  Because this partition is being
made with Disk Revision 20, MAKE will enable -FORMAT to remove the DBS file.

WARNING:  By removing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

OK to continue with MAKE? NO
MAKE aborted.
OK,
```

Remaking a Head Zero Partition with a Size Change: In this two- part example, the user remakes a disk that either was not split previously or was split with a different number of paging/crash dump records. This causes MAKE to display the message about the different size which will cause the DSKRAT to be of a different size and thus cause MAKE to move the DBS file and RMA. MAKE must then enable –FORMAT to create a new DBS file.

Initially, the user enters an invalid number of paging records (the number is larger than the number of available records) and MAKE reprompts. The user decides to go ahead and let MAKE format the partition and create a new DBS file. The user should then remake all other partitions on this spindle using the -FORMAT option.

```
OK, MAKE -DISK 462 -PART OPRTNS -DT MODEL_4860 -SPLIT
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Please specify how you want to -SPLIT this partition.
Maximum number of paging (or CDD) records available is 51389.
Number of paging/CDD records? 51400
Invalid number of paging/CDD records (51400).
Maximum number of paging (or CDD) records available is 51389.
Number of paging/CDD records? 12000

Partition 462 appears to have been previously made with a different size.
The MAKE you have specified will destroy the existing DBS file.

MAKE will enable -FORMAT to create a new DBS file.

WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

OK to continue with MAKE? Y

You can make this partition with Dynamic BadSpotting (-DBS) either ON or OFF.

A partition made with -DBS ON (-IC) ...
     - CAN be used on an IDC1 controller;  Dynamic BadSpotting WILL be done
     - CAN be mirrored, and CAN be used for CDD if split
     - Cannot be used on 4004 or 4005 controllers, which do not support
       Dynamic BadSpotting

A partition made with -DBS OFF (-AC) ...
     - Can be used on an IDC1 controller, but Dynamic BadSpotting is NOT done
     - Can be used on 4004 or 4005 controllers, which NEVER do
       Dynamic BadSpotting
     - CANNOT be mirrored, and CANNOT be used for CDD

Please specify "ON" or "OFF" ("Q" to quit): ON

Making 2 head partition:     OPRTNS              (Disk Revision 22.1)
Disk type:                   MODEL_4860
Number of tracks:            1379
Sectors per track:           19
Sectoring mode:              REVERSE
Dynamic Badspot handling:    ON   (-IC)

File system records:         40402
Paging records:              12000
```

```
Processing flaw map on device 462.
Processing of flaw map completed.

Beginning format.
Format completed.

Badspots on device 462:
        128 known badspots (from badspot file/s, flaw map, or terminal input)
          0 new badspots found during badspot checking or file system I/O
Total of 128 badspots mapped out of device 462.

Partition OPRTNS created successfully.
OK,
```

In the second part of this example, the user splits a disk and uses the MAX argument to the –SPLIT option when MAKE prompts for the number of paging/crash dump records resulting in a change of size of the DSKRAT. Note that the user specifies the –AC option which is obsolete but supported and which is the same as –DBS OFF. The user decides against continuing and MAKE aborts.

```
OK, MAKE -DISK 462 -PAR BIO101 -DT MODEL_4860 -SPL -AC
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Please specify how you want to -SPLIT this partition.
Maximum number of paging (or CDD) records available is 51389.
Number of paging/CDD records? MAX

Partition 462 appears to have been previously made with a different size.
The MAKE you have specified will destroy the existing DBS file.

MAKE will enable -FORMAT to create a new DBS file.

WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

OK to continue with MAKE? NO
MAKE aborted.
OK,
```

Remaking a Head Zero Partition Which Has a Corrupted DSKRAT: In this three part example, the user encounters a message about an invalid DSKRAT. This may be because of running MAKE on a spindle that has never been used on a Prime system, because of a corrupted DBS file, or because the user may have broken out of MAKE at an earlier time corrupting the DSKRAT.

If the user had specified both the –FORMAT and –NEW_DISK options, MAKE would not display the error and warning messages and would have continued.

The DBS file is not readable because of the corrupted DSKRAT, so MAKE queries the user after displaying the messages and the user decides to quit, aborting MAKE.

Because the DSKRAT is corrupted, MAKE cannot tell if Dynamic Badspot Handling was previously enabled on this spindle. If it was, you should –FORMAT all partitions on the spindle, as recommended, in order to remove old badspot remappings. If Dynamic Badspot Handling was not enabled (and you are *sure* of this), you need not format the other partitions. MAKE assumes that there is risk to other partitions on the spindle whenever the DBS file is not readable and the partition MAKE is creating is not the whole spindle.

```
OK, MAKE -DISK 462 -PART ACCTNG -DT MODEL_4845  -REV 22 -FMT
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Invalid DSKRAT on partition 462.
Cannot determine if there is an active Dynamic BadSpot file on this spindle.

If this partition has previously been made with -DBS ON (-IC), the -FORMAT
operation you have specified will cause the DBS file to be rebuilt.

WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

OK to continue with MAKE? QUIT
MAKE aborted.
OK,
```

In the second part of this example, the user omits the –FORMAT option. Because MAKE cannot read the DBS file, MAKE recommends the use of the –FORMAT option to rebuild the DBS file, if the partition was previously made with –DBS ON. MAKE queries the user as to use of –FORMAT and continues when the user responds YES.

```
OK, MAKE -DISK 462 -PART ACCNTG -DT MODEL_4845  -REV 22
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Invalid DSKRAT on partition 462.
Cannot determine if there is an active Dynamic BadSpot file on this spindle.

If this partition was previously made with -DBS ON (-IC), you are strongly
recommended to make it with -FORMAT in order to rebuild the DBS file.


WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  If this partition is formatted,
          these other partitions should be remade with -FORMAT.
```

```
OK to enable -FORMAT? [Y,N,Q] Y

You can make this partition with Dynamic BadSpotting (-DBS) either ON or OFF.

A partition made with -DBS ON (-IC) ...
     - CAN be used on an IDC1 controller;  Dynamic BadSpotting WILL be done
     - CAN be mirrored, and CAN be used for CDD if split
     - Cannot be used on 4004 or 4005 controllers, which do not support
       Dynamic BadSpotting

A partition made with -DBS OFF (-AC) ...
     - Can be used on an IDC1 controller, but Dynamic BadSpotting is NOT done
     - Can be used on 4004 or 4005 controllers, which NEVER do
       Dynamic BadSpotting
     - CANNOT be mirrored, and CANNOT be used for CDD

Please specify "ON" or "OFF" ("Q" to quit): ON
Making 2 head partition:     ACCTNG              (Disk Revision 22)
Disk type:                   MODEL_4845
Number of tracks:            848
Sectors per track:           19
Sectoring mode:              REVERSE
Dynamic Badspot handling:    ON  (-IC)
Partition size in records:   32224

Processing flaw map on device 462.
Processing of flaw map completed.

Beginning format.
Format completed.

Badspots on device 462:
         128 known badspots (from badspot file/s, flaw map, or terminal input)
           0 new badspots found during badspot checking or file system I/O
Total of 128 badspots mapped out of device 462.

Partition ACCNTG created successfully.
OK,
```

The third part of this example shows the effect of using the –NO_QUERY option and the –REV 20 option. MAKE displays the warnings, noting that the user should remake the partition with –FORMAT to remove any existing DBS file because Dynamic Badspot Handling does not exist at Rev. 20. MAKE also notes that the user will risk data loss on any other partitions on the spindle because of removal of the DBS file and should also remake the other partitions with the –FORMAT option.

```
OK, MAKE -DISK 462 -PART LAB.EX -DT MODEL_4845 -REV 20 -NO_QUERY
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]
```

```
Invalid DSKRAT on partition 462.
Cannot determine if there is an active Dynamic BadSpot file on this spindle.

If this partition was previously made with -DBS ON (-IC), you are strongly
recommended to remake it with -FORMAT in order to remove the DBS file.

WARNING:  By removing the DBS file you will risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

Making 2 head partition:     LAB.EX           (Disk Revision 20)
Disk type:                   MODEL_4845
Number of tracks:            848
Sectors per track:           19
Sectoring mode:              REVERSE
Partition size in records:   32224
Processing flaw map on device 462.
Processing of flaw map completed.

Badspots on device 462:
         4 known badspots (from badspot file/s, flaw map, or terminal input)
         0 new badspots found during badspot checking or file system I/O
Total of 4 badspots mapped out of device 462.

Partition LAB.EX created successfully.
OK,
```

Examples of Running MAKE on Non-head Zero Partitions

The following examples are of running MAKE on the non-head zero partition of spindles that support Dynamic Badspot Handling. These examples emphasize these points:

- All partitions on a spindle must be created with the same Dynamic Badspot Handling mode.
- The head zero partition must be created first.
- The head zero partition should be assigned when you create other partitions on the spindle.

When you run MAKE on disks that support Dynamic Badspot Handling at Rev. 21 and later and you use the –DBS option on the head zero partition, you determine whether Dynamic Badspot Handling should be enabled (ON or –IC) or disabled (OFF or –AC) on all partitions on the spindle.

When you use the –DBS option on a non-head zero partition, MAKE interprets that as specifying how the head zero partition was created, either with –DBS ON or –DBS OFF. If the head zero partition is assigned and valid, specifying the

–DBS option is unnecessary as MAKE always creates the non-head zero partition in the same mode as the head zero partition. This is the only case in which there is a defined default –DBS mode. If the head zero partition is inaccessible or invalid, MAKE interprets the –DBS option as specifying to MAKE the mode of the head zero partition.

Note If MAKE is run standalone and the head zero partition was created with –DBS ON, MAKE cannot create the non-head zero partition. There is one exception to this rule, when the non-head zero partition is split, illustrated by the example on page 5-96.

Normal Remaking of a Non-head Zero Partition: In this example, the user did not specify a –DBS option; MAKE determines that the head zero partition was created with –DBS ON enabling Dynamic Badspot Handling on the entire spindle and thus creates this non-head zero partition with –DBS ON.

```
OK, MAKE -DISK 10462 -PAR GROUP1 -DT MODEL_4860
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition (device 462) has been made with -DBS ON (-IC);
Dynamic BadSpotting is enabled on this spindle.

This partition will be made with -DBS ON.

Making 2 head partition:    GROUP1          (Disk Revision 22.1)
Disk type:                  MODEL_4860
Number of tracks:           1379
Sectors per track:          19
Sectoring mode:             REVERSE
Dynamic Badspot handling:   ON  (-IC)
Partition size in records:  52402

Processing flaw map on device 10462.
Processing of flaw map completed.

Badspots on device 10462:
         2 known badspots (from badspot file/s, flaw map, or terminal input)
         0 new badspots found during badspot checking or file system I/O
Total of 2 badspots mapped out of device 10462.

Partition GROUP1 created successfully.
OK,
```

Remaking a Non-head Zero Partition With Conflicting DBS Modes: In this example, the user specifies a –DBS argument that conflicts with the setting of the head zero partition, which was –DBS ON enabling Dynamic Badspot Handling on the entire spindle. If the user allows MAKE to continue, MAKE will create this non-head zero partition with –DBS ON. MAKE aborts when the user quits.

```
OK, MAKE -DISK 10462 -PART GROUP2 -DT 4845 -DBS OFF
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition (device 462) has been made with -DBS ON (-IC);
Dynamic BadSpotting is enabled on this spindle.

To make this partition (device 10462) with -DBS OFF (-AC) you must first
remake the head zero partition with -DBS OFF.

This partition will be made with -DBS ON.
OK to continue with MAKE? QUIT
MAKE aborted.
OK,
```

If the user specifies the –NO_QUERY option in the above case, MAKE simply aborts due to the conflicting –DBS specifications.

```
OK, MAKE -DISK 10462 -PART GROUP2 -DT MODEL_4845 -DBS OFF -NO_QUERY
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition (device 462) has been made with -DBS ON (-IC);
Dynamic BadSpotting is enabled on this spindle.

To make this partition (device 10462) with -DBS OFF (-AC) you must first
remake the head zero partition with -DBS OFF.

MAKE aborted because of invalid option specified with -NO_QUERY:
This partition cannot be made with -DBS OFF.
ER!
```

Remaking a Non-head Zero Partition With a Pre-Rev. 21 Head Zero Partition: In this example, the user specifies –DBS ON but the head zero partition is a pre-Rev. 21 partition and thus cannot have Dynamic Badspot Handling either enabled or disabled because Dynamic Badspot Handling does not exist for a pre-Rev. 21.0 partition. Thus, Dynamic Badspot Handling cannot be enabled on this spindle. MAKE will create this partition with –DBS OFF if allowed to continue. The user should remake the head zero partition as a Rev. 22.1 format partition and then remake the non-head zero partition.

```
OK, MAKE -DISK 10462 -PAR OLDER -DT 4860 -DBS ON
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition (device 462) has been made with Disk Revision 20;
Dynamic BadSpotting is NOT enabled on this spindle.

To make this partition (device 10462) with -DBS ON (-IC) you must first
remake the head zero partition with -DBS ON.

This partition will be made with -DBS OFF.
OK to continue with MAKE?
```

Remaking a Non-head Zero Partition With Conflicting Revisions: In this example, the user remakes the non-head zero partition with a disk revision of 20 on a spindle that has the head zero partition in Dynamic Badspot Handling mode thus enabling Dynamic Badspot Handling on the entire spindle. MAKE allows the user to do this but it is not recommended; all partitions on a spindle should be in the same Dynamic Badspot Handling mode and of the same revision.

```
OK, MAKE -DISK 10462 -PAR BADREV -DT MODEL_4860 -REV 20
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

NOTE:  This partition (10462) is being made as Disk Revision 20, but the
       head zero partition (462) is Disk Revision 22.1, made with -DBS ON
       (Dynamic BadSpotting is enabled on this spindle).

Making 2 head partition:    BADREV           (Disk Revision 20)
Disk type:                  MODEL_4860
Number of tracks:           1379
Sectors per track:          19
Sectoring mode:             REVERSE
Partition size in records:  52402

Processing flaw map on device 10462.
Processing of flaw map completed.

Badspots on device 10462:
        2 known badspots (from badspot file/s, flaw map, or terminal input)
        0 new badspots found during badspot checking or file system I/O
Total of 2 badspots mapped out of device 10462.

Partition BADREV created successfully.
OK,
```

Remaking a Non-head Zero Partition With a Corrupt Head Zero Partition: In this example, MAKE finds that the head zero partition is corrupt and cannot, therefore, determine if there is an active DBS file. MAKE recommends that the head zero partition be created first. If allowed to continue, MAKE will create this partition with –DBS OFF.

If the user knows that Dynamic Badspot Handling was previously enabled on this spindle (that is, all partitions on the spindle were made with –DBS ON), MAKE recommends that this partition be created with the –FORMAT option to remove any remapped record pointers.

The best course of action would be to remake the head zero partition in the Dynamic Badspot Handling mode desired and then create the non-head zero partitions in the same mode. The user quits out of MAKE.

```
OK, MAKE -DISK 10462 -PART STUDNT -DT MODEL 4845
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Invalid DSKRAT on head zero partition (462).
Cannot determine if there is an active Dynamic BadSpot file on this spindle.

It is strongly recommended that you make the head zero partition (462)
BEFORE making this partition (10462).

Because the integrity of Dynamic BadSpotting on this spindle is in doubt,
this partition will be made with -DBS OFF (-AC).

If the head zero partition has previously been made with -DBS ON (-IC),
you are strongly recommended to make this partition with -FORMAT in order
to remove any old Dynamic BadSpot remappings.

OK to enable -FORMAT? [Y,N,Q] QUIT
OK,
```

If the user had specified the –DBS ON and –NO_QUERY options in the above example, as in the example below, MAKE would abort after displaying the warnings. Because the –NO_QUERY option does not allow the user to change the –DBS argument or quit out of the session, MAKE does not continue. If the user had not specified a –DBS option, MAKE could default to –DBS OFF.

```
OK, MAKE -DISK 10462 -PAR STUDNT -DT 4845 -DBS ON -NO_QUERY
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Invalid DSKRAT on head zero partition (462).
Cannot determine if there is an active Dynamic BadSpot file on this spindle.

It is strongly recommended that you make the head zero partition (462)
BEFORE making this partition (10462).

Because the integrity of Dynamic BadSpotting on this spindle is in doubt,
this partition must be made with -DBS OFF (-AC).
```

```
If the head zero partition was previously made with -DBS ON (-IC),
you are strongly recommended to remake this partition with -FORMAT in order
to remove any old Dynamic BadSpot remappings.

MAKE aborted because of invalid option specified with -NO_QUERY:
This partition cannot be made with -DBS ON (-IC).
ER!
```

Remaking a Non-head Zero Partition Without Head Zero Assigned: In this example, the user runs MAKE on a non-head zero partition but neglects to assign the head zero partition. MAKE needs the head zero partition assigned in order to determine the Dynamic Badspot Handling mode of the spindle. MAKE queries the user to continue and the user does not wish to continue. MAKE then pauses to allow the user to assign the head zero partition. After assigning the head zero partition, the use can enter START to continue MAKE from this point. Prime strongly recommends that the user assign the head zero partition before processing non-head zero partitions.

```
OK, MAKE -DISK 10462 -PAR GROUP2 -DT 4860
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition is not assigned;  cannot determine if there is an
active Dynamic BadSpot file on this spindle.

You are strongly recommended to QUIT from MAKE now, and assign the head zero
partition, if possible.  You should NOT continue with MAKE unless you are
certain that EITHER the head zero partition was NOT made with -DBS ON (-IC),
OR it was made with -DBS ON and the DBS file contains no badspots on this
partition (10462).

OK to continue with MAKE? NO
MAKE paused.  (Assign the head zero partition and type START to continue.)
ER!
```

If the user decides to continue rather than pausing to assign the head zero partition (not recommended), MAKE displays the following and requires the user to enter the Dynamic Badspot Handling mode of the head zero partition or to quit. The user should quit and assign the head zero partition.

```
OK, MAKE -DISK 10462 -PAR GROUP2 -DT 4860
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition is not assigned;  cannot determine if there is an
active Dynamic BadSpot file on this spindle.

You are strongly recommended to QUIT from MAKE now, and assign the head zero
partition, if possible.  You should NOT continue with MAKE unless you are
certain that EITHER the head zero partition was NOT made with -DBS ON (-IC),
```

```
OR it was made with -DBS ON and the DBS file contains no badspots on this
partition (10462).

OK to continue with MAKE? YES

You should make this partition with the same -DBS option as was used to
make the head zero partition on this spindle.

Was the head zero partition made with -DBS ON (-IC) or -DBS OFF (-AC)?
Please specify "ON" or "OFF" ("Q" to quit):
```

The user should quit at this point, assign the head zero partition, and enter START to continue MAKE.

If the user had specified the -NO_QUERY option in this example, MAKE aborts because the head zero partition is not assigned and no -DBS option is specified.

```
OK, MAKE -DISK 10462 -PAR GROUP2 -DT 4860
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition is not assigned;  cannot determine if there is an
active Dynamic BadSpot file on this spindle.

You should make this partition with the same -DBS option as was used to
make the head zero partition on this spindle.

Insufficient options with -NO_QUERY.  -DBS option needed.
MAKE aborted.
ER!
```

Remaking a Non-head Zero Partition With Conflicting DBS Modes: In this example, the user assigns the head zero partition and then specifies the incorrect -DBS argument. The second part of this example shows the effect of the -NO_QUERY option on this case. In this part, MAKE queries the user to continue, noting that MAKE will create the partition in the same mode as the head zero partition, which is assigned allowing MAKE to determine the mode.

```
OK, MAKE -DISK 10462 -PAR SYS_24 -DT MODEL_4845 -DBS OFF
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition (device 462) has been made with -DBS ON (-IC);
Dynamic BadSpotting is enabled on this spindle.

To make this partition (device 10462) with -DBS OFF (-AC) you must first
remake the head zero partition with -DBS OFF.

This partition will be made with -DBS ON.
OK to continue with MAKE? QUIT
```

```
MAKE aborted.
OK,
```

With the -NO_QUERY option, MAKE aborts, first noting that the user should remake the head zero partition to remove the DBS file before creating the non-head zero partition.

```
OK, MAKE -DISK 10462 -PAR TEST2 -DT MODEL_4845 -DBS OFF -NO_QUERY
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition (device 462) has been made with -DBS ON (-IC);
Dynamic BadSpotting is enabled on this spindle.

To make this partition (device 10462) with -DBS OFF (-AC) you must first
remake the head zero partition with -DBS OFF.

MAKE aborted because of invalid option specified with -NO_QUERY:
This partition cannot be made with -DBS OFF.
ER!
```

Examples of Use of the -NO_QUERY Option

-NO_QUERY is intended primarily for use with phantoms (see page 5-57), when creating partitions on disk types that support Dynamic Badspot Handling, to suppress the queries caused by

- Not having the head zero partition assigned when creating a non-head zero partition
- MAKE using -FORMAT to change the state of the DBS file when creating a head zero partition

If you use -NO_QUERY with FORMAT or -FORMAT_OK, MAKE may proceed in a way that could risk loss of data on other partitions on the spindle. MAKE will, however, display warnings about these conditions and may abort.

Example of MAKE Asking Permission to Override User's -DBS Option: Without -NO_QUERY, MAKE would normally query the user to continue after finding a conflicting -DBS argument.

```
OK, MAKE -DISK 10462 -PART ADMIN -DT MODEL_4845 -DBS OFF
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition (device 462) has been made with -DBS ON (-IC);
Dynamic BadSpotting is enabled on this spindle.

To make this partition (device 10462) with -DBS OFF (-AC) you must first
remake the head zero partition with -DBS OFF.
```

```
This partition will be made with -DBS ON.
OK to continue with MAKE? QUIT
MAKE aborted.
OK,
```

When the user specifies –NO_QUERY, MAKE aborts because of the conflicting –DBS arguments.

```
OK, MAKE -DISK 10462 -PART ADMIN -DT 4845 -DBS OFF -NO_QUERY
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition (device 462) has been made with -DBS ON (-IC);
Dynamic BadSpotting is enabled on this spindle.

To make this partition (device 10462) with -DBS OFF (-AC) you must first
remake the head zero partition with -DBS OFF.

MAKE aborted because of invalid option specified with -NO_QUERY:
This partition cannot be made with -DBS OFF.
ER!
```

Example of MAKE Needing –FORMAT to Continue: In this example, the head zero partition is assigned and was created with –DBS ON. The user specifies –DBS OFF potentially changing the Dynamic Badspot Handling mode of the spindle and risking loss of data on other partitions. MAKE must enable –FORMAT to deactivate the DBS file. Without –NO_QUERY, MAKE prompts to be sure the user wants to do this.

```
OK, MAKE -DISK 462 -PAR ADMIN2 -DT MODEL 4860 -DBS OFF
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

An activated DBS file exists on partition 462.  If you proceed with
-DBS OFF (-AC), MAKE will enable -FORMAT to deactivate the DBS file.

WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle. After this partition has been made,
          these other partitions should be remade with -FORMAT.

Are you sure you want to make this partition with -DBS OFF?
```

The user can enter Y, N, or Q after considering the alternatives.

If the user specifies –NO_QUERY on the command line in the above example, MAKE aborts because the –FORMAT option is needed to deactivate the DBS file. MAKE aborts when –FORMAT is *not* specified because formatting entails risk and MAKE will not proceed with a risky operation without permission. That is, with –NO_QUERY the –FORMAT option is construed as permission to proceed with a risky operation, if risk exists.

```
OK, MAKE -DISK 462 -PAR ADMIN2 -DT MODEL_4860 -DBS OFF -NO_QUERY
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

An activated DBS file exists on partition 462.  To proceed with -DBS OFF
(-AC), you must specify -FORMAT to allow MAKE to deactivate the DBS file.

WARNING:  By disturbing the DBS file you will risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

Insufficient options with -NO_QUERY.  -FORMAT option needed.
MAKE aborted.
ER!
```

Example of MAKE Needing Both –DBS and –FORMAT to Continue: In this example, the user specifies the –SPLIT option which changes the size of the head zero partition's file system, requiring the DBS file to be recreated in a new location. MAKE thus needs to format the partition and this risks loss of data on all other partitions. Because the user did not specify the –DBS option, MAKE needs to know whether Dynamic Badspot Handling should be enabled or disabled. MAKE cannot continue because this information is needed.

```
OK, MAKE -DISK 462 -PAR TESTNG -DT MODEL_4845 -SPLIT MAX -NO_QUERY
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Partition 462 appears to have been previously made with a different size.
This MAKE will destroy the existing DBS file.

To proceed, you must specify -FORMAT to allow MAKE to create a new DBS file.

WARNING:  By disturbing the DBS file you will risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

You can make this partition with Dynamic BadSpotting (-DBS) either ON or OFF.

A partition made with -DBS ON (-IC) ...
     - CAN be used on an IDC1 controller;  Dynamic BadSpotting WILL be done
     - CAN be mirrored, and CAN be used for CDD if split
     - Cannot be used on 4004 or 4005 controllers, which do not support
       Dynamic BadSpotting

A partition made with -DBS OFF (-AC) ...
     - Can be used on an IDC1 controller, but Dynamic BadSpotting is NOT done
     - Can be used on 4004 or 4005 controllers, which NEVER do
       Dynamic BadSpotting
Insufficient options with -NO_QUERY.  -FORMAT and -DBS needed.
MAKE aborted.
ER!
```

Example of the Use of –FORMAT_OK: –FORMAT_OK allows MAKE to enable the –FORMAT option if necessary. You use –FORMAT_OK only when making a head zero partition on a disk type that supports Dynamic Badspot Handling and that has an existing DBS file. Without –FORMAT_OK in this situation, MAKE aborts. Use of –FORMAT_OK allows MAKE to proceed. –FORMAT_OK is intended to be used with phantoms when there is no data of any value on any partition on the spindle. This example shows the use of –FORMAT_OK in a CPL file run as a phantom.

```
OK, SLIST M462.CPL
COMO M462.COMO
ASSIGN DISK 462
MAKE -DISK 462 -PAR TOMATO -DT 4845 -DBS OF -NO_QUERY -FORMAT_OK
COMO -END
OK, PH M462.CPL
PHANTOM is user 41
OK,
Phantom 41: Normal logout at 11:18
Time used: 00h 00m connect, 00m 02s CPU, 00m 48s I/O.
SLIST M462.COMO
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

An activated DBS file exists on partition 462.  Because you have specified
-DBS OFF (-AC), MAKE will enable -FORMAT to deactivate the DBS file.

WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

Making 2 head partition:     TOMATO          (Disk Revision 22.1)
Disk type:                   MODEL_4845
Number of tracks:            848
Sectors per track:           19
Sectoring mode:              REVERSE
Dynamic Badspot handling:    OFF  (-AC)
Partition size in records:   32224

Processing flaw map on device 462.
Processing of flaw map completed.

Beginning format.
Format completed.

Badspots on device 462:
        4 known badspots (from badspot file/s, flaw map, or terminal input)
        0 new badspots found during badspot checking or file system I/O
Total of 4 badspots mapped out of device 462.

Partition TOMATO created successfully.
OK,
```

Examples of Controller Not Supporting Dynamic Badspot Handling

If the disk type supports Dynamic Badspot Handling but the controller does not, for example, a 4005 controller, and you use the –DBS ON argument, MAKE detects this as an error and requires you to create the partition in Nondynamic Badspot Handling mode (–DBS OFF or –AC). This can happen with SMD disks which support Dynamic Badspot Handling but which can be moved to a nonintelligent controller or this can happen if cables are accidentally switched.

Head Zero Partition: The user specifies –DBS ON but the controller that this disk is connected to does not support Dynamic Badspot Handling. MAKE determines that this is the case and then prompts the user to create the partition with –DBS OFF.

```
OK, MAKE -DISK 2360 -DT SMD -PAR LIBRY -DBS ON
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Disk is connected to a controller that does not support Dynamic BadSpotting.
Option -DBS ON (-IC) ignored.
This partition will be made with -DBS OFF (-AC).

A partition made with -DBS OFF (-AC) ...
     - Can be used on an IDC1 controller, but Dynamic BadSpotting is NOT done
     - Can be used on 4004 or 4005 controllers, which NEVER do
       Dynamic BadSpotting
     - CANNOT be mirrored, and CANNOT be used for CDD

OK to continue with MAKE? NO
MAKE aborted.
OK,
```

In this related example, the user includes the –NO_QUERY option with the –DBS ON. The DSKRAT is corrupt so MAKE cannot determine if Dynamic Badspot Handling is enabled on the spindle. MAKE determines that the controller does not support Dynamic Badspot Handling and aborts because of the invalid –DBS option.

```
OK, MAKE -DISK 462 -DT MODEL_4845 -PAR LIBRY -DBS ON -NO_QUERY
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Invalid DSKRAT on partition 462.
Cannot determine if there is an active Dynamic BadSpot file on this spindle.

If this partition was previously made with -DBS ON (-IC), you are strongly
recommended to make it with -FORMAT in order to rebuild the DBS file.
```

```
WARNING: By disturbing the DBS file you will risk loss of data on all other
         partitions on the spindle.  If this partition is formatted,
         these other partitions should be remade with -FORMAT.

Disk is connected to a controller that does not support Dynamic BadSpotting.
This partition must be made with -DBS OFF (-AC).

A partition made with -DBS OFF (-AC) ...
     - Can be used on an IDC1 controller, but Dynamic BadSpotting is NOT done
     - Can be used on 4004 or 4005 controllers, which NEVER do
       Dynamic BadSpotting
     - CANNOT be mirrored, and CANNOT be used for CDD

MAKE aborted because of invalid option specified with -NO_QUERY:
This partition cannot be made with -DBS ON (-IC).
ER!
```

Non-head Zero Partition: The user specifies –DBS ON but the controller does not support this option. MAKE first attempts to determine the DBS mode of the head zero partition by prompting the user because the head zero partition is not assigned and then determines that the controller will not support the –DBS ON option and aborts.

```
OK, MAKE -DISK 10462 -PAR TEST2 -DT MODEL 4845 -DBS ON
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition is not assigned;  cannot determine if there is an
active Dynamic BadSpot file on this spindle.

You are strongly recommended to QUIT from MAKE now, and assign the head zero
partition, if possible.  You should NOT continue with MAKE unless you are
certain that EITHER the head zero partition was NOT made with -DBS ON (-IC),
OR it was made with -DBS ON and the DBS file contains no badspots on this
partition (10462).

OK to continue with MAKE? Y

You should make this partition with the same -DBS option as was used to
make the head zero partition on this spindle.

Was the head zero partition made with -DBS ON (-IC) or -DBS OFF (-AC)?
Please specify "ON" or "OFF" ("Q" to quit): ON

Disk is connected to a controller that does not support Dynamic BadSpotting.
This partition cannot be made with -DBS ON (-IC).
ER!
```

In this related example, the user runs MAKE on a non-head zero partition using the –DBS ON option but the head zero partition is not assigned. Because MAKE lacks access to the head zero partition, MAKE cannot guarantee that it will make a valid partition in the case where the spindle has –DBS ON and the DBS file contains badspots on this partition. The user finally enters the –DBS OFF argument at the prompt allowing MAKE to continue and create the partition. The user should quit and assign the head zero partition so that MAKE can guarantee creating a valid partition, whatever the circumstances.

```
OK, MAKE -DISK 10462 -PAR TEST2 -DT MODEL_4845 -DBS ON
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition is not assigned;  cannot determine if there is an
active Dynamic BadSpot file on this spindle.

You are strongly recommended to QUIT from MAKE now, and assign the head zero
partition, if possible.  You should NOT continue with MAKE unless you are
certain that EITHER the head zero partition was NOT made with -DBS ON (-IC),
OR it was made with -DBS ON and the DBS file contains no badspots on this
partition (10462).

OK to continue with MAKE? YES

You should make this partition with the same -DBS option as was used to
make the head zero partition on this spindle.

Was the head zero partition made with -DBS ON (-IC) or -DBS OFF (-AC)?
Please specify "ON" or "OFF" ("Q" to quit): OFF

Making 2 head partition:    TEST2            (Disk Revision 22.1)
Disk type:                  MODEL_4845
Number of tracks:           848
Sectors per track:          19
Sectoring mode:             FORWARD
Dynamic Badspot handling:   OFF  (-AC)
Partition size in records:  32224
     .    .    .
     .    .    .
     .    .    .
```

Example of Entering Known Badspots

This example shows the use of the –QUERY_BADSPOTS option and illustrates entering known badspots and the verification of the input data.

The partition is first placed in the Assignable Disks Table and assigned. At the end of the session, the partition is unassigned and removed from the Assignable Disks Table. The user first enters badspots by record number. One record number is within the DSKRAT and MAKE ignores that input. The user enters a record number that is outside this partition and MAKE ignores that also. The user then verifies the data entered so far and then enters more badspots that are outside the partition by track, head, and sector. Verification shows them to be outside this partition.

After the known badspots are entered, MAKE checks for additional badspots.

```
OK, DISKS 40760
OK, ASSIGN DISK 40760
OK, MAKE -DISK 40760 -PART FRUITY -DT SMD -DBS OFF -QRY
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Making 2 head partition:     FRUITY            (Disk Revision 22.1)
Disk type:                   SMD
Number of tracks:            823
Sectors per track:           9
Sectoring mode:              FORWARD
Dynamic Badspot handling:    OFF  (-AC)
Partition size in records:   14814

Badspots may be entered as record numbers, or as track/head/sector.
Use record number format? Y
Record numbers are OCTAL.  Enter a record number of 0 to finish.
Record = 10204
Record = 2
Cannot add badspot at record = '2 (track = 0, head = 8, sector = 2)
  A record below or within the DSKRAT (record <= '3) cannot be a badspot.
Record = 113456
Record number '113456 is too big, maximum for device 40760 is '34735.
Record = 13456
Record = 0
Verify data? Y

   Track       Head       Sector        Record
   -----       ----       ------      ----------
*     0          8          3                '3
*   234          9          7            '10204
*   329          9          3            '13456

Note:  Badspots on device 40760 are marked above with a "*".

List of badspots OK? NO
You may delete or retain badspots entered so far.
OK to delete? NO

Badspots may be entered as record numbers, or as track/head/sector.
Use record number format? NO
Enter (decimal) track, head, and sector.
Enter a track, head, and sector of 0 to finish.
```

```
Track = 111
Head = 5
Sector = 7
Track = 112
Head = 5112
Sector = 7
Track = 0
Head = 0
Sector = 0
Verify data? Y

   Track        Head        Sector           Record
   -----        ----        ------         -----------
     111          5            7
     112          5            7
*      0          8            3                 '3
*    234          9            7             '10204
*    329          9            3             '13456

Note:  Badspots on device 40760 are marked above with a "*".

List of badspots OK? Y

Checking for badspots (level = 1).
         .        .        .
         .        .        .
         .        .        .

OK, UNASSIGN DISK 660
OK, DISKS NOT 660
OK,
```

Example of Creating a Split Partition

The following example illustrates the use of the MAKE command with the –SPLIT option to make a paging partition or a crash dump disk. See the –SPLIT option on page 5-15 for other examples.

The user puts the head zero and this partition in the Assignable Disks Table, assigns the partitions, and invokes MAKE with the –SPLIT option. After entering an invalid number of records (more records than are available), the user enters the MAX argument to specify the maximum number of records available to use for paging.

```
OK, DISKS 2360
OK, DISKS 40760
OK, ASSIGN DISK 2360
OK, ASSIGN DISK 40760
```

```
OK, MAKE -DISK 40760 -PAR PAGING -DT SMD -LEV 0 -SPLIT -NQ
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition (device 2360) has been made with -DBS OFF (-AC);
Dynamic BadSpotting is NOT enabled on this spindle.

This partition will be made with -DBS OFF.

Please specify how you want to -SPLIT this partition.
Maximum number of paging/CDD records available is 14806.
Number of paging/CDD records? 14860
Invalid number of paging/CDD records (14860).
Maximum number of paging/CDD records available is 14806.
Number of paging/CDD records? MAX

Making 2 head partition:    PAGING             (Disk Revision 22.1)
Disk type:                  SMD
Number of tracks:           823
Sectors per track:          9
Sectoring mode:             FORWARD
Dynamic Badspot handling:   OFF   (-AC)

File system records:        8
Paging/CDD  records:        14806

No BADSPT file found on device 40760.
Continuing with MAKE.

Total of 0 badspots mapped out of device 40760.

Partition PAGING created successfully.
OK, UNASSIGN DISK 40760
OK, UNASSIGN DISK 2360
OK, DISKS NOT 40760
OK, DISKS NOT 2360
OK,
```

The method of file record allocation was not specified, so MAKE determines the CPU type and the disk controller type and sets the allocation method to forward sectoring. MAKE determines the DBS mode of the assigned head zero partition and sets this partition to the same mode.

Example of Copying Badspots From Another Partition

In the example, the user specifies the –REPORT option and MAKE reports the progress of badspot checking. The user also specifies the –LIST_BADSPOTS option and MAKE lists the badspots at completion of the process. Two badspots are found by MAKE on this partition; the remaining badspots listed are on the other partitions. The user copies badspots from two other partitions, one of which is assigned (2360) and the other is added to the system (PINCER). After entering the partitions to copy from, the user enters a carriage return at the next prompt and MAKE reprompts until the user enters a 0.

```
OK, MAKE -DISK 40760 -DT SMD -DBS OFF -COPY_BADSPOTS -RPT -LST
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Partition name? STUDNT
Specify pdev(s) or partition(s) from which badspots are to be copied.
You may specify up to 4 devices in all.  Enter "0" (zero) to finish.
Enter pdev or partition: 2360
Copying badspots from:  2360
Enter pdev or partition: PINCER
Copying badspots from:  2360, PINCER
Enter pdev or partition:
Enter pdev or partition: 0
2 devices specified.  OK to continue with MAKE? Y

Making 2 head partition:    STUDNT              (Disk Revision 22.1)
Disk type:                  SMD
Number of tracks:           823
Sectors per track:          9
Sectoring mode:             FORWARD
Dynamic Badspot handling:   OFF  (-AC)
Partition size in records:  14814

No BADSPT file found on device 40760.
Continuing with MAKE.

4 badspots added from BADSPT file on device 2360.

Unable to find partition PINCER.
Ignoring -COPY_BADSPOTS option.  Continuing with MAKE.

Checking for badspots (level = 1).
Badspot checking now  5% complete.
Badspot checking now 10% complete.
        .        .        .
        .        .        .
        .        .        .
New badspot detected writing record = '25245
    (track = 606, head = 9, sector = 0)

New badspot detected writing record = '25307
    (track = 608, head = 8, sector = 7)
        .        .        .
        .        .        .
        .        .        .
Badspot checking now 90% complete.
```

```
Badspot checking now 95% complete.
Done checking for badspots.  2 new badspots detected.

List of badspots:

   Track        Head        Sector            Record
   -----        ----        ------          -----------
     455          0            4
     332          1            7
      10          4            2
     469          5            3
*    608          8            7              '25307
*    606          9            0              '25245

Note:  Badspots on device 40760 are marked above with a "*".

Badspots on device 40760:
         0 known badspots (from badspot file/s, flaw map, or terminal input)
         2 new badspots found during badspot checking or file system I/O
Total of 2 badspots mapped out of device 40760.

Partition STUDNT created successfully.
OK,
```

The method of file record allocation was not specified, so MAKE determines the CPU type and the disk controller type and sets the allocation method to forward sectoring based on a CPU in the 9950 class with a nonintelligent disk controller, as indicated in Table 10-2.

Booting MAKE.SAVE

Rev. 23.3 MAKE will run under PRIMOS or it may be booted to run by itself without PRIMOS. The latter mode of operation, sometimes referred to as running standalone, is useful for creating new systems or for recovering from halts. Standalone MAKE does not support Dynamic Badspot Handling and you cannot use the –DBS ON option (with one exception - see page 5-96) because MAKE cannot determine what type of controller you have in the system nor can an intelligent controller be downline loaded until PRIMOS is running.

MAKE is run standalone by giving the Rev. 23.3 system boot the pathname of MAKE when the system boot prompts for the runfile pathname. The pathname of Rev. 23.3 MAKE is CMDNC0>MAKE.SAVE. MAKE may be booted from disk or from MAGSAV-format magnetic tape.

Note See your CPU handbook for an explanation of the BOOT command and its options.

When booting MAKE to run standalone, these restrictions apply:

- You can only boot from disk drive units 0 through 3 on controllers with addresses of 22_8, 23_8, 26_8, and 27_8.
- Standalone MAKE cannot determine the CPU or controller type and, thus, cannot determine the correct –SECTOR argument so always prompts for this information except on the Model 7210 controller.
- You cannot use the –COPY_BADSPOTS option with the *name* argument because there is no file system until PRIMOS is running.
- The erase and kill characters are the double quote (") and the question mark (?), respectively. (Under PRIMOS, MAKE uses the user's current erase and kill characters.)

When MAKE is run under PRIMOS, command line options are typed immediately following the MAKE command and on the same command line. When MAKE is run standalone, MAKE prompts you for the command line options because the boot does not accept command line options following the pathname. See your CPU handbook for an explanation of the BOOT command and its options.

When MAKE is running standalone, the CPU halts when MAKE has finished. To run MAKE again at this point, you need only enter the VCP command SYSCLR at the `CP>` prompt and then enter the RUN command; MAKE will then restart. This is useful if you run MAKE first with the –HELP option and then want to run MAKE to create a partition.

Booting MAKE.SAVE From Disk

The following examples of using MAKE by booting from disk could also be accomplished by booting MAKE from tape.

Example of Creating a Head Zero Partition Using –DBS ON: When you use the –DBS ON option with standalone MAKE, MAKE informs you that the ON argument will be ignored and prompts to check that you want to make the head zero partition with –DBS OFF. You can answer YES and allow MAKE to use –DBS OFF and then later convert to –DBS ON (–IC) with FIX_DISK, answer NO and allow MAKE to abort, or quit out of MAKE. If you must create the partition enabling Dynamic Badspot Handling on the spindle, you will have to use FIX_DISK later to convert to Dynamic Badspot Handling mode.

In the BOOT 10314 command, the 10000_8 switch is set as MAKE is booted from the partition on drive 1 and controller address 26_8, the same device that the boot is read from. Because the 4000_8 switch is not set, the `RUN FILE TREENAME=` prompt is displayed and you enter the pathname of MAKE.

After MAKE starts, you are prompted for command line options. You should enter all the options and their arguments that you want to use. If you enter no options, MAKE will prompt you for the pdev, the partition name, and the disk type. For other options that expect arguments, MAKE will prompt you if the arguments are missing or invalid. Once you enter the command line options, the operation of MAKE running standalone is identical to operation under PRIMOS.

```
CP> SYSCLR
DPM006: Central Processor System initialization completed.
        28 Feb 1992 15:26:29 Friday
CP> BOOT 10314
DPM007: System Booting, please wait.
[CPBOOT Rev. 10.2 Copyright (c) 1990 Prime Computer, Inc.]
[BOOT Rev. 23.3 Copyright (c) 1992 Prime Computer, Inc.]

RUN FILE TREENAME=CMDNC0>MAKE.SAVE

BOOTING FROM 001062 CMDNC0>MAKE.SAVE

[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]
Enter command line options: -DISK 2360 -PAR TEST -DT SMD -DBS ON

Dynamic BadSpotting is not supported by standalone MAKE.
Option -DBS ON (-IC) ignored.
This partition will be made with -DBS OFF (-AC).

A partition made with -DBS OFF (-AC) ...
     - Can be used on an IDC1 controller, but Dynamic BadSpotting is NOT done
     - Can be used on 4004 or 4005 controllers, which NEVER do
       Dynamic BadSpotting
     - CANNOT be mirrored, and CANNOT be used for CDD

OK to continue with MAKE?
```

If you wish to continue, enter YES.

Note that you cannot use the –DBS ON (–IC) option when you run MAKE standalone. If you attempt to create a disk in Dynamic Badspot Handling mode when running MAKE standalone, MAKE informs you that you cannot and defaults to –DBS OFF and prompts to see if you want to continue. If you do not answer YES, you must then start over. When running standalone, MAKE also prompts for the type of record allocation scheme you want.

The reason MAKE cannot create a –DBS ON mode partition is because it cannot determine the type of disk controller or downline load the controller when running standalone. MAKE must prompt for sectoring information because it cannot determine the type of CPU in your system and set the record allocation properly.

You can specify the –DBS OFF option when running MAKE standalone; it is unnecessary, however, because MAKE cannot use –DBS ON when running standalone.

Example of Creating a Head Zero Partition With an Existing DBS File: In this example, the user first specifies the –USAGE option and, when the CPU halts, the user enters RUN to re-execute MAKE. The user then specifies –DBS ON and MAKE ignores the ON argument because the user is running MAKE standalone. The head zero partition was previously created with –DBS ON. Thus, MAKE must enable –FORMAT to deactivate the DBS file and then use –DBS OFF to create the partition. The user can let MAKE proceed and deactivate the DBS file or can quit out of MAKE.

```
CP> SYSCLR
DPM006: Central Processor System initialization completed.
        28 Feb 1992 15:26:29 Friday
CP> BOOT 10314
DPM007: System Booting, please wait.
[CPBOOT Rev. 10.2 Copyright (c) 1990 Prime Computer, Inc.]
[BOOT Rev. 23.3 Copyright (c) 1992 Prime Computer, Inc.]

RUN FILE TREENAME=CMDNC0>MAKE.SAVE

BOOTING FROM 001062 CMDNC0>MAKE.SAVE

[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]
Enter command line options: -USAGE

Usage:

MAKE  {-HELP | -USAGE}

MAKE  [-DiSK [pdev]]                 [-DBS [ON | OFF]]
      [-PARtition [name]]            [-badspot_LEVel [level]]
      [-Disk_Type [type]]            [-CoPY_badspots [pdev | partition]]
      [-disk_REVision [rev]]         [-QueRY_badspots]
      [-SPLit [records]]             [-LiST_badspots]
      [-BAUd_rate [baud]]            [-No_FLaw_map]
      [-RePorT]                      [-map_UNCorr]
      [-No_Query]                    [-SECtor [FORward | REVerse]]
      [-ForMaT | -Format_OK]         [-MIN_extent_size [records]]
      [-INIt | -No_INit]             [-MAX_extent_size [records]]
      [-NEW_disk]

DPM400: CPU halted at 036354: 103775
        28 Feb 92 15:27:04
CP> SYSCLR
DPM006: Central Processor System initialization completed.
        28 Feb 1992 15:27:06 Friday
CP> RUN
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]
Enter command line options: -DISK 462 -PAR TEST -DT 4845 -DBS ON
```

```
Dynamic BadSpotting is not supported by standalone MAKE.
Option -DBS ON (-IC) ignored.
This partition will be made with -DBS OFF (-AC).

A partition made with -DBS OFF (-AC) ...
     - Can be used on an IDC1 controller, but Dynamic BadSpotting is NOT done
     - Can be used on 4004 or 4005 controllers, which NEVER do
       Dynamic BadSpotting
     - CANNOT be mirrored, and CANNOT be used for CDD

An activated DBS file exists on partition 462.  If you proceed with
-DBS OFF (-AC), MAKE will enable -FORMAT to deactivate the DBS file.

WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

OK to continue with MAKE? QUIT
MAKE aborted.
DPM400: CPU halted at 036354: 103775
        28 Feb 92 09:12:44 Friday
CP>
```

The CPU halts when MAKE aborts or finishes. If MAKE is allowed to continue, the user should run MAKE on all other partitions on the spindle using the –FORMAT option, either running MAKE standalone or under PRIMOS.

Example of Creating a Non-head Zero Partition on a DBS Spindle: In this example, the head zero partition has Dynamic Badspot Handling on and thus enabled on the entire spindle. Even if you specify –DBS OFF or do not specify the –DBS option, standalone MAKE cannot create this partition because the head zero partition must first be created with Dynamic Badspot Handling off (–AC). MAKE must abort in this case and you would have to remake the head zero partition with –DBS OFF and –FORMAT.

```
CP> SYSCLR
DPM006: Central Processor System initialization completed.
        28 Feb 1992 15:26:29 Friday
CP> BOOT 10314
DPM007: System Booting, please wait.
[CPBOOT Rev. 10.2 Copyright (c) 1990 Prime Computer, Inc.]
[BOOT Rev. 23.3 Copyright (c) 1992 Prime Computer, Inc.]

RUN FILE TREENAME=CMDNC0>MAKE.SAVE

BOOTING FROM 001062 CMDNC0>MAKE.SAVE

[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]
Enter command line options: -DISK 10462 -PAR TEST2 -DT 4860
```

```
Head zero partition (device 462) has been made with -DBS ON (-IC);
Dynamic BadSpotting is enabled on this spindle.

Dynamic BadSpotting is not supported by standalone MAKE.

To make this partition (device 10462) with -DBS ON (-IC),
run MAKE under PRIMOS.

To make this partition (device 10462) with -DBS OFF (-AC),
you must first remake the head zero partition with -DBS OFF.

This partition cannot be made with either -DBS ON or -DBS OFF
in this MAKE session.
MAKE aborted.
DPM400: CPU halted at 036354: 103775
        28 Feb 92 09:12:44 Friday
CP>
```

Creating a Paging Partition with –DBS ON: Normally standalone MAKE does not allow you to create a partition with –DBS ON. There is one exception to this rule, intended to be used *only* in the case of a system with only one spindle containing a command device made with –DBS ON and a paging device which needs to be remade. In this situation, MAKE allows you to remake the non-head zero paging device with –DBS ON.

Standalone MAKE cannot add new badspots to the DBS file because it cannot download the IDC1 controller. For this reason, the –QRYBAD and –BADLEV options are ignored. However, the flaw map and badspot files are read so that MAKE can avoid using any known badspots in creating the file system.

```
CP> SYSCLR
DPM006: Central Processor System initialization completed.
        29 Mar 1992 09:06:29 Sunday
CP> BOOT 10314
DPM007: System Booting, please wait.
[CPBOOT Rev. 10.2 Copyright (c) 1990 Prime Computer, Inc.]
[BOOT Rev. 23.3 Copyright (c) 1992 Prime Computer, Inc.]

RUN FILE TREENAME=CMDNC0>MAKE.SAVE

BOOTING FROM 001062 CMDNC0>MAKE.SAVE
Enter command line options: -DISK 10462 -PAR PAGER -DT 4860 -SPLIT -QRY -LEV 1
-RPT
[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]

Head zero partition (device 462) has been made with -DBS ON (-IC);
Dynamic BadSpotting is enabled on this spindle.

This partition will be made with -DBS ON.
```

```
WARNING: Dynamic BadSpotting is not supported by standalone MAKE.
         You should make this partition under PRIMOS if at all possible.

MAKE is assuming that this split partition is intended for paging.
Badspot checking/input will not be done.  Badspots will be ignored.

OK to continue with MAKE? Y

Which sectoring scheme would you like?
Enter "F" for Forward Sectoring, "R" for Reverse Sectoring: F

Please specify how you want to -SPLIT this partition.
Maximum number of paging (or CDD) records available is 52395.
Number of paging/CDD records? 12000

Making 2 head partition:       PAGER          (Disk Revision 22.1)
Disk type:                     MODEL_4860
Number of tracks:              1379
Sectors per track:             19
Sectoring mode:                FORWARD
Dynamic Badspot handling:      ON  (-IC)

File system records:           40402
Paging records:                12000

Processing flaw map on device 10462.
179 badspots added from flaw map on device 10462.

0 badspots added from DBS file on device 462.

Partition PAGER created successfully.
DPM400: CPU halted at 036354: 103775
        29 Mar 92 09:12:44 Sunday
CP>
```

Booting MAKE.SAVE From Magnetic Tape

The following example illustrates booting MAKE from magnetic tape to create a system. First, the Operator boots CMDNC0>MAKE.SAVE from tape drive 1 on controller 0 to create and format the partition. The pathname of MAKE must be entered exactly as it was saved by MAGSAV. The –PARTITION option is not entered on the command line; MAKE then prompts for the partition name. When MAKE finishes, the CPU halts.

This example illustrates the method you would use to restore a system from magnetic tape. You would follow the following steps, which are also outlined in your CPU handbook.

1. Boot MAKE from a boot tape to create the command device (COMDEV) as in this example.
2. Boot MAKE from tape again to create a paging partition using the -SPLIT option.
3. Restore your command device software from a backup tape using MTRESUME to resume MAGRST from your boot tape.
4. Boot your system from disk.

```
CP> SYSCLR
DPM006: Central Processor System initialization completed.
        22 May 92 09:08:45 Friday
CP> BOOT 10205
DPM007: System booting, please wait.
[CPBOOT Rev. 10.2 Copyright (c) 1990, Prime Computer, Inc.]
[BOOT Rev. 22.1 Copyright (c) 1992, Prime Computer, Inc.]

RUN FILE TREENAME=CMDNC0>MAKE.SAVE

BOOTING FROM MT1 CMDNC0>MAKE.SAVE

[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]
Enter command line options: -DISK 660 -DT 4730

Partition name? BIO101

Making 2 head partition:     BIO101          (Disk Revision 22.1)
Disk type:                   MODEL_4730
Number of tracks:            13
Sectors per track:           254
Sectoring mode:              FORWARD
Partition size in records:   7112

Partition BIO101 created successfully.

DPM400: CPU halted at 025175: 103775
        22 May 92 09:27:36 Friday
CP>
```

See your CPU handbook for further information and examples of booting PRIMOS and MAKE and other utilities from magnetic tape.

Repairing Partitions – FIX_DISK

6

This chapter discusses the FIX_DISK command, the PRIMOS utility for repairing disk partitions. The chapter presents an itemized list of the uses of FIX_DISK, discusses all of the FIX_DISK command-line options, and shows examples of running FIX_DISK.

You should use FIX_DISK under PRIMOS to repair disks of any revision. Rev. 23.3 FIX_DISK can repair Rev. 20.0 and later disk partitions, which have hashed directories, as well as pre-Rev. 20.0 disk partitions, which do not have hashed directories.

Note It is recommended that you delete any pre-Rev. 23.3 versions of FIX_DISK. This is to avoid the possibility of mistakenly invoking a pre-Rev. 23.3 version and thereby getting unexpected results. The Rev. 23.3 version of FIX_DISK resides in CMDNC0 as FIX_DISK.SAVE. When you invoke FIX_DISK, a copyright line with the revision stamp is displayed.

Using FS_RECOVER

This chapter describes the procedures and considerations for manually running FIX_DISK. If your command device (COMDEV) is damaged, you must repair it by manually running FIX_DISK at the supervisor terminal. For other partitions, you can either manually run FIX_DISK or you can use FS_RECOVER to determine what partitions should have FIX_DISK run on them and to automate the repair process. Prime recommends that you routinely use FS_RECOVER whenever PRIMOS indicates that you should run FIX_DISK. For complete details on using FS_RECOVER, see *Using FS_RECOVER.*

What Does FS_RECOVER Do?

The main goal of FS_RECOVER is to reduce file system recovery time following a system crash. FS_RECOVER can assess the general state of your file system and provide an automated interface to FIX_DISK. If your system did crash and you took a crash dump, you can use FS_RECOVER to read and analyze the crash dump.

You can also use FS_RECOVER without a crash dump. You can use FS_RECOVER to make a generalized assessment of the state of your locally added partitions. If any of these partitions are damaged, you can have FS_RECOVER set up for automated FIX_DISK the same way it does for a crash dump recovery analysis.

FS_RECOVER determines

- Which partitions need to be fixed *immediately*
- Which partitions need fixing that can be *deferred* to a more convenient time
- Which partitions are not damaged or were unaffected by a crash

FS_RECOVER determines which partitions are damaged and which partitions are *clean*, that is, do not cause PRIMOS to generate a warning message at the time the partition is added, or mounted.

FS_RECOVER also determines the correct FIX_DISK options for those partitions that must be fixed and provides an automated facility for running FIX_DISK.

What Is FIX_DISK?

FIX_DISK is an Operator command that

- Reads every physical record that is in use on a disk or partition, including records in files, directories, and segment directories
- Checks the quota information on partitions
- Checks that the information in each record header is consistent with the directory that contains the record
- Checks the DSKRAT file for discrepancies
- Checks ACLs
- Checks file system pointers

When FIX_DISK identifies any error on a partition, FIX_DISK displays an appropriate error message. See Appendix B for an explanation of all FIX_DISK messages.

An important feature of FIX_DISK is its repair facility. FIX_DISK can

- Repair mismatched pointers
- Correct quota information
- Either replace defective records with new empty records or truncate files containing defective records
- Delete defective files
- Rebuild a defective DSKRAT file

FIX_DISK has other uses as well. It can

- Add new badspot information on disks where PRIMOS handles badspots
- Convert disks made as pre-Rev. 21.0 partitions to Rev. 19.0, Rev. 20.0, or Rev. 21.0 partitions
- Convert disks made as Rev. 22.0 partitions to Rev. 22.1 standard partitions
- Change the maximum and minimum CAM file extent sizes set on Rev. 22.0 and subsequent revision partitions
- Remap equivalence blocks created by PSR
- Check a partition to see whether it needs repair
- Rapidly repair robust partitions containing CAM files
- Determine if a partition was shut down properly and, if it was not, warn you that you should run FIX_DISK
- Enable dynamic badspot handling so that partitions may be mirrored
- Display the dynamic badspot (DBS) file
- Change the method of file record allocation

How Does FIX_DISK Work?

FIX_DISK performs these tasks, as described in the following paragraphs, when you command it to repair partitions by using the –FIX option.

- Locates DSKRAT errors
- Locates quota errors
- Handles the BADSPT file or the DBS file and the RMA

- Converts pre-Rev. 21.0 partitions to Rev. 21.0, pre-Rev. 20.0 partitions to Rev. 20.0, or pre-Rev. 19.0 partitions to Rev. 19.0
- Converts Rev. 22.0 partitions to Rev. 22.1 standard partitions

Locating DSKRAT Errors

In order to read each physical record currently in use, FIX_DISK must traverse the entire logical file structure of a partition. While traversing, FIX_DISK creates its own **Record Availability Table (RAT)**, which is updated and checked against the existing Disk Record Availability Table (DSKRAT) for each record. If FIX_DISK notes a discrepancy in comparing its own RAT to the DSKRAT, it displays an error message.

Locating Quota Errors

On Rev. 19.0 and subsequent revision partitions (on which the quota information is maintained), FIX_DISK compares the count of records used for directories, subdirectories, and files against the actual number of records within the directory. An error message is generated if there is an error in the directory's count of records that have been used.

Converting a Pre-Rev. 21.0 Partition

FIX_DISK can convert a pre-Rev. 21.0 partition

- From Rev. 18.0 to either Rev. 19.0, Rev. 20.0, or Rev. 21.0
- From Rev. 19.0 to Rev. 20.0 or Rev. 21.0
- From Rev. 20.0 to Rev. 21.0.

The options that accomplish these conversions are –CONVERT_19, –CONVERT_20, and –CONVERT_21.

To convert a partition, FIX_DISK does these three things:

- Initializes the quota information
- Upgrades the current badspot file to the proper revision format
- Creates the proper revision stamp in the DSKRAT

When you use FIX_DISK to convert a pre-Rev. 20.0 partition to Rev. 20.0 or to Rev. 21.0, existing directories do not become hashed. Only ACL directories created *after* the conversion will be hashed directories. The MFD cannot be converted to a hashed directory by FIX_DISK. For these and other reasons, you should use MAKE to convert pre-Rev. 20.0 partitions to Rev. 21.0 and you must use MAKE to convert pre-Rev. 22.0 partitions to Rev. 22.1 format. Converting to Rev. 21.0 also involves

- Selecting Dynamic Badspot Handling (–DBS ON or –IC) mode or Nondynamic Badspot Handling (–DBS OFF or –AC) mode for the entire spindle on those disks that support Dynamic Badspot Handling
- Changing the DSKRAT to Rev. 21.0 format
- Updating the revision stamp

However, remember that, since the MFD cannot be converted to a hashed directory by FIX_DISK and other directories do not become hashed without further steps, it is recommended that you use Rev. 23.3 MAKE to convert pre-Rev. 22.0 partitions to Rev. 22.1 format, following the steps discussed in Chapter 5, Formatting Disks - MAKE.

Since Rev. 21.0 partitions have the same format as Rev. 22.1 standard partitions, it is not necessary to convert Rev. 21.0 partitions to Rev. 22.1 format unless

- You wish to use robust partitions
- You wish to have unlimited CAM file extents
- You wish to set the per-partition minimum and maximum CAM file extent sizes
- You wish to bring the DSKRAT header information and the revision stamp on the Rev. 21.0 partition to Rev. 22.1

If you want to use robust partitions, you must use Rev. 23.3 MAKE to convert partitions to Rev. 22.1. (See Chapter 7 for information on robust partitions.)

Handling the BADSPT File or the DBS File and the RMA

For disks on nonintelligent (4005) controllers or Model 6580 (IDC1) intelligent controllers, all badspots encountered are added either to the static badspot file (BADSPT) or to the dynamic badspot file (DBS), if they exist. If no badspot file exists, a new badspot file is created if there is room on the partition. If FIX_DISK encounters an equivalence section in a partition's badspot file, FIX_DISK maps bad records to their equivalence records and then sets the file system pointers to those records. When the file system structure has been completely traversed, the equivalence section is deleted.

Whether you select Dynamic Badspot Handling (–DBS ON or –IC) mode or Nondynamic Badspot Handling (–DBS OFF or –AC) mode, FIX_DISK must activate or deactivate the DBS file and the RMA. In Nondynamic Badspot Handling (–DBS OFF) mode, FIX_DISK must also allow PRIMOS access to the static badspot file, BADSPT, if it exists.

When Should You Use FIX_DISK?

You should run FIX_DISK in these situations:

- If you suspect that the file structure is damaged or that the quota system is damaged (indicated by a warning message from PRIMOS)
- If PRIMOS tells you that the system was not properly shut down
- If a problem exists with attaching to a directory or using a file (for example, a database file)
- If a message from PSR indicates that an equivalence block has been created
- After a power failure or other outage that shuts down the system
- When you wish to change controller modes, Dynamic Badspot Handling (–DBS ON) mode or Nondynamic Badspot Handling (–DBS OFF) mode
- When you wish to change the maximum and minimum CAM file extent sizes
- When you wish to convert a Rev. 22.0 partition to a Rev. 22.1 standard partition and use unlimited CAM file extents

---

WARNING If you suspect a problem with the disk drive, disk pack, or disk controller, you should *not* run FIX_DISK with the –FIX option until the hardware is checked. A hardware problem could cause FIX_DISK to erroneously delete files and directories.

Table 6-1 presents recommendations as to when you should run FIX_DISK. See the discussion in the section FIX_DISK and Robust Partitions in Chapter 7 for more detail and explanations concerning these recommendations. Also see the discussion in Chapter 1 relating to FIX_DISK and file system states.

Table 6-1. Recommendations for Using FIX_DISK

Type of Halt	*Standard Partition*	*Robust Partition*
Forced shutdown – successful	No FIX_DISK unless message from ADDISK; then use fast FIX_DISK	No FIX_DISK unless message from ADDISK; then use fast FIX_DISK
Forced shutdown – unsuccessful – no Locate flush	Full FIX_DISK	Fast FIX_DISK
Trapped Halt – DMX completed	Full FIX_DISK	Fast FIX_DISK
ECCU and IDC1 – DMX completed	Full FIX_DISK	Fast FIX_DISK
ECCU and NDC – DMX not completed	Full FIX_DISK	Fast FIX_DISK or Full FIX_DISK (See note on page 7-29.)

Note In the above table, IDC1 refers to the Model 6580 intelligent disk controllers and NDC refers to nonintelligent disk controllers.

Why Run FIX_DISK as Part of System Backups?

System backups are an important aspect of protecting the data on your system against accidental loss. It is equally important that all data are backed up correctly. By running FIX_DISK before backing up a partition, you ensure the integrity of the backup procedure itself. The partition you are backing up should be clean such that you do not encounter any errors that may cause a record or a directory to be inaccessible.

When badspot handling has taken place on a partition after you back up the partition to disk by using PSR, running FIX_DISK on the backup partition reduces the amount of time it takes to recover from a loss when you use the backup partition.

How Does PRIMOS Warn You to Run FIX_DISK?

Whenever a partition is added to the system, PRIMOS checks to see if its file structure has been damaged or the partition was not properly shut down. When PRIMOS starts up the partition, it displays the message

```
Starting up revision 22.1 partition "NAME".
```

If it finds that the file structure may be damaged, it displays one of the following messages.

```
*** Disk "diskname" was not shutdown properly, Run FIX_DISK.  ***
*** Disk Write Errors detected, Run FIX_DISK On Disk "diskname" ***
*** Disk DTA/DTM Errors detected, Run FIX_DISK On Disk "diskname" ***
*** Disk Return Record Errors detected, Run FIX_DISK On Disk "diskname" ***
*** Disk CRA Mismatch Errors detected, Run FIX_DISK On Disk "diskname" ***
```

Generally, disks are added during system cold start. Therefore, you should look at the output generated by the system during each cold start to determine if you should run FIX_DISK. You should look at the output whenever you add a disk at other times for the same reason.

If you attempt to add a robust partition that was not properly shut down because of a system halt and, thus, has a damaged file system, PRIMOS displays this message:

```
*** Robust Partition diskname has not been properly shutdown.
*** Fast Fix_disk has to be run before it can be added.
```

You should run fast FIX_DISK in this case. However, if it is necessary to access the data immediately, you can use the -FORCE option of ADDISK to access the data for read-only operations. See Chapter 7 for additional details.

How Will You Know Whether There Are Problems With a Disk?

Occasionally, users may receive the following error messages, which indicate problems with a disk:

```
Pointer mismatch found.   (not the same as POINTER_FAULT$)
The directory is damaged.
Directory too large.
Bad DAM file.
Bad truncate of segment directory.
Segment directory error.
The file is too long.
Too many subdirectory levels.
```

The following error message at the supervisor terminal also indicates a disk problem:

```
Disk format does not support this revision of PRIMOS.
```

In addition, messages indicating disk read-errors and write-errors may appear at the supervisor terminal or in the system event log file. For example:

```
*** Message from Prime product LOG DISK, user GEORGE on ENPUB2
   (Severity information, logged at 29 NOV 91 14:27:08
DISK READ ERROR DEVICE NUMBER = 003460 (OCT)
(4005 CONTROLLER 0 UNIT 0)
CRA = 00000010356 (OCT) RCRA = 00000000000 (OCT)
CYLINDER = 34 HEAD = 5 RECORD = 5 (9 SECTORS PER TRACK)
STATUS (ACCUM) = 120011 (OCT) STATUS (LAST) = 120011 (OCT) RETRIES = 10
(UNCORRECTED)
```

When such error messages appear, it may be desirable to run FIX_DISK on the disk on which the error occurred. However, the error message `The directory is damaged` may appear when a user references several different directories or, particularly, different partitions on the same disk drive. In this case, have a representative from PrimeService check the drive, disk, and controller before you run FIX_DISK. This check ensures that the problem is not in the hardware.

What to Do Before Running FIX_DISK

You run FIX_DISK to repair a defective file structure. There must be no users on the partition when you repair it. In this case, follow the procedures described in Chapter 4 to shut down the partition, warning users that the partition you will repair is to be shut down. Then use the disk repair procedure, which is illustrated in Figure 6-1 and explained in the next section.

SCSI Disk Support

PRIMOS at Rev. 23.2 and later supports several new SCSI disk drives and also supports the ICOP+ protocol for enhanced performance on SCSI disks. Do not use any options related to badspot handling on these disks because badspots are handled in a different manner. See Chapter 5 for a discussion of SCSI disks.

The Head Zero Partition of Disks Connected to a Model 6580 (IDC1) Disk Controller

The head zero partition (the partition containing surface 0, or head 0) of a Rev. 21.0 or later physical disk, or spindle, connected to a Model 6580 (IDC1) disk controller defines how all the partitions on the spindle are to be treated by the disk controller with respect to Dynamic Badspot Handling. See Chapter 5 and Chapter 8 for a more thorough discussion.

The following disks are capable of having Dynamic Badspot Handling occurring on them when they are connected to a Model 6580 (IDC1) disk controller.

SMD (300MB and 80MB)

68MB	MODEL_4475
158MB	MODEL_4735
160MB	MODEL_4845
600MB	MODEL_4860

The Procedure for Running FIX_DISK

If the partition you are repairing is the command device, follow the left-hand path in 6-1: log out all users, stop the Login server by using the STOP_LSR command, and stop DSM by using the STOP_DSM command. (See the discussion of the –COMDEV option on page 6-20, Repair or Use the Command Device, for details.) Then use FIX_DISK with the –COMDEV option and PRIMOS automatically performs all the procedures illustrated on the right-hand path in Figure 6-1 from shutting the disk down to starting it up again. However, it may be desirable to cold start the system when you repair the command device and FIX_DISK finishes so that all system servers are automatically restarted and any other servers and phantoms are started by inclusion of the appropriate commands in your PRIMOS.COMI file.

If you are not repairing the command device, perform the procedures shown in the right-hand path of Figure 6-1 at the supervisor terminal: shut down and assign the partition; run FIX_DISK; unassign and start the partition. Note that you can invoke the SHUTDN, DISKS, and ADDISK commands only from the supervisor terminal but you can run FIX_DISK at a user terminal.

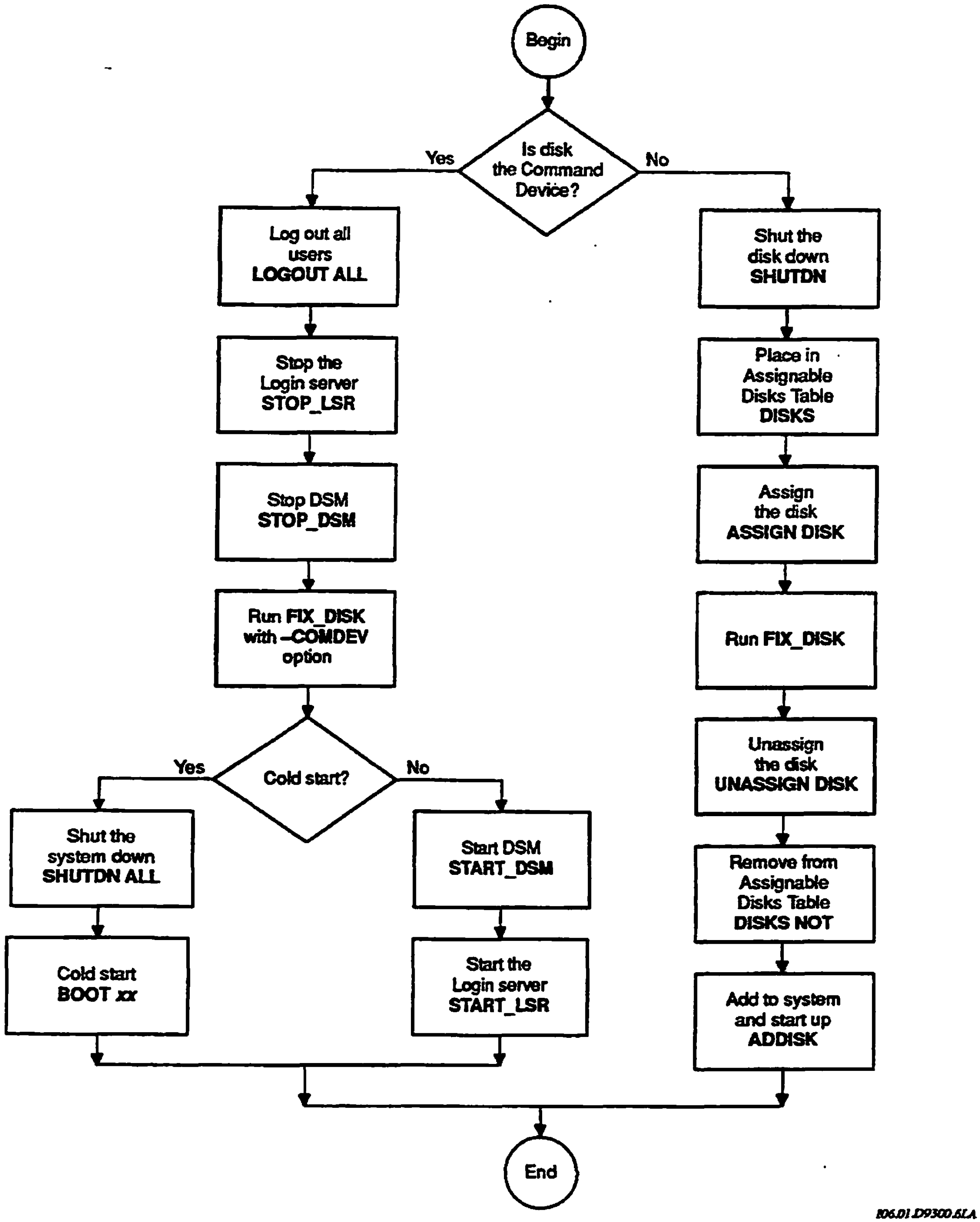


Figure 6-1. Disk Repair Using FIX_DISK

Options Available for Running FIX_DISK

FIX_DISK resides in CMDNC0 as FIX_DISK.SAVE. You invoke FIX_DISK with this command format:

FIX_DISK –DISK *pdev* [*options*]

pdev is the physical device number of the partition you are repairing. See Chapter 3 for directions on constructing physical device numbers. If you are going to repair a partition that is presently running, use the STATUS DISKS command to list the pdevs of all your partitions and to determine the pdev of the partition you want to repair.

Ten categories of command-line options are used with FIX_DISK:

- Options that specify repair tasks and revision conversions
- Options that select capabilities of disk controllers
- An option that checks to see that the system was properly shut down
- An option that specifies the command device as the target
- Options for use with pre-Rev. 22.1 formats
- An option that displays the dynamic badspot (DBS) file
- Options that modify terminal display
- Options that set the size of CAM file extents
- Options that change the method of file record allocation
- An option that gives you online help

Deciding Which Options to Use

Because SCSI disks associated with the Model 7210 disk controller (SDIC) downloaded with ICOP+ software handle badspots within the disk subsystem, no badspot file or dynamic badspot file exists on these disks. Therefore, FIX_DISK ignores the –DBS OFF and –DBS ON options and you should not use any options related to badspot handling on these disks. In addition, sectoring on these disks is forward with an interleave factor of 1 and this cannot be changed, thus FIX_DISK ignores the –SECTOR (–ODI and –RDI) options.

To summarize, do not use the following options on these SCSI disks:

–ADD_BADSPOT	–DBS
–DUMP_DBS	–LIST_BADSPOT
–SECTOR	

A summary of each command-line option is listed below and the options are discussed in detail on the following pages in the same order. Study these items and write down the options that you wish to use. You *must* specify either –DISK pdev or –DISK pdev –COMMAND_DEVICE on the command line. The FIX_DISK options are also summarized in Appendix F of this book for your convenience.

–DISK

Specifies the physical device number (pdev) of the partition. See page 6-14.

–FIX –DUFE –SUFE –UFD_COMPRESSION –NUMBER_OF_RETRIES

Specifies the extent of the repair operation. See page 6-15.

–ADD_BADSPOT

Adds new badspots to the badspot file. See page 6-17.

–DBS OFF (–AC) –DBS ON (–IC)

Selects either Nondynamic Badspot Handling or Dynamic Badspot Handling mode on disks that support these modes. See page 6-17.

–AUTO_TRUNCATION –MAX_NESTED_LEVEL

Truncates deeply nested directories. See page 6-19.

–CHECK

Checks to see if the partition needs to be repaired. See page 6-20.

–COMDEV

Repairs or use the command device. See page 6-20.

–CONVERT_19 –CONVERT_20 –CONVERT_21

Converts partitions created under older revisions forward to Rev. 19.0 from pre-Rev. 19.0, to Rev. 20.0 from pre-Rev. 20.0, and to Rev. 21.0 from pre-Rev. 21.0. See page 6-22.

–CONVERT_22.1

Converts Rev. 22.0 partitions to Rev. 22.1 format standard partitions. See page 6-23.

–DISK_TYPE

Specifies the physical disk type. See page 6-24.

–DUMP_DBS

Displays the DBS file. See page 6-25.

–FAST

Rapidly repairs a partition. See page 6-26.

-HELP

Gets online help with a display of FIX_DISK options. See page 6-27.

-INTERACTIVE

Repairs a defective or missing DSKRAT. See page 6-27.

-LEVEL -LIST_FILE

Specifies the amount of terminal output. See page 6-28.

-LIST_BADSPOTS

Lists badspots from the static badspot file, BADSPT. See page 6-29.

-MAX_EXTENT_SIZE -MIN_EXTENT_SIZE

Sets the maximum and minimum extent sizes for CAM files. See page 6-30.

-NO_QUOTA

Does not convert a pre-Rev. 19.0 partition. See page 6-30.

-SECTOR FORWARD (-ODI) REVERSE (-RDI)

Specifies the method of file record allocation. See page 6-30.

-TRUNCATE

Specifies the disposition of files containing badspots. See page 6-31.

The following options are obsolete but are supported. Their replacements are shown in the right column.

Obsolete Options	*Replacement*
-INTELLIGENT_CONTROLLER (-IC)	-DBS ON
-ALL_CONTROLLER (-AC)	-DBS OFF
-OVERRIDE_DEFAULT_INTERLEAVE (-ODI)	-SECTOR FORWARD
-RESTORE_DEFAULT_INTERLEAVE (-RDI	-SECTOR REVERSE

After studying the above options, invoke FIX_DISK with the list of options you wish to use. FIX_DISK performs its functions and then returns to PRIMOS. As FIX_DISK executes, it displays messages describing its progress and any problems it encounters.

Specify the Physical Device Number: You must use **-DISK** to specify the physical device number (pdev) of the partition on which FIX_DISK is to operate. (See Chapter 3 for information on determining physical device numbers or use the STATUS DISKS command to see the pdev if the partition is presently running.) Follow -DISK with the physical device number. -DISK and the physical device number are required *must* be specified on the command line in this format:

```
OK, FIX_DISK -DISK 61260
```

Unless you specify –COMMAND_DEVICE (–COMDEV, as explained on page 6-20), FIX_DISK responds as if you have already shut down and assigned the partition you are repairing. If you have not assigned the partition, FIX_DISK displays a message to that effect and aborts:

```
DISK pdev NOT ASSIGNED
ER!
```

If you do not include the pdev, FIX_DISK aborts with this message:

```
Bad Physical Device Number.  (cl_par)
FIX_DISK aborted
ER!
```

Caution Do not forget to include –DISK *preceding* the pdev on the command line. If the –DISK option is inadvertently omitted or entered after the pdev, unpredictable behavior occurs. If this happens, type Ctrl–P to stop the program. If you have assigned only one disk (the one to be repaired) to your terminal, there is little chance that any harm has occurred. (The chances are greater if you have assigned two or more disks.) Issue the command RELEASE_LEVEL –ALL (RLS –ALL), then invoke FIX_DISK again. Include –DISK on the command line in the correct place, followed immediately by the pdev. It is not necessary to use –DISK with the –HELP option if you only want help.

Specify the Extent of the Repair Operation: If you do not want FIX_DISK to attempt repairs, do not specify the –FIX option. FIX_DISK then reports inconsistencies, but does not attempt to rectify them. Messages from FIX_DISK concerning bad file pointers, record addresses, and file structures indicate that files or directories may be deleted or truncated if you do specify –FIX.

Caution Do not use the –FIX option if you suspect that the disk drive itself is faulty. First run FIX_DISK without using the –FIX option. If no unrecoverable disk read-errors or write-errors are reported at the supervisor terminal, run FIX_DISK again, using the –FIX option. If FIX_DISK encounters unrecoverable disk read-errors or write-errors and you used the –FIX option, FIX_DISK may delete or truncate files.

If you do want FIX_DISK to make repairs or changes, you must include the –FIX option on the command line. You probably want to include the –UFD_COMPRESSION option (abbreviation –CMPR) and the option –DUFE (delete unknown file entries) or –SUFE (save unknown file entries) as well. For example:

```
OK, FIX_DISK -DISK 61260 -FIX -UFD_COMPRESSION -DUFE
```

This command line tells FIX_DISK to

- Make all needed repairs (–FIX)
- Compress unused space in directory records (–UFD_COMPRESSION)
- Delete all unrecognizable file entries (–DUFE)

You cannot use both –SUFE and –DUFE on the same command line; if you do, FIX_DISK aborts with this error message:

```
Both -DUFE and -SUFE can not be specified at the same time.
```

Caution If you run a version of FIX_DISK older than the revision of the partition you are repairing (not recommended), use the –SUFE option because there may be new file types on the more recent revision of the file system that are unrecognizable to the older version of FIX_DISK. The –SUFE option results in saving any unrecognizable file entry. If you do not use the –SUFE option, the default is –DUFE (delete unknown file entries). Be sure that you are running Rev. 23.3 FIX_DISK.

Note that the default is –DUFE (delete unknown file entries). Be sure that your partition is recently backed up before running FIX_DISK and that you are using Rev. 23.3 FIX_DISK.

You can specify the number of times that FIX_DISK is to attempt to read a record with the **–NUMBER_OF_RETRIES** option (abbreviation **–NUMRTY**). FIX_DISK normally makes nine physical adjustments to attempt to read a record and does this two times; that is, the default is two retries. You can change the default with the –NUMRTY option. A practical maximum may be four retries. With an intelligent disk controller, FIX_DISK displays one error message for each retry. With a nonintelligent disk controller, FIX_DISK displays nine error messages with each retry.

You must specify the –FIX option if you want FIX_DISK to make changes to the partition. An error message is displayed and FIX_DISK aborts if you do not use –FIX with the following options.

–ADD_BADSPOT	–DBS (–IC, –AC)
–CONVERT_19	–MAX_EXTENT_SIZE
–CONVERT_20	–MIN_EXTENT_SIZE
–CONVERT_21	–SECTOR
–CONVERT_22.1	–INTERACTIVE
–UFD_COMPRESSION	

If you do not specify –FIX with other options, FIX_DISK reports any errors encountered but does not attempt to repair the errors.

Add New Badspots to the Badspot File: Use the –ADD_BADSPOT option (abbreviation –ADBADS) to add new badspots to the static badspot file (BADSPT) if the partition is in –DBS OFF mode or to the dynamic badspot (DBS) file if the partition is in –DBS ON mode. Use a command line like this:

FIX_DISK –DISK *pdev* –FIX –ADD_BADSPOT *record_1 . . . record_16*

Enter record numbers (*record_1 . . .*) in octal. A maximum of 16 badspots may be entered on one command line with the –ADD_BADSPOT option. If you attempt to enter more than 16 badspots on one command line or if you enter a number that is not octal, an error message is displayed and FIX_DISK aborts. The –FIX option must be specified with the –ADBADS option. You must use the –ADBADS option to add uncorrectable errors that occur on paging partitions.

The badspot file is updated by FIX_DISK after you enter badspots with the –ADBADS option. When you cold start the system or when you add the partition to the system with the ADDISK command, the DSKRAT is changed appropriately so that PRIMOS does not use the records marked as badspots in the case of a –DBS OFF mode partition or PRIMOS is directed to use RMA records in the case of a –DBS ON mode partition.

Set the Partition's Disk Controller Mode: In order to have Dynamic Badspot Handling or mirroring on Rev. 21.0 and later partitions, the disk drive on which the partition's spindle is located must be connected to an intelligent disk controller (IDC1) and the spindle (all partitions on the spindle) must be in Dynamic Badspot Handling (–IC) mode.

Note For a complete discussion of Dynamic Badspot Handling, see Chapter 8. For a discussion of mirroring and the mirroring commands, see Chapter 9.

To switch the controller mode of a Rev. 21.0 or later partition on a spindle that supports Dynamic Badspot Handling, use the options –DBS ON (–IC) and –DBS OFF (–AC). For example, to switch the mode of a Rev. 22.1 format partition from Nondynamic Badspot Handling (–DBS OFF) to Dynamic Badspot Handling (–DBS ON) so that the partition is capable of being mirrored, use a command line like this:

```
OK, FIX_DISK -DISK 7462 -DT MODEL_4845 -FIX -DBS ON
```

In order to switch the DBS mode from OFF to ON, the spindle must be capable of Dynamic Badspot Handling and must be connected to a downloaded Model 6580 intelligent disk controller (IDC1). In addition, all partitions on the spindle must be in the same DBS mode.

The following disk types are capable of Dynamic Badspot Handling:

SMD (300 and 80MB)	
68MB	MODEL_4475
158MB	MODEL_4735
160MB	MODEL_4845
600MB	MODEL_4860

In order to switch the mode of any partition, the head zero partition of that spindle must be in the mode to which you wish to switch the other partitions. If it is not, an error message is displayed and FIX_DISK aborts:

```
OK, FIX_DISK -DISK 172462 -DT 600MB -FIX -DBS ON
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 06/28/92.  Time: 15:32.
Partition name is SWTPEA

First partition must be in same mode as the conversion on this partition
FIX_DISK aborted
ER!
```

To switch the mode of a Rev. 21.0 or later partition to allow the partition to be used with nonintelligent disk controllers (4005), use the –DBS OFF argument. The partition can then be used on a disk drive connected to a nonintelligent disk controller or an intelligent disk controller in 4005 mode but it cannot be mirrored unless it is connected to an intelligent disk controller and is in Dynamic Badspot Handling (–DBS ON) mode. Use a command line like this to select Nondynamic Badspot Handling (–DBS OFF) mode:

```
OK, FIX_DISK -DISK 141020 -FIX -DBS OFF
```

When you use the FIX_DISK –DBS ON argument to switch a Rev. 22.1 format partition to Dynamic Badspot Handling (–DBS ON) mode, the head zero partition on that spindle must be in –DBS ON mode and the head zero partition must be assigned to you. This is because remapped records must be retrieved by FIX_DISK from the head zero partition without interference from another process. The only way to do this is to have the head zero partition shut down and assigned. If you are switching or using the command device, use the –COMDEV option (see page 6-20).

Note If the head zero partition is the command device (COMDEV), use the –COMDEV *pdev* option to have FIX_DISK shut it down and assign it to you. (Remember to stop and restart DSM, the Login server, and any other phantoms.) In addition, if you are switching a partition other than the head zero partition, the head zero partition must be in the same mode to which you are switching the other partition. Also, you cannot switch from Nondynamic Badspot Handling (–DBS OFF) mode to Dynamic Badspot Handling (–DBS ON) mode unless the disk is connected to a downloaded intelligent disk controller (IDC1). FIX_DISK leaves a Rev. 22.1 format partition in its current mode unless you specifically instruct FIX_DISK to select either –DBS OFF mode or –DBS ON mode.

Truncate Deeply Nested Directories: FIX_DISK may encounter directories that are nested very deeply (more than 99 levels deep). Normally when this happens, FIX_DISK aborts with the following error message, allowing you to consult with the owner of the offending directory tree and take appropriate action.

```
Ufd nesting exceeds maximum specified.
```

However, if you want FIX_DISK to automatically delete such directories at the 99th level, include the –AUTO_TRUNCATION option (abbreviation –AT) on the command line, as follows:

```
OK, FIX_DISK -DISK 61260 -FIX -AUTO_TRUNCATION
```

In addition, you may change the allowable maximum number of nested levels by including the –MAX_NESTED_LEVEL *n* option (abbreviation –MAX) on the command line. *n* is a decimal number from 0 to 3855, inclusive. If FIX_DISK encounters a directory nesting greater than the decimal *n* value you specify, FIX_DISK aborts, or, if –AUTO_TRUNCATION was specified, FIX_DISK deletes the directories that exceed the maximum nesting level. For example:

```
OK, FIX_DISK -DISK 61260 -FIX -MAX_NESTED_LEVEL 20
```

Here, the specified maximum nested level is 20 and FIX_DISK aborts if directories are nested deeper than that level and you did not include the –AUTO_TRUNCATION option. If you also include the –AUTO_TRUNCATION option, FIX_DISK deletes directories nested deeper than that level. If no maximum level *n* is specified with the –MAX_NESTED_LEVEL option or if you do not use the –MAX_NESTED_LEVEL option, the maximum level defaults to 99. If you specify an invalid number for *n* (that is, outside the range of 0 to 3855), this error message is displayed:

```
Invalid max nested level, set to max of 3855
```

Check to See If the Partition Needs to Be Repaired: The -CHECK option allows you to determine if the partition has been shut down improperly (for example, due to a system crash). The -CHECK option causes FIX_DISK to examine two bits in the DSKRAT. One of these bits is set to 0 when the partition is added to the system by the ADDISK command and is set to 1 when the partition is properly shut down with the SHUTDN command. The other bit is set to 0 if the partition is re-added without running FIX_DISK on it after having been shut down improperly. If FIX_DISK finds that either or both of these bits are 0, a message is displayed warning you to run FIX_DISK. For example:

```
OK, FIX_DISK -DISK 44063 -CHECK
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 03/24/89.  Time: 14:44.
Partition name is BOOKS

This is a revision 22.1 partition.
Disk uses reverse sectoring with -DBS ON.
Disk not properly shutdown.  Full fix disk will be run.
FIX_DISK finished
ER!
```

If you suspect that the partition was not shut down properly, you can specify the -FIX option with the -CHECK option to make the necessary repairs.

Repair or Use the Command Device (COMDEV): If the specified disk is the command device (logical device 0 as indicated by a STATUS DISKS command), you *must* include the -COMMAND_DEVICE option (abbreviation -COMDEV) on the command line, as follows:

```
OK, FIX_DISK -DISK 1060 -COMMAND_DEVICE -FIX
```

Including this option instructs FIX_DISK to automatically perform the steps shown in the right-hand path of Figure 6-1 on page 6-11: shut down the command device, assign it, repair it, unassign it, and start it up again. In addition, FIX_DISK restores the event-logging state on pre-Rev. 21.0 systems and reestablishes any priority ACL that may have been set on the command device. FIX_DISK does this because shutting down the command device disables all event logging and removes any priority ACL set on the command device.

For Rev. 21.0 and later systems, DSM does the logging; therefore, you must stop DSM before running FIX_DISK on the command device or on any partition that contains the top-level directory DSM* or on any partition that contains the DSM logging directories. Restart DSM when FIX_DISK finishes to restart the event logging mechanism. While FIX_DISK is operating on the command device or on the partition containing DSM* or the logging directories, normal event logging does not take place but event messages appear at the supervisor terminal. If DSM* is on a partition other than the command device, messages are logged in the undelivered log file DSM*>LOGS>UMH>UNDELIVERED.LOG when FIX_DISK operates on the command device.

When using the –COMMAND_DEVICE option, you *must* run FIX_DISK from the supervisor terminal because partitions can be shut down and started only from the supervisor terminal. In addition, you should log out all users and phantoms before running FIX_DISK because they rely on the availability of the command device.

Before using FIX_DISK with the –COMDEV option, stop the Login server by using the STOP_LSR command and stop DSM by using the STOP_DSM command. Also stop any other system servers and phantoms that depend on the availability of the command device. When the command device or any partition is shut down, all files on that partition are closed. In the case of the command device, this includes the run files for EPFs. Thus, when a process attempts to access memory, it fails because the EPF is unmapped.

Note An error can occur while you are running FIX_DISK on the command device causing FIX_DISK to stop. If this happens, the command device is now shut down. Recover by adding the command device and re-initializing your command environment by using the ICE command. Then restart FIX_DISK by reentering your command line.

When FIX_DISK finishes and the partition restarts, issue the START_LSR command to restart the Login server and issue the START_DSM command to restart DSM and event logging. Also restart any other system phantoms. However, if you shut down the system when FIX_DISK finishes and then cold start, the Login server and other system servers are automatically restarted and DSM is also automatically restarted if you have the START_DSM command in your PRIMOS.COMI file. Cold starting the system is recommended after running FIX_DISK on the command device.

There are two reasons for using the optional *pdev* argument with the –COMDEV option:

1. If all three of the following are true:

   - You are converting a partition to Dynamic Badspot Handling (–DBS ON) mode or Nondynamic Badspot Handling (–DBS OFF) mode or you are converting a partition from a pre-Rev. 21.0 format to Rev. 21.0.
   - The partition you are converting is not the head zero partition.
   - The head zero partition is the command device (COMDEV).

2. If the head zero partition is the command device and you want to display the DBS file while repairing another partition on the same spindle.

The following example command line illustrates the first reason.

Assume you are switching modes of the second partition (pdev = 32060) of a disk and the head zero partition is the COMDEV (pdev = 1460). Use a command line like this:

```
OK, FIX_DISK -DISK 32060 -FIX -DBS ON -COMDEV 1460
```

Using the –COMDEV option causes the command device to be shut down and assigned to you and prevents access to the file system on the command device by other user processes while FIX_DISK is making the conversion. Use of the –COMDEV option also allows FIX_DISK necessary access to the DBS file and the RMA. When FIX_DISK finishes, you must restart system servers and phantoms or cold start the system, as noted above.

Convert the Revision of a Partition: You cannot use FIX_DISK to convert partitions to Rev. 22.0 or from pre-Rev. 22.0 to Rev. 22.1; you must use MAKE to convert existing pre-Rev. 22.0 partitions to Rev. 22.1. (See Converting Partitions in Chapter 5 and the discussion earlier in this chapter on page 6-4 under Converting a Pre-Rev. 21.0 Partition.) You can convert Rev. 22.0 partitions to Rev. 22.1 standard partitions with FIX_DISK; see page 6-23.

To convert a pre-Rev. 21.0 partition to Rev. 21.0, use the command-line option –CONVERT_21. You must use the –FIX option with any of the convert options (–CONVERT_19, –CONVERT_20, –CONVERT_21). If you use more than one of the convert options on the same command line, FIX_DISK displays an error message and aborts.

Caution In order to have FIX_DISK convert a partition to Rev. 21.0 format, the head zero partition (the partition that contains surface 0) on a spindle that supports Dynamic Badspot Handling must be converted before any other partition on that spindle can be converted. This is necessary because the DBS file and the RMA must be on the head zero partition and FIX_DISK must have access to them in order to repair other partitions on the spindle. If you convert the head zero partition, you must convert all partitions on the spindle.

To convert partitions other than the head zero partition on a spindle that supports DBS to Rev. 21.0, you must shut down and assign the head zero partition of that spindle to yourself. You must shut down the head zero partition and assign it because FIX_DISK must have access to the DBS file and the RMA in order to convert the other partitions and the DBS file and the RMA are on the head zero partition. To convert to Rev.21.0, use a command line like this:

```
OK, FIX_DISK -DISK 2062 -FIX -CONVERT_21 -DISK_TYPE MODEL_4711
```

The pdev used here (2062) indicates that this is the first partition so it is already shut down and assigned to you in order to run FIX_DISK. Note that you must specify the –DISK_TYPE option with the –CONVERT_21 option. (See page 6-24.) You should also specify the mode of the partition, either Dynamic Badspot Handling (–DBS ON) or Nondynamic Badspot Handling (–DBS OFF), if the spindle supports DBS. (See page 6-18 for the list of disk types that support DBS.)

If you do not specify –DBS ON or OFF, FIX_DISK determines the type of disk controller the disk is associated with (intelligent (IDC1) or nonintelligent) and sets the mode accordingly.

To convert a disk made as a pre-Rev. 20.0 partition to a Rev. 20.0 partition, include the –CONVERT_20 and –FIX options on the command line, as follows:

```
OK, FIX_DISK -DISK 1060 -FIX -CONVERT_20
```

If a BADSPT file exists, it is converted to the Rev. 20.0 format. All quota information is initialized, warning and error messages related to quotas are not displayed, and a new revision stamp is created. Existing directories are not hashed, however. (See the discussion on conversions on page 6-4.) The –FIX option must be used with the –CONVERT_20 option.

To convert a pre-Rev. 19.0 partition to a Rev. 19.0 partition, use the –CONVERT_19 and –FIX options. You cannot use the –CONVERT_22.1, –CONVERT_21, –CONVERT_20, and –CONVERT_19 options on the same command line.

Note It is recommended that you use MAKE to convert the format of pre-Rev. 20.0 partitions to Rev. 20.0 or to Rev. 21.0 and pre-Rev. 22.0 partitions to Rev. 22.0 or to Rev. 22.1. This is because of directory and file attributes introduced starting at Rev. 20.0. You must use MAKE to convert any partition to a Rev. 22.1 format partition that will become a robust partition. See Chapter 7 for details.

Convert a Rev. 22.0 Partition to Rev. 22.1: You can convert a Rev. 22.0 partition to a Rev. 22.1 format standard partition. A Rev. 22.1 partition can contain CAM files with an unlimited number of extents. The option –CONVERT_22.1 allows you to convert a Rev. 22.0 partition to Rev. 22.1 format by updating the revision stamp in the DSKRAT of the partition. Earlier revisions than Rev. 22.0 cannot be converted to Rev. 22.1 with FIX_DISK; you must use MAKE to convert earlier revision partitions to Rev. 22.1 by creating them as Rev. 22.1 format partitions. In addition, if you want to create Rev. 22.1 format robust partitions, you must have the MAKE_ROBUST utility (see Chapter 7) and you must create the partition in Rev. 22.1 format with MAKE.

You must use the –FIX option with –CONVERT_22.1 to convert the partition. It is not necessary to convert the head zero partition first, but it is good practice to do so and to convert all partitions on a spindle to the same revision. Also, it is not necessary to have the head zero partition assigned to you or to use the –DISK_TYPE option because all the conversion does is update the revision stamp in the partition's DSKRAT.

Prior to Rev. 22.1, a CAM file could have a maximum of 340 extents, which should be more than sufficient for a well-organized disk partition. However, this limit of 340 extents can be reached on fragmented partitions and with large databases. (See Chapter 7 for a discussion of fragmentation of partitions.) When the limit is reached, the operation that is extending the file is stopped and the file must be copied or the partition remade to reduce the number of extents in the file.

A CAM file performs best when it has a minimal number of extents and the size of each extent is as large as possible. For example, a file of a single extent of five physical records performs better than a file of five extents containing one physical record each. Thus, it is possible for an already poorly allocated CAM file to become even less optimal by increasing the number of extents allowable in the file. You should monitor large CAM files with the LEM command. If the number of extents is growing large and the LCB command indicates sufficient contiguous space is available to reduce the number of extents, you should copy the file with the COPY command, delete the original file, and change the name of the copy back to the original filename. If sufficient contiguous space is not available on the partition, you can save the contents of the partition to magnetic tape with a logical backup utility (such as MAGSAV), recreate the partition by using MAKE, and restore the files with a complimentary logical utility (such as MAGRST).

Specify the Physical Disk Type: You must use the –DISK_TYPE option (abbreviation –DT) with the –CONVERT_21 option. This is necessary because FIX_DISK must create the DBS file and the RMA on the head zero partition of a spindle that supports Dynamic Badspot Handling. (See page 6-18.) The amount of disk space FIX_DISK uses for these files depends on the disk type.

If you do not specify the –DISK_TYPE option with –CONVERT_21, FIX_DISK displays this message and aborts:

```
For a Rev 21 conversion, the -disk_type model MUST be specified.
```

If you specify –DISK_TYPE but do not specify the type or specify an invalid type, FIX_DISK displays a list of valid types and aborts. For example:

```
OK, FIX_DISK -DISK 102062 -FIX -CONVERT_21 -DISK_TYPE
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date:03/24/92.  Time: 14:48.
Invalid disk model name; Valid disk types are:

        SMD                 80 or 300 MB removable
        CMD                 Cartridge module device
        68MB                68 megabyte fixed media
        158MB               158 megabyte fixed media
        160MB               160 megabyte fixed media
        600MB               600 megabyte fixed media
        MODEL_4475          300 megabyte fixed media
```

```
          MODEL_4714          84 megabyte fixed media
          MODEL_4711          60 megabyte fixed media
          MODEL_4715          120 megabyte fixed media
          MODEL_4735          496 megabyte fixed media
          MODEL_4719          258 megabyte fixed media
          MODEL_4845          770 megabyte fixed media
          MODEL_4721          328 megabyte fixed media
          MODEL_4860          817 megabyte fixed media
          MODEL_4729          673 MB fixed media
          MODEL_4730          215 MB fixed media
          MODEL_4731          421 MB SCSI fixed media
          MODEL_4732          1.34 GB SCSI fixed media

FIX_DISK aborted
ER!
```

Select a valid type from the left column above and reenter the command line, specifying the disk type with the –DISK_TYPE option.

Display the DBS File: You may wish to display the dynamic badspot (DBS) file and save the display in a COMO file. This may be useful if the DBS file is somehow damaged. You can then use FIX_DISK or MAKE to manually reenter the badspots.

To display the DBS file, use the FIX_DISK option **–DUMP_DBS** (abbreviation **–DDBS**). The format of the display is shown here; a detailed display is shown in the section Examples of Running FIX_DISK, later in this chapter.

```
OK, FIX_DISK -DISK pdev -DUMP_DBS

DBS file version version_number.
DBS file has number_of_records records.
Number of badspots = number_of_badspots.
File last modified by Primos/controller.

Badspot                       Remap Record
Cyl,  Head,  Sector           Cyl,  Head,  Sector
 -----------------             -----------------
a,     b,     c                 r,     s,     t
d,     e,     f                 u,     v,     w
```

pdev in the command line is the pdev of the head zero partition on which the DBS file is located. If the file has not been modified, the line `File last modified by` is not displayed. The fields in the other lines are filled in appropriately. The letters a, b, c, and so on represent cylinder, head, and sector numbers of the badspots and their remapped records.

In order to display the DBS file of a spindle, you must have the head zero partition of that spindle assigned to you. If the first partition is the command device, use the –COMDEV *pdev* option in a command line like the following, where *pdev* is the pdev of the command device, in this case 460:

```
OK, FIX_DISK -DISK 10460 -DUMP_DBS -COMDEV 460
```

Use of this option will shut down the command device and assign it to you as explained under the discussion of the –COMDEV option on page 6-20. Remember to stop and start DSM, the Login server, and any other phantoms when you use the –COMDEV option or cold start the system when FIX_DISK finishes.

You can have the DBS file displayed while you are repairing some other partition on the disk. The DBS file is displayed when FIX_DISK finishes the repair operation. To display the DBS file without specifying the –FIX option, use the pdev of the head zero partition where the DBS file resides. If the disk is in Nondynamic Badspot Handling (–DBS OFF) mode, the DBS file is displayed but it may not be up-to-date; it is updated in Dynamic Badspot Handling (–DBS ON) mode.

Rapidly Repair a Partition: If the partition you wish to repair with FIX_DISK is a robust partition (described in Chapter 7), you can use the –FAST option to speed up processing by FIX_DISK.

Running FIX_DISK with the –FAST and –FIX options is referred to as fast FIX_DISK and running FIX_DISK with the –FIX option but without the –FAST option is referred to as full FIX_DISK.

The purpose of the –FAST option is to save time in repairing a robust partition that has experienced a system halt and was thus improperly shut down. Specifying the –FAST option tells FIX_DISK to check directory entries, including CAM file extent maps, available space (the DSKRAT), and the quota system on robust partitions. When you specify the –FAST option, FIX_DISK uses the extent maps to determine which records are in use in order to rebuild this information if necessary. Since FIX_DISK does not read each record header in this case, it is much faster than a full FIX_DISK. Thus, if it is more important to access data as soon as possible after a halt and have the partition available, use fast FIX_DISK. You may use full FIX_DISK to ensure the integrity of all record headers at a more convenient time.

The –FAST option is not applicable to SAM files, since, by definition, SAM files are built sequentially and FIX_DISK must access them by reading every record of a SAM file in sequence. Thus, no time is saved.

You can use fast FIX_DISK on a standard partition on which the quota system may be damaged but no other problems exist; fast FIX_DISK checks and repairs the quota system in this case. For example, when running FIX_DISK without the –FIX option, you may get a message from FIX_DISK indicating that the quota system may be incorrect. If the standard partition was improperly shut down and you use the –FAST option, FIX_DISK ignores the –FAST option and full FIX_DISK is run.

You can use the –FAST option without the –FIX option to see if it is necessary to repair the partition. However, in order to repair a partition, you must use the –FIX option as shown here:

```
OK, FIX_DISK -DISK 100461 -FIX -FAST
```

Caution

Since using the –FAST option results in FIX_DISK checking only the file structures noted above, the integrity of all record headers in these files is not checked. Thus, some potential problems may not be detected unless you run full FIX_DISK; that is, without the –FAST option.

It is recommended that you always run full FIX_DISK (without the –FAST option) on standard (nonrobust) partitions with the one exception related to the quota system noted above. See Table 6-1 on page 6-6 or Table 7-1 for complete recommendations on running Fast FIX_DISK.

Get Online Help With a Display of FIX_DISK Options: To list the FIX_DISK options with a brief explanation of each option, use this command:

```
OK, FIX_DISK -HELP
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 05/24/92.  Time: 14:29.
USAGE: FIX_DISK -DISK pdev [options]
OPTIONS:
```

The options and a synopsis of each is then displayed. You may use this command without assigning any disks and without the –DISK option and you may use it from any terminal if you want only help.

A summary of the FIX_DISK command-line options appears in Appendix F, for your convenience, and in the *Operator's Guide to System Commands.*

Repair a Defective or Missing DSKRAT: If you have already attempted to run FIX_DISK, it may have displayed one of the following error messages and aborted:

```
The file structure of DSKRAT is bad.

The number of heads is different.
It should be YY is XX

The physical record size is different.
It should be YY is XX

The DSKRAT header has the wrong length.
It should be YY is XX

The partition cannot be handled by this version of
FIX_DISK.
```

If one of these messages appears, rerun FIX_DISK and specify the options –INTERACTIVE (abbreviation –INT) and –FIX on the command line, as follows.

```
OK, FIX_DISK -DISK 61260 -FIX -INTERACTIVE
```

These options cause FIX_DISK to ask you questions (which you can answer either YES or NO) when it discovers a defective or missing DSKRAT, rather than to abort. The answers you supply enable FIX_DISK to construct a correct DSKRAT. The –FIX option must be used with the –INTERACTIVE option. See Example of Reconstructing the DSKRAT on page 6-40 in the examples section of this chapter.

Specify the Amount of Terminal Output: Normally, while FIX_DISK is running, it displays the name of the top-level directory being processed, as follows:

```
BEGIN CLEOPATRA
```

Then, when it finishes that top-level directory, it displays

```
END CLEOPATRA recs
```

Here, *recs* is the total number of 2048-byte records in use in the directory.

FIX_DISK starts the output with `BEGIN MFD` and ends the output with `END MFD` *recs*, because all top-level directories have the MFD as their parent directory. Because listing all of the top-level directories on the disk may take time, you may wish to disable this feature. To disable it, include the –LEVEL option on the command line with a decimal argument of 0, as follows:

```
OK, FIX_DISK -DISK 61260 -FIX -LEVEL 0
```

When you use this option as shown, FIX_DISK displays the BEGIN MFD and END MFD messages, but does not display any other BEGIN or END messages.

If you do not specify –LEVEL on the command line or if you specify –LEVEL without specifying a numerical value, the default is –LEVEL 1. Level one means to output only top-level directories in the MFD via `BEGIN` and `END` messages. Specifying a level value higher than 1 causes `BEGIN` and `END` messages to be displayed for top-level directories, their subdirectories, and so on, to the specified level. (To assist you in understanding the directory structure on a disk, FIX_DISK indents the `BEGIN` and `END` messages by two spaces for each level. (See the examples at the end of this chapter.)

Note Do not confuse the -LEVEL option with the -MAX_NESTED_LEVEL option. -LEVEL affects how much information is displayed at your terminal, but does not otherwise affect FIX_DISK. -MAX_NESTED_LEVEL, followed by a decimal number, specifies the maximum allowable nested level of directories (see page 6-19).

If you wish FIX_DISK to display all the filenames of the files in all of the directories to the level you specify or to the default level, include the -LIST_FILE option (abbreviation -LF) on the command line. The filenames are included under the names of their parent directories. (See the example on page 6-47.)

List Badspots: Normally, the only information FIX_DISK prints concerning badspots is the number of lost records they represent. If you would like to see more information on badspots on an -DBS OFF mode partition or a partition that was formerly an -DBS OFF mode partition and that has a BADSPT file, include the -LIST_BADSPOTS option (abbreviation -LB) on the command line, as follows:

```
OK, FIX_DISK -DISK 1060 -LIST_BADSPOTS
```

FIX_DISK then lists all known badspots from the BADSPT file, followed by a listing of any equivalence blocks, in a display before the `BEGIN MFD` message. For each known badspot, the track, head, and sector are listed. For each equivalence block, the record numbers of both the badspot and the remapped (alternate) record are listed. In addition, the track, head, and sector numbers are also listed for the badspots and their equivalence blocks. For example:

```
OK, FIX_DISK -DISK 4422 -LIST_BADSPOTS
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 06/01/92.  Time: 16:48.
Partition name is TEMP1

This is a revision 22 partition.
Disk uses reverse sectoring with -DBS OFF.
The BADSPT file has 38 entries.

Track = 39 Head = 11 Sector = 5
Track = 79 Head = 6 Sector = 5
Track = 146 Head = 9 Sector = 7
Track = 160 Head = 23 Sector = 8
  .  .  .
  .  .  .
  .  .  .
```

For disks having many badspots, listing of all of the badspots may take a considerable amount of time, particularly at a low-speed supervisor terminal.

Set or Change the Maximum and Minimum Extent Sizes for CAM Files: At Rev. 22.0, you can set the maximum and minimum extent sizes for CAM files. An extent size is the number of contiguous 2048-byte records in one extent of the CAM file. (See Chapter 7 for a discussion of setting extent sizes.) FIX_DISK has two options to allow you to change or set the extent sizes: the –MAX_EXTENT_SIZE option (abbreviation –MAXSIZ) and the –MIN_EXTENT_SIZE option (abbreviation –MINSIZ).

You can initially set these sizes when you create the partition with MAKE by using similar MAKE options. If you do not use those MAKE options, PRIMOS uses the default number of records shown in Table 6-2 to set the extent sizes, depending on whether the partition is a robust partition or a standard partition.

Table 6-2. Default Maximum and Minimum Extent Sizes

	Robust Partition	*Standard Partition*
Maximum	256	32
Minimum	64	16

If you decide to change these extent sizes after the partition is created, use the FIX_DISK options in this format:

FIX_DISK –DISK *pdev* –MAXSIZ *size* –MINSIZ *size* –FIX

size is a decimal number representing the minimum or maximum extent size (number of contiguous records in an extent) to be used by the algorithm that PRIMOS uses to extend CAM files. The minimum size cannot be zero (0) and cannot be greater than the maximum. The maximum size can be any value up to 32767.

Do Not Convert a Pre-Rev. 19.0 Partition: To run FIX_DISK on a partition made as a pre-Rev. 19.0 partition without converting it, include the –NO_QUOTA option (abbreviation –NQ) on the command line:

```
OK, FIX_DISK -DISK 1060 -FIX -NO_QUOTA
```

Disabling of quota checking is necessary because pre-Rev. 19.0 partitions have no quota information, and error messages are displayed if FIX_DISK is run on a pre-Rev. 19.0 partition without the –NO_QUOTA option.

Set the Record Allocation Direction and Interleave Factor: The recommended method of file record allocation for Rev. 20.0 and later standard partitions depends on the combination of CPU and disk controller in your system. The interleave factor is 3 with forward sectoring for all pre-Rev. 20.0 partitions and for Rev. 22.1 robust partitions. (See Chapter 10 for an explanation of sectoring and interleave factors.) Reverse sectoring with an interleave factor of 1 is supported starting at PRIMOS Rev. 20.0.

Table 10-2 presents the recommended record allocation schemes for standard partitions based on the combination of the type of disk controller and the type of CPU in your system. Basically, the method is forward sectoring with an interleave factor of 3 for systems with a nonintelligent disk controller and a CPU in the 9950 class, and reverse sectoring with an interleave factor of 1 for all other combinations of CPU and intelligent disk controller (IDC1). MAKE and FIX_DISK can determine the CPU and controller combination and set the allocation method accordingly. For all SCSI disks connected to the Model 7210 controller, the method is forward sectoring with an interleave factor of 1 and you cannot change it.

You can change these recommended methods on standard partitions by using the **–SECTOR** option (abbreviation –SEC) with the appropriate argument. To set the interleave factor to 3 with forward sectoring on a standard partition, use the **FORWARD** argument (abbreviation **FOR**) as in this command line:

```
OK, FIX_DISK -DISK 11061 -FIX -SECTOR FORWARD
```

Use the **REVERSE** argument (abbreviation REV) in a command line like this to set the interleave factor to 1 with reverse sectoring:

```
OK, FIX_DISK -DISK 11061 -FIX -SECTOR REVERSE
```

You cannot use –SECTOR on a robust partition; if you do, FIX_DISK displays the message `You may not change the default interleaving on a robust partition.` and ignores the option. The interleave factor remains 3 with forward sectoring on the robust partition. You also cannot use –SECTOR on SCSI disks.

Specify the Disposition of Files Containing Badspots: When FIX_DISK finds a record that it cannot read, it creates a null record (a record filled with zeros) on a good portion of the disk and appends the remaining records of the original file to the null record. If more than one sequential record is missing and the file is a SAM file, then FIX_DISK creates two null records and displays an error message to indicate that the correct number of records is not known. If you prefer to have FIX_DISK truncate a file when it encounters a badspot in any file or when it encounters an uninitialized record in a CAM file on a robust partition, include the **–TRUNCATE** option (abbreviation **–TRU**) as follows:

```
OK, FIX_DISK -DISK 61260 -FIX -TRUNCATE
```

When a file is truncated, the part of the file that is located at and beyond the file pointer is eliminated from the file. If the file pointer is positioned at the beginning of the file, all of the information in the file is removed, but the filename remains in the file directory. If you intend to truncate a file, you should have a recent backup copy of that file to restore for the file owner.

What to Do After Running FIX_DISK

After FIX_DISK has finished, you should get the repaired partition back into service by adding, or starting up, the partition. To do this, follow the steps shown on the right side of Figure 6-1 on page 6-11. If you are using FIX_DISK as part of a disk backup, continue with the system preparation procedure as described in the *Operator's Guide to Data Backup and Recovery* after you run FIX_DISK.

Note If you ran FIX_DISK on the command device by using the –COMDEV option and the supervisor terminal origin directory is a password directory, it is advisable to cold start the system. Until the system has been cold started, the supervisor terminal has no origin directory. You can use the ORIGIN –SET command to attach the supervisor terminal process to CMDNC0 or to its origin directory if it is different from CMDNC0. For example, to set the supervisor terminal process origin directory to CMDNC0, use the command ORIGIN –SET CMDNC0.

Examples of Running FIX_DISK

This section presents some examples of running FIX_DISK to point out uses and purposes of various options and to demonstrate typical displays produced by FIX_DISK. In cases where only the command line is shown, it is assumed that the disk has been shut down, placed in the Assignable Disks Table, and assigned to the user from the supervisor terminal, as shown in Figure 6-1.

Example of Normal FIX_DISK Display

In the example below, the partition BEEBLE is specified by its pdev (22660), 0 levels of directories are to be displayed (–LEVEL), directories are to be compressed (–UFD_COMPRESSION), and necessary disk modifications are to be made (–FIX).

```
OK, SHUTDOWN 22660
OK, DISKS 22660
OK, ASSIGN DISK 22660
OK, FIX_DISK -DISK 22660 -LEVEL 0 -UFD_COMPRESSION -FIX
FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 04/05/92.  Time: 02:37.
Partition name is BEEBLE

This is a revision 22.1 partition.
Disk uses reverse sectoring with -DBS ON.
BEGIN MFD
END MFD 29404
```

```
74070 records in partition
29404 records used
    0 records lost
44666 records left
    1 records compressed
DSKRAT OK
FIX_DISK finished
OK, UNASSIGN DISK 22660
OK, DISKS NOT 22660
OK, ADDISK 22660
Starting up revision 22.1 partition "BEEBLE".
OK,
```

FIX_DISK identifies the partition name; determines the PRIMOS revision, the mode (–DBS ON) and the method of record allocation (reverse) of the partition; summarizes crucial data concerning record usage; checks the DSKRAT and finds it to be proper; and signals the completion of its operation. No errors are indicated. No compression of directories takes place. The line `records lost` refers to record loss caused by badspots; it does not indicate file truncation or record compression.

Example of Error Handling by FIX_DISK

It is quite possible that FIX_DISK may find problems on the partition it examines. Such problems are indicated by error messages. For a complete listing of FIX_DISK error messages, see Appendix B.

The next two examples illustrate how FIX_DISK handles errors. In the first example, 0 levels of directories are printed.

```
OK, FIX_DISK -DISK 1060 -LEVEL 0 -UFD_COMPRESSION -FIX
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 04/05/92.  Time: 03:42.
Partition name is ZAPHOD

This is a revision 22 partition.
Disk uses reverse sectoring with -DBS ON.
BEGIN MFD
ACL at word 513 does not point at a file or access category!
ACL is deleted!
ACL at word 4044 does not point at a file or access category!
ACL is deleted!
Access Category BOOKS.ACAT does not reference an ACL!
Access Category is deleted!
File CMDNC0 does not reference an ACL or Access Category!
Changed to default ACL pointer
File CHAPTER does not reference an ACL or Access Category!
Changed to default ACL pointer
File LATE does not reference an ACL or Access Category!
```

```
Changed to default ACL pointer
File PAYOFF does not reference an ACL or Access Category!
Changed to default ACL pointer
MFD
The Directory Used count is bad.  It should be 67 instead of 71.
MFD>PAYOFF
END MFD 29404

74070 records in partition
29404 records used
    0 records lost
44666 records left
    1 records compressed
DSKRAT UPDATED!
FIX_DISK finished
OK,
```

In the above example, partition 1060 is identified as ZAPHOD; two ACLs in the MFD are deleted; the access category BOOKS.ACAT is deleted; four files are changed to the default ACL; the Directory Used count in PAYOFF is found to be inconsistent and is corrected; one record is freed by compression of directories; the DSKRAT is updated; and operation is returned to PRIMOS.

Example of Conversion and Displaying Directories

In the following example, FIX_DISK explores the entire disk and prints to file system level 9 (the limit specified by –LEVEL). The use of the option –CONVERT_21 indicates that this partition is a pre-Rev. 21.0 partition and FIX_DISK is to convert it to a Rev. 21.0 partition. The first partition on this disk is assigned to the user process because FIX_DISK must have access to the DBS file and the RMA. In order to do the conversion, you must specify the –DISK_TYPE option. FIX_DISK needs to know the disk type in order to properly size the DBS file and the RMA. Since the method of record allocation is not specified with the –SECTOR option, and the partition originally is a Rev. 20.0 partition with forward record allocation and an interleave factor of 3, it retains that method of record allocation.

```
OK, ASSIGN DISK 1660
OK, ASSIGN DISK 61660
OK, FIX_DISK -DISK 30462 -LEVEL 9 -CONVERT_21 -CMPR -FIX
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 05/10/92.  Time: 16:25.
For a Rev 21 conversion, the -disk_type model MUST be specified.
FIX_DISK aborted
ER! FIX_DISK -DISK 30462 -LEVEL 9 -CONVERT_21 -CMPR -FIX -DT SMD

[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 05/10/92.  Time: 16:26.
```

```
Partition name is UNICRN

This is a revision 21 partition.
Disk uses reverse sectoring with -DBS OFF.
The quota system may be incorrect.
The BADSPT file has 89 entries.
BEGIN MFD
  BEGIN CMDNC0
    BEGIN RUN
    END RUN 8
    BEGIN SYS.SIGNOFF.SEG
    END SYS.SIGNOFF.SEG 14
  END CMDNC0 3536
  BEGIN OEDPUS
The father pointer is bad.
It should be 5332 is 5335
Bad record address = 53340 BRA = 5340 Father = 5332 Type = 0
File is deleted!
MFD>OEDPUS>ATE>0

Zero record added to file.
MFD>OEDPUS>ATE

The Directory Used count is bad.  It should be 14 instead of 147.
MFD>OEDPUS

  END OEDPUS 85
  BEGIN OSPREY
  END OSPREY 1001
  BEGIN COEPH
  END COEPH 8907
  BEGIN CLNNOS
    BEGIN HYBRIS
      BEGIN CTHSYS
      END CTHSYS 16
    END HYBRIS 99
  END CLNNOS 650
END MFD 13527

59724 records in partition
13527 records used
    6 records lost
46191 records left
    1 records compressed
DSKRAT UPDATED!
FIX_DISK finished
OK,
```

In the above example, FIX_DISK tries to examine the MFD and all directories. Terminal output is set to level 9. FIX_DISK finds an error in the top-level directory OEDPUS and corrects it by adding a record of zeros to the file; the correction is reported; a record is freed by compressing a directory, thus removing empty space on the partition; record information is displayed. The DSKRAT is updated, indicating that the disk was converted to a Rev. 21.0 partition.

Example of Using the –COMDEV Option

On a single-partition system, running FIX_DISK with the –COMMAND_DEVICE (–COMDEV) option results in output similar to the following. (Before issuing the FIX_DISK command line, use the STOP_DSM command to stop DSM and use the STOP_LSR command to stop the Login server. For information on these commands, see the *Operator's Guide to System Commands*.)

```
OK, MAXUSR 0
OK, STOP_DSM
[STOP_DSM Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
[2 Apr 92 15:48:54 Sunday]
DSM shutdown in progress.
Warning: SYSTEM_MANAGER terminated.  Event logging will not take place!
OK, STOP_LSR
Really? YES
Phantom 97: Normal logout at 15:49
Time used: 241h 26m connect, 00m 25s CPU, 00m 30s I/O
OK, FIX_DISK -DISK 460 -UFD_COMPRESSION -FIX -COMDEV -DUFE -LIST_BADSPOTS
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 04/02/92.  Time: 15:52.
Partition name is TOPDOG

This is a revision 22.1 partition.
Disk uses reverse sectoring with -DBS OFF.
Disk not properly shutdown.  Full fix_disk will be run.
The quota system may be incorrect.
The badspot section has 2 entries.

Track = 0 Head = 6 Sector = 7
Track = 0 Head = 6 Sector = 0

BEGIN MFD
ACL at word 733 does not point at a file or access category!
ACL is deleted!
File SPOOLQ does not reference an ACL or Access Category!
Changed to default ACL pointer
  BEGIN SAD
  END SAD 19
  BEGIN CMDNC0
  END CMDNC0 2337
  BEGIN DOS
```

```
  END DOS 21
  BEGIN TOOLS
  END TOOLS 69
  BEGIN SYSCOM
  END SYSCOM 164
  BEGIN LIBRARIES*
  END LIBRARIES* 1297
  BEGIN HELP*
  END HELP* 279
  BEGIN BACKUP*
  END BACKUP* 707
  BEGIN SYSOVL
  END SYSOVL 151
  BEGIN BATCHQ
  END BATCHQ 78
  BEGIN PRIMENET*
  END PRIMENET* 67
  BEGIN SYSTEM
  END SYSTEM 1083
  BEGIN EMACS*
  END EMACS* 1275
  BEGIN LOGREC*
  END LOGREC* 74
  BEGIN DOWN_LINE_LOAD*
  END DOWN_LINE_LOAD* 180
  BEGIN SEGRUN*
  END SEGRUN* 252
  BEGIN DIRECV
  END DIRECV 33
  BEGIN SPOOLQ
  END SPOOLQ 130
END MFD 8216

14814 records in partition
 8216 records used
    0 records lost
 6598 records left
    0 records compressed
DSKRAT OK
FIX_DISK finished
Starting up revision 22 partition "TOPDOG".
OK, START_LSR
OK, START_DSM

[START_DSM Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
[2 Apr 89 16:28:45 Sunday]
DSM initialization started.
OK,
*** DSMSR (user 99 on ENPUB) at 16:29
DSM is now in a steady state.
```

When you use the –COMDEV option, the partition is automatically shut down and assigned at the start and automatically unassigned and restarted at the conclusion of FIX_DISK. The partition is identified as a Rev. 23.3 partition; the existing badspots are listed; the directories are listed to the default level with the total records used in each; the disposition of all records is displayed at the end; and the DSKRAT is found to be in order. When FIX_DISK finishes, the partition automatically restarts because the –COMDEV option is used.

The START_LSR command is issued next to restart the Login server and the START_DSM command to start DSM and the event logging process. Any other system servers should also be started. If you cold start the system when FIX_DISK finishes, the Login server and other system servers are automatically restarted and DSM is automatically restarted if the START_DSM command is in the PRIMOS.COMI file.

Example of Running FIX_DISK Without –FIX

In the following example, FIX_DISK is run with the –LIST_BADSPOTS option but without the –FIX option to list the badspots and any errors on the partition. This indicates what the problems are on the partition and the user can decide if full FIX_DISK should be run. FIX_DISK is then run with the –ADD_BADSPOT and –FIX options, to add the new badspot and repair the partition.

```
OK, SHUTDN DISK 1066
OK, DISKS 1066
OK, ASSIGN DISK 1066
OK, FIX_DISK -DISK 1066 -LIST_BADSPOTS
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 04/05/92.  Time: 06:58.
Partition name is GOLD

This is a revision 22 partition.
Disk uses forward sectoring with -DBS OFF.
The badspot section has 1 entries.

Track = 290 Head = 2 Sector = 3 Bad record = '24335.

BEGIN MFD
  BEGIN CMDNC0
  END CMDNC0 1234
  BEGIN DOS
  END DOS 112
END MFD 1353

29628 records in partition
 1353 records used
    1 records lost
```

```
28274 records left
    0 records compressed
DSKRAT MISMATCH!
FIX_DISK finished
OK, FIX_DISK -DISK 1066 -ADD_BADSPOT 22222 -FIX
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 04/05/92.  Time: 07:04.
Partition name is GOLD

This is a revision 22 partition.
Disk uses forward sectoring with -DBS OFF.
The badspot section has 1 entries.

BEGIN MFD
  BEGIN CMDNC0
  END CMDNC0 1234
  BEGIN DOS
  END DOS 112
Processing add badspot request for '22222
Added to BADSPT file!
END MFD 1353

29628 records in partition
 1353 records used
    2 records lost
28273 records left
    0 records compressed
DSKRAT UPDATED!
FIX_DISK finished

OK, UNASSIGN DISK 1066
OK, DISKS NOT 1066
OK, ADDISK 1066
OK,
```

In the preceding example, FIX_DISK first lists the badspot 24335_8, which already exists in the file BADSPT. It checks the DSKRAT file and finds that it does not agree with the record allocation information generated by FIX_DISK. FIX_DISK is then invoked again, with the –FIX and –ADD_BADSPOT options specified. On the second pass, the badspot information and the DSKRAT file are updated.

Example of Record Truncation

FIX_DISK is invoked with the –TRUNCATE and –NUMBER_OF_RETRIES options in this example. FIX_DISK truncates any bad records, but tries four times (rather than the default of two) to read the bad records. A problem is found in a file, the file is truncated, and the DSKRAT is updated.

```
OK, ASSIGN DISK 1066
OK, FIX_DISK -DISK 1066 -TRUNCATE -NUMBER_OF_RETRIES 4 -FIX
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 04/05/92.  Time: 08:12.
Partition name is GOLD

This is a revision 22 partition.
Disk uses forward sectoring with -DBS ON.
The badspot section has 2 entries.

BEGIN MFD
2 files point to the same record!
bad record address = 24335
BRA = 24044 Father = 1
File truncated.
MFD>J
The Directory Used count is bad.  It should be 11367 instead of 12101.
MFD
END MFD 11367

14814 records in partition
11367 records used
    1 records lost
 3446 records left
    0 records compressed
DSKRAT UPDATED!
FIX_DISK finished
OK,
```

Example of Reconstructing the DSKRAT

The following example demonstrates the use of the –INTERACTIVE option to repair a damaged DSKRAT. First the user runs FIX_DISK, attempts to display the DBS file, and finds out that the partition is corrupted. The next step is to run FIX_DISK with the –FIX and –INTERACTIVE options to repair the DSKRAT. FIX_DISK prompts for information necessary to reconstruct the DSKRAT, for the correct disk type, and for the correct revision for the partition. (See Appendix B, FIX_DISK Messages, for a complete description of the prompts.) FIX_DISK displays error messages as it processes the file system and corrects the errors.

```
OK, FIX_DISK -DISK 462 -DUMP_DBS
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 02/09/92.  Time: 08:54.
Partition name is RASCAL

The DSKRAT header has wrong length.  (RAT_CK)
The number of heads is different.
```

```
The physical record size is different.  (RAT_CK)
This is a revision 18 partition.
FIX_DISK aborted
ER! FIX_DISK -DISK 462 -FIX -INTERACTIVE
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 02/09/92.  Time: 08:55.
Partition name is RASCAL

The DSKRAT header has wrong length.  (RAT_CK)
The number of heads is different.
The physical record size is different.  (RAT_CK)
This is a revision 18 partition.
Fix DSKRAT header? YES
Physical Disk? 462

Valid disk model names:
                SMD              80 or 300 MB removable
                CMD              Cartridge module device
                68MB              68 MB fixed media
                158MB            158 MB fixed media
                160MB            160 MB fixed media
                600MB            600 MB fixed media
                MODEL_4475       300 MB fixed media
                MODEL_4714        84 MB fixed media
                MODEL_4711        60 MB fixed media
                MODEL_4715       120 MB fixed media
                MODEL_4735       496 MB fixed media
                MODEL_4719       258 MB fixed media
                MODEL_4845       770 MB fixed media
                MODEL_4721       328 MB fixed media
                MODEL_4860       817 MB fixed media
                MODEL_4729       673 MB fixed media
                MODEL_4730       215 MB fixed media
                MODEL_4731       421 MB fixed media
                MODEL_4732       1.34 GB fixed media
Please enter model name of disk...  SMD
Split Disk? NO
DISK  FILE-RECORDS PAGING-RECORDS (DECIMAL)
462    14814         0
Parameter OK? YES
This must be a rev 20, 21, 22. or 22.1 partition.
Is this a rev 22.1 partition? NO
Is this a rev 22 partition? YES
Is this a robust partition? NO
Forward Sectoring ? NO
Reverse Sectoring  (default)? <Return>
Please answer "YES" or "NO"? YES
Intelligent controller mode  (default)? YES
Partition not shutdown properly during the previous session; please run
fix_disk!
The BADSPT file has 11 entries.
```

```
BEGIN MFD
Bad file type: special bit not set!
The Beginning Record Address (BRA) pointer (0), is bad, it points to
a record that belongs to another file.
System file is bad - ignored!
The current record address (CRA) is bad.
  It should be 2 is 6366.
  Bad record address = 2  BRA = 2    Father = 1    Type = 0
MFD>BOOT
  BEGIN FALCON
  END   FALCON  57
  BEGIN RAVEN
  END   RAVEN   60
  END   RAVEN   60
  BEGIN PETREL
  END   PETREL  404
  BEGIN GANNET
  END   GANNET  4699
Unable to move record, rewriting to attempt fix it up.
END   MFD  5224
14814 records in partition
 5224 records used
    1 records lost
 9590 records left
    0 records compressed
DSKRAT UPDATED!
FIX_DISK finished
OK,
```

Example of Checking for Proper Shutdown of a Partition

In this example, you run FIX_DISK on the command device with the –CHECK option. First, stop DSM and the Login server. You should also shut down any other servers or phantoms that may need to use the command device. You must also use the –COMDEV option, which automatically shuts down and assigns the partition before FIX_DISK is run and then unassigns and starts the partition when FIX_DISK finishes. Messages concerning the quota system and the necessity of running FIX_DISK are displayed, but the partition is not repaired because the –FIX option is not used.

Run FIX_DISK again with the –FIX option to repair the command device. When FIX_DISK finishes, DSM, the Login server, and any other phantoms must be started or the system may be shut down and then cold started.

```
OK, FIX_DISK -DISK 1260 -COMDEV -CHECK
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 04/01/92.  Time: 18:03.
Partition name is TOPDOG
```

```
This is a revision 23.3 partition.
Disk uses forward sectoring with -DBS ON.
The quota system may be incorrect.
FIX_DISK finished.
Starting up revision 22.1 partition "TOPDOG"
 *** Disk "TOPDOG" was not shutdown properly, Run FIX_DISK. ***

OK, FIX_DISK -DISK 1260 -COMDEV -FIX
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 04/01/92.  Time: 18:30.
Partition name is TOPDOG

This is a revision 22.1 partition.
Disk uses forward sectoring with -DBS OFF.
The quota system may be incorrect.
The badspot section has 2 entries.

BEGIN MFD
  BEGIN CMDNC0
  END CMDNC0 2354
  BEGIN PRIRUN
  END PRIRUN 1659
  BEGIN BACKUP*
  END BACKUP* 379
  BEGIN DOWN_LINE_LOAD*
  END DOWN_LINE_LOAD* 100
END MFD 4453
29628 records in partition
 4453 records used
    2 records lost
25173 records left
    0 records compressed
DSKRAT OK
FIX_DISK finished
Starting up revision 22 partition "TOPDOG"
OK,
```

Example of Changing Controller Modes

In this example, the second partition of a physical disk is switched from Nondynamic Badspot Handling (–DBS OFF) mode to Dynamic Badspot Handling mode by using the –DBS ON option.

```
OK, DISKS 10460
OK, ASSIGN DISK 10460
OK, FIX_DISK -DISK 10460 -FIX -LEVEL 0 -DBS ON
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 04/01/92.  Time: 18:35.
```

```
Partition name is TEST2

The first partition needs to be assigned for this conversion
FIX_DISK aborted
ER! ASSIGN DISK 460
OK, FIX_DISK -DISK 10460 -FIX -LEVEL 0 -DBS ON
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 04/01/92.  Time: 18:38.
Partition name is TEST2

This is a revision 22.1 partition.
First partition must be in the same mode as the conversion on this partition
FIX_DISK aborted

ER! FIX_DISK -DISK 460 -FIX -LEVEL 0 -DBS ON
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 04/01/92.  Time: 18:40.
Partition name is SYSA29

This is a revision 22.1 partition
Disk uses reverse sectoring with -DBS ON.
The DBS file has 0 entries

BEGIN MFD
END MFD 1119

14814 records in partition
 1119 records used
    0 records lost
13695 records left
    0 records compressed
DSKRAT UPDATED!
FIX_DISK finished

OK, FIX_DISK -DISK 10460 -FIX -LEVEL 0 -DBS ON
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 04/01/92.  Time: 18:42.
Partition name is TEST2

This is a revision 22.1 partition
Disk uses reverse sectoring with -DBS ON.
The DBS file has 0 entries

BEGIN MFD
END MFD 246
14814 records in partition
  246 records used
    0 records lost
14568 records left
    0 records compressed
DSKRAT UPDATED!
```

```
FIX_DISK finished
OK, UNASSIGN DISK 460
OK, UNASSIGN DISK 10460
OK, DISKS NOT 460 10460
OK, ADDISK 460 10460
OK,
```

First the disk is assigned to the user and FIX_DISK is invoked. The first partition on this physical disk was not assigned to the user and FIX_DISK aborts. The first partition must be available to FIX_DISK in order to have the DBS file available. The first partition is then assigned and FIX_DISK is invoked again. This time the mode switching cannot take place because the first partition must be in the same mode to which the user is switching the second partition. The user then switches the mode of the first partition and then that of the second partition. When FIX_DISK finishes, the user unassigns the two partitions and adds them to the system. These partitions can be added, or started, on this system because they are associated with an intelligent disk controller.

Example of the DBS File Display

The following example shows the display of the DBS file when the –DUMP_DBS option is used. In this case, the DBS file is on the first partition of a file system disk that is not the command device; therefore it is not necessary to use the –COMDEV option. The display is generated during the normal course of repairing the partition. the first partition must be assigned to you or FIX_DISK displays the message `Cannot access RAT on disk 0` at the end of its normal display.

```
OK, ASSIGN DISK (2460 12460)
OK, FIX DISK -DISK 12460 -DUMP DBS -FIX -LEVEL 0
[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 04/03/92.  Time: 18:42.
Partition name is TPLAB

This is a revision 22.1 partition
Disk uses forward sectoring with -DBS OFF.
The DBS file has 5 entries.

The BADSPT file has 5 entries.

BEGIN MFD
END   MFD  2345
74070 records in partition
 2345 records used
   18 records lost
71707 records left
    0 records compressed
```

```
DSKRAT UPDATED!

DBS file version 1
DBS file has 1 record
Number of badspots = 5
File last modified by PRIMOS.

Badspot                 Remap Record
Cyl, Head, Sector      Cyl, Head, Sector
713,  7,  0           0,  0,  8
778,  1,  4           0,  1,  8
806,  6,  7           0,  1,  1
807,  1,  1           0,  1,  2
810,  1,  3           0,  1,  3
FIX_DISK finished
OK,
```

The user specifies the –LEVEL 0 option to suppress display of directories under the MFD. Use of the –FIX option causes the record allocation information in the DSKRAT to be corrected. The DBS file is listed when FIX_DISK finishes repair tasks; the file contains information for five badspots. It was last updated by PRIMOS since the disk is associated with a nonintelligent disk controller. The badspots and their remapped records are listed by cylinder, head, and sector addresses.

Invoking FIX_DISK From Magnetic Tape

If the command device or the CMDNC0 directory becomes damaged so that the FIX_DISK utility is inaccessible, you can invoke FIX_DISK from magnetic tape using the MTRESUME command. Remember to assign the tape drive first. MTRESUME is fully documented in the *Operator's Guide to System Commands.*

When you invoke FIX_DISK from magnetic tape, use the MTRESUME option –COMMAND_LINE_OPTIONS (abbreviation –CMDOPT) to specify to FIX_DISK the details of the repair. For example, to repair a damaged command device while deleting all unknown file entries and compressing directories, use a command line like this:

```
OK, MTRESUME MT1 FIX_DISK.SAVE -CMDOPT -DISK 1060 -COMDEV -FIX -DUFE -CMPR
```

The output then appears as follows:

```
This is a revision 20 MAGSAV tape.
Date: 04-04-88
Revision: 20
Reel: 1
Name: REPAIR

[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 06/05/92.  Time: 02:37
          .  .  .
          .  .  .
          .  .  .
```

In this case, FIX_DISK.SAVE is run from magnetic tape. After the dialog from the tape portion of the procedure, the FIX_DISK messages appear as they normally do. The –COMDEV option shuts down and assigns the partition.

Note When you boot PRIMOS from tape and then invoke FIX_DISK with the MTRESUME command, you cannot repair a disk in dynamic badspot handling (–DBS ON) mode. This is because intelligent disk controllers are downloaded from disk when PRIMOS is booted from disk. Therefore you must use the –DBS OFF option if you are repairing a –DBS ON mode partition. You can convert back to –DBS ON mode after you repair the partition and reboot PRIMOS from disk.

If you

- Use MTRESUME and FIX_DISK in the above manner and you boot PRIMOS from tape to do this and
- Have a CPU that does not have a diagnostic processor or that has a VCP III or earlier virtual control panel

you must also use the PRIMOS SETIME command to set the date and time before invoking FIX_DISK.

Example of Invoking FIX_DISK.SAVE From Tape

FIX_DISK may be invoked from magnetic tape by using the MTRESUME command and specifying the pathname of FIX_DISK as it was saved by MAGSAV on the tape. In this example, FIX_DISK is invoked from magnetic tape on tape drive MT0 while PRIMOS is running. FIX_DISK is instructed to repair the partition, compress directories, delete unknown file entries, and display directories down to the tenth level and list all files in those directories.

```
OK, MTRESUME MT0 FIX_DISK.SAVE -CMDOPT -DISK 30662 -FIX -DUFE -CMPR -LEVEL
-LIST_FILE
```

```
This is a revision 20 MAGSAV tape.
Date:     04-04-88
Revision: 22
Reel:     1
Name:     REPAIR

[FIX_DISK Rev. 23.3 Copyright (c) 1992, Prime Computer, Inc.]
Date: 06/05/92.  Time: 16:49.
Partition name is SWTPEA
This is a revision 22 partition.
Disk uses forward sectoring with -DBS ON.
BEGIN MFD
        SWTPEA
        BOOT
        DAISY
        MAYFLOWER
  BEGIN FLOWER
          DAISY
    BEGIN IRISES
            PETALS
      BEGIN ROSES
              PISTILS
        BEGIN PEONY
                SEPALS
          BEGIN TULIP
                  YELLOW
                  MAGENTA
                  FUCHSIA
            BEGIN CFOILS
                    YELLOW
            END   CFOILS  340
          END   TULIP  1680
        END   PEONY  3361
      END   ROSES  5384
    END   IRISES  7740
  END   FLOWER  10517
END   MFD  10592


14814 records in partition
10592 records used
    0 records lost
 4222 records left
    0 records compressed
DSKRAT OK
FIX_DISK finished
OK,
```

The FIX_DISK display lists all of the files in directories down to the subdirectory CFOILS and lists the records used in each directory.

7 Robust Partitions

This chapter discusses the following topics:

- Understanding robust partitions and what they provide
- Understanding the –FAST option of FIX_DISK
- Understanding the robust partition file system
- Evaluating the use of robust partitions
- Creating robust partitions
- Administering robust partitions
- Using MAKE, FIX_DISK, ADDISK, and mirroring in conjunction with robust partitions

Before you decide to convert your disk partitions to the robust format, please take a few minutes to read this chapter carefully. Then answer the questions in the section Evaluating the Use of Robust Partitions and fill out the Robust Partitions Evaluation Form.

What Is a Robust Partition?

A **robust partition** is a new type of disk partition at Rev. 22.1. Robust partitions reduce the time that it takes to recover from a system halt. All files and segment directory subfiles on a robust partition are physically stored as CAM files. The CAM file structure allows the –FAST option of FIX_DISK to quickly check the extent map and verify the physical structure of the CAM file. This same capability is not available on a **standard** (nonrobust) partition. (See Chapter 1 and Figures 1-2, 1-3, and 1-4 for descriptions of file types.)

Another major advantage of robust partitions is that PRIMOS advises you whenever the result of a system halt requires you to run FIX_DISK on a partition. PRIMOS cannot require you to run FIX_DISK on a standard partition after a system halt nor can FIX_DISK indicate when it should be run except in the case of an incorrect quota system.

What Robust Partitions Can Provide

Robust partitions offer several advantages that can significantly reduce the length of time that is required for you to resume normal operations after a system halt. Some of these advantages derive from the robust partition structure. A few of the advantages are based upon the inherent characteristics of CAM files. The purpose of this subsection is to explain the nature of the advantages that robust partitions offer.

Advantages: Advantages of using robust partitions include

- System availability is improved because some halts do not require FIX_DISK to be run and others require only fast FIX_DISK (FIX_DISK –FAST) in place of full FIX_DISK.
- PRIMOS tells you whether or not you must run FIX_DISK on the partition when you use the ADDISK command. This saves you the time of running FIX_DISK unnecessarily.
- Robust partitions can improve upon your ability to resume operations after some system halts.
- File deletions and truncations are faster since it is necessary to read only extent maps rather than every data record.
- Writing out full records using the PRIMOS subroutine PRWF$$ is 50% faster.
- Robust partitions offer a faster record access mechanism for some environments.
- Robust partitions offer the most advantage when you have large files or segment directories with large subfiles.

Because the design of robust partitions specifically improves the ability to recover from a system halt, the disk format is less likely to suffer from some types of directory corruption that can occur on a standard partition. Because of the file system structure implemented on a robust partition, fast FIX_DISK can verify the integrity of the user directories. This can greatly reduce the length of time that is required to run FIX_DISK. As a result, you can quickly check the directory structure.

Logical File Types: Robust partitions introduce a new concept called logical file typing. All files stored on a robust partition are physically stored as CAM files. Although you might open a file with a logical file type of SAM, PRIMOS physically creates the file as a CAM file. This is transparent to all higher levels of software and allows you to move existing applications to a robust partition without modification. This logical-to-physical mapping also allows PRIMOS to more tightly control the file structure on a robust partition, without changing the logical appearance of that file structure.

Because every file and every segment directory subfile on a robust partition is physically stored as a CAM file, there is less likelihood that a file will be damaged by a corrupt record header chain. Since CAM file data records are not chained through the record headers, corruption of a data record header does not cause the remainder of the file to be lost. Also, the extent map mechanism means that fast FIX_DISK is able to detect file structure corruption very quickly by checking the extent map.

Record Errors: The introduction of robust partitions offers a new method of responding to a corrupted data record. On a standard disk partition, a pointer mismatch (e$ptrm) error occurs if the record header chaining is corrupt. This error is fatal to the application and can be corrected only by running FIX_DISK. This same error can occur on a robust partition, but PRIMOS reports it as an uninitialized block (e$zero) and re-initializes the data record header, filling the data record with nulls. Although, there is now a null data record, the file can still be accessed without requiring you to run FIX_DISK to correct the error. If the application detects this error, it can take its own corrective action, which may include a data-management rollback procedure to correct the data integrity of the database. (Prime DBMS, Prime® ORACLE®, MIDASPLUS™, and PRISAM™ all treat the uninitialized block as a fatal error; the application fails and returns to PRIMOS.)

Record Access: Robust partitions also offer a faster record access mechanism for some environments. Typically, a large CAM file provides faster data access than a large DAM file. This is noticeable when you have multiple users accessing the same file simultaneously and when the file is larger than 512 disk records (1 megabyte). This faster access can be an advantage if your application does not already use CAM files.

File Deletion: Deleting a large file is always significantly faster on a robust partition than on a standard partition. Two files cannot claim the same data record on a robust partition. On a standard partition, PRIMOS must verify that all of the records within the file actually belong to the file. Verification is not necessary on a robust partition.

Restrictions on the Use of Robust Partitions

There are a few restrictions on when you can use a robust partition.

Shutdowns: Because the ADDISK command checks a robust partition, you must run FIX_DISK if the partition was not cleanly shut down. This can be inconvenient if you do not regularly run FIX_DISK after every system halt. Forcing you to run FIX_DISK in this case, however, provides better assurance of file structure integrity.

In order to reduce the time necessary to recover from a system halt, you need to use the –FAST option of FIX_DISK (fast FIX_DISK). Fast FIX_DISK checks the directory structure and CAM file extent maps only. Please refer to the section Understanding the Concept of Recoverability and then read through the section Understanding the –FAST Option of FIX_DISK, below.

Booting: The boot procedure can only access files stored as SAM files. All files on a robust partition are stored as CAM files. The PRIMOS boot procedure cannot access any file stored on a robust partition. This means that you should not convert your command partition to a robust partition. This also means that you cannot use a robust partition as an alternate boot device.

Disk Space Required: Sometimes a robust partition requires more disk space than a standard partition to store the same amount of data. A SAM file on a standard partition contains only data records. When you move the SAM file to a robust partition, the file requires an additional record for the extent map. This means that a file that was stored as a single-record SAM file on a standard partition becomes a two-record CAM file on a robust partition. You must allow enough additional space for the conversion.

The amount of additional space required depends on the file type. ACLs and ACATs do not require additional space. Each SAM file requires one additional disk record for an extent map. DAM files might not require any additional disk space. CAM files do not require any additional disk space. Remember however, that CAM files allocate data records in blocks called extents. There are occasions when PRIMOS appends unused data records to the end of a CAM file. These records occupy additional disk space. Generally, you can minimize all of these considerations by placing only large database files on a robust partition.

The size of a segment directory is not significant to the discussion about robust partitions. The segment directory structure is itself not changed. Size considerations instead focus on the size of the individual subfiles within a segment directory. To determine the size of subfiles, see the discussion of segment directories in the section Robust Partition File System Objects.

Directories: The directory structure itself is changed on a robust partition. A Rev. 22.1 standard partition uses a hashed directory structure. A robust partition uses a linear directory structure. Entries should consist of only a small number of large files in each directory on the robust partition to maintain the directory search time.

Sectoring: Robust partitions do not support reverse sectoring. Whenever you convert a partition to the robust format, sectoring is automatically set to forward.

Accessing Rev. 22.1 Format Disks: Rev. 22.1 and later disks are a new format. To locally access either standard or robust Rev. 22.1 format partitions, you must be running Rev. 22.1 or later PRIMOS. You can access Rev. 22.1 format partitions remotely on a network, however, such as through PRIMENET. This means that you should not reboot your local system to an earlier version of PRIMOS. Insure that all of the PRIMOS upgrade has been successfully completed before you begin the conversion to robust partitions.

Null Records: Finally, understand that PRIMOS can insert a null-filled data record into your database as a result of a system halt. This rare event would cause a fatal error on a standard partition.

Understanding the Concept of Recoverability

You should understand one essential concept before deciding whether or not to use robust partitions. Robust partitions improve upon **recoverability**, or your ability to resume operations after a system halt. Similar to FIX_DISK, robust partitions do not offer any protection against disk corruption; they offer only an improvement in your ability to detect disk corruption. This is one reason why it is important to use robust partitions only for files that an application-level data verification routine can properly check.

In many cases, you can find a degree of data integrity corruption by running full FIX_DISK. This is not the reason for running FIX_DISK; FIX_DISK was designed to check file system integrity and does not check data integrity. Nevertheless, many locations rely on FIX_DISK to indicate whether the data integrity of a file has been compromised. This appears to work on a standard partition because full FIX_DISK detects corrupted data record headers. The assumption is made that if the data record headers are not corrupt, the data records are probably not corrupt either. Some of this ability to detect data corruption is lost when fast FIX_DISK (FIX_DISK –FAST) is used on a robust partition because fast FIX_DISK will not read any data record headers and therefore cannot verify the validity of the data record headers. Used properly, fast FIX_DISK offers the advantage of rapidly repairing your partitions but this can only be an advantage when you have an alternative process in place to verify data record integrity.

Robust partitions offer help in minimizing the inconvenience caused by a hardware failure, which can cause data loss. Table 7-1, Recommendations for Using FIX_DISK, later in this chapter, summarizes types of system halts and the necessary action to properly respond to those halts. These recommendations are applicable to systems using either standard or robust partitions. From an examination of Table 7-1, you can see that robust partitions offer the advantage of effectively utilizing the –FAST option of FIX_DISK (fast FIX_DISK) for those system halts that are trapped and processed through the PRIMOS slow-halt mechanism.

Understanding the –FAST Option of FIX_DISK

The –FAST option of FIX_DISK (fast FIX_DISK) allows the System Operator to quickly verify the integrity of the file structure. FIX_DISK does not provide any check on the integrity of the data contained within the files. Only a utility that understands the data management application can verify the data within a file.

This section explains the functionality of fast FIX_DISK on a robust partition and then briefly compares the functionality when you run fast FIX_DISK on a standard partition.

Both robust partitions and standard (nonrobust) partitions support the –FAST option. The –FAST option is less useful, however, on a standard partition because it can be used only if the partition was cleanly shut down.

FIX_DISK Action: FIX_DISK acts identically on the file system directory structure on both standard and robust partitions whether you enable the –FAST option or not. FIX_DISK checks the entire directory structure and verifies the integrity of every directory and segment directory entry. Use of fast FIX_DISK, however, limits the degree of verification on files within directories.

Use of fast FIX_DISK also limits the degree of verification of subfiles within a segment directory. This is an important technical detail. A segment directory is a special type of directory structure that contains a set of subfiles. All of the data is contained within the subfiles. Like any directory, there is a directory header that contains all of the information about the contents of that directory. A segment directory can contain many subfiles. Both full and fast FIX_DISK verify every directory header and every segment directory header. Use of the –FAST option allows FIX_DISK to provide directory structure verification more quickly.

When fast FIX_DISK completes without finding any mismatches, it has checked that the directory structure is intact and that the correct number of disk records have been allocated for the data files. You cannot be sure, however, that the data records actually have the correct data within them. To verify the data record content, you must run a verification routine of a data management package on any data management files.

Full FIX_DISK provides one additional level of verification that fast FIX_DISK does not provide. Full FIX_DISK reads every data record header within every file. Full FIX_DISK then verifies that the record header is properly initialized. Do not, however, rely on FIX_DISK as an indicator of the integrity of the data in a disk record.

Full and Fast FIX_DISK Comparison: To better understand the benefits robust partitions offer, we must distinguish between CAM file functionality and robust partition functionality. Full FIX_DISK processes a CAM file identically whether it is on a robust partition or on a standard partition. The operation of fast FIX_DISK depends whether the CAM file is on a standard partition or on a robust partition. On a standard partition, fast FIX_DISK verifies the last two data records within every CAM file extent. On a robust partition, fast FIX_DISK verifies only the extent map.

On a robust partition, all files are automatically stored as CAM files. Through the logical file typing mechanism, the physical file type is transparent to all higher levels of software. It is the physical typing, however, that is important to FIX_DISK.

In order for FIX_DISK to know which disk records a physical SAM file on a standard partition uses, FIX_DISK must check every record because SAM files do not have an index or an extent map. When FIX_DISK encounters a SAM file, it must read a record header, find the pointer to the following record, and then repeat the process. Thus, both full FIX_DISK and fast FIX_DISK must read through the entire SAM file. PRIMOS physically stores all SAM files as CAM files on a robust partition and, thus, FIX_DISK needs to check only the extent map.

In conclusion, the –FAST option is available on both standard and robust partitions. Fast FIX_DISK verifies the full directory structure on both standard and robust partitions. You can run fast FIX_DISK on a standard partition only when the partition has been cleanly shut down. If you need to run FIX_DISK on a regular basis, robust partitions can reduce the time required.

Understanding the Robust Partition File System

The robustness of a robust partition is transparent to nearly all software. Robust partitions introduce a new concept called **logical file typing**. In previous revisions of PRIMOS there were three types of physical files. A file could be a physical SAM file, a physical DAM file, or a physical CAM file. This physical typing determines exactly how the file is strung together to make it an entity. Robust partitions separate the physical file structure from the logical, or application-level, file structure. (See Chapter 1 for a description of the physical file types.)

Every file that is created on a robust partition is physically organized as a CAM file. This means that every file on a robust partition has an extent map that tells PRIMOS where the actual data records are stored. All of this is transparent to higher levels of software. LD, for example, reports the existence of SAM, DAM, and CAM files on a robust partition. If your application opens a SAM file on a robust partition, it appears to be a SAM file. This is the logical file type and it determines which application-level operations are possible. Underneath the application, however, PRIMOS converts the operations into the proper steps to access the correct data record in the physical CAM file that actually exists.

How Do You Know If a Partition Is Robust?

Because of logical file typing, a robust partition can be completely transparent to the system users. There are three commands that you, as the System Operator, can use to determine which partitions are robust.

You can determine if a local partition is robust by looking at the status information for the partition with the STATUS DISKS command or with the LIST_DISKS command. The display includes an indication that a partition is robust, as shown in the following example.

```
OK, STATUS DISKS

Disk      Ldev      Pdev      System    Robust
GROUP1              0         4461
OATIOS              1         4260
BIGDSK              2         4463                Robust
```

The LD command is another useful tool for determining whether a partition is robust. Unlike the STATUS DISKS and LIST_DISKS commands, the LD command reports robustness on both local and remote disks. The following example shows information that LD reports on a robust partition.

```
OK, ATTACH <BIGDSK>MFD
OK, LD

BIGDSK>MFD (LUR access) Robust

4 Files.

BADSPT BOOT BIGDSK DYNBSP

12 Directories

BATCH     CMDNC0    CRUSHES   DOS
DSA.ENM   FED       JANE      LEPRUN
MFD       BULL_BOARD          R1040A    WORD*

OK,
```

Robust Partition File System Objects

The file system objects supported on a robust partition consist of the following subset of the objects that are supported on a standard PRIMOS partition:

- CAM files (with logical SAM, DAM, and CAM file typing)
- Segment directories
- Linear user directories
- EPFs
- Access Control Lists (ACLs)
- Access Categories (ACATs)

This subset represents all of the PRIMOS file system objects that are found on a standard disk partition except for the physical SAM and DAM files. Robust partitions support logical SAM and DAM files only as physical CAM files.

Also, robust partitions support only the linear directory structure. Robust partitions do not support hashed directories. This subset of entry types supported by robust partitions is best for use with a few large files.

CAM Files: CAM files are extent-based files. An extent is a variable number of contiguous 2048-byte disk records. (See Figure 1-4 and the description of CAM files in Chapter 1.) The addresses of the CAM file extents are indexed in the extent maps that exist for each CAM file. Each extent map for a CAM file can index a maximum of 340 extents. The use of CAM files requires that additional memory be allocated for extent maps because an extent map table resides in memory for each open CAM file. The amount of memory required varies since the number of extent maps and, thus, the size of the extent map table is determined by the number of extents in the file.

A Rev. 22.1 standard or robust partition can contain CAM files with a large number (16,381) of extents. Prior to Rev. 22.1, a CAM file could have a maximum of 340 extents and, thus, only one extent map. The previous limit of 340 extents can be reached on fragmented partitions and with large databases. (See the subsection Fragmentation of Partitions later in this chapter for a discussion of fragmentation.) When the limit is reached, the operation that is extending the file stops and you must either copy the file or remake the partition to reduce the number of extents in the file. With unlimited CAM file extents available at Rev. 22.1, it is unlikely that CAM files will reach the limit, but fragmentation can occur if there are many extents in a file.

File System Objects: Other than the special files created by MAKE, CAM files are the only nondirectory file system objects on a robust partition. When you first create the partition, MAKE builds the special files BOOT, DSKRAT, DYNBSP, and a badspot file. A standard partition supports nondirectory file system objects of the types SAM, DAM, or CAM. A robust partition supports only the CAM physical type of nondirectory file system objects. Because SEGSAM and SEGDAM entries are actually a type of directory structure, these structures are also supported on a robust partition. All subfiles within a segment directory, however, are created as CAM physical file types.

Whenever a user creates or copies a nondirectory file system object on a robust partition, PRIMOS automatically organizes the file physically as a CAM file. This physical typing is distinct from the logical typing. The logical file type remains that which was set by the subroutine or application that creates the file. The logical file type can be SAM, DAM, or CAM but the physical file type will always be CAM on a robust partition.

The physical file typing is transparent to the software that accesses the file. The file continues to behave as if it were a file of the logical type. Even EPFs, which are DAM files on standard partitions, are physically organized as CAM files on robust partitions. If you check the file type through a subroutine call or by using LD -DETAIL, the file type listed is always the logical file type. Robust partitions are thus able to support all three nondirectory logical file types even though all of the files are physically CAM files.

Copying Files: You can copy CAM files to robust partitions, both locally and remotely. You can copy CAM files *from* robust partitions, both locally and remotely, as long as the partition you are copying the CAM file to is Rev. 20.0 format (or later). You can copy CAM files to pre-Rev. 20.0-format disks as long as you use the –SAM or –DAM conversion options when you invoke the COPY command. This same conversion process is also available through the COPY command when copying a CAM file to a Rev. 20.0 or later standard partition.

CAM File Management: PRIMOS subroutines allow management of CAM files that have a CAM logical file type. See the for discussions of these subroutines:

- CF$EXT to move a CAM file's physical end-of-file
- CF$REM to return information about a CAM file's physical layout on disk
- CF$SME to set a CAM file's extent length value

Segment Directories: Segment directories exist on both standard and robust partitions at Rev. 22.1. Both partition types support both SEGSAM and SEGDAM directory types. Segment directories are used by a number of data management applications and products, such as Prime INFORMATION™.

On a robust partition, the subfiles in a segment directory (either SEGSAM or SEGDAM) are always physically organized as CAM files. The segment directory, however, retains the logical type of either SEGSAM or SEGDAM. Thus, it is not necessary to change existing applications employing segment directories for use on a robust partition.

It is important to know the size of the individual subfiles within a segment directory before you move the segment directory onto a robust partition. You can use either LD or SIZE to determine the size of the subfiles within a segment directory. An example of determining the size of the subfiles is shown below.

```
OK, LD ROB.P>MIDASPLUS>DEMO>@@ -SIZE

<SOFTWR>ROB.P>MIDASPLUS>DEMO (ALL access)

5 Files.
name            size  type  rbf
---------------------------------------------------------------
0               1     sam
1               2     sam
185             4     dam


OK, SIZE ROB.P>MIDASPLUS>DEMO>@@
1 record in sam file "ROB.P>MIDASPLUS>DEMO>0" (336 halfwords)
2 records in sam file "ROB.P>MIDASPLUS>DEMO>1" (1088 halfwords)
4 records in dam file "ROB.P>MIDASPLUS>DEMO>185" (3640 halfwords)
OK,
```

If these files are moved to a robust partition, the SAM files will each use one more record for extent maps but the DAM file will not need any more records. There will then be eight records in SAM files and four records in the DAM file on the robust partition.

Directories: All directories on robust partitions are linear (that is, sequential) as opposed to hashed. Directories on Rev. 22.1 standard partitions are hashed directories. Access to hashed directories is faster than access to linear directories. Thus, users who have directories with many files should consider the change in directory access speed before moving to robust partitions.

Linear directories on robust partitions are just like ACL directories on standard partitions, except that entries are stored in a linear rather than hashed fashion. Robust partition linear directories support all the same attributes supported by standard partition ACL directories, including the DTA (date/time accessed) and DTC (date/time created) attributes.

Performance Characteristics of the Robust File System

The robust partition format does not directly affect the performance of any applications. There are three indirect effects, however, that might alter the performance characteristics of an application accessing data on a robust partition.

- Robust partitions do not support hashed directory structures. Therefore, performance can degrade if the directory contains a large number of files. See the subsection Robust Partition File System Objects, above.
- Robust partitions do not support reverse sectoring. Thus, there are specific situations when this can cause performance to degrade. See the subsection Restrictions on the Use of Robust Partitions, above.
- All other performance characteristics of robust partitions are extensively tied to the use of CAM files.

Some of the performance characteristics are

- CAM files offer improved performance when used in the proper situations. Typically, a CAM file offers better performance than a SAM or DAM file when the file is larger than one megabyte (512 disk records). This is a very generalized statement and refers specifically to the lowest level file. Within a segment directory, each individual subfile should be larger than one megabyte.
- Generally CAM files perform better than SAM or DAM files when a large number of users simultaneously access random records in the file. Simultaneous access is normally how users access a database file. This is generally not how a CAD application accesses a file, however.

- A single user reading a CAM file sequentially might notice slower performance than that available from a SAM or DAM file on a standard partition.
- If your application can call PRWF$$ to write 2048-byte disk records, a CAM file is faster.
- File truncation and file deletion are always faster on a robust partition because the robust structure allows the data records to be released without being unthreaded, or verified. Therefore, the DSKRAT is updated but the individual data records do not need to be read.
- Appending a large block of data can be faster on a CAM file. Because each extent is pre-allocated, a large append operation needs to extend the file only once. This benefit is visible, however, only if your application writes the entire block of data in one operation.

Prime Data Management Packages on a Robust Partition

This section discusses the effect of robust partitions on Prime data management packages.

DBMS: Most file structure issues that affect other applications do not affect DBMS. DBMS uses pre-allocated data files that do not tend to grow or shrink except during special administrative operations. Thus, there is less likelihood that the types of errors that fast FIX_DISK can detect will corrupt a DBMS file. Additionally, DBMS already takes advantage of the performance improvements available from CAM files.

If it is important that you run FIX_DISK after every system halt, robust partitions can save you time.

Prime INFORMATION™: For Prime INFORMATION files, the dynamically hashed file type is best suited for robust partitions. Dynamically hashed Prime INFORMATION files are structured as segment directories with a few relatively large subfiles. (Refer to the previous subsection, Restrictions on the Use of Robust Partitions, to determine the size of the subfiles within a segment directory.)

You can use all other types of Prime INFORMATION files on a robust partition. Remember, however, that every SAM file and every SAM subfile requires one additional disk record for the extent map. If you have Prime INFORMATION statically hashed files, which are SAM files, the sizes of those files will double on a robust partition.

It is necessary to check the data integrity of the Prime INFORMATION files after running fast FIX_DISK. You can verify the data contained within the Prime INFORMATION file using the FIX.FILE command VERIFY option. FIX_DISK checks only file system structure.

MIDASPLUS™: Before you move a MIDASPLUS segment directory to a robust partition, check the size of the subfiles. (Refer to the previous subsection, Restrictions on the Use of Robust Partitions, for information on determining the size of the subfiles within a segment directory.) If the subfiles are less than a few hundred disk records, you might see a performance degradation if you convert to robust partitions. If the subfiles are larger, however, you might find a performance improvement when you convert to robust partitions.

One distinguishing feature of MIDASPLUS is the index block splitting process. If you have had problems with corruption of index subfiles after a system halt, robust partitions might reduce the number of these problems because CAM files requires fewer updates to add a new record to the end of the file.

Remember to check the data integrity of the MIDASPLUS files after running fast FIX_DISK. FIX_DISK checks only the file system structure. (Please review the section Understanding the –FAST Option of FIX_DISK, above.) Use the CREATK command, selecting the COUNT option, to quickly check the data integrity of a MIDASPLUS subfile.

Prime® ORACLE®: Most of the file structure issues that affect other applications do not affect Prime ORACLE. Prime ORACLE uses pre-allocated data files that do not tend to grow or shrink except during special administrative operations. Thus, there is less likelihood that the types of errors that fast FIX_DISK can detect will corrupt a Prime ORACLE file. After considering these factors, base your decision on whether you need to run fast FIX_DISK after every system halt. If you do, robust partitions can save you time.

PRISAM™: PRISAM INDEXED and PRISAM SEQUENTIAL files are good candidates for use on a robust partition. These files tend to expand. On a standard disk partition, the process of adding an additional record to the end of the file can be left incomplete if there is a system halt. Because CAM files require fewer steps to add a record, there is less likelihood that the operation will be incomplete.

PRISAM RELATIVE files are pre-allocated. Most of the file structure issues that affect other applications do not affect pre-allocated PRISAM files. Pre-allocated data files do not tend to grow or shrink except during special administrative operations. This means there is less likelihood that the types of errors that fast FIX_DISK can detect will corrupt a pre-allocated file.

PRISAM already takes advantage of the performance improvements available from CAM files on a standard disk partition. It is unlikely that robust partitions will noticeably impact the performance of PRISAM.

After considering these factors, base your decision on whether you need to rapidly run FIX_DISK after every system halt. If you do, robust partitions can save you time.

Non-Prime Data Management Files

If you have a data management package purchased from a company other than Prime, check with that company before converting to robust partitions. If this is not possible, try to use the previous section as a guide. Look for the important indicators. Consider these items:

- Is the file pre-allocated? Generally, pre-allocated files do not derive as much benefit from robust partitions.
- Are the subfiles large? Larger (more than 512 records) files and subfiles can improve in performance on a robust partition.
- Are there problems with corruption caused by system halts? Although robust partitions cannot prevent this corruption, robust partitions allow you to find the problem more quickly.
- Does the application have a tool to verify the integrity of the data within the file? This is an important tool that is useful on all partition types. Be aware that full FIX_DISK might have been used as a substitute for running a proper database verification routine. If this is the case, then you must change your procedures when you use robust partitions. (See the discussion in Understanding the –FAST option of FIX_DISK.)

Other Files

Other files are all of those files that are not specifically tied to a data management package. These other files might include text files, mail files, spooler files, source code, runfiles, and system configuration files. In general, these files are not appropriate for a robust partition. However, you can use the following guidelines to help in determining whether or not these files should reside on a robust partition.

- Is the file large? Larger (more than 512 records) files and subfiles can improve in performance when converted from SAM or DAM to CAM by being copied to a robust partition. Smaller files can show a performance degradation.
- SAM files gain one extra disk record when converted to CAM files. This means that a one-record SAM file occupies two records when placed on a robust partition. Generally, you do not want to use the robust partition for spooler files, batch jobs, FTS queues, or other applications that have a large number of small files.
- Are there problems with corruption caused by system halts? Although robust partitions cannot prevent this corruption, robust partitions allow you to find the problem more quickly.

- Do you understand that the tradeoff offered by using the –FAST option of FIX_DISK means that the data records are not checked? Data management packages generally have a special tool to perform this integrity check. File system objects like mailboxes, source code, and system configuration files might be left in a corrupted state for a long time before a check is made to insure their integrity. You must decide if fast FIX_DISK is appropriate for these files.

Evaluating the Use of Robust Partitions

This section poses some questions about your file system and your use of FIX_DISK. If you answer NO to any of the first four questions, robust partitions are not appropriate for your system. If you answer YES to all four questions, please continue.

1. Do you want to consistently use the –FAST option with FIX_DISK to improve the MTTR (mean-time-to-recover) for your system? (Please read through the section Understanding the –FAST Option of FIX_DISK before answering this question.)
2. Do you run FIX_DISK on at least one disk partition after most system halts?
3. Do you want to know when it is not necessary to run FIX_DISK after a system halt?
4. Are you willing to run FIX_DISK on a robust partition every time the ADDISK command indicates that it is required?

Robust partitions are most beneficial when you properly prepare to use them. There are some disk partitions you will not want to convert to the robust format. Use the following questions and the Robust Partitions Evaluation Form at the end of this chapter to help you decide which disk partitions are best suited to the robust format.

1. Which disk partitions do you consistently run FIX_DISK on after a system halt? Make a list of those partitions on the Robust Partitions Evaluation Form. List the physical device number (pdev) and the name of the partition. You require both the pdev and the partition name when you convert a partition to the robust format.
2. Fill in the appropriate column to indicate whether or not you intend to boot from a disk partition. Since you cannot boot from a robust partition, you cannot convert some partitions to robust partitions. This includes your command partition and might include other partitions if you frequently need to boot from a partition that is not the normal command partition.

3. Indicate the percent full that the AVAIL command reports for each partition. Some free space must exist on a disk partition in order to successfully complete the conversion process. The amount of additional space that is required depends on the size and type of files that are on the existing partition. For this infomation, review the paragraph Disk Space Required within the earlier section Restrictions on the Use of Robust Partitions.
4. List the names of any data management packages that use this disk partition. A data management package can be a development environment supplied by Prime, such as Prime INFORMATION and Prime ORACLE, or a data management package might be an application that your company developed. If you are unsure of the answer to this question, read the section Prime Data Management Applications on a Robust Partition, above.
5. Can the data management package verify the integrity of the data records on the robust partition? A software package with an integral transaction-recovery feature probably meets this need. Prime supplies this feature with DBMS, PRISAM, and Prime ORACLE. Other packages, such as Prime INFORMATION and MIDASPLUS, have functions available that you can use to verify the integrity of the data records.

   Indicate on the Evaluation Form whether or not you use the data verification functionality offered with the data management package.
6. Decide whether or not you want to convert this disk partition into a robust partition. This partition is a good candidate for conversion into a robust partition if all of the following statements are true:
   - You have answered yes to the first four questions, above.
   - You do not intend to boot from this partition.
   - This partition has sufficient space.
   - Any data management package that may use this partition can effectively make use of the features offered by robust partitions.
7. Fill in the last column on the Evaluation Form to indicate the appropriateness of this disk partition for conversion into the robust format.
8. If you have decided that you want to use robust partitions, contact your Prime Support Analyst (call the Prime 800 number) and discuss the Robust Partitions Evaluation Form and any other details of your use of robust partitions.

Creating Robust Partitions

This section describes how to convert existing partitions to robust partitions and how to create new robust partitions.

Converting Existing Partitions

You can use the FIX_DISK option –CONVERT_22.1 with the –FIX option to convert existing Rev. 22.0 partitions to Rev. 22.1-format standard partitions. This results in updating the revision stamp in the DSKRAT of the partition and allows you to create CAM files with an unlimited number of extents. You cannot, however, convert the resulting Rev. 22.1-format standard partition to a robust partition.

In order to convert existing Rev. 22.0 partitions to Rev. 22.1-format *robust* partitions, you must use MAKE and then the MAKE_ROBUST utility (discussed below) as well as available logical backup utilities such as MAGSAV and MAGRST, or COPY.

To convert existing pre-Rev. 22.0 partitions to Rev. 22.1-format standard partitions, you also must use MAKE. For example, use this procedure to convert a partition to a Rev. 22.1-format standard partition.

1. Use MAGSAV to save the data from the existing partition to tape.
2. Create the Rev. 22.1 standard partition by using Rev. 23.3 MAKE.
3. Use MAGRST to restore the data to the newly created partition if you do not intend to convert this partition to a robust partition.

Alternatively, you may create a Rev. 22.1-format standard partition, convert it to a robust partition and copy data to it from another partition using the COPY command or restore the data by using MAGRST. The file system organizes all files copied to the robust partition as CAM files but preserves their logical file type. All file directories on the robust partition will be linear directories.

You may mirror Rev. 22.1-format standard and robust partitions in the normal manner as discussed at the end of this chapter.

The process for converting any partition to a Rev. 22.1-format standard or robust partition is summarized here.

Existing Partition	*Rev. 22.1 Partition*	*Use*
pre-Rev. 22.0	standard	MAKE
pre-Rev. 22.0	robust	MAKE followed by MAKE_ROBUST
Rev. 22.0	standard	FIX_DISK (cannot convert to robust)
Rev. 22.0	standard	MAKE
Rev. 22.0	robust	MAKE followed by MAKE_ROBUST

Creating New Robust Partitions

MAKE_ROBUST is the utility that is necessary to create robust partitions. MAKE_ROBUST converts Rev. 22.1-format standard partitions, which you create with MAKE, to robust partitions. The partition that you want to convert to a robust partition must be a Rev. 22.1-format standard partition that has not been altered in any way. It must contain the standard file system objects that MAKE creates on the partition, as described in the first section of this chapter. The standard partition must also include the ACLs that are placed on the MFD by MAKE and that, by default, are in effect for all the file system objects on the partition. MAKE_ROBUST rejects any partition that is not a Rev. 22.1-format standard partition or one that has been altered in any way.

To obtain MAKE_ROBUST, see item 12 in a previous section Evaluating the Use of Robust Partitions.

Follow this procedure to create a Rev. 22.1-format robust partition:

1. Place the partition in the Assignable Disks Table with the DISKS command and assign it with the ASSIGN DISK command.
2. Run MAKE to create a Rev. 22.1 standard partition.
3. Run MAKE_ROBUST to convert the standard partition to a robust partition.
4. Unassign the disk with the UNASSIGN DISK command and remove it from the Assignable Disks Table with the DISKS NOT command.

MAKE_ROBUST does not replace MAKE; rather MAKE_ROBUST augments MAKE. MAKE_ROBUST does not perform any of the functions of MAKE, such as creating file system objects or doing badspot checking. MAKE_ROBUST simply alters portions of the file system structure to tell PRIMOS that the partition is robust and enforces forward sectoring as the method of file record allocation. You can also specify the robust partition minimum and maximum extent sizes with MAKE_ROBUST. If you do not specify these values, MAKE_ROBUST sets them to the default values (64 for the minimum and 256 for the maximum) whether or not you previously set them with MAKE.

MAKE_ROBUST Syntax

The format of the MAKE_ROBUST command is

MAKE_ROBUST ***pdev*** **[*options*]**

The arguments and options are as follows.

pdev

Tells MAKE_ROBUST the physical device number of the partition that you are converting to a robust partition. You may use *pdev* by itself or you may use –DISK *pdev* as in the MAKE command line.

–HELP

Displays the command line usage and a brief description of each of the options.

–MIN_EXTENT_SIZE [*size*]
–MINSIZ

Specifies the minimum extent *size* for CAM files on this Rev. 22.1 robust partition. If you do not specify the minimum extent size, MAKE defaults to 64 records for minimum extent sizes on robust partitions.

–MAX_EXTENT_SIZE [*size*]
–MAXSIZ

Specifies the maximum extent *size* for CAM files on this Rev. 22.1 robust partition. If you do not specify the maximum extent size, MAKE defaults to 256 records for maximum extent sizes on robust partitions.

MAKE_ROBUST Messages

Special messages indicate that MAKE_ROBUST is unable to convert the partition or is unable to interpret the command line. Appendix E explains these messages.

Administration of the Files on a Robust Partition

This section discusses the allocation of CAM file records on a robust partition and the possible fragmentation of robust partitions.

File Allocation on Robust Partitions

The default allocation algorithm for CAM files on all partitions is designed to reduce disk fragmentation and wasted space in files. (See the section Fragmentation of Partitions later in this chapter.) The algorithm uses the minimum and maximum extent sizes specified for the partition. You can specify the minimum and maximum extent sizes using MAKE and FIX_DISK options. The default extent sizes for Rev. 22.1 robust partitions are 64 records for the minimum and 256 records for the maximum. The default extent sizes for Rev. 22.0 and later standard partitions are 16 records for the minimum and 32 records for the maximum. The minimum and maximum extent sizes cannot be set equal to zero or equal to each other. The maximum value for extent size is 32,767 records.

You can set the extent sizes on any Rev. 22.0 and later partition. You cannot, however, set extent sizes on pre-Rev. 22.0 partitions. The default minimum and maximum extent sizes for CAM files on pre-Rev. 22.0 partitions are 16 records and 32 records, respectively. These defaults are small enough to result in keeping the amount of time required to extend a CAM file to a practical minimum (since all the records on pre-Rev. 22.0 partitions must be initialized) and still allow files to grow larger on a pre-Rev22.0 partition. Initialization consists of clearing out any header information that may be in the records, such as pointers to other records if these records were previously used in a file. CAM file records on Rev. 22.1 robust partitions are not initialized when they are allocated.

The Allocation Algorithm: The allocation algorithm allocates blocks, or groups, of records in graduated sizes based on the size of the file. Allocation requests equal the current file size until the maximum extent size is reached; as a result, the file size doubles at each extension. After the maximum size is reached, a block of records equal to the maximum extent size is allocated. This means that the size of a file generally grows in the following increments (using the default maximum extent size of 256 records for a robust partition): 1, 2, 4, 8, 16 . . . 256, 512, 768, 1024 . . .. When the file is larger than the maximum extent size (256 in this case), it grows in increments of maximum extent size records.

In the default algorithm, whenever a file is extended, any free contiguous space at the end of the file is added to the file, even if it is less space than the requested number of records or less than the minimum extent size. If there is not sufficient free contiguous space at the end of the file, additional extents are allocated until the requested size is reached. If the initial request is greater than the minimum extent size, all the additional extents are at least the minimum extent size.

A user may optionally specify an extent size for individual files by using the COPY –MXL option or by using the CF$SME system subroutine in an application program. This may be done with a large file, for example, so that fewer records are allocated on the next extension, thus saving disk space and reducing fragmentation. If the extent size is specified in this way, the default allocation algorithm is not used. Instead, whenever a file is extended, any contiguous space at the end of the file is used and an additional extent is added so that the total new space in the file is equal to the extent size specified.

Minimum Extent Size: The minimum extent size is used when there is not enough free contiguous space for PRIMOS to either double the file size or to allocate the maximum extent size when extending a file. This helps to prevent disk fragmentation and helps to keep the number of extents in a file to a minimum by ensuring that the file is not extended by using a small number of records. A CAM file performs best when it has a minimum number of extents and when the size of each extent is as large as possible. For example, if a file is to be extended by 50 records and the minimum extent size is one record, the extension could occur in 50 one-record extents. If the minimum extent size is 25 records, the file is extended in a maximum of 2 extents. Under certain circumstances, the default allocation algorithm, minimum extent size, and extent lengths are ignored. These circumstances are discussed in the later section Fragmentation of Partitions.

When PRIMOS extends a file and the number of free contiguous records available is not equal to either the current file size or to the maximum extent size, PRIMOS uses the largest block of free contiguous records that is equal to or larger than the minimum extent size. If no such block exists, the user sees the message `Insufficient free contiguous blocks` and PRIMOS cannot extend the file. In this case, files must be deleted in order to free the necessary space. In addition, the disk may be too fragmented so that you must reduce the fragmentation, as discussed in the section Fragmentation of Partitions later in this chapter.

Allocation Example: As an example of how the allocation algorithm works, assume there is a file that currently has a block of 512 records in one extent and is followed by a block of 50 records of available space. (See Figure 7-1.) Also assume that the default minimum and maximum extent size limits of 64 and 256 records are in effect and that the partition has additional unused space of 800 contiguous records.

When additional records are needed for the file, the default algorithm requests the default maximum extent size, 256 records, since the file is currently larger than 256 records. The 50 records of free space at the end of the file are allocated for the file and the current extent is expanded to include these records. To satisfy the original request, 206 records must still be allocated. Since there is sufficient space (the 800 contiguous records), the 206 records are allocated from the 800 contiguous records in one more extent. The result is that the file now has 768 records in two extents. If there had been no available extent of at least 64 records (the minimum extent size), no additional extents would have been allocated; the file size would be only 562 records (512 + 50) in one extent.

Extent Size Defaults: The defaults for minimum and maximum extent size were chosen to provide a user with an environment that allows both large files and small files to exist and not use excessive space. You can change the default minimum and maximum extent sizes to adjust to specific situations. For example, if all the files on a partition are extremely large (say 10,000 records each), the maximum extent size can be made larger to reflect the anticipated size of files. If fragmentation of files is a concern, the minimum extent size can be increased. You change the default minimum and maximum extent sizes on an existing partition by using FIX_DISK.

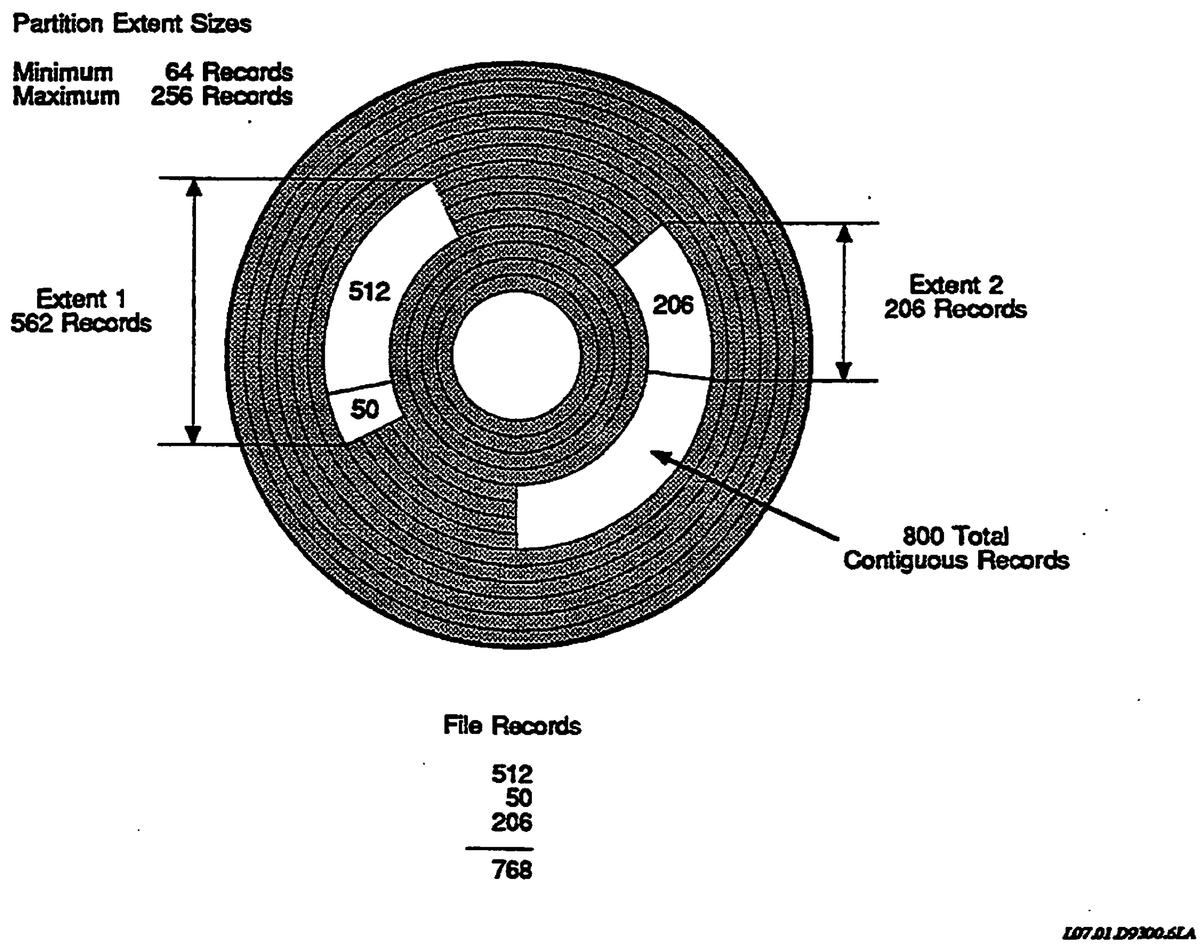


Figure 7-1. Allocation Example – Robust Partition

Setting Extent Sizes

To ensure the optimum performance of robust partitions and CAM files, you need to consider several issues. When you use large files, you should set the minimum extent size to accommodate extension of files in larger pieces. If the extent size is small, allocation occurs more frequently, requiring more extents for the file and potentially leading to severe fragmentation of the partition.

To further increase performance, applications should write to the end of the file in increments of a record (2048 bytes). Applications should also buffer records internally and then write them as a whole, since the logical end-of-file is then updated less often. Thus, extents for large CAM files should be multiple records, which means that a file may have unused records because the physical end-of-file is beyond the logical end-of-file. (The SIZE command shows this difference; see the example in Chapter in the section Example of Creating a Robust Partition.)

An application can also set an extent size for individual CAM files on Rev. 20.0, Rev. 21.0, and Rev. 22.0 partitions. This size is used as the allocation size whenever the application extends a file. You can also set extent sizes on Rev. 22.1 standard and robust partitions; this is effective for all file extensions done by PRIMOS when individual file extent size is not set. However, if an application explicitly requests (through CF$EXT) that a number of records be allocated for a file, the file extent size that is set for the partition is not used.

Fragmentation of Partitions

Robust partitions may become fragmented after extensive use (for example, if users create, extend,and delete files frequently). That is, the free, or unused, blocks of contiguous records remaining may be small and randomly placed on the partition. In this case, if a user attempts to add to a file, there may not be enough contiguous space to do so even though the remaining space appears to contain enough records. If this happens, PRIMOS returns an error stating that there are insufficient free contiguous records or, in certain circumstances, PRIMOS adds the records to the file and displays a warning message at the supervisor terminal stating that the partition is highly fragmented.

To help you determine the degree to which the partition is fragmented, use the LIST_CONTIGUOUS_BLOCKS command (abbreviation LCB). The format of this command is

LIST_CONTIGUOUS_BLOCKS [{ *partition_name* | **–LDEV** *ldev* }] [*options*]
LCB

Use the following arguments and options.

partition_name
–LDEV ***ldev***

Use the name of any added partition as shown in the listing produced by the STATUS DISKS command or use –LDEV to enter the partition's logical device number. *ldev* is the *decimal* logical device number. (You can use the PRIMOS command TYPE and the function [OCTAL *number*] to convert the octal ldev to decimal. For example, if the $ldev_8$ is 17 and you enter `TYPE [OCTAL 17]`, you get 15 as the decimal result.) If you do not give a *partition_name* or a logical device number (*ldev*), the MFD of the current attach point is used.

–DISPLAY *number*
–DSP

Display the sizes of the decimal *number* of fragments, or blocks, of free contiguous space larger than 32 records or larger than a size specified by the –FREE option. The default number of blocks is 20 and the range of the number of blocks that you can specify is from 6 through 1024. If you specify a number outside of this range, LCB uses the default number of blocks.

–FREE *size*

Specify the minimum *size*, in decimal number of records, of the largest blocks of free contiguous space that you want displayed. The default minimum size is 32 records and the range of *size* that you can specify is from 0 through 9999 records. If you specify a size outside of this range, LCB uses the default size.

–HELP

Displays the usage of the command and presents a brief description of the options.

The following information about the partition is displayed by default by the LIST_CONTIGUOUS_BLOCKS command:

- Sizes of the 20 largest blocks, or fragments, of free contiguous space larger than 32 records
- Total number of fragments of free contiguous blocks of records
- Minimum and maximum extent sizes
- Total number of records in the partition
- Total number of free records (available space)
- Percentage full (records used, total minus free, divided by total records)

You can change both the number of blocks and the minimum size of those blocks of free contiguous space to be displayed by using the –DISPLAY and –FREE options.

The following example shows the output displayed by the LCB command.

```
OK, LCB INTEG
[LCB Rev.  23.3 Copyright (c) 1992 Prime Computer, Inc.]
+---------------------------------------------------------------+
! Partition: INTEG                                              !
+---------------------------------------------------------------+
! Largest 20 blocks of free                  2973       335     !
! contiguous space larger                    2211       282     !
! than 32 records.                           2035       280     !
!                                            1976       195     !
!                                            1894       191     !
!                                            1854       146     !
!                                            1588       109     !
!                                             599        90     !
!                                             401        86     !
!                                             335        50     !
+-------------------------------+-------------------------------+
!                               !                               !
! Number of fragments :  99     ! Total records   : 238896      !
! Minimum extent size :  50     ! Free records  :    18319      !
! Maximum extent size : 200     ! Percentage full : 92.33%      !
!                               !                               !
+-------------------------------+-------------------------------+
OK,
```

Note that the display of the LCB command allows you to determine the present maximum and minimum extent sizes for a partition.

Criteria for Determining Fragmentation: Use the information below to establish criteria for determining whether your partitions are highly fragmented:

- The number and size of free contiguous space larger than 32 records
- The number and size of fragments as deduced from the display
- The operations you are doing in the file system, such as creating and extending large files or small files or compiling programs

For example, from the previous LCB display, a user could create a 10,000-record file on the INTEG partition, using the first five blocks of free contiguous space even though the partition is presently 92% full. There would then still be 94 fragments of free contiguous space and 11 of them are larger than 100 records each.

A different situation exists in the example of the following partition.

```
OK, LCB FOXTST
[LCB Rev.  23.3 Copyright (c) 1992 Prime Computer, Inc.]
+-----------------------------------------------------------+
! Partition: FOXTST                                         !
+-----------------------------------------------------------+
! Largest 20 blocks of free              176        85      !
! contiguous space larger                175        84      !
! than 32 records.                       152        83      !
!                                        133        80      !
!                                        108        72      !
!                                        104        70      !
!                                         99        62      !
!                                         97        51      !
!                                         91        50      !
!                                         87        49      !
+-----------------------------+-----------------------------+
!                             !                             !
! Number of fragments : 3958  ! Total records    : 125920   !
! Minimum extent size :   16  ! Free records     :  16497   !
! Maximum extent size :   32  ! Percentage full  :  86.89%  !
!                             !                             !
+-----------------------------|-----------------------------+
OK,
```

The 20 largest blocks of free contiguous space are all less than 200 records, there are 3958 fragments, and the partition is about 87% full. If a user creates a 2000-record file, it will use more than the existing 20 blocks of free contiguous space. The disk is highly fragmented and is a candidate for running the procedure discussed in the following section to compact it.

Under certain circumstances, PRIMOS displays the following message at the supervisor terminal when a partition is highly fragmented:

`*** From PRIMOS: Partition` *`name`* `has little free contiguous space`

This message means that there are no blocks of free contiguous space larger than 30 records on the partition. Under these unusual conditions, PRIMOS grants requests for new CAM files or new CAM extents by using a secondary allocation algorithm but ignores any specified minimum number of records for the new extent. At least one and at most 30 records are allocated, depending on the individual request.

This behavior continues as long as the partition remains in the highly fragmented state. While in this state, the error `Insufficient free contiguous blocks` is not generated by PRIMOS in response to any request for a new CAM file or for a new CAM extent for an existing file. However, if you do not relieve the condition, the partition may run out of space altogether and the error `Disk full` is generated.

To relieve this condition, see the next section, What to Do if Partitions Are Fragmented. Once the condition is relieved, the first time a new CAM file or extent is created, PRIMOS detects that the condition has been relieved. When this occurs, PRIMOS displays the following message at the supervisor terminal:

```
*** From PRIMOS: Partition name has regained free contiguous space
```

At this point, PRIMOS reverts to using the standard allocation algorithm, described earlier in this chapter.

What to Do if Partitions Are Fragmented: If your partition is highly fragmented, you should do the following:

1. Save the data on the partition to tape or to another disk by using a logical backup utility, such as MAGSAV or COPY.
2. Run MAKE on the fragmented partition with any necessary options.
3. If the partition is to be a robust partition, use the MAKE_ROBUST utility to convert to robust partition format.
4. Restore the data to the newly created partition by using a logical backup utility complementary to the one used in step 1.

If you have recently run full FIX_DISK on the fragmented partition so that you are confident that all badspots have been detected, you may use the MAKE –BADLEV 0 option to reduce the time it takes MAKE to recreate the partition. Do *not* use the –NEW_DISK option as you do not want to make a new badspot file and lose the existing badspot file. (See Chapter 5.)

FIX_DISK and Robust Partitions

Typically, system halts involving robust partitions require that you run FIX_DISK, generally fast FIX_DISK (FIX_DISK with the –FAST option). You must run FIX_DISK after halts that involve machine checks, power failures, and other critical failures. Halts that result in a clean shutdown during halt processing do not require you to run FIX_DISK. Approximately 25% of halts are in this latter category. (See Table 7-1.)

When to Run FIX_DISK

When you add a robust partition, PRIMOS notifies you whether it is necessary to run FIX_DISK. If you use ADDISK and the robust partition is not corrupted, the partition is added. If the robust partition is corrupted, it is not added and a message is returned requesting that FIX_DISK be run on the partition:

```
*** Robust Partition diskname has not been properly shutdown.
*** Fast Fix_disk has to be run before it can be added.
```

The partition is then not accessible. (For exceptions, see the following section ADDISK and Robust Partitions.)

If the halt results in an unrecoverable read or write error on the partition and it is necessary to immediately correct this situation, you should run full FIX_DISK to find the disk record that is in error and add it to the badspot file. Do not run fast FIX_DISK in this situation unless the disk is associated with an intelligent controller and is in -DBS ON mode. Fast FIX_DISK detects the error on robust partitions only if the error is in directory entries or CAM file extent maps.

Once FIX_DISK finishes, the partition is robust again and can be added for both reading and writing. You should run fast FIX_DISK periodically to remove directory entries and data records that are not in use. Use full FIX_DISK to detect and remove badspots from partitions. If an unrecoverable error occurs, you can use either fast or full FIX_DISK to add the badspot to the badspot file.

Whenever a partition is improperly shut down, as in a system halt, the possibility of errors occurring on the partition exists. The only way to correct these errors is to run FIX_DISK. On Rev. 22.1 robust partitions, you can use fast FIX_DISK to recover directory information and repair errors.

Summary of Recommendations for Running FIX_DISK

Table 7-1 summarizes recommendations for running full FIX_DISK (FIX_DISK -FIX) or fast FIX_DISK (FIX_DISK -FIX -FAST) for both standard and robust partitions. Halts correspond to the types discussed in your CPU handbook in the chapter on handling halts and hangs. If you cold start after a halt, PRIMOS reminds you to run FIX_DISK if the disk needs repairing. If you warmstart and the warm start is successful but the disk needs repair, you are not reminded, but you should follow the recommendations in Table 7-1. You can use Resident Forced Shutdown (RFS) after a halt to attempt a successful shutdown.

The terms in the left column of Table 7-1 have these meanings:

- A successful Forced Shutdown means that the system locate buffers were flushed. An unsuccessful Forced Shutdown means that the system locate buffers were not flushed. The message at your supervisor terminal indicates whether the Forced Shutdown was successful or not.
- IDC refers to downloaded intelligent disk controllers; that is, disk controllers that are operating as intelligent disk controllers. (A message appears on the supervisor terminal at cold start when an intelligent disk controller is downloaded.)
- NDC refers to nonintelligent disk controllers (such as Model 4005 disk controllers) and intelligent disk controllers that are not downloaded and are thus operating as nonintelligent controllers.

The major difference between the ECCU and Immediate halts on machines with an intelligent disk controller (IDC) and those with a nonintelligent disk controller (NDC) is that the IDC successfully writes the record to the disk while the NDC may not successfully write the record out to disk and you may have an unrecoverable error.

The types of halts are defined in your CPU handbook.

Caution An unrecoverable error that is not detectable by fast FIX_DISK could exist in a file in cases of disk controller failure, some processor failures, or errors in the file system software. However, since these occurrences are rare, you may use fast FIX_DISK (as summarized in Table 7-1) on a robust partition to improve recovery time. You can run full FIX_DISK at a convenient time.

Table 7-1. Recommendations for Using FIX_DISK

Type of Halt	*Standard Partition*	*Robust Partition*
Forced shutdown – successful	No FIX_DISK unless message from ADDISK; then use fast FIX_DISK	No FIX_DISK unless message from ADDISK; then use fast FIX_DISK
Forced shutdown – unsuccessful – no Locate flush	Full FIX_DISK	Fast FIX_DISK
Trapped Halt – DMX completed	Full FIX_DISK	Fast FIX_DISK
ECCU and IDC – DMX completed	Full FIX_DISK	Fast FIX_DISK
ECCU and NDC – DMX not completed	Full FIX_DISK	Fast FIX_DISK or Full FIX_DISK (See note in Chapter 7.)

Note With ECCU and Immediate halts and a nonintelligent disk controller (NDC), it is possible to have a corrupted data record on the disk if the NDC does not successfully write the record to disk. Fast FIX_DISK does not detect the corrupted data record on a robust partition if it is not one of the directory or extent map records. Full FIX_DISK detects the corrupted record and attempts to fix the record. If the record cannot be repaired and restored, full FIX_DISK either replaces the record with all zeros or, if you specify the –TRUNCATE option, truncates the file at the bad record.

If you use fast FIX_DISK and the corrupted record is a data record, a subsequent attempt to access the bad record (which is not fixed by fast FIX_DISK) results in an unrecoverable read error. If this happens, your application software may handle the error; if it does not, you should run either fast or full FIX_DISK with the –ADD_BADSPOT option to add the record as a badspot and then recover the record from backup.

FIX_DISK Options

FIX_DISK has an option, –CONVERT_22.1, that allows you to convert a Rev. 22.0 standard partition to a Rev. 22.1 format standard partition capable of supporting CAM files with unlimited numbers of extents as discussed earlier in this chapter under Converting Existing Partitions. If you convert a partition to a Rev. 22.1 format standard partition with FIX_DISK, you cannot then convert it to a robust partition.

FIX_DISK also has two options to support CAM files on robust partitions: –MIN_EXTENT_SIZE (abbreviation –MINSIZ) and –MAX_EXTENT_SIZE (abbreviation –MAXSIZ). These options set or change the extent sizes for Rev. 22.1 format robust and standard partitions in the same manner as described for MAKE. Thus you can change the maximum and minimum extent sizes when you run FIX_DISK.

ADDISK and Robust Partitions

If you attempt to add a robust partition that was not properly shut down because of a system halt and, thus, has a damaged file system, PRIMOS displays this message:

```
*** Robust Partition diskname has not been properly shutdown.
*** Fast Fix_disk has to be run before it can be added.
```

If it is necessary to immediately access the data on a robust partition that has a damaged file system, you can do so with the ADDISK –FORCE option. The –FORCE option forces a robust partition to be added even though the file system on the partition is no longer consistent. When a partition is added in this way, it is write-protected so that the data is available but no corruption can occur; that is, the PROTECT argument of ADDISK is implicitly invoked with –FORCE in this case. Adding a partition in this way is not considered normal operational procedure. When you do add the partition in this way, PRIMOS displays this message:

```
*** Proper shutdown of Robust Partition diskname did not take place.
*** The Partition will be write protected.
```

A partition added with the –FORCE option is write-protected because the consequences of forcing a robust partition to be added without running FIX_DISK on it are potentially serious. Errors such as doubly allocated records,which result in pointer mismatch errors, are not detectable on a robust partition during file system operations. This type of error could result in a large loss of data if a user updates files having this error after the partition is forcibly added. The only way to resolve this error is to run FIX_DISK on the robust partition before it is added to the system. After FIX_DISK is run, you can add the partition normally.

Mirroring Robust Partitions

Robust partitions may be mirrored. However, keep in mind the following results of mirroring partitions.

- If the primary partition is a standard partition, the secondary partition is also a standard partition after the copy server finishes.
- If the primary partition is a robust partition, the secondary partition is also a robust partition after the copy server finishes.

See Chapter 9 for complete information on disk mirroring.

ROBUST PARTITIONS EVALUATION FORM

Rather than filling in the original copy of this form, we suggest that you make a photocopy to work on.

Name: ______________________________ Date: ______________________________

Yes/No Do you run FIX_DISK after most system halts?

Yes/No Do you want to know when to run FIX_DISK?

Yes/No Are you willing to run FIX_DISK when required?

Yes/No Do you want to use FIX_DISK –FAST regularly?

Physical Device Number	*Partition Name*	*Use for Boot?*	*Percent Full*	*Data Management Packages*	*Record Verify Used?*	*Robust Conversion Decision*

8 Dynamic Badspot Handling

Generally, all physical disks, or spindles, have badspots. Previously, **static badspots** (badspots that exist on a disk when it is first partitioned) either have been found by MAKE during disk partitioning, entered by the Operator when running MAKE or FIX_DISK, or read by MAKE from a vendor flaw map. **Dynamically occurring badspots** (those static badspots that are marginally defective and missed by MAKE or those badspots that are actually developing due to disk media degradation both of which cause errors when the disk is in operation) have been handled by FIX_DISK only when a partition is shut down for repair, not while the partition is in operation.

At Rev. 21.0 and later, an intelligent disk controller (that is, a disk controller that is microprocessor-based) can handle both static badspots and dynamically occurring badspots. This process of **Dynamic Badspot Handling** allows PRIMOS to access an apparently error-free partition. Dynamic Badspot Handling can only be done by Model 6580 intelligent disk controllers and is done on an entire spindle (all partitions on the spindle).

This chapter discusses the following subjects in relation to Dynamic Badspot Handling:

- Requirements
- Disk error handling
- Rev. 21.0 and later disks
- Physical copying of disks
- Operator commands

Note The discussion in this chapter is pertinent to the Model 6580 (IDC1) intelligent disk controller only. The Model 7210 SCSI intelligent disk controller (SDIC) downloaded with ICOP+ software (SDIC_DISK.DL) handles badspots on its disks in a manner that is transparent to the user. Badspots on these disks are remapped by the SCSI disk drive. In addition, when you are creating or repairing a SCSI disk on a SDIC controller, MAKE and FIX_DISK ignore the -DBS ON/OFF (-IC and -AC) options.

Requirements for Dynamic Badspot Handling

To activate Dynamic Badspot Handling

- You must be using Rev. 21.0 or later PRIMOS.
- You must have an appropriate revision Model 6580 intelligent disk controller (IDC1) in the system.
- The Model 6580 intelligent disk controller (IDC1) must be downloaded with ICOP software (IDC1.DL) and in Dynamic Badspot Handling (DBS) mode.
- The partition must be created with –DBS ON (–IC).

Only these disk types are capable of having Dynamic Badspot Handling occurring on them:

SMD (300 or 80MB)

68MB	MODEL_4475 (315MB)
158MB	MODEL_4735 (496MB)
160MB	MODEL_4845 (770MB)
600MB	MODEL_4860 817MB)

Dynamic Badspot Handling is not supported with a nonintelligent disk controller or with an intelligent disk controller (IDC1) in nonintelligent mode. The conversion of a partition to Dynamic Badspot Handling (–DBS ON or –IC) mode from Nondynamic Badspot Handling (–DBS OFF or –AC) mode can be done only with an IDC1. Conversions of partitions from Dynamic Badspot Handling (–DBS ON) mode to Nondynamic Badspot Handling (–DBS OFF) mode can be done with any disk controller.

All pre-Rev. 22.1 format disks can be used on Rev. 21.0 and later systems. All disks processed by Rev. 21.0 and later MAKE or converted to Rev. 21.0 or later format by FIX_DISK contain the appropriate files for Dynamic Badspot Handling. At Rev. 21.0 and later, an IDC1 can handle all badspots by using the dynamic badspot (DBS) file.

DBS files are inactive on Rev. 21.0 and later disks connected to nonintelligent disk controllers or IDC1 disk controllers in nonintelligent mode. FIX_DISK marks the badspots as being in use so that PRIMOS does not write to them.

Use FIX_DISK to convert between the two modes, Dynamic Badspot Handling (–DBS ON) and Nondynamic Badspot Handling (–DBS OFF), of Rev. 21.0 and later disks. Although –DBS ON mode disks cannot be handled on a nonintelligent disk controller, a –DBS OFF mode (nonintelligent-disk-controller-compatible) disk can be used normally on the intelligent disk controller. However, it can neither be mirrored or used as a crash dump disk while in –DBS OFF mode nor does Dynamic Badspot Handling occur on it. (See Chapter 9 for a discussion of mirroring.)

On either type of disk controller (intelligent or nonintelligent), use MAKE and FIX_DISK as shown in Table 8-1 to get the Rev. 21.0 or later partition in the mode that you want, either Dynamic Badspot Handling (–DBS ON) or Nondynamic Badspot Handling (–DBS OFF).

Table 8-1. Using MAKE and FIX_DISK for –DBS ON and –DBS OFF Modes

Partition	*DBS Mode Desired*	*Utility*	*Remarks*
New	ON	MAKE	Must have Intelligent Controller Model 6580 (IDC1)
New	OFF	MAKE	
Existing	OFF to ON	FIX_DISK	Must have Intelligent Controller Model 6580 (IDC1)
Existing	ON to OFF	FIX_DISK	

Note You cannot add a Dynamic Badspot Handling (–DBS ON) mode partition to a file system if that partition is on a disk drive connected to a nonintelligent disk controller. You must convert it to –DBS OFF mode first.

Disk Errors

Errors that occur when a partition is in operation; that is, dynamically occurring badspots, are of two basic classes: **write errors** and **read errors**. Write errors, which are detected at the time of the write, prevent the data from being written to the disk.This type of error is usually due to a corrupted or marginal physical record header.

Read errors are of two types: **correctable** (or recoverable) and **uncorrectable**. PRIMOS handles the correctable type by attempting to read the record a specified number of times until the data is successfully read. If the read is not successful, PRIMOS attempts error correction. If either of these methods is successful, PRIMOS retrieves the correct data and does nothing about the error. The next time this record is accessed, PRIMOS may encounter an uncorrectable error. PRIMOS handles the uncorrectable type by simply displaying an error message and not retrieving the data.

Dynamic Badspot Handling is intended to address these two cases:

- Write errors
- Correctable read errors

Error Handling

Model 6580 (IDC1) intelligent disk controllers handle disk errors caused by badspots on the disk in the following ways.

Handling Disk Errors with Badspot Remapping: An IDC1 disk controller handles badspots by a technique known as **remapping**. Remapping is a way of keeping a list of all records on the disk that are bad, or unusable. A record may either be known to be bad initially (for example, from the flaw map), or found to be bad dynamically. In either case, once identified, a bad record is no longer used; the controller avoids using the bad record and stores the data that would otherwise reside in the bad record in a known good record allocated from a pool kept for this purpose.

This technique is called remapping because of the way the controller transparently diverts a reference to a known bad record address to the alternative good record address. PRIMOS continues to use the bad record addresses as if they were good; the controller checks every record address it is given against its list of bad records and diverts as necessary. In this way, PRIMOS sees an apparently error-free disk. Because references to the bad records are diverted, the bad records are said to be mapped out (in MAKE's badspot summary, for example).

The pool of good records is known as the **remap area**, or RMA. There is one RMA per spindle, and it always resides on the head zero partition. The list of known bad record addresses is kept in the DBS file, of which there is also one per spindle on the head zero partition.

Adding a New Badspot: When the IDC1 is told or determines that a record address is bad, it adds the bad record address to the DBS file.

The DBS file is essentially a large table containing as many entries as there are RMA records. Each entry in the table contains a known bad record address and the RMA record to which the badspot is mapped. Table (DBS file) entries representing spare RMA records contain the RMA record address and an empty slot for a bad record address.

For a new badspot, the controller scans the DBS file for an entry representing a spare RMA record. The bad record address is inserted in that entry. This effectively adds the bad record address to the controller's list of bad record addresses that have been remapped. (The controller also maintains a copy of the DBS file's remapping table in its memory, for speed of access).

Every time an entry is created, the controller also writes the address of the good (RMA) record into the header of the bad record. This is a backup mechanism. The information written in this way, known as a **remap pointer**, can be used when the IDC1 is not running in DBS mode (for example, during a crash dump to disk, when PRIMOS is not running and the IDC1 cannot be downloaded). The remap pointers enable remapped records to be accessed even when the IDC1 is not in DBS mode. The remap pointers can only be removed by formatting the disk, which is why MAKE enables –FORMAT to reinitialize Dynamic Badspot Handling.

Write Errors and Correctable Read Errors: In the case of a write error, the IDC1 allocates an unused RMA record, as described above, and writes the data to the RMA record. The controller also links the bad record to the remapped record by writing the RMA record address in the header of the bad record.

In the case of the correctable read error, the IDC1 determines whether a record needs to be remapped either by retrying the read operation a specified number of times or by determining that error correction must be done. The controller then remaps the data to an unused record and writes the corrected data into the unused record. The controller then sets pointers in the bad record pointing to the remapped record. It is these pointers that must be removed by MAKE with the –FORMAT option when the DBS file is rebuilt or removed.

Uncorrectable Read-errors: Uncorrectable read-errors that occur dynamically cannot be handled by the controller. The data for the record containing an uncorrectable read-error is lost and the controller cannot recover it. If the controller were to remap the record immediately, the controller could access the new record but the data would be incorrect. Thus, an uncorrectable read-error is left for FIX_DISK to repair either when it normally scans the file system or when you use the –ADD_BADSPOT option. You must use the –ADD_BADSPOT option if the uncorrectable error occurs on a paging partition.

Although the controller cannot handle an uncorrectable read-error, it is probable that the first time the badspot is encountered, it produces a correctable read-error. In this case the controller remaps it. Thus, very few uncorrectable read-errors will be encountered.

Summary: To summarize, the purpose of Dynamic Badspot Handling is

- To provide Model 6580 (IDC1) intelligent disk controllers with enough information to effectively handle static badspots and dynamically occurring badspots
- To provide a mechanism to convert Rev. 21.0 and later disks to Dynamic Badspot Handling (–DBS ON) mode and to reverse this process
- To provide an apparently error-free disk medium to PRIMOS necessary for disk mirroring

Rev. 21.0 and Later Disks and Dynamic Badspot Handling

Rev. 21.0 and later disks contain two entities related to Dynamic Badspot Handling: the dynamic badspot (DBS) file and the remapped area (RMA). These files are located on the head zero partition of a spindle (the partition that contains the first surface, surface 0, of the spindle). These files are not visible and are not displayed in a listing of the MFD produced by an LD command.

There is also no badspot file named BADSPT on a disk with Dynamic Badspot Handling unless the disk is or was a Rev. 21.0 or later disk in Nondynamic Badspot Handling (-DBS OFF) mode and badspots exist on it. There is a file named DYNBSP in the MFD of the head zero partition of all Rev. 21.0 and later disks in either mode, whether or not badspots exist on the disk. This file is used only for security of the DBS file. It provides access checking of the DBS file by a specific ACL and provides a lock to control writing to the DBS file.

The addresses of all known badspots on an entire spindle are stored in the DBS file by MAKE or added to the DBS file by FIX_DISK. Any badspots encountered subsequently (dynamic badspots) are added to the DBS file by the disk controller or by FIX_DISK if the error is unrecoverable. The disk controller determines when to add a badspot to the DBS file based on the number of reattempted read operations and whether error correction is necessary.

If the DBS file becomes 80% or more full, the disk contains an unacceptable number of physical defects and a head crash is probably imminent. You are notified of a possible hardware problem by the following message from ADDISK when you add a partition on the disk to the system:

```
(Dynamic badspot remapping area is at least 80 percent full.
Disk may have a hardware problem.  Please have it checked.)
```

MAKE and FIX_DISK display similar messages. If you receive this message, contact PrimeService.

If the DBS file becomes full, the RMA is also full and no new badspots can be added, although existing mapped badspots are still handled normally.

Need for Conversion to Rev. 22.1-format Disks

It is necessary to convert partitions to Rev. 21.0 or later in Dynamic Badspot Handling (-DBS ON) mode to enable disk mirroring (see Chapter 9). To convert pre-Rev. 22.0-format partitions to Rev. 22.1 format, you must use MAKE. Rev. 22.0 partitions can be converted to Rev. 22.1 format with either MAKE or FIX_DISK.

Prior to Rev. 21.0, when badspots were marked in use in the DSKRAT, two partitions could not appear identical to PRIMOS unless they each had no badspots or unless they each had badspots in the same locations. (Both of these events are highly unlikely.) At Rev. 21.0 and beyond,with Dynamic Badspot Handling, the IDC1 makes entire spindles appear error-free to PRIMOS. The IDC1 does this by redirecting all attempted accesses to a badspot to a remapped record. Remapping is implemented on pre-Rev. 21.0 partitions when you use the –CONVERT_21 option of FIX_DISK to convert them to Rev. 21.0. FIX_DISK modifies the record header to indicate that the record is remapped.

Nonintelligent disk controllers cannot read the modified record headers. To prevent a nonintelligent disk controller from attempting to access badspots, FIX_DISK marks all badspots listed in the DBS file as being in use in the DSKRAT. Thus, the fact that the nonintelligent controller cannot read the modified header does not matter.

FIX_DISK can convert a Rev. 21.0 or later partition created in Dynamic Badspot Handling (–DBS ON) mode to make it compatible with nonintelligent disk controllers. FIX_DISK can also restore Rev. 21.0 and later partitions, which are compatible with nonintelligent disk controllers, to Dynamic Badspot Handling (–DBS ON) mode.

Recreating Rev. 21.0 or Later Partitions

If you want to create pre-Rev. 21.0 partitions on a disk that has been partitioned as a Rev. 21.0 or later disk in Dynamic Badspot Handling (–DBS ON) mode, you must use the –FORMAT option of MAKE to reinitialize the record headers. This is necessary because of differences in the way the IDC1 handles badspots. Thus, you must use the –FORMAT option of MAKE in these circumstances:

- The partition is Rev. 21.0 or later format and is in Dynamic Badspot Handling (–DBS ON) mode and you are recreating it either as a pre-Rev. 21.0 partition or as a new Rev. 21.0 or later partition.
- The disk is new or has not been used on a Prime system before.

When you recreate a Rev. 21.0 or later head zero partition, you are prompted to be sure that you want to use MAKE on existing partitions. This is because, when you change the geometry of the head zero partition or disable Dynamic Badspot Handling on the spindle, the DBS file and the remapped area are disturbed in a way that affects all partitions on the spindle. (See the following section discussing the first, or head zero, partition.) If you specify the –FORMAT option at this time, the prompt appears as follows:

```
A valid DBS file exists on partition pdev.  You have specified -FORMAT,
which will cause the DBS file to be rebuilt.

WARNING:  By disturbing the DBS file you risk loss of data on all other
          partitions on the spindle.  After this partition has been made,
          these other partitions should be remade with -FORMAT.

OK to continue with MAKE?
```

If you do not specify the –FORMAT option, MAKE enables the –FORMAT option in this case and prompts as follows, including the above warning:

```
Partition pdev appears to have been previously made with a different size.
The MAKE you have specified will destroy the existing DBS file.

MAKE will enable -FORMAT to create a new DBS file.

OK to continue with MAKE?
```

It is necessary to use the –FORMAT option in this case because the record headers contain remapping information and this information must be changed to reflect the new circumstances.

You should not use a version of MAKE that is at a revision earlier than the revision of the partitions you are recreating (for example, Rev. 20.0 MAKE on Rev. 21.0 and later partitions). Use Rev. 23.3 MAKE to create all partitions, no matter what revision you want the partition to be.

Reconstructing a Damaged Dynamic Badspot (DBS) File

If the DBS file somehow becomes inaccessible (very unlikely), you will have to do one of the following:

- Reenter the badspots manually
- Use MAKE to reconstruct the DBS file through normal badspot checking
- Use MAKE to reconstruct the DBS file from a vendor flaw map written on the disk (for disks that have this feature)

If it becomes necessary to rebuild the DBS file by one of the above means, you must remake all partitions on the spindle. For this reason, you must have a good backup procedure for your partitions.

The disk continues to function on an IDC1 if the DBS file is inaccessible, but if a new badspot occurs, the disk will stop functioning, or, if you want to convert to Nondynamic Badspot Handling(–DBS OFF) mode, you will be unable to.

The Head Zero Partition of a Rev. 21.0 or Later Spindle

The **head zero, or first, partition** (the partition containing surface 0) of a Rev. 21.0 and later spindle defines how all the partitions on that spindle are to be treated by the IDC1. If a Rev. 21.0 or later spindle with a head zero partition in Dynamic Badspot Handling (–DBS ON) mode has a partition on it that is Rev. 20.0 or pre-Rev. 20.0, that partition automatically has Dynamic Badspot Handling occurring on it.

The Rev. 20.0 or pre-Rev. 20.0 partition can continue to operate with an IDC1, but if pre-Rev. 21.0 utilities (such as MAKE, FIX_DISK, or COPY_DISK) are used on it, the utilities will fail. In addition, the Rev. 20.0 or pre-Rev. 20.0

partition will not function on a disk drive with a nonintelligent controller. Therefore, all partitions on a disk must be in the same mode: either Dynamic Badspot Handling (–DBS ON) or Nondynamic Badspot Handling (–DBS OFF). It is also recommended that all partitions on a disk be of the same revision.

WARNING Do not use a pre-Rev. 23.3 version of MAKE on any partition of a Rev. 22.1 format disk that has Dynamic Badspot Handling (–DBS ON) occurring on it. Do not use any version of MAKE with the –FORMAT option on any partition of a DBS spindle unless you remake all other partitions on the spindle, starting with the head zero partition. If you do either, the entire spindle (all partitions on the spindle) may be corrupted because MAKE overwrites the DBS file in these cases. Consequently, the data for records that initially were to be written to badspots on any partition of the spindle, and that were thus written to the RMA, are lost. Be sure all the data on all partitions on a spindle is backed up before using MAKE.

In addition, do not change the number of surfaces on an existing Rev. 22.1-format head zero partition. If you do, the basic pdev and geometry of the first partition are changed and all other partitions on that spindle lose any data that was remapped to the RMA. You must then run MAKE with the –FORMAT option on all partitions on the spindle.

Changing the Geometry of the Head Zero Partition: The head zero partition contains the DBS file and the RMA for all badspots on the entire spindle. If the DBS file and the RMA are damaged, any files having remapped records lose those records because the pointers to those records, or the records themselves, are lost. In addition, the DBS file and the RMA occupy a group of contiguous records immediately following the DSKRAT on the head zero partition and if the DSKRAT changes in size, the DBS and RMA must be rebuilt in a new location. The DBS file and the RMA use a small amount of disk space, which is unavailable to the file system in both modes, –DBS ON and OFF.

In order to change the geometry, or the basic pdev, of the head zero partition, follow this procedure:

1. Back up all files to magnetic tape or to disk.
2. Remake all partitions on the spindle starting with the head zero partition and continuing in order.
3. Restore all your files from the backup.

Handling Badspots in the RMA: Badspots that are physically in the RMA are also listed in the DBS file, which provides a record of them and prevents access to them. Badspots that occur dynamically in the RMA are handled in the same way; the original badspot (which was initially remapped to the RMA) is now remapped to a new remapping record, and the old remapping record (which is now a badspot) is mapped to itself to prevent any future access to it. Therefore, even if several records in the RMA degrade, no loss of performance results because the controller does not have to go indirectly to several records. It goes instead directly to the most recent remapping record.

Access by MAKE and FIX_DISK to the Head Zero Partition: MAKE and FIX_DISK require access to the head zero partition of a spindle when they are working on other partitions of that spindle because remapped records must be retrieved from the head zero partition without interference from the file system. The only way to do this is to have the head zero partition shut down and assigned. Thus, you must assign the head zero partition on the spindle to yourself when you use MAKE on other partitions on the spindle or when you use FIX_DISK to perform these tasks:

- Convert pre-Rev. 21.0 partitions to Rev. 21.0 or later format
- Convert a partition from Dynamic Badspot Handling (–DBS ON) mode to Nondynamic Badspot Handling (–DBS OFF) mode

In addition to assigning the head zero partition, the disk controller must be an IDC1 when you use FIX_DISK to do the following:

- Convert a pre-Rev. 21.0 partition to a Rev. 21.0 partition in Dynamic Badspot Handling (–DBS ON) mode
- Convert a Rev. 21.0 or later partition from Nondynamic Badspot Handling (–DBS OFF) mode to Dynamic Badspot Handling (–DBS ON) mode

Note If the head zero partition is the command device (COMDEV), use the FIX_DISK –COMDEV option to shut down and assign the head zero partition. (Remember to stop the Login server and DSM first.)

Using PSR, COPY_DISK, PHYSAV, and PHYRST

Pre-Rev. 21.0 versions of the utilities COPY_DISK, PHYSAV, and PHYRST do not recognize the DBS and RMA files that exist on Rev. 21.0 or later revision partitions. Thus, these utilities copy the entire disk, overwriting the DBS file of the target disk and causing the RMA to contain meaningless data.

Caution Use only versions of COPY_DISK, PHYSAV, and PHYRST to copy files to and from Rev. 21.0 and later partitions that are compatible with the revision of the partition with which you are dealing. Rev. 21.0 and later versions of these utilities do not copy the DBS file and RMA to the target partition and, thus, do not corrupt the DBS and RMA files of the target partition.

At Rev. 23.0, COPY_DISK, PHYSAV, and PHYRST were replaced by the PSR utility and they are no longer on the master disk. At Rev. 23.0 and later, you should use PSR for any physical copies.

Valid Data Transfers

COPY_DISK, PHYSAV, and PHYRST and their replacement, PSR, make data transfers between valid combinations of disk controllers and disk modes. Only these data transfers can be made:

- *From* a Dynamic Badspot Handling (–DBS ON) mode partition on an IDC1 *to* a Nondynamic Badspot Handling (–DBS OFF) mode partition on either type of disk controller
- *From* any Nondynamic Badspot Handling (–DBS OFF) mode partition *to* a second, third, or higher Dynamic Badspot Handling (–DBS ON) mode partition (a partition starting at other than surface 0 – not the head zero partition)
- *From* any Nondynamic Badspot Handling (–DBS OFF) mode partition *to* a head zero Dynamic Badspot Handling (–DBS ON) mode partition (a partition starting at surface 0)

If you copy from a non-head zero partition to a head zero partition, be sure the head zero partition has enough space for the data *plus* the DBS file and RMA, which are already on the head zero partition. The head zero partition must also already be a Rev. 21.0 or later partition of the correct geometry (that is, having the same number of surfaces as the partition being copied).

Invalid Data Transfers

You *cannot* make the following data transfers:

- *From* a Rev. 21.0 or later partition *to* a head zero partition that is not already of the same revision and the correct geometry (having the same number of surfaces as the partition being copied from)
- *From* a partition in Dynamic Badspot Handling (–DBS ON) mode connected to a nonintelligent disk controller *to* any partition
- *From* a non-head zero partition *to* a head zero partition

If you attempt the latter copy, you will see the error message `Cannot copy [restore] to a partition with a larger dynamic badspot remap area` and the copy utility will abort.

Effect of Dynamic Badspot Handling on Operator Commands

The three operator commands, ADDISK, MAKE, and FIX_DISK, are affected by Dynamic Badspot Handling as discussed in the following paragraphs.

ADDISK and Dynamic Badspot Handling

If you attempt to use ADDISK to add a Dynamic Badspot Handling (–DBS ON) mode partition from a disk drive with a nonintelligent disk controller, an error message is displayed stating that you must run FIX_DISK to convert the disk to –DBS OFF (–AC) in order to allow access to it. The partition is not added to the system. For example:

```
OK, ADDISK 40463
*** Disk 40463 - A Dynamic Badspot Handling disk cannot be
started on this controller.  Run FIX_DISK to convert to
an -All_Controller format disk.
ER!
```

If you add a Dynamic Badspot Handling (–DBS ON) mode partition to the system using ADDISK and the RMA has reached 80% or more of capacity, a warning message is displayed. This message warns you that an unusual number of dynamic badspots have occurred and that the disk may be approaching a head crash or some other hardware problem and should be checked. The disk is added, however. For example:

```
OK, ADDISK 103024
Dynamic badspot remapping area is at least 80 percent full.
Disk may have a hardware problem.  Please have it checked.
Starting up Revision 21 partition "EAGLE".
OK,
```

If you receive this message, contact PrimeService.

MAKE and Dynamic Badspot Handling

You use MAKE to create partitions having Dynamic Badspot Handling. If you do not specify either –DBS ON or –DBS OFF and the disk type supports Dynamic Badspot Handling, MAKE informs you that you can make this partition with –DBS either ON or OFF and informs you of the consequences of using either ON or OFF. Partitions made with –DBS ON can be mirrored and can be used as crash dump disks whereas partitions made with –DBS OFF cannot.

Use Nondynamic Badspot Handling (–DBS OFF) mode for removable disks (80MB and 300MB SMDs), provided that you do not want to mirror them, because these disks can be moved from one controller to another. Use Dynamic Badspot Handling (–DBS ON) for all other disks connected to an IDC1 so that the controller capability is used. These disks are the following:

68MB	158MB
160MB	600MB
MODEL_4475 (315MB)	MODEL_4735 (496MB)
MODEL_4845 (770MB)	MODEL_4860 (817MB)

When you use MAKE, you must create the head zero partition of a spindle (the partition that contains surface 0) before you create any other partition on that spindle. You create the head zero partition of the spindle first because the DBS file and the RMA must be on the head zero partition and the DBS and RMA are used for badspot handling on all other partitions of the disk. Prior to Rev. 21.0, it made no difference in what order you partitioned disks except if you wished to keep redundant badspot files. However, it is always good practice to start with the head zero partition and proceed in order and it is mandatory that you do this with partitions of the above spindles created for Dynamic Badspot Handling with the Model 6580 (IDC1) disk controller. See also the discussion starting on page 5-4.

The format of the MAKE command to create partitions having Dynamic Badspot Handling is

MAKE –DISK *pdev* –DISK_TYPE *type* –DBS ON

Use the correct disk type with the –DISK_TYPE option or when MAKE prompts you for this information. MAKE needs to know the type of spindle to get the disk geometry information in order to build the correct size DBS file and RMA. You may also use the –FORMAT and –NEW_DISK options if this spindle has not been used on a Prime system before. (See the previous section Creating Pre-Rev. 21.0 Partitions.) If you do not specify the –FORMAT option when changing the geometry of the partition, MAKE will display a message about enabling –FORMAT. The –FORMAT option has special ramifications on partitions supporting Dynamic Badspot Handling because of its role in reinitializing the modified record headers. Special messages having to do with formatting are illustrated on page 8-7.

To create a pre-Rev. 22.1 format partition, use the –DISK_REVISION option with either –DBS OFF or –DBS ON. If you use the –DBS OFF or –DBS ON mode switches with a –DISK_REVISION parameter less than 21, MAKE displays an error message and reprompts for the disk revision. For example:

```
OK, MAKE -DISK 42022 -DISK_REVISION 20 -DBS ON

[MAKE Rev 23.3  Copyright (c) 1992, Prime Computer, Inc.]
Options -DBS ON (-IC) and -DBS OFF (-AC)
are not available with
a Disk Revision earlier than 21.

Disk revision?
```

If you do not specify the –DISK_REVISION option with either –DBS ON or –DBS OFF, MAKE creates a Rev. 22.1-format partition.

FIX_DISK and Dynamic Badspot Handling

FIX_DISK leaves a partition in its current mode unless you specifically instruct FIX_DISK to convert to either Nondynamic Badspot Handling (–DBS OFF) mode or to Dynamic Badspot Handling (–DBS ON) mode. You can also have FIX_DISK display a summary of the DBS file.

Caution In order to have FIX_DISK convert a pre-Rev. 21.0 partition to Rev. 21.0 or later format, you must convert the head zero partition (the partition that contains surface 0) before you convert any other partition on that spindle. This is necessary because the DBS file and the RMA must be on the head zero partition and FIX_DISK must have access to them to repair other partitions on the disk.

FIX_DISK does the following with respect to Dynamic Badspot Handling:

- Converts a pre-Rev. 21.0 partition to a Rev. 21.0 partition having either Dynamic Badspot Handling (–DBS ON) mode or Nondynamic Badspot Handling (–DBS OFF) mode
- Switches a Rev. 21.0 or later Dynamic Badspot Handling (–DBS ON) mode partition to a Nondynamic Badspot Handling (–DBS OFF) mode partition
- Switches a Rev. 21.0 or later Nondynamic Badspot Handling (–DBS OFF) mode partition to a Dynamic Badspot Handling (–DBS ON) mode partition

---

Note To convert a partition to Dynamic Badspot Handling (–DBS ON) mode, the spindle must be connected to a Model 6580 (IDC1) disk controller.

You can use either MAKE or FIX_DISK to convert a partition from Rev. 20.0 to Rev. 21.0 format. You can use only MAKE to create a partition as a Rev. 22.0 format partition because there is no FIX_DISK option to convert partitions to Rev. 22.0. You can convert a Rev. 22.0 partition to a Rev. 22.1 standard partition with FIX_DISK. In order to convert any partition to a robust partition, you must use MAKE to create a Rev. 22.1-format standard partition first. See Chapter 7 for details on robust partitions.

For simple conversions involving only a change in controller mode and involving no disk geometry, or basic pdev, changes, FIX_DISK is fastest to use. To convert a partition from Rev. 19.0 or earlier to Rev. 21.0 format with FIX_DISK, it is necessary to convert to Rev. 20.0 first, because of hashed directories, and then convert to Rev. 21.0 format. See Chapter 5 for a discussion of converting pre-Rev. 21.0 partitions to Rev. 21.0 or later partitions.

If you are converting a partition from pre-Rev. 21.0 to Rev. 21.0 with the –CONVERT_21 option and you do not specify the controller mode with either –DBS OFF or –DBS ON, FIX_DISK determines the type of disk controller that is connected to the disk drive and sets the partition controller mode accordingly. Rev. 21.0 format partitions are compatible with Rev. 22.0- and Rev. 22.1-format standard partitions.

FIX_DISK Conversion to Nondynamic Badspot Handling (–DBS OFF) Mode: Use FIX_DISK to convert a partition that is in Dynamic Badspot Handling (–DBS ON) mode to a partition in Nondynamic Badspot Handling (–DBS OFF) mode. To make this conversion, FIX_DISK must relocate file records, which were remapped to the RMA in Dynamic Badspot Handling mode, to the partition containing the parent file.

Note To convert a partition, you must use the ASSIGN command to assign to yourself the head zero partition (the partition containing surface 0) of the spindle and the partition to be converted. If the head zero partition is the command device, use the –COMDEV option to shut down and assign it.

To convert to –DBS OFF mode, use the FIX_DISK format

FIX_DISK –DISK *pdev* –FIX –DBS OFF

FIX_DISK Conversion to Dynamic Badspot Handling (–DBS ON) Mode: To convert a Rev. 22.1-format Nondynamic Badspot Handling (–DBS OFF) mode partition to a format that allows full utilization of Dynamic Badspot Handling and mirroring on an IDC1 disk controller, use FIX_DISK to convert and allow access to the badspots defined in the DBS file.To convert a partition to Dynamic Badspot Handling (–DBS ON) mode, the spindle must be connected to a Model 6580 (IDC1) disk controller.

To convert to –DBS ON mode, use the FIX_DISK format:

FIX_DISK –DISK *pdev* –FIX –DBS ON

Note You must use the FIX_DISK –COMDEV option followed by the pdev of the command device under the following conditions:

- You are converting a partition to –DBS ON mode, either from a pre-Rev. 21.0 format or by switching a partition from –DBS OFF mode and
- The partition you are converting or switching is not the head zero partition (the partition containing the first surface of the spindle) but
- The head zero partition is the command device (COMDEV – ldev 0)

You should stop the Login server and DSM before using the –COMDEV option.

For example, if you are switching a Rev. 22.1-format partition from –DBS OFF mode to –DBS ON mode and the partition is the second partition on a spindle that contains the command device as logical device 0 (ldev 0), specify the –COMDEV option in a command line like this to tell FIX_DISK where to find the command device where the DBS file is located.

```
OK, FIX_DISK -DISK 71020 -FIX -DBS ON -COMDEV 3420
```

FIX_DISK Display of DBS File

You can display the dynamic badspot (DBS) file and save the display in a COMO file. This can be useful if the DBS file is somehow damaged because you can then use FIX_DISK or MAKE to manually reenter the badspots.

To display the DBS file, use the FIX_DISK option –DUMP_DBS (abbreviation –DDBS) in this command format:

FIX_DISK –DISK *pdev* –DUMP_DBS

In order to display the DBS file of a disk, you must have the head zero partition of that disk assigned to you. You can display the DBS file without specifying the –FIX option as in the example above; in this case, use the pdev of the head zero partition.

You can also have the DBS file displayed while you are repairing some other partition on the spindle. It is displayed when FIX_DISK finishes the repair operation. If the disk is in Nondynamic Badspot Handling (–DBS OFF) mode, the DBS file is displayed but it may not be up-to-date; it is updated in Dynamic Badspot Handling (–DBS ON) mode.

After FIX_DISK has finished repairing files or updating the badspot file, the DBS file is displayed in the following tabular format.

```
DBS file version version_number.
DBS file has number_of_records records.
Number of badspots = number_of_badspots.
File last modified by controller/Primos.      /* No Line if no changes

Badspot                         Remap Record
Cyl,  Head, Sector              Cyl,  Head, Sector
------------------              ------------------

   badspot_1                       remap_1
   badspot_2                       remap_2
   badspot_3                       remap_3
      .                               .
      .                               .
      .                               .
```

Messages Associated With Dynamic Badspot Handling

When the disk controller remaps a record, a disk error message is displayed at the supervisor terminal. (See Appendix D for the message format.) At the end of the message is the comment `Bad record remapped.` indicating that the controller has remapped the record.

If you specify the FIX_DISK –DBS ON option and you are repairing the head zero partition, the following message is displayed:

```
Activating DBS file
```

Otherwise, this message is displayed:

```
Deactivating DBS file
```

Appendices A, B, and C list other messages associated with Dynamic Badspot Handling.

9 Disk Mirroring

This chapter discusses **disk mirroring**, the use of two logically equivalent partitions with identical data written to them.

For many applications, any circumstance that causes processing of data to stop is undesirable. In the case of a disk failure, recovering from the failure requires downtime. The application must be stopped, the disk must be repaired or replaced, and recovery procedures must be run to restore the system to its original state.

Purpose of Mirroring

The purpose of disk mirroring is to increase system availability by making it possible to process with pairs of logical disks. These logical disks are equivalent such that if one fails, the other is an exact duplicate and is available for use. The transition to the use of the duplicate disk is automatic.

Disk mirroring allows PRIMOS to

- Mirror partitions on different disk drive units (which thus have different disk drive unit numbers) of the same disk controller
- Mirror partitions on disk drive units that have the same disk drive unit numbers but are on different disk controllers
- Mirror partitions on different disk drive units of different disk controllers
- Continue processing on one partition if the other fails
- Copy a partition as a background process while the partition pair is being mirrored (a catch-up copy)

When you mirror partitions, all records written to a partition, called the **primary partition**, are also written to another partition, called the **secondary partition**. Thus, all write operations are duplicated.

Reading of records is not duplicated. Reading is split so that the records in the first half of the partition are read from the primary partition and the records in the second half are read from the secondary partition. This process reduces the average time it takes to read a record (compared to reading all records from one of the partitions) because the average seek time is reduced.

Mirroring Requirements

The requirements for disk mirroring are as follows:

- Both the primary partition and the secondary partition must be in disk drives associated with downloaded intelligent disk controllers that are capable of dynamic badspot handling; that is, the Model 6580 (IDC1) for SMD disks and the Model 7210 SCSI disk/tape controller downloaded with ICOP+ for SCSI disks.

Note If the primary partition and the secondary partition are on different disk controllers, the controller is eliminated as a common point of failure. In addition, performance improves when you are mirroring partitions on different disk controllers.

- The two partitions must be Rev. 21.0 or later partitions.
- The two partitions must be in Dynamic Badspot Handling (–DBS ON) mode if they are associated with a Model 6580 (IDC1) disk controller or they must be on a Model 7210 SCSI disk/tape controller downloaded with ICOP+ so that Dynamic Badspot Handling can take place on them.
- The two partitions must be on the same model disk; that is, they must be on the same physical disk, or spindle, types.
- The two partitions must be identical with respect to size (number of surfaces) and position (starting surface number) on the spindles. (They thus will have identical basic pdevs before the pdev is modified for disk drive unit number and disk controller address.)
- A maximum of 128 partitions can be mirrored at one time; that is, there can be a maximum of 64 pairs of mirrored partitions.
- Assigned partitions cannot be mirrored.
- It is not possible to mirror both the paging portion and the file system portion of a split partition. Generally this means that only the paging portion can be mirrored because you start the paging mirror at system startup by a configuration directive. In addition, if the paging portions of two partitions are mirrored, it is *not* possible to add the file system portion of either partition with the ADDISK command.

- One or more of the following directives must be in the configuration file. (See the section Configuration Directives for Mirroring below.)

```
MIRROR
COMDVM pdev
PAGINM pdev1 [. . . pdev8]
```

- You can mirror robust partitions; however, the type of partition that results (either standard or robust) depends on what the primary partition is. See Mirroring and Robust Partitions in Chapter 7.

Since the catch-up copy facility in the mirroring process makes a physical copy of the primary partition that you want to mirror to the secondary partition, the resulting secondary partition becomes the same revision (either Rev. 21.0, Rev. 22.0, or Rev. 22.1) and the same type of partition (standard or robust) as the primary partition.

Performance

If you mirror one partition of a spindle, you should mirror every partition on that spindle for best performance. In addition, configure each of the two partitions of a mirrored pair on different disk controllers, if possible. This provides better reliability and performance because if mirrored partitions, and thus their spindles, are associated with a single controller, the controller can be a single failure point for both partitions.

Caution You can mirror only some of the logical partitions on a spindle. However, doing this will have a negative performance impact if there is much activity on the nonmirrored partitions. It is thus strongly recommended that you mirror all the partitions on a spindle if you plan to mirror any partitions on that spindle.

Configuration Directives for Mirroring

You can activate mirroring or initiate mirroring of the command device (COMDEV - ldev 0) or the paging devices at system startup by the use of directives in the configuration file. If you want to mirror any partitions, at least one of the following directives must be in the configuration file:

```
MIRROR
COMDVM pdev
PAGINM pdev1 [. . . pdev8]
```

Using MIRROR

Use MIRROR if neither the command device (COMDEV) nor any of the paging devices is to be mirrored at system startup but you want to be able to mirror partitions after startup. If the MIRROR directive is in the configuration file, you can then mirror any partitions after system startup.

If either or both of the directives COMDVM and PAGINM are in the configuration file, mirroring is turned on at cold start for the command device (COMDEV) in the case of the COMDVM directive and for the paging devices in the case of the PAGINM directive. In addition, these directives have the same effect as the MIRROR directive; that is, file system partitions can be mirrored after system startup.

Using COMDVM

Use COMDVM to initiate mirroring of the command device (COMDEV) at system startup. The COMDVM directive must follow the COMDEV directive in the configuration file. For example:

```
COMDEV 2420 COMDVM 2520
```

You specify the pdev on which to mirror the COMDEV with the COMDVM directive. If the specified pdev cannot be used for any reason, a message is displayed and the COMDEV starts without mirroring. You can start mirroring of the COMDEV after system startup by using the MIRROR_ON command (described later in this chapter).

Using PAGINM

The PAGINM directive must follow the PAGING directive to mirror paging devices at system startup. It is not possible to set up mirroring in conjunction with the pre-Rev. 21.0 PAGDEV or ALTDEV directives.

Use the PAGINM directive to specify the pdevs (*pdev1 . . . pdev8*) on which to mirror the corresponding pdevs specified with the PAGING directive. If a pdev used in the PAGING directive does not have a mirror, a 0 must be used in the corresponding position in the PAGINM directive. For example:

```
PAGING 1020 100421 71023 100463
PAGINM 1060 0 71065 0
```

In this example, the second (100421) and fourth (100463) paging partitions are not mirrored, whereas device 1020 is mirrored on an identical partition on disk drive 0 associated with disk controller 1 at address 26_8 (1060), and device 71023 is mirrored by an identical partition on disk drive 2 associated with disk controller 1 at address 26_8 (71065).

If a mirrored paging partition specified by *pdev* cannot be used for any reason, a message is displayed and the associated paging partition starts without a mirror. You can still start a mirror for the paging partitions by using the MIRROR_ON command after the system has started; a catch-up copy then starts.

Catch-up Copy

The catch-up copy facility is a mechanism for making a *physical* copy of the primary partition on the secondary partition while mirroring is in effect. The catch-up copy facility allows a mirrored pair to be set up after a primary partition has begun operation. In this case, you are prompted to be sure you want the catch-up copy to start. This facility also allows two partitions that are started together but are not identical to be updated so that they are identical.

WARNING The copy server copies *from* the primary partition to the secondary partition. Be sure that you know which is the primary and which is the secondary partition when you use the MIRROR_ON command. You could destroy current data if you use the command incorrectly.

If your primary partition is not up-to-date (it is the older of the two mirrored partitions), change the positions of the pdevs in the MIRROR_ON command so that the secondary is now the primary. If you start up the mirrored pair with the older partition as the primary, the copy server copies the older data to the secondary. The more recent data of the secondary is thus destroyed. It is especially important to change the positions of the pdevs in this case if the MIRROR_ON command is in your PRIMOS.COMI file and you cold start your system.

If two partitions to be mirrored are started simultaneously and the partitions have different shutdown stamps indicating that the partitions were not shut down at the same time and, thus, are not identical, the catch-up copy facility starts automatically. However, paging partitions are not copied when they are started simultaneously at cold start. In all cases, the shutdown stamp is set at shutdown and cleared at startup.

The catch-up copy is accomplished by a server while the partitions are on line. The server reads a record from the primary partition, writes that record to the secondary partition, and then advances to the next record on the primary partition. The server activity is prevented from interfering with reading and writing by PRIMOS on the mirrored partitions. Until the catch-up copy is complete, records are read by PRIMOS from the primary partition only. If the catch-up copy facility fails for any reason (for example, the copy server logs out), the mirror is disabled.

In place of using the catch-up copy facility, you can use PSR with the –COPY option to make the two partitions physically identical. However, in this case, the partitions must remain shut down until the copy is complete.

Startup of the Catch-up Copy Facility

The copy server phantom performs the catch-up copy. The copy server is started by the following sequence of actions:

1. The MIRROR_ON command is issued at the supervisor terminal or a configuration directive (such as COMDVM) that establishes a mirror is read at cold start.
2. The shutdown stamps from the DSKRAT of each partition in the mirrored pair are compared if the partitions are started simultaneously. (No shutdown check is made on paging partitions.) If the shutdown stamps are identical or if the two partitions are paging partitions being mirrored at cold start, there is no need to do a catch-up copy and mirroring starts.

   If the shutdown stamps are not identical or if the two partitions are not started simultaneously, step 3 is carried out.
3. You are queried to be sure that mirroring is desired. This is necessary because *the catch-up copy overwrites whatever data is on the secondary partition* with data from the primary partition.
4. The copy server process starts, updating the secondary partition of the mirrored pair.
5. A message is displayed upon successful completion of the copy. If the copy fails, a message is displayed indicating this failure and the mirror is disabled (breaks).

Effect of Mirroring on Operator Commands

Mirroring affects the STATUS command and the processing of the PRIMOS.COMI file and results in messages from the ADDISK and SHUTDN commands as discussed in the following paragraphs.

The STATUS DISKS Display

As shown in the example below, when you issue the STATUS DISKS command, a table of mirroring information is displayed if mirroring is in effect for any partitions on your system. The column headings having to do with mirroring are displayed if a mirroring directive is in your PRIMOS.COMI file whether or not mirroring is presently being done on any partitions.

The pdev of the primary partition is displayed in the `Pdev` field of the STATUS DISKS display unless a problem on the primary partition has caused an automatic switch to the secondary partition. In that case, the pdev of the secondary partition is displayed in the `Pdev` field. The status of each mirrored partition, either `Active` or `Inactive`, is displayed in the `State` field indicating which partition is being used. `Active, copying` means that a catch-up copy is in progress and `Active, copy needed` means that a catch-up copy is necessary but the catch-up copy process is not running yet.

If you issue the STATUS ALL command at the supervisor terminal, mirroring information is displayed for paging partitions; otherwise it is not. The format in either case is the same as in this example:

```
                                            Mirror
                                            Primary Secondary  State
Disk    Ldev  Pdev   System  Robust         ---------------------------------
ADMINS   0    4420                            4420       4622  Active
PURCHS   1  122420                          122420     122462  Active, copying
HUMRES   2   23463                           23423      23463  Inactive, primary off
ACCTGS   3   14321                           14321      14327  Inactive, secondary off
PAYROL   4   14121
MAINT    5  120421                          120421     120721  Active, copy needed
ENGRG1   6           ENG
BLDGS    7           ENG
GRNDS   10           ENG
```

Using the ADDISK Command

You use the ADDISK command to start partitions and to make file system partitions known to the system. ADDISK checks to be sure that a partition to be added is not currently in use in a mirrored pair. If the partition is in use in a mirror, the disk is not added and a message is displayed at the supervisor terminal:

```
PDEV pdev conflicts with assigned or paging or mirrored device.   (addisk)
```

If you attempt a warm start on your system and it is successful, you may encounter an error subsequent to the warm start when adding or mirroring disks that are associated with an intelligent disk controller. You may see the following message or a similar message from the MIRROR_ON command.

```
Could not OPEN PDEV pdev due to controller or device errors.   (addisk)
```

If this message does appear, wait until one of the following messages appears at the supervisor terminal.

```
DLL and init ICOP complete (cntrllr_address) - (disk_init)
DLL of 7210 controller complete.  (cntrllr_address) - (disk_init)
```

The intelligent disk controller is now downloaded again and you can now add or mirror the disks that you previously attempted to add or mirror.

This problem occurs only on disk drives associated with intelligent disk controllers and only on those disk drives where all the logical disks (partitions) on a spindle were shut down prior to the warm start.

Using the SHUTDN Command

You use the SHUTDN command to take a partition out of the file system. After you issue the SHUTDN command, PRIMOS checks to be sure that the partition to be shut down is not currently in use in a mirrored pair. If the partition is being mirrored, the partition is not shut down and the following message is displayed at the supervisor terminal:

```
Must use MIRROR_OFF to shut down mirrored PDEV pdev.  (shutdn)
```

In this case, use the MIRROR_OFF command (described later in this chapter) to shut down the mirrored partition. Note, however, that the SHUTDN ALL command shuts down mirrored partitions as part of the system shutdown process.

The PRIMOS.COMI File

If you initiate mirroring of the command device at startup with the COMDVM configuration directive or initiate mirroring of other file system partitions at startup by placing the MIRROR_ON command in the PRIMOS.COMI file, and the mirrored partitions are not identical such that a catch-up copy must start, you are queried with the following message to be sure that you want to continue:

```
A catch-up copy of primary device pdev1 to secondary device pdev2 will be
started.  Are you sure you want to continue?
```

The PRIMOS.COMI file stops and waits for your response. After you enter your response, PRIMOS.COMI continues with the startup process.

Note Although the PRIMOS.COMI file stops at this point, messages from the startup of other system servers and phantoms may cause the above prompt to scroll off the screen of a video display terminal. If you have the MIRROR_ON command in your PRIMOS.COMI file and your system appears to be hung, enter a return. If the PRIMOS.COMI file did stop for this reason, it will now continue. Mirroring does not start but you can manually start the mirroring process with the MIRROR_ON command later. In the case of mirroring the command device with the COMDVM directive, the prompt is issued before processing of the PRIMOS.COMI file.

PRIMOS Mirroring Commands

You can activate disk mirroring at system startup by using configuration directives or, while the system is running, by using the MIRROR_ON command. To turn off mirroring, use the MIRROR_OFF command. You can issue the MIRROR_ON and MIRROR_OFF commands, which are described in this section, only from the supervisor terminal.

Caution Do not use the MIRROR_ON or the MIRROR_OFF command while in the RESUS environment because they may request terminal input. See the *DSM User's Guide* for details.

The MIRROR_ON Command

To start the mirroring process, use the following command format.

MIRROR_ON *pdev1 pdev2* [{ -HELP | -PRIORITY_SELECT | -PRIVATE | -REPLACE }]

The arguments and options are described below.

pdev1 pdev2

Specifies the physical device numbers of the two partitions that are to be mirrored. The geometry of the two partitions must be the same; that is, they must have the same basic pdev. *pdev1* and *pdev2* cannot be assigned partitions; if either is assigned, an error message is displayed.

pdevs are not needed with the –HELP option. The requirements for mirroring partitions are described under Mirroring Requirements, earlier in this chapter.

pdev2 must not be known to the system, that is, it must not be a currently added file system partition, an assigned partition, or a paging partition, and it must not be mirrored currently. A message is displayed at the supervisor terminal if *pdev2* is part of a currently mirrored pair.

pdev1 can be unknown (in which case it is assumed to be a file system partition) or it can be a known (added) file system or paging partition that is not currently being mirrored.

–HELP
–H

Displays command usage similar to the above syntax. For an online explanation of the options, type HELP MIRROR_ON.

–MOUNT_PATH
–MP ***pathname***

Specifies a logical mount point for the partition.The mount point may be any existing, local directory in the tree structure except for an MFD. The mount point may also be in the root directory.

–PRIORITY_SELECT
–PRISEL

Allows you to forcibly mirror a local dual-ported disk, which was added to another system, after the other system has halted. A dual-ported disk is physically connected to two CPUs through the dual-porting option package but can be accessed by only one CPU at a time. If the CPU having the logical connection to the dual-ported disk halts, the other CPU can then access the disk. This provides for greater file system availability. Do not use this option if the other system is running.

WARNING Do not attempt to priority select a dual-ported disk if the other system has control of the disk drive. If you do, you may corrupt the data on the disk. You should only priority select a disk if your system has control of the dual-ported disk or if the other system halted while it had control of the dual-ported disk.

–PRIVATE

Allows the addition of a partition on the local system without it being able to be accessed by remote systems with remote file access (RFA).

–REPLACE

Forces all systems in the common file system name space to reference the system where you issue this MIRROR_ON command for the specified disk partition in place of the system where the partition was originally added. You must use –REPLACE whenever you use the –PRIORITY_SELECT option if the Name Server is running.

Caution Do not use the –REPLACE option unless you are certain that the disk was moved from one system to another without being explicitly shut down first.

The MIRROR_ON command is valid only if one or more of the following directives is in the configuration file when the system is started:

```
MIRROR
COMDVM pdev
PAGINM pdev1 [...  pdev8]
```

The following prompt is issued when a request is made to mirror a pair of partitions that are not identical, based either on their shutdown stamps or on not being started or stopped simultaneously.

```
A catch-up copy of primary device pdev1 to secondary device pdev2 will be started.
Are you sure you want to continue?
```

If you want the mirroring process to start and a catch-up copy to proceed as indicated, answer YES but be aware that the primary is always copied to the secondary. (See the warning in the Catch-up Copy section). If you start mirroring by using a COMINPUT file with the MIRROR_ON command, the COMI file stops and awaits your response from the supervisor terminal. The COMI file resumes processing after you respond.

Note While PRIMOS is awaiting your response, users cannot log in and the system may appear to be hung. When you use the MIRROR_ON command, you should promptly answer the above prompt.

The maximum number of partitions that can be mirrored is 128. If you attempt to start more than 64 mirrored pairs of partitions, an error message is displayed at the supervisor terminal.

The MIRROR_OFF Command

To stop the mirroring process, use the following command format.

MIRROR_OFF ***pdev1 pdev2*** { –SHUT_BOTH [–FORCE] | –SHUT_PRIMARY | –SHUT_SECONDARY | –HELP }

You must specify one, and only one, of the four options. If the copy server is active (that is, a catch-up copy of the primary partition to the secondary partition is in progress), you cannot use the –SHUT_PRIMARY option but you can use either the –SHUT_SECONDARY or the –SHUT_BOTH option. If you use either the –SHUT_SECONDARY option or the –SHUT_BOTH option, PRIMOS attempts to log out the copy server. If PRIMOS cannot log out the copy server, an error message is displayed. You must then log out the copy server and reissue the MIRROR_OFF command.

pdev1 pdev2

Specify the physical device numbers of the two partitions that are presently mirrored and that you want to shut down as a mirrored pair. Not needed with the –HELP option.

pdev1 and *pdev2* must be a mirrored pair; if they are not, an error message is displayed at the supervisor terminal. *pdev1* must be the primary partition and *pdev2* must be the secondary partition; if they are not, an error message is displayed at the supervisor terminal. Use the STATUS DISKS command or the LIST_DISKS –LOCAL command to list the pdevs of all currently mirrored partitions.

–SHUT_BOTH
–SB

Turns off mirroring and shuts down both partitions, giving them identical shutdown stamps. If mirroring was already disabled, an error message is displayed at the supervisor terminal and the two partitions do not get identical shutdown stamps.

This option is invalid with paging partitions. Paging partitions can be shut down only at system shutdown with the SHUTDN ALL command. If paging partitions are specified, an error message is displayed.

–FORCE
–F

Operates like the –FORCE option of the SHUTDN command. –FORCE makes it possible to shut down a partition even if problems prevent it from being read. Use only with MIRROR_OFF –SHUT_BOTH.

When you use the –FORCE option and disk errors occur, the shutdown stamps may not be updated; thus, the next attempt to mirror the pair requires a catch–up copy. The display of disk error messages immediately after MIRROR_OFF –SHUT_BOTH –FORCE indicates that the shutdown stamps are not identical on both partitions. The partition for which the error message occurs is now the older partition and the other is the most up-to-date and should now be used as the primary partition.

–SHUT_PRIMARY
–SP

Turns off mirroring, if it was on, and shuts down the primary partition. A subsequent attempt to mirror this primary partition and secondary partition pair requires a catch-up copy because the shutdown stamps cannot match since the two partitions were not shut down at the same time.

Caution The primary partition is now the older of the two and a catch-up copy results in the primary being copied to the secondary. Thus, make the secondary partition the primary partition in a subsequent MIRROR_ON command.

The –SHUT_PRIMARY option is valid only if mirroring is on or if the primary partition is inactive; otherwise an error message is displayed at the supervisor terminal. The secondary partition is now indistinguishable from an ordinary file system or paging partition. However, the primary partition is left in an indeterminate state because a user may have been writing to a file and the file is not saved. Using the –SHUT_PRIMARY option in this way is the same as using the SHUTDN command to shut down a file system partition without warning and logging out users so that they can close all files. If you use this option, also use normal shutdown procedures.

If read requests or write requests are pending on the primary partition after five seconds, an error message is displayed. This should rarely occur. If it does, this command option can be reissued until it works.

–SHUT_SECONDARY
–SS

Turns off mirroring, if it was on, and shuts down the secondary partition. A subsequent attempt to mirror this primary partition and this secondary partition requires a catch-up copy because the shutdown stamps cannot match since the partitions were not shut down at the same time. This option is valid only if the the mirror is on or if the secondary partition is inactive; otherwise an error message is displayed. The primary partition is now indistinguishable from an ordinary file system or paging partition. However, the secondary partition is left in an indeterminate state because a user may have been writing to a file and the file is not saved.

Using the –SHUT_SECONDARY option in this way is the same as using the SHUTDN command to shut down a file system partition without warning and logging out users so that they can close all files. If you use this option, also use normal shutdown procedures.

If read requests or write requests are pending on the secondary partition after five seconds, an error message is displayed. This should rarely occur. If it does, this command option can be reissued until it works.

–HELP
–H

Displays command usage similar to the above syntax. For an online explanation of the options, type HELP MIRROR_OFF.

When you use the –SHUT_PRIMARY or the –SHUT_SECONDARY option, PRIMOS displays the following message and prompt to be sure that you intend to break the mirror.

```
The mirror of primary device pdev1 and secondary device pdev2 will be broken.
Are you sure you want to continue?
```

While PRIMOS is awaiting your response, users cannot log in and the system may appear to be hung. When you use the MIRROR_OFF command, you should promptly answer the above prompt.

Errors

If one of the two mirrored partitions fails, all records are read from the remaining good mirrored partition (the **survivor**). Similarly, records are written only to the survivor. This assumes that the mirrored partitions are identical and a catch-up copy is not in progress at the time of the failure.

The failure of one of the mirrored partitions and disabling of the mirroring process is referred to as **breaking the mirror**. Users are not aware of the transition. However, the STATUS DISKS display and messages displayed at the supervisor terminal indicate that the transition occurred. Failures on the remaining partition are handled like failures on any unmirrored partition.

As noted previously, mirroring can be done only on disks connected to intelligent disk controllers that are capable of dynamic badspot handling such as the Model 6580 (IDC1) and the Model 7210 SCSI disk/tape controller running ICOP+. This means that PRIMOS does not encounter correctable read errors, correctable write errors, or uncorrectable write errors.

If PRIMOS does encounter one of the above error conditions (such as during the initial stage of a cold start or briefly after a warm start, before the controller is downloaded), PRIMOS does the following:

- Handles correctable read errors and write errors but does not break the mirror
- Handles uncorrectable write errors and breaks the mirror
- Handles uncorrectable read errors, but breaks the mirror if uncorrectable read errors occur on both partitions of the mirrored pair

The following failures break the mirror:

- An attempt to write to a write-protected disk
- An uncorrectable write error

- An uncorrectable read error while PRIMOS is attempting to access a record from both partitions of the mirrored pair
- Any failure of the copy server that performs the catch-up copy

Very few errors should occur on mirrored partitions because dynamic badspot handling is active on these partitions. Other situations that require recovery action, such as a system halt, occur infrequently. In general, then, the recovery procedures outlined below should rarely be necessary.

Error Recovery for Write Errors

If an uncorrectable write error occurs, the mirror is disabled, or broken, and the system continues to operate on the good partition. The uncorrectable write error occurs only under the following conditions. Solutions are described in the following paragraphs.

The first three items apply only to disks associated with the Model 6580 (IDC1) disk controller, and not to SCSI disks associated with the Model 7210 (SDIC) disk controller running ICOP+.

- The write-protect switch is set.
- The controller is in Nondynamic Badspot Handling mode.
- The DBS/RMA of the partition is full.
- Physical problems, such as loose cables, exist.

What to Do If the Write-protect Switch Is Set: Perform the following steps for every partition on the spindle where a write error occurred.

1. Turn off the write-protect switch.
2. Use MIRROR_OFF to take the partition having the error out of the mirror.
3. Use MIRROR_ON to restart the mirror. A catch-up copy will start.

What to Do If the Controller Is In Nondynamic Badspot Handling (–DBS OFF) Mode: The IDC1 controller should be in Nondynamic Badspot Handling mode only during the initial stage of a cold start and for less than a minute after a warm start. You can determine that the write error occurred while the controller was in Nondynamic Badspot Handling mode by using the DISPLAY_LOG command on the system event log file. If this is the case, perform the following steps for every partition on the spindle where a write error occurred.

1. Take the damaged partition out of the mirror with MIRROR_OFF.
2. Restart the mirror with MIRROR_ON. A catch-up copy will start. The record where the write error occurred is remapped by the IDC1 controller in Dynamic Badspot Handling (–DBS ON) mode.

What to Do if the DBS/RMA of the Partition Is Full: The following message probably was displayed when the system was started and the partition was added to the system:

```
Dynamic badspot remapping area is at least 80 percent full.
```

This message indicates that the observed uncorrectable write error occurs because the area where the dynamic badspots are remapped is nearly full. Verify that this area is full by invoking FIX_DISK with the –DUMP_DBS option on the first partition. A full DBS/RMA area indicates unacceptable quality of the spindle and you should not use it. Install a replacement disk. You can reestablish the mirror by using the MIRROR_ON command.

Note Whereas a removable disk pack (SMD) can be replaced easily, replacing an FMD requires shutting down all the disk drives on the same controller. Strategic use of multiple disk controllers in conjunction with mirroring makes it possible to shut down all the drives on one controller and replace a drive without shutting down the system. Thus, be sure that mirrored partitions are on different disk controllers if possible.

There is a remote possibility that the area used by the SDIC SCSI disk controller could fill up. If it does, a message will be logged containing logical status word 5 (LSW5) as explained in Appendix D. If this does occur, it indicates that the disk has other problems, such as a physical problem, and you may need to call your PrimeService representative.

What to Do if Physical Problems Exist: If an uncorrectable write error occurs and the IDC1 disk controller is in Dynamic Badspot Handling (–DBS ON) mode and the remapped area is not full, the problem is probably physical. The occurrence of several uncorrectable write errors on more than one partition further suggests a physical problem on disks associated with either the IDC1 or the SCSI disk controller. If this is the case, follow this procedure:

1. Take the damaged partition out of the mirror with MIRROR_OFF.
2. Correct the problem.
3. Restart the mirror with MIRROR_ON. A catch-up copy will start.

Perform these steps for every partition on the spindle where a write error occurred.

Error Recovery for Read Errors

Uncorrectable read errors are not handled by the dynamic badspot mechanism. However, since uncorrectable read errors generally start out as correctable read errors and since dynamic badspot handling remaps correctable read errors, very few uncorrectable read errors should be encountered. If PRIMOS encounters an uncorrectable read error on one of the mirrored partitions, PRIMOS retrieves the data from the other mirrored partition and attempts to write the data to the first partition. This process causes the bad record to be added to the badspot file and the data is remapped to another location. The mirror is not broken unless the same record is unreadable on both partitions of the mirrored pair. In the latter case, use the following procedure.

What to Do If the Mirror Breaks With an Uncorrectable read error for the Same Record on Both Partitions: If the mirror should break while the partition pair has an uncorrectable read error on both partitions on the same record (a very unlikely event), do the following:

1. Use the MIRROR_OFF command with the –SHUT_BOTH option to stop the mirror and use the SHUTDN *pdev* command to shut down both partitions so that no new data can be written to the primary partition.
2. Run FIX_DISK on one of the partitions. On non-SCSI disks (disks *not* associated with the SDIC disk controller), use the –ADD_BADSPOT option to add the bad record to the badspot file. Keep a list of any files that were damaged as reported by FIX_DISK.
3. Run FIX_DISK on the other partition of the pair. On non-SCSI disks (disks *not* associated with the SDIC disk controller), again use the –ADD_BADSPOT option to add the bad record to the badspot file. Keep a list of any files that were damaged on that partition as reported by FIX_DISK. The damaged files are the same files on both partitions.
4. Recover the damaged files from a recent backup to magnetic tape or other media to the primary partition. Inform the owners of the damaged files of the backup.
5. Use the MIRROR_ON command to restart the mirror; a catch-up copy will start.

Note If any records were written to files after the mirror broke, the data from the secondary partition will be stale, or not up-to-date, since writing occurs only on the primary partition after the mirror is turned off. The secondary partition is made up-to-date when the mirror is restarted with the MIRROR_ON command.

Catch-up Copy Failure

Any uncorrectable read error or write error encountered by the catch-up copy process causes the mirror to break. If the error occurs while *writing* a record to the secondary, recover by following the above procedures for write errors. If the error occurs while *reading* a record from the primary, recover by running FIX_DISK on the primary partition and, on non-SCSI disks (disks *not* associated with the SDIC disk controller), use the –ADD_BADSPOT option to add the bad record to the badspot file. In both cases, use the MIRROR_OFF command before using the recovery procedures. The mirror can then be restarted with the MIRROR_ON command and the catch-up copy starts.

If the mirror breaks due to some other problem or condition encountered by the copy server (for example, the copy server was logged out at the supervisor terminal), the mirror can be restarted with the MIRROR_ON command and the catch-up copy starts.

System Halts

If, during a system halt, PRIMOS is able shut down in an orderly manner, mirrored partitions will have the same shutdown stamp and can be started without any special recovery action. If PRIMOS is not able to shut down in an orderly manner, the shutdown stamps will not match and a catch-up copy will start when the mirror is turned on, even if no errors are revealed when FIX_DISK is run on the primary partition.

Mirroring Messages

Informational and error messages associated with the mirroring process and the mirroring commands are presented and discussed in Appendix C.

10 Record Allocation and Sectoring

PRIMOS uses two methods for allocating disk sectors, or records, for files: forward sectoring and reverse sectoring. This chapter describes these methods.

PRIMOS Record Allocation

The PRIMOS disk record allocation algorithm maintains a logical ordering of disk records. The algorithm uses bits stored in the DSKRAT to represent the entire set of records on a partition. Records are grouped in logical sets of 16 for allocation.

The actual physical location of these records on the disk is dependent on the physical configuration of the disk, such as the number of surfaces in the partition, the number of cylinders, or tracks, on the disk, and the number of sectors in a track. Records within the group of 16 are allocated to one or more tracks. When a given track is filled, the track on the next surface is used. The entire cylinder is allocated in this manner. When this cylinder is fully allocated, the next cylinder is used.

Records are allocated in ascending order. The records are interleaved with a preset interleave factor. The **interleave factor** is the number of sectors that are skipped before reading or writing the next record. The interleave factor allows PRIMOS enough time to process the record just read and then to issue a request for the next sequential record to be read before the sector where the next record is located comes under the read head (the read head is **on sector**).

Forward Sectoring

In **forward sectoring**, PRIMOS allocates a physical sector for a record, then skips two sectors and allocates the next sector. This skipping is referred to as the interleave factor and is used to account for the time it takes the disk controller to write one record and return to the disk to write the next record. By that time, the next sector under the disk read/write head should be three ahead of the last sector. For example, with forward sectoring, if the first record is written to physical sector 5 (see A in Figure 10-1), the second record is written to physical sector 8.

Forward sectoring records on disk is suited to nonintelligent disk controllers and to the Model 7210 (SDIC) intelligent controller on SCSI disks with pseudo-geometry. The nonintelligent disk controller is not capable of buffering records. A different scheme is useful for intelligent disk controllers that can **buffer**, or **cache**, records (temporarily store) that pass under the read head before or after the read head arrives on the desired record. For intelligent disk controllers, records are allocated to a file so as to maximize the number of records of the file lying on the same physical track that passes under the read head before or after the desired record, depending on the intelligent controller. These file records are logically contiguous to the desired record.

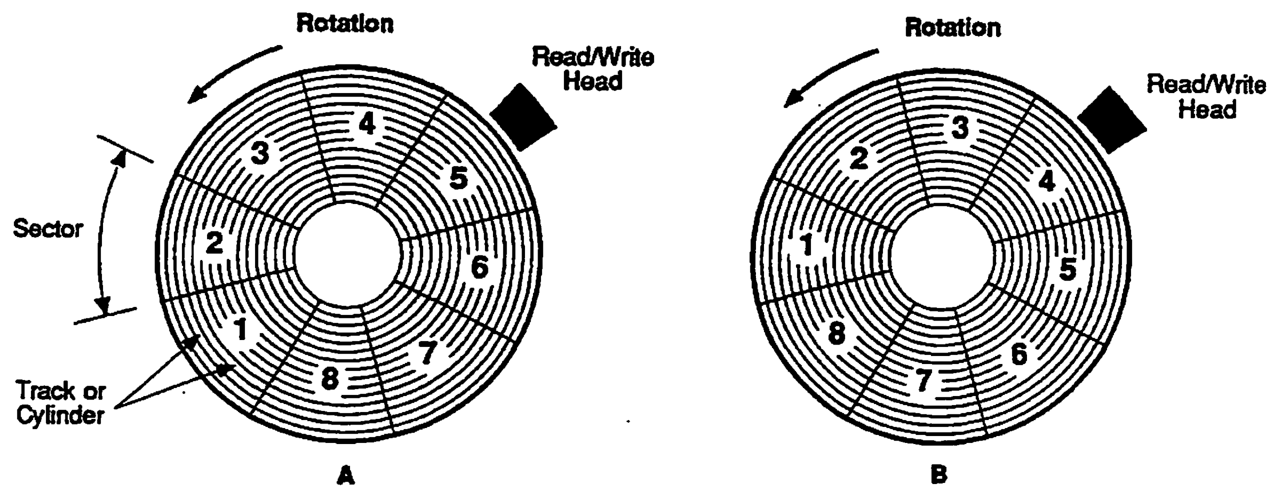
Because the Model 7210 (SDIC) controller on SCSI disks buffers, or caches, all records on a track, or cylinder, starting with the desired record, it uses forward sectoring with an interleave factor of 1 and this method of file record allocation is the default and cannot be changed by using MAKE or FIX_DISK. For example, as shown in Figure 10-1, if the desired record in part A is record 5, the controller caches all records on the track starting with record 5. Thus records are allocated starting with physical record 5 and continuing contiguously with records 6, 7, 8, 1, 2, 3, and 4; thus the interleave factor is 1 and the allocation direction is forward.

Reverse Sectoring

Reverse sectoring is an alternative method used by PRIMOS to allocate records for files and is designed to take advantage of a Model 6580 (IDC1) intelligent controller's ability to buffer records. The IDC1 caches records from the time it arrives on sector until it reads the desired record. In this scheme, the interleave factor is based on the maximum number of sectors on a track of the disk's surface minus 1. For example, if a track contains 8 records, the reverse sector interleave factor for that disk is 8 minus 1, or 7.

For example, as shown in Figure 10-1, a disk with 8 sectors per track is to be allocated in reverse order. The disk is spinning in a counterclockwise direction as you look at it. If the first sector allocated is physical sector 5 as shown in A of Figure 10-1, the next sector to be allocated will be sector 5 plus the interleave factor of 1 less than the number of sectors per track, or sector 5 plus 7 sectors, which is sector 4 as shown in B of Figure 10-1. After sector 4, the next sector to be allocated is sector 4 plus 7 sectors, or sector 3. This scheme then continues until all available sectors on a disk are allocated.

If you follow this allocation scheme, you will note that sectors are allocated in a "counting down" fashion: that is, 5, 4, 3, 2, 1, and so on. Thus it appears that they are allocated in reverse order: hence the name. Also, since the records are allocated contiguously, the interleave factor appears to be 1. The reason for this allocation scheme is to make logical file records contiguous when they are written so that, when the IDC1 gets a record (say the logical record that is stored in physical sector 8), it will also get and buffer, or cache, the records stored in physical sectors 1 through 7 as they pass under the read head. If these records logically follow the file record stored in physical sector 8, they are available for processing and the controller will not have to go to the disk to get them, thus saving processing time.



LJ10.01 D9300.6LA

Figure 10-1. Record Allocation

Allocation Order

The order in which disk records are allocated for a file is structured to optimize retrieval of records by intelligent disk controllers, that is, disk controllers that are microprocessor-based and are capable of buffering data. Records are allocated to maximize the number of sequential records that are in the disk controller's memory. When you want to read a record, the controller reads the records that logically follow or precede the one you want before or after getting to your record, depending on the controller. These other records in the controller's memory are then available to be used and do not have to be read again from disk.

Thus, the way in which disk records are laid out on disk takes advantage of disk controllers that can buffer records. Disk I/O performance is improved on disks using intelligent disk controllers by taking full advantage of the controller's buffering capabilities. However, disks using nonintelligent disk controllers do not suffer any performance degradation brought about by optimizing the allocation of disk records for intelligent controllers.

Note It is recommended that you always use reverse sectoring with CPUs not in the 9950 class (CPUs in the 9950 class consist of the 4000, 5000, and 6000 series CPUs, the 2850, the 2950, the 9755, and those with model numbers numerically equal to or larger than 9950, such as 9955) and if the majority of your standalone applications do little actual processing of the retrieved records.

Record Allocation Dependencies

The allocation of records by PRIMOS is as follows:

- If the partition is pre-Rev. 20.0, allocate records in forward order using an interleave factor of 3.
- If the partition is a Rev. 20.0 or later standard partition and the file type is CAM, allocate records in forward order using an interleave factor of 3.
- If you are using pre-Rev. 21.0 PRIMOS, allocate records in forward order using an interleave factor of 3.
- If the partition is a Rev. 22.1 robust partition, allocate records in forward order using an interleave factor of 3.
- If the partition is a standard (nonrobust) partition created with Rev. 21.0 or later MAKE or is converted to Rev. 21.0 format with FIX_DISK, allocate records using reverse sectoring and an interleave factor of 1.

These dependencies are summarized in Table 10-1.

Table 10-1. PRIMOS Record Allocation

Dependency	*Allocation Order*	*Interleave Factor*
Pre-Rev. 20.0 partition	Forward	3
Rev. 20.0 and later standard partitions and CAM files	Forward	3
Using pre-Rev. 21.0 PRIMOS	Forward	3
Robust partitions	Forward	3
Create as or convert to Rev. 20.0 or later standard partition	--- See Table 10-2 ---	

Note If you are creating robust partitions (by using the MAKE_ROBUST utility), the method of file record allocation is set by MAKE_ROBUST to forward. No reverse sectoring takes place on robust partitions. If you use the –SECTOR option of FIX_DISK on robust partitions, FIX_DISK informs you that this option cannot be used on robust partitions.

Allocation Direction for Standard Partitions

At PRIMOS Rev. 21.0 and later, the default record allocation direction and the default disk record interleave factor after you create a standard (nonrobust) partition with MAKE or convert a pre-Rev. 21.0 partition to Rev. 21.0 format with FIX_DISK, depend on the disk controller and CPU combination (see Table 10-2). Reverse sectoring with an interleave factor of 1 optimizes the sequential retrieval of records on an IDC1 intelligent disk controller. You or your System Administrator may change the method of record allocation and the interleave factor on standard partitions by using the MAKE and FIX_DISK options –SECTOR FORWARD (–ODI) or –SECTOR REVERSE (–RDI).

PRIMOS determines the combination of disk controller type and CPU type and does the following for standard (nonrobust) partitions:

- For a CPU in the 9950 class and an intelligent disk controller, use reverse sectoring with an interleave factor of 1. CPUs in the 9950 class consist of the 4000, 5000, and 6000 series CPUs, the 2850, the 2950, the 9755, and those with model numbers numerically equal to or larger than 9950, such as 9955.
- For a CPU in the 9950 class and a nonintelligent disk controller, use forward sectoring with an interleave factor of 3.
- For SCSI disks with pseudo-geometry on a SDIC controller, the default is forward sectoring with an interleave factor of 1.
- For all other CPUs and either a nonintelligent or an IDC1 controller, use reverse sectoring with an interleave factor of 1.

These conditions are summarized in Table 10-2.

Table 10-2. PRIMOS Default Record Allocation

Controller	*CPU*	*Allocation Direction/ Interleave Factor*
Intelligent (IDC1)	Less than 9950 class	Reverse/1
Nonintelligent (4005)	Less than 9950 class	Reverse/1
Intelligent (IDC1)	9950 class	Reverse/1
Intelligent (SDIC/SCSI)	9950 class	Forward/1
Nonintelligent (4005)	9950 class	Forward/3

When you create Rev. 20.0 or later standard partitions by using MAKE or convert partitions to Rev. 20.0, Rev. 21.0, or Rev. 22.1 by using FIX_DISK, those utilities determine the CPU and disk controller combination. They then set a bit in the DSKRAT header to indicate the record allocation direction according to the above recommendations. This direction then becomes the record allocation direction for that combination. Thus, depending on the controller and CPU combination, record allocation could be either forward or reverse.

Note FIX_DISK sets the record allocation bit only when you use the -CONVERT_21 and -CONVERT_22.1 options; it does not set the record allocation bit at any other time.

SAM and DAM File Management

You may allocate disk records for both SAM and DAM files in either forward order using an interleave factor of 3 or in reverse order using an interleave factor of 1 on Rev. 20.0 or later standard partitions. On SDIC controllers with SCSI disks, all records are forward allocated with an interleave factor of 1 and cannot be changed.

Setting the Interleave Factor

You do not have to convert partitions to Rev. 22.1 format in order to use Rev. 23 PRIMOS or to benefit from performance gains of allocating disk records in reverse order with an IDC1. Both SAM and DAM files on a Rev. 20.0 or later standard partition may be allocated in reverse order with an interleave factor of 1. Only Rev. 21.0 and later PRIMOS recognize reverse allocation. However, a file may have records allocated in either order or in both orders and you may use Rev. 20.0 PRIMOS to read records from that file. PRIMOS gets a record by its address, which is determined after record allocation.

To set the interleave factor and the direction of sectoring on a pre-Rev. 21.0 partition, you can do one of the following:

- Run Rev. 23.3 MAKE to create the partition as a Rev. 20.0 or later format partition.
- Run Rev. 23.3 FIX_DISK to convert a pre-Rev. 21.0 partition to Rev. 21.0 or a pre-Rev. 20.0 partition to Rev. 20.0.
- Indicate the allocation direction and the interleave factor on a Rev. 20.0 or later partition by using the MAKE or the FIX_DISK option -SECTOR with either the FORWARD or the REVERSE argument except on robust partitions or on SCSI disks with pseudo-geometry.

Performing one of the above procedures allows you to benefit from the allocation of disk records in reverse order with an interleave factor of 1.

You must use Rev. 21.0 or later PRIMOS on a partition in which the record allocation is reverse order with an interleave factor of 1 in order to realize performance improvement.

Which Method of Record Allocation Should You Use?

Certain standalone applications, which do little processing of the data being read from or written to disk, perform better on CPUs in the 9950 class with nonintelligent disk controllers if the disk has forward record allocation with an interleave factor of 3 rather than reverse record allocation with an interleave factor of 1. Examples of such applications are MAGSAV, MAGRST, COPY, and FUTIL (assuming they are run standalone).

Some of the applications noted, such as MAGSAV, attain an interleave factor of 3. Thus some systems, which back up large databases to tape using MAGSAV, need a method of changing the interleave factor while running PRIMOS with nonintelligent controllers and a 9950 class CPU.

You can change the interleave factor on particular Rev. 20.0 and later standard (nonrobust) partitions either when you create the partition by using MAKE or when you repair it by using FIX_DISK.

It is not necessary to convert Rev. 20.0 and later partitions to Rev. 22.1 format partitions in order to change the disk record interleave factor. To change the disk record interleave factor, use the –SECTOR options of MAKE or FIX_DISK on non-SCSI standard partitions.

If you convert partitions to Rev. 22.1 format, you cannot use a pre-Rev. 22.1 version of PRIMOS with the Rev. 22.1 FORMAT partition.

Note You cannot locally add a Rev. 22.1 partition to a system that is running a pre-Rev. 22.1 version of PRIMOS because of the changes in the header of the Rev. 22.1 DSKRAT. However, you may add a Rev. 22.1 format partition remotely, using PRIMENET, to a system running pre-Rev. 22.1 PRIMOS.

Appendices

MAKE Messages

A

Special messages indicate that MAKE is unable to create the disk according to the options specified on the command line or according to other input. This appendix lists the messages from MAKE in alphabetical order in three categories and presents a brief explanation of each message. The categories are (1) command-line parsing, (2) those associated with use of the –NO_QUERY option, and (3) badspot-related messages.

If a variable, such as a number, is the first word in the message, the second word is used for alphabetizing. Variable names in messages are italicized, for example, *pdev*.

For a discussion of MAKE and its command-line options, see Chapter 5. For a summary of the FIX_DISK messages, see Appendix B.

Command Line Parsing

`-DBS must be followed by ON or OFF.  Option ignored.`

You specified an invalid argument to the –DBS option. The only arguments you can use are ON and OFF. MAKE ignores the argument and continues but will later prompt for ON or OFF if this is a DBS-supporting spindle.

`Device` *`pdev`* `is not assigned.`
`MAKE paused to allow device assignment.  Type START to continue.`

The partition must be assigned to your process in order for you to run MAKE on it. Issue the ASSIGN DISK *pdev* command and then enter START to continue MAKE. MAKE also displays this message if you use *pdev* with the –COPY_BADSPOTS option but the *pdev* is not assigned.

`Device` *`xxxxx`* `has already been specified.`

You specified either a diskname or a pdev more than once with the –COPY_BADSPOTS option.

`Disk type?`

Either you did not include the –DISK_TYPE option or you did not include a *type* with the option. Enter a *type* or press Return for a list of valid types.

`Disk type` *`type`* `cannot be used on this controller.`

You specified a disk *type* that is incompatible with the disk controller that is connected to this spindle. Respecify the disk type at the prompt or check that the pdev you used is correct.

`Disk type` *`type`* `is incompatible with Disk Revision` *`rev`*
`Disk Revision must be 20, 21, 22 or 22.1.`

The disk *type* you specified cannot be used with a disk revision earlier than Rev. 20. Respecify the disk type and disk revision at the prompts.

`Input "`*`name`*`" is not a valid partition name.`

The *name* you specified with the –COPY_BADSPOTS option or at the prompt or with the –COPY_BADSPOTS option was too long or contained invalid characters. Specify a name of up to six characters using only alphabetic characters, integers, and the special characters # $ & – _ . /. The first character must be alphabetic; it must not be a digit or dash (–).

`Input "`*`xxxxxx`*`" is not a valid physical device number.`

You entered an invalid physical device number. The request for a physical device number is repeated. Physical device numbers are octal numbers, can contain only the digits 0 through 7, can be no more than six digits, and the basic pdev must contain 20_8. See Chapter 3 for a complete guide to the construction of physical device numbers for disks, or partitions.

`Input "`*`xxxxxx`*`" resembles neither pdev nor partition name.`

You entered either an invalid physical device number or an invalid partition name with –COPY_BADSPOTS. Reenter the correct format for a pdev or a partition name either in the command line or at the prompt.

`Invalid badspot level (`*`nn`*`).  Must be a decimal number 0 to 4.`

You entered a badspot level checking number other than 0, 1, 2, 3, or 4 with the –BADSPOT_LEVEL option. Enter the proper value at the prompt.

`Invalid baud rate (`*`nnn`*`).  Must be 110, 300, 1200, or 9600.`

The baud rate you specified with the –BAUD option was not one of the numbers shown in the message. Enter a proper value at the prompt.

`Invalid -COPY_BADSPOTS argument: "`*`arg`*`"`
`(`*`reason`*`)`

The argument *arg* that you used with the –COPY_BADSPOTS option is invalid for the *reason* given. *reason* is explained later in this section of this appendix.

`Invalid disk revision "rev". Must be 18, 19, 20, 21, 22 or 22.1.`

You entered an invalid number for *rev* with the –DISK_REVISION option. MAKE prompts for one of the valid numbers listed. You can use a decimal point and a zero with the integer revisions, for example, 22.0.

`Invalid disk type "type".`

The disk *type* you specified with the –DISK_TYPE option was not one of the valid disk types or was misspelled. Enter a return at the prompt for a list of valid prompts.

`Invalid maximum extent size. Must be in range 1 to 32767.`
`Invalid minimum extent size. Must be in range 1 to 32767.`

You did not specify a valid CAM file extent size with either the –MINSIZ or –MAXSIZ options. Enter a valid value at the prompt.

`Invalid number of paging/CDD records (nnnnnn).`

The number (*nnnnnn*) you specified with the –SPLIT option or at the prompt exceeds the number of records available. Enter a valid number or use the MAX argument.

`Invalid partition name "name".`

You used an invalid partition name either when initially naming the partition with the –PARTITION option or at the prompt. The partition name may contain a maximum of six characters. The first character must not be a digit or dash (–); the name can contain only alphabetic characters, digits, and the special characters _ # $ & * – . /.

`Minimum extent size (min) cannot be greater than maximum (max).`

The minimum extent size for CAM files must be less than or equal to the maximum extent size. Either you entered a value for minimum extent size that was larger than that of the maximum extent size or you did not enter a maximum extent size value. Reenter the values at the appropriate prompts.

`Must be on same spindle as device xxxxx.`

The device you are copying badspots from with the –COPY_BADSPOTS option must be a partition on the same spindle as the partition you are creating. Check the argument for the option.

`Option "opt" used more than once!`

An option that takes an argument cannot be used more than once on a command line with the exception of –COPY_BADSPOTS. MAKE aborts. Reenter the command line with only one instance of the offending option.

`Options -DBS ON (-IC) and -DBS OFF (-AC) cannot be used together.`

You cannot specify both the –DBS arguments ON (–IC) and OFF (–AC) on the same command line. Reenter the command line.

`Options -DBS ON and -DBS OFF (-IC and -AC) are not available with a Disk Revision earlier than 21.`

Dynamic Badspot handling on disks connected to an IDC1 controller cannot be used with a revision earlier than Rev. 21. Reenter the disk revision at the prompt.

`Options -INIT and -NO_INIT cannot be used together.`

You used both the –INIT and –NO_INIT options on the command line. Restart MAKE and use only one of these options.

`Options -MINIMUM_EXTENT_SIZE and -MAXIMUM_EXTENT_SIZE are not available with a Disk Revision earlier than 22.`

You cannot specify CAM file extent size on a disk at a revision earlier than Rev. 22. Reenter the disk revision at the prompt.

`Partition name?`

You did not specify the partition name with the –PARTITION option or you did not use that option. Enter the name here.

`Partition name "`*`diskname`*`" is too long: maximum is 6 characters.`

A partition name can be no more than six characters. See the `Invalid partition name` message for the contents of a valid partition name.

`PDEV and disk type do not agree. Disk type `*`disk_type`*` has `*`xx`*` heads, whereas PDEV `*`pdev`*` specifies `*`yy`*` heads starting at head offset `*`zz`*`.`

Check the disk type and the *pdev* that you specified. Your *pdev* and disk *type* do not match in the number of heads.

`Pdev `*`pdev`*` is invalid, number of heads cannot be zero.`

The *pdev* you specified is for a partition with the number of heads, or surfaces, equal to 0 and is thus invalid. Recheck your parameters for determining the *pdev*.

`Physical device?`

You did not specify the physical device number (pdev) with the –DISK option. Enter the pdev here.

`-SECTOR FORWARD (-ODI) and -SECTOR REVERSE (-RDI) cannot be used together.`

You can specify only one method of sectoring. Reenter the command line with only one instance of –SECTOR and with the proper argument (FORWARD or REVERSE) or with one of –ODI or –RDI.

`-SECTOR FORWARD and -SECTOR REVERSE (-ODI and -RDI) are not available with a Disk Revision earlier than 20.`

Reverse and forward sectoring cannot be used with a revision earlier than Rev. 20. Reenter the disk revision at the prompt.

`-SECTOR FORWARD and -SECTOR REVERSE (-ODI and -RDI) are ignored with a model 7210 controller.`

Reverse and forward sectoring cannot be used with SCSI disks connected to a Model 7210 SDIC disk controller. The –SECTOR option is ignored by MAKE in this case and MAKE continues.

`-SECTOR option must be followed by FORWARD or REVERSE.`

You did not include the appropriate argument with the –SECTOR option. MAKE prompts for the argument.

`Standalone MAKE cannot access device` *pdev.*

The *pdev* you specified is that for a controller address of 24_8, 25_8, 45_8, or 46_8 and/or for a disk drive unit number of 4 through 7. Standalone MAKE can access disks on controllers 22_8, 23_8, 26_8, and 27_8 only and on disk drive units 0 through 3 only.

MAKE may also display this message when you are copying badspots with the –COPY_BADSPOTS option for the same reason as above. Check the argument for the option.

`Standalone MAKE cannot access device by name.`

You used *diskname* with the –COPY_BADSPOTS option and standalone MAKE can access disks by pdev only because disks are not added to the system when MAKE is run in standalone mode.

`This is a pre-formatted disk:  -FORMAT ignored.`

You used the –FORMAT option with SCSI disks connected to a Model 2382 or Model 7210 disk controller. MAKE ignores the –FORMAT option.

`Too many -COPY_BADSPOTS devices specified (limit is` *xx*`).`

You can use –COPY_BADSPOTS a maximum of four times or use a maximum of four arguments with the –COPY_BADSPOTS option on a command line. If you use more than four occurrences, MAKE aborts.

`Unable to get controller type.`

MAKE failed to get the controller type from PRIMOS. Run MAKE again and if the error persists, call PrimeService.

Messages Related to –NO_QUERY

```
Insufficient options with -NO_QUERY.  -FORMAT option needed.
Insufficient options with -NO_QUERY.  -DBS option needed.
Insufficient options with -NO_QUERY.  -DBS OFF needed.
Insufficient options with -NO_QUERY.  -FORMAT and -DBS needed.
Insufficient options with -NO_QUERY.  -FORMAT and -DBS OFF needed.
```

If you include the –NO_QUERY option either on the command line or in a CPL or COMI file to run MAKE as a phantom, you must include the correct –DBS and –FORMAT options and arguments to allow MAKE to proceed. If you do not, MAKE aborts. You can run MAKE first without the –NQ option to see what other options are required. See the discussions in Chapter 5 on pages 5-32 and 5-57.

Badspot-related Messages

`A record below or within the DSKRAT (record <= 'xx) cannot be a badspot.`

Always preceded by a `Cannot add badspot` message. May occur during the badspot input dialog or while reading a (probably corrupt) BADSPT or DBS file from either the disk being made, a device specified with the –CPY option, or a flawmap. In all cases the badspot is ignored.

A badspot has been found within records 0_8 through *xx*. Records 0 through *xx*, which contain the bootstrap file, DSKRAT, and MFD, must not contain any badspots. A disk with flaws in these locations is not usable by PRIMOS. MAKE aborts.

You may also get this message if you use the wrong disk type argument with the –DISK_TYPE option. This message is then followed by disk error messages in that case. To recover, rerun MAKE being sure to specify the correct disk type from the table in Chapter 5.

Another possible cause of this error is that the disk has not yet been formatted. If this really is a new disk and you did not specify –FORMAT on the command line, reenter the command line and include the –FORMAT option and, possibly, the –NEW_DISK option. If the error occurs again, contact PrimeService.

`BADSPT file on device xxxxxx is not a SAM file!`

The BADSPT file on the device that you are copying badspots from is not a SAM file and is therefore assumed by MAKE to be invalid. MAKE continues without attempting to read the file.

```
BADSPT file on device xxxxxx is invalid.
```

The existing BADSPT file on the device that you are creating is corrupt. MAKE continues.

```
BADSPT file on partition xxxxxx is invalid.
Ignoring -COPY_BADSPOTS option.  Continuing with MAKE.
```

The BADSPT file on the device that you are copying badspots from is corrupt. MAKE continues.

```
BADSPT file on partition xxxxxx is invalid.
nn badspots added from BADSPT file on partition xxxxxx.
Continuing with MAKE.
```

The BADSPT file on the device that you are creating or copying badspots from is corrupt but MAKE was able to read a portion and add the badspots. MAKE continues.

```
Because of badspots the file system allowance is too small.
Specify a -SPLIT that allows at least nnn file system records.
```

Badspots on this disk make it impossible for MAKE to achieve the –SPLIT argument you specified. Rerun MAKE with a –SPLIT argument of *nnn* or larger. If numerous new badspots have been reported, first have the disk checked for a possible hardware problem.

```
Cannot add badspot at record = 'xxxxxxx
    (track = xx, head = yy, sector = zz)
  reason
```

MAKE cannot add another badspot during input with the –QUERY_BADSPOTS option for the *reason* given. *reason* is explained in this section of the appendix.

```
Cannot add badspot at track = xx, head = yy, sector = zz
  reason
```

MAKE cannot add another badspot during input with the –QUERY_BADSPOTS option for the *reason* given. *reason* is explained in this section of the appendix.

```
Cannot add known badspot at record = 'xxxxxxx
    (track = xx, head = yy, sector = zz)
  reason
```

MAKE cannot add another badspot read from an existing source such as a BADSPT or DBS file or a flaw map for the *reason* given. *reason* is explained in this section of the appendix.

`Cannot add known badspot at track = `*`xx`*`, head = `*`yy`*`, sector = `*`zz`*
  `reason`

MAKE cannot add another badspot read from an existing source such as a BADSPT or DBS file or a flaw map for the *reason* given. *reason* is explained in this section of the appendix.

`Cannot add new badspot at record = '`*`xxxxxxx`*
    `(track = `*`xx`*`, head = `*`yy`*`, sector = `*`zz`*`)`
  `reason`

MAKE cannot add another badspot that MAKE found while checking for badspots or while creating the file system for the *reason* given. *reason* is explained in this section of the appendix.

`reason`
`Cannot determine if there is an active Dynamic BadSpot file on this spindle.`

reason precedes this message and is a message documented elsewhere in this section. This and the related messages occur only when making a DBS-supporting spindle because MAKE needs to know whether an active DBS file is present on the spindle in order to correctly make the partition you specified. The action you need to take next is made clear by the messages and prompts MAKE displays next.

`Cannot read DSKRAT on head zero partition (`*`pdev`*`).`
`Cannot read DSKRAT on partition `*`pdev`*`.`

There was a disk error trying to read record 2. Either the disk is new and needs formatting, or there is a hardware problem.

`DSKRAT pointer mismatch.`

While reading a DSKRAT record which it has previously written, MAKE finds the record to have become invalid. MAKE displays this error message and aborts. Reenter your command line, being sure that you have entered the correct options and arguments. If the problem persists, call PrimeService.

`Dynamic Badspot remapping area is at least 80 percent full.`
`Disk may have a hardware problem. Please have it checked.`

If the DBS file and RMA become 80% or more full, the disk contains an unacceptable number of badspots and a head crash is probably imminent. Contact PrimeService.

`Fatal MAKE error: Disk I/O error while formatting.`

An unexpected error. A disk error occurred while MAKE is trying to format a track on the disk. This is probably a hardware problem. If the problem is not obvious (for example, someone took the disk off-line or damaged the cables), get the disk error message reported by PRIMOS and call PrimeService.

`Fatal MAKE error: Disk I/O error while writing` *`xxxxxx`*`.`
`Status =` *`xx`*`.` *`error_message`*

Some unexpected error occurred. *xx* is a standard PRIMOS error code and *error_message* is a standard PRIMOS message.

`Fatal MAKE error: Disk I/O error while writing` *`xxxxxx`*`.`
`Status = 48. Device not assigned.`

The partition you are creating became unassigned at the supervisor terminal. Assign the partition and reenter the MAKE command line.

`Fatal MAKE error: Disk I/O error while writing` *`xxxxxx`*`.`
`Status =` *`xx`*`.` *`error_message`*

Some unexpected error occurred. *xx* is a standard PRIMOS error code and *error_message* is a standard PRIMOS message.

`File system expanded by new badspot; paging space decreased to xxxxxx records.`

Informational message only. You specified a very large –SPLIT argument, meaning a very small file system, and MAKE was forced to make the file system larger and the paging or crash dump area smaller to cope with the badspot. MAKE continues.

`Flaw map is bad. Continuing with MAKE.`

MAKE found the vendor flaw map on the partition to be corrupt. The disk may have a hardware problem. MAKE continues and checks for badspots to level four.

`Flaw map on device` *`pdev`* `is unreadable. Continuing with MAKE.`

MAKE is unable to find a head within the partition being made that contains a complete and readable flaw map. MAKE continues and will check badspots to a badspot level of four.

`Illegal xxx number for this disk type.`

xxx is `track`, `head`, or `sector`. Always preceded by a `Cannot add badspot` message. Can occur whenever MAKE is reading badspots from a BADSPT file, DBS file, or flaw map. This message can occur if you have specified the wrong –DISK_TYPE or if MAKE is reading a BADSPT file that was originally created for a different disk type (for example, you used –CPY *name* to read a BADSPT file from the wrong place). If none of these are the reason, it is likely that the badspot source is invalid.

`Invalid DBS file on head zero partition (`*`pdev`*`).`
`Invalid DBS file on partition `*`pdev`*`.`

Applies to an active DBS file on the spindle. A readable and largely valid DSKRAT was read but the DBS file appears damaged. These errors may be expected if you know the file system on the disk has been damaged. MAKE continues.

`Invalid DSKRAT on head zero partition (`*`pdev`*`).`
`(DSKRAT specifies `*`x`*` heads, wrong for pdev `*`pdev`*`. Check your pdevs.)`
`Cannot determine if there is an active Dynamic BadSpot file on this spindle.`

Occurs under PRIMOS when making a non-head zero partition. The head zero partition DSKRAT is valid, but specifies a number of heads (*x*) that conflicts with the head zero *pdev* you have assigned. You have probably assigned the wrong head zero pdev or the head zero pdev you have assigned is not the one the head zero partition was made with or you have accidentally specified a non-head zero pdev, some of whose surfaces overlap the head zero partition.

`Invalid DSKRAT on head zero partition (`*`pdev`*`).`
`(DSKRAT specifies `*`x`*` heads, conflicts with pdev `*`pdev`*`. Check your pdevs.)`
`Cannot determine if there is an active Dynamic BadSpot file on this spindle.`

Occurs standalone only when making a non-head zero partition. The head zero partition DSKRAT is valid but specifies a number of heads (*x*) that looks wrong because it specifies a head zero pdev that overlaps the non-head zero pdev you are making. Same probable reasons as above.

`Invalid DSKRAT on head zero partition (`*`pdev`*`).`
`Invalid DSKRAT on partition `*`pdev`*`.`

Applies to an active DBS file on the spindle. Record 2 was read, but does not represent a valid DSKRAT. These errors may be expected if you know the file system on the disk has been damaged. MAKE continues.

`MFD on device `*`pdev`*` is corrupted.`

The MFD on the device that you are copying badspots from is invalid and MAKE cannot tell if a BADSPT file exists. MAKE continues.

`New badspot detected writing record = '`*`xxxxxxx`*
`(track = `*`xx`*`, head = `*`yy`*`, sector = `*`zz`*`)`

This is the normal way MAKE reports a new badspot detected during the badspot checking phase or during the final file system creation phase. The record is added to the BADSPT or the DBS file and reflected in the badspot summary at completion.

`New BADSPT file is full.`

Always preceded by a `Cannot add badspot` message. May occur at any time where MAKE is adding to the list of known badspots that MAKE uses to create the new BADSPT or DBS file. Indicates that the supported limit of the number of badspots is exceeded. The badspot is ignored. The supported limit is 2047 badspots in the case of a DBS file or 2047 entries in the case of the BADSPT file. If you have this many badspots, you probably have a hardware problem and a head crash is probably imminent. Have the hardware checked before making the disk.

`No BADSPT file found on device` *`pdev`*`.  Continuing with MAKE.`

MAKE did not find the BADSPT file on the device from which you are copying badspots. If you use the –REPORT option, MAKE also displays this message if an existing BADSPT file is not found on the partition you are creating. MAKE continues.

`No flaw map found on device` *`pdev`*`.  Continuing with MAKE.`

MAKE did not find any readable records on any surface of the partition in the cylinder where the vendor flaw map was expected to reside. This could be caused by an incorrect –DISK_TYPE argument. MAKE continues and checks for badspots to level four.

`Processing flaw map on device` *`pdev`*`.`
`Processing of flaw map completed.`

These messages are displayed in the normal course of creating a disk partition when you use the –DISK_TYPE option with either the MODEL_4735, MODEL_4845, or the MODEL_4860 argument and you do not use the –NO_FLAW_MAP option. If you use the –REPORT option, MAKE also displays *`nn`* `badspots added from flaw map on device` *`pdev`*.

`Record number '`*`bignumber`*` is too big, maximum is '`*`maxrecord`*`.`

The badspot you entered by record number exceeds the largest valid record number on the partition. Reenter a valid record number.

`Unable to completely read BADSPT file on device` *`pdev`*`.`
`Continuing with MAKE.`

The BADSPT file on the device that you are copying badspots from is corrupt but MAKE was able to read a portion and add the badspots. MAKE continues.

`Unable to completely read DBS file on device` *`pdev`*`.`
`Continuing with MAKE.`

The DBS file on the device that you are copying badspots from is corrupt but MAKE was able to read a portion and add the badspots. MAKE continues.

`Unable to find partition` *`name`*`.`
`Ignoring -COPY_BADSPOTS option.  Continuing with MAKE.`

The device you specified with the –COPY_BADSPOTS option to copy badspots from is not added. MAKE continues.

```
Unable to open BADSPT file on partition pdev.
Ignoring -COPY_BADSPOTS option.  Continuing with MAKE.
```

MAKE is unable to open the BADSPT file on the device that you are copying badspots from. Possible reasons are lack of access, file in use, or file is damaged. MAKE continues.

```
Unable to read BADSPT file on device pdev.
Continuing with MAKE.
```

```
Unable to read BADSPT file on partition name.
Continuing with MAKE.
```

MAKE is unable to read the BADSPT file on the partition you are creating or on the device from which you are copying badspots because the file is invalid or damaged or or a disk error prevented MAKE from reading the first record of the file. MAKE continues.

```
Unable to read DBS file on device pdev.
Continuing with MAKE.
```

MAKE is unable to read the DBS file on the partition you are creating because the file is invalid or damaged or a disk error prevented MAKE from reading the first record of the file. MAKE continues.

```
Unexpected error sending badspot to controller.  Status = xx (ADBSP$)
Unable to make partition with -DBS ON.
For remedial action, refer to Operator's Guide to File System
Maintenance.
MAKE aborted.
```

MAKE adds badspots to the DBS file by calling the PRIMOS gate ADBSP$ to pass them to the controller. It is the controller that modifies the DBS file. This error reports failure of the ADBSP$ call for various possible reasons including these:

- The DYNBSP file on the head zero partition is in use by someone else, or it is damaged or inaccessible in some way. PRIMOS Rev. 23.2 and earlier have a known problem: if you are making a non-head zero partition and the head zero partition is added (not assigned) and has a name of less than six characters, you get the above error. Shut down the head zero partition using the -RENAME option of SHUTDN to change its name to have six characters and assign it again.
- The controller may have just fallen out of DBS mode; this problem should be accompanied by a disk error in the log.

In all cases, the error aborts MAKE. You should try running MAKE again and, if the error persists, call PrimeService. This message can occur when new badspots are added in or close to the DBS file. Try remaking the disk with -FORMAT to force the DBS file to be rebuilt.

This error only prevents using –DBS ON. Running MAKE on the same disk with –DBS OFF would succeed, provided all partitions on the spindle have the same mode.

```
WARNING:  This partition cannot be mirrored (because the DBS file
          has been moved to avoid a badspot at record '4).
```

In order to be mirrored, the disk pair must have their DBS files in identical locations. A badspot coinciding with the first record of one of the DBS files violates this requirement because it causes the DBS file to be moved.

FIX_DISK Messages

B

This appendix contains error messages generated by the FIX_DISK command. Many of the messages displayed by FIX_DISK are informational, that is, they tell you of any errors encountered. Many of these messages are generated due to failure of other software and are not generated or caused by users.

If the repair operation results in the truncation or deletion of files or records, you can restore these files or records from backup copies. Thus, it is important to have a good backup procedure. (See the *Operator's Guide to Data Backup and Recovery*.)

The messages below are in alphabetical order. If the article *The* or a variable, such as a number, is the first word in the message, the second word is used for alphabetizing. Exceptions to this are the first three messages, which have *1st*, *2nd*, and *3rd* as the second words. An explanation follows each message. Variable names and numerical output in the messages are italicized, for example, *category_name*. The name of the module issuing the message may appear in parentheses following the message.

For a discussion of FIX_DISK and its command-line options, see Chapter 6. For a summary of the FIX_DISK command-line options, see Appendix F.

If you specify the –FIX option on the command line, FIX_DISK repairs any correctable errors or inconsistencies. If you do not specify the –FIX option, FIX_DISK reports any errors but does not correct them.

FIX_DISK Error Messages

`The 1st file entry of the MFD is not DSKRAT`

FIX_DISK checks that the first entry in the MFD is DSKRAT. If this entry is missing and the –FIX and –INTERACTIVE options were not specified on the command line, FIX_DISK aborts.

`The 2nd file entry of the MFD file is not MFD`

FIX_DISK checks that the second entry in the MFD is MFD. If this entry is missing, FIX_DISK aborts. Remake the partition by using MAKE.

`The 3rd file entry of the MFD is not BOOT.`

FIX_DISK checks that the third entry in the MFD is BOOT. If this entry is missing, FIX_DISK aborts. Remake the partition by using MAKE.

`Access Category` *`category_name`* `does not reference an ACL!`

The ACL pointer of an access category does not point to a valid ACL. If you specify the –FIX option, the access category is deleted, and all objects that it protects revert to default protection.

`Access Category` *`category_name`* `is not pointed at by ACL it points to!`

The ACL pointer of an access category points to an ACL that doesn't point back to it. If you specify the –FIX option, the access category is deleted, and all objects that it protects revert to default protection.

`Access Category` *`category_name`* `points outside the directory!`

The ACL pointer of an access category points outside the directory. If you specify the –FIX option, the access category is deleted, and all objects that it protects revert to default protection.

`ACL at word` *`XX`* `does not point at a file or access category!`

The owner pointer of an ACL does not point to a file or access category. If you specify the –FIX option, the ACL is deleted.

`ACL at word` *`XX`* `is not pointed at by object it points to!`

The owner pointer of an ACL points to an object that does not point back to it. If you specify the –FIX option, the ACL is deleted.

`ACL at word` *`XX`* `points outside the directory!`

The owner pointer of the ACL is pointing to something in a different directory. If you specify the –FIX option, the ACL is deleted.

`Activating DBS file`

This message indicates a conversion to –IC mode or a conversion to a Rev. 21.0 head zero partition in –IC mode; the DBS file is activated.

`Added to BADSPT file!`

The record you added with the –ADD_BADSPOT option is added.

`Added to the DBS file!`

The record you added with the –ADD_BADSPOT option is added to the DBS file.

`Already exists in BADSPT file!`

The BADSPT file already contains information on the bad record that was found, even though that bad record is also in use. FIX_DISK continues.

`Attempt to re-run FIX_DISK to add this badspot.`

For a discussion of this message, see the message `FIX_DISK is unable to read badspot . . . .`

`Back pointer in free list at XX is YY, it should be ZZ.`

The backward pointer in the free list does not point back to the previous free entry. The file system maintains a list of free, or vacant, directory entries. These are entries that have been deleted from the directory and thus freed for use. To save space in the partition, you can remove these free entries by using the –UFD_COMPRESSION (–CMPR) option to compress the directory. This message indicates that the directory is damaged. Use the –FIX option to repair the directory.

`The backward pointer is bad.`
`It should be YY instead of XX!`
`Bad record address = XX BRA = YY Father = ZZ Type = N`

The backward pointer of a record does not point back to the previous record of a file. In the case of the first record of a file, the backward pointer is not zero. If you specify the –FIX option, the backward pointer is fixed to point to the previous record when the BRA word of this record matches the first record address of this file. If the BRA word of this record does not match the first record address of the file, the file is truncated. You can restore the file by using a backup procedure.

`The backward pointer of the next record does not point back to the pre-remap record!`

The backward pointer of a record does not point back to the previous record, which has been remapped to a location that does not contain a badspot. In the case of the first record of a file, the backward pointer is not zero. If you specify the –FIX option, the backward pointer is fixed to point to the previous record when the BRA word of this record matches the first record address of this file. If the BRA word of this record does not match the first record address of the file, the file is truncated. You can restore the file by using a backup procedure.

`Bad data count XX in segdir YY!`

The segment directory contains a different number of records than the record header indicates it should contain. If you specify the –FIX option, the header is fixed.

`Bad entry length at record XX, word YY.`

A record has been corrupted and does not have a valid length. Use the –FIX option to repair the record.

`Bad equivalence section!`

The equivalence section of the badspot file is not in the correct format and is ignored. Other messages, such as `Single zero record added...` will be displayed. Use the -FIX option to delete the equivalence section.

`Bad file type: illegal 1st 8 bits!`
`It should be` *XX* `instead of` *YY*.
`Bad record address =` *XX* `BRA =` *YY* `Father =` *ZZ* `Type =` *N*

A correctable error is found in the header of a directory file or a SAM segment directory file indicating an incorrect file type. FIX_DISK corrects the error.

`Bad file type: special bit not set!`
`Bad record address =` *XX* `BRA =` *YY* `Father =` *ZZ* `Type =` *N*

The special bit in the BOOT, MFD, or DSKRAT files has not been set. If you specify the -FIX and -INTERACTIVE options, the DSKRAT file is rebuilt. Otherwise you may have to create the partition again by using MAKE.

`Bad forward pointer (correctable).`

Contiguous Access Method (CAM) files are in use on this pre-Rev. 22.0 partition. Use the -FIX option to correct the error.

`Bad forward pointer (uncorrectable).`

Contiguous Access Method (CAM) files are in use on this pre-Rev. 22.0 partition. This error is uncorrectable; consequently, if you use the -FIX option, you may receive a message indicating that the file has been truncated or deleted. In either case, restore the file by using a backup procedure.

`Bad header in BADSPT file, ignored!`

The header of the BADSPT file is incorrect. FIX_DISK continues, but badspot handling is disabled.

`Bad length (`*XX*`) on index block, entry deleted.`

A table of records in a directory is kept for access to the directory. The index of the table is corrupted. The table affects how fast the entries are accessed. This error condition means that access will be slower if you use the -FIX option because this option causes deletion of the index block. Delete this directory and restore it by using a backup procedure.

`Bad "minimum extent length" value (correctable).`

Contiguous Access Method (CAM) files are in use on this pre-Rev. 22.1 partition. The value of the extent length, or number of records in a CAM file, is less than ten. Use the -FIX option to correct the error.

`Bad number of extents value (correctable).`

Contiguous Access Method (CAM) files are in use on this pre-Rev. 22.0 partition. The number of blocks of contiguous records found in the file header is either greater than the maximum allowed or it is less than zero. Use the –FIX option to correct the error.

`Bad Physical Device Number (cl_par)`

The physical device number that is specified in the command line is incorrect. Reenter the command line and use a correct pdev.

`Bad record address:` *`XX`* `BRA =` *`YY`* `Father =` *`ZZ`* `Type =` *`file_type`*

Record address *XX* in the record header is bad. The Beginning Record Address (BRA), father pointer (FRA), and file type are also displayed. If you specify the –FIX option, the address is corrected, if possible. Otherwise the file is truncated. You can restore the file by using a backup procedure.

`BADSPT: Track =` *`nn`* `Head =` *`nn`* `Sector =` *`nn`* `Record = 'XX`

You specified the –LIST_BADSPOTS option and the badspots are listed.

`The BADSPT file cannot be read, ignored!`

BADSPT file was not found in the current partition. Badspot handling is disabled.

`The BADSPT file is bad, ignored!`

The BADSPT file that is found by FIX_DISK is bad. Badspot handling is disabled.

`The Beginning Record Address (BRA) pointer is bad.`
`It should be` *`XX`* `is` *`YY`*`.`
`Bad record address =` *`XX`* `BRA =` *`YY`* `Father =` *`ZZ`* `Type =` *`N`*

The Beginning Record Address (BRA) pointer of the records within the file (except the first record) should point to the first record of the file. If you specify the –FIX option, the BRA pointer is fixed.

`The Beginning Record Address (BRA) pointer` *`XX`* `is bad, it points to a record that belongs to another file.`
`Bad record address =` *`XX`* `BRA =` *`YY`* `Father =` *`ZZ`* `Type =` *`N`*

In checking the integrity of a record header, FIX_DISK finds that the Beginning Record Address (BRA) pointer points outside the current file. If you specify the –FIX option, the file is truncated. You can restore the file by using a backup procedure.

`Both -IC and -AC conversion modes can not be specified at the same time.`

You cannot specify more than one of the controller mode selection options (–DBS ON or OFF, or –AC or –IC) on the same command line. Reenter the command line with the one selection you want.

`Both -SECTOR FORward (-ODI) and -SECTOR REVerse (-RDI) cannot be specified.`

When selecting the type of record allocation, you have used both selection options on the command line. Only one of these options is permitted. Reenter the command line with the one selection option you want to use.

`Both -DUFE and -SUFE can not be specified at the same time.`

Since the –DUFE and –SUFE options have opposite effects, they cannot both be specified on the same command line. Reenter the command line specifying one or the other. If you specify neither, –DUFE (delete unknown file entries) is the default.

`CAM file(s) in DBS/RMA area prevent conversion to rev 21 disk. Please save & delete these files, and restore following the conversion.`

Just before this message appears, you see the message `The following CAM file . . .` and that message contains the pathname of the offending files. When converting a partition to Rev. 21.0 with –CONVERT_21, a CAM file, *pathname*, is found in the area of the head zero partition where it is necessary for FIX_DISK to create a DBS file and the RMA. FIX_DISK aborts. If you want to convert this partition to Rev. 21.0, save the offending files to tape, delete them from the disk, convert the disk, and then restore the CAM files you saved and deleted.

`Cannot access RAT on disk` *`pdev`*`.`

In trying to read the DBS file, FIX_DISK could not find the DSKRAT on the head zero partition, *pdev*. Be sure that the head zero partition is added to the system or is assigned to you at the supervisor terminal.

`Cannot add badspot to DBS file.  Controller code:` *`DD`*

The DBS file may be full, the head zero partition may not be added, or the disk controller may have encountered a problem (the disk may be write-protected). FIX_DISK aborts. The code returned by the disk controller is also displayed (*DD*). If the head zero partition is added and the disk drive is not write-protected, you may have a hardware problem; call PrimeService.

`Cannot convert a revision` *`nn`* `disk to revision` *`mm`* `disk.`

You cannot revert the revision of a partition using FIX_DISK; that is you cannot, for example, convert a Rev. 21.0 partition to a Rev. 20.0 partition and you cannot convert any revision partition to Rev. 22.0 with FIX_DISK. You can convert from an older revision forward to a more recent revision, up to

Rev. 21.0, with the –CONVERT_19, –CONVERT_20, and –CONVERT_21 options.

`Cannot convert a revision nn disk to revision 21 disk.`

The only partitions that can be converted to Rev. 21.0 are pre-Rev. 21.0 partitions. You have attempted to convert a Rev. 21.0 or later partition to Rev. 21.0.

`Cannot convert a revision nn disk to revision 22.1 disk.`

The only revision partition that you can convert to Rev. 22.1 with FIX_DISK is a Rev. 22.0 partition. To convert an earlier revision partition to Rev. 22.1, you must use MAKE.

`Cannot determine controller type (RCINF$)`

The controller type cannot be determined; a nonintelligent controller is therefore assumed.

`Cannot find comdev in disk list!`

–COMDEV was specified as an option on the command line, but the partition specified is not added to the system. Use STATUS DISKS to verify the pdev of the COMDEV (logical device 0) and reenter the command line with the correct pdev.

`Cannot FIX_DISK a -DBS ON (-IC) disk on a non-intelligent controller!`

You cannot run FIX_DISK on a Rev. 21.0 or later partition in Dynamic Badspot Handling (–DBS ON) mode that is not associated with an IDC1 disk controller except to switch controller modes. You can use only the –FIX option and either –DBS ON or –DBS OFF.

`Cannot process BADSPOT for records less than XX!`

A certain number of records is reserved at the beginning of a partition for system information. The size of this area depends on the size of the partition. Badspots cannot occur in this region. This message indicates that a badspot falls in this reserved region. If this region contains an uncorrectable badspot, PRIMOS cannot use the partition.

`Cannot read DBS file!`

Either the head zero partition is not assigned to you or the DBS file is corrupted. In the case of a corrupted DBS file, FIX_DISK will reconstruct it if you are presently running FIX_DISK on the head zero partition. If you are not presently running FIX_DISK on the head zero partition, do so and FIX_DISK will reconstruct the DBS file.

`Cannot read record XX of DBS file.  Will rebuild DBS and RMA area.`

A bad record is encountered while FIX_DISK is reading the DBS file. FIX_DISK attempts to reconstruct the DBS file and the RMA if this is the head zero partition. If it is not the head zero partition, run FIX_DISK on the head zero partition.

`Cannot update DBS file!`

In trying to update the DBS file, FIX_DISK could not read the file. Be sure that the head zero partition is added to the system or is assigned to you at the supervisor terminal.

`Cannot use -DBS option (-IC/-AC)  on a Pre-Revision 21 disk!`

You can change the controller mode and use dynamic badspot handling and mirroring only on Rev. 21.0 and later partitions.

`Cannot write DBS file!`

The head zero partition may not be available so that FIX_DISK can write to the DBS file. Be sure that the head zero partition is assigned to you and that the disk drive where the head zero partition is located is not write-protected. It is also possible that there is a badspot in the area where the DBS file and the RMA must go.

`Cannot write updated DBS file!`

The head zero partition may not be available so that FIX_DISK can write to the DBS file. Be sure that the head zero partition is assigned to you and that the disk drive where the head zero partition is located is not write-protected.

`Changed to default ACL pointer.`

Every file must point either to the default ACL, to a specific ACL, or to an access category. Every access category must point to a specific ACL. Every ACL must point back either to a file or to an access category. If a file or an access category does not meet these criteria and you specify the –FIX option, that file's ACL pointer is set to the default.

```
Command Line Error.  (get_arg)
For more information type "fix_disk -help"
```

You have made an error on the command line. Reenter the command line and use correct syntax or use the –HELP option of FIX_DISK to get help.

```
The current record address (CRA) is bad.
It should be XX is YY.
Bad record address = XX BRA = YY Father = ZZ Type = N
```

The Current Record Address (CRA) pointer of this record does not match the current record. If you specify the –FIX option, the CRA is corrected.

`The DAM index is too long to represent the DAM file!`
`The DAM index is too long to represent the data records!`

There are fewer data records in a DAM file than the DAM file's index indicates. If you specify the -FIX option, the index is truncated. You can restore the file by using a backup procedure.

`The data count in the first record of the UFD is too small.`

The information needed by PRIMOS to access this file or record does not exist. Use the -FIX option to correct the data count.

`Date and time not set.  Type START or REN to re-enter.`

FIX_DISK must have the date and time set in PRIMOS in order to run. This is usually not a problem because the date and time are set when you ordinarily boot PRIMOS either by PRIMOS reading the diagnostic processor clock or by explicitly setting the date and time with the PRIMOS SETIME command at the end of the boot and configuration processes. If you use the MTRESUME command, however, to repair a damaged command device or for some other reason and, to do so, you boot PRIMOS from tape on an older machine, you must use the PRIMOS SETIME command to set the date and time before invoking FIX_DISK.

`DBS file is corrupted.  Run FIX_DISK on partition` *`pdev`*`.`

This is not the head zero partition, but the DBS file version or the DBS file header or the DBS file entry size on the head zero partition is corrupted. Run FIX_DISK on the head zero partition *pdev*.

`DBS file is corrupted.  Will rebuild DBS and RMA area.`

This is the head zero partition and the DBS file version or the DBS file header or the DBS file entry size is corrupted. FIX_DISK attempts to reconstruct the DBS file and the RMA.

`The DBS file cannot be read, ignored!`

The DBS file was not found in the current partition. See the previous message.

`The -DBS option requires an argument (ON or OFF).`

You must specify one of the -DBS arguments with the -DBS option, ON to turn on Dynamic Badspot Handling on the spindle or OFF.

`Deactivating DBS file`

This message indicates a conversion to -DBS OFF mode or a conversion to a Rev. 21.0 head zero partition in -DBS OFF mode; the DBS file is deactivated.

`Directory is longer than 64K!`

The maximum size of a directory is 64K halfwords. If a directory exceeds this limit, truncation occurs if you specify the -FIX option. The directory information is probably corrupted and not really longer than 64K words. You can restore the directory from a backup.

`The Directory Used count is bad. It should be` *`YY`* `instead of` *`XX`*`.`

The directory-used count for this directory does not match the directory-used count that is calculated by FIX_DISK. (The directory file itself and all the files in it are counted.) If you specify the –FIX option, the directory-used count is corrected.

`Disk is full!` *`filename`* `file entry cannot be created.`

There is insufficient space on the partition for FIX_DISK to write the directory entry for the BADSPT file or the DBS file that was just created. FIX_DISK continues, but badspot handling is disabled. Make room on the partition by deleting some files and then rerun FIX_DISK.

`Disk is full!` *`filename`* `file is incomplete.`

There is insufficient space on the partition for FIX_DISK to write more information into the BADSPT file or the DBS file. The appropriate badspot file then cannot contain any new badspot information. Make room on the partition by deleting some files and then rerun FIX_DISK.

`Disk is full!` *`filename`* `file cannot be created.`

There is insufficient space on the partition for FIX_DISK to write the BADSPT file or the DBS file. The appropriate badspot file then does not exist. FIX_DISK continues, but badspot handling is disabled. Make room on the partition by deleting some files and then rerun FIX_DISK.

`Disk is full! Cannot allocate another record! (fix_disk)`

There is insufficient space on the partition for FIX_DISK to allocate another record for badspot processing. Make room on the partition by deleting some files and then rerun FIX_DISK.

`Disk is in -`*`mode`*`_Controller mode with` *`dir`* `Allocation`
`Disk is in` *`dir`* `allocation mode.`

If the partition is a Rev. 21.0 or later partition, FIX_DISK tells you the controller mode (ALL or INTELLIGENT) and the type of record allocation (Forward or Reverse); otherwise only the type of record allocation is displayed.

`Disk not properly shutdown. Full fix disk will be run.`

FIX_DISK determines from a bit in the RAT that the disk was not shut down properly and the partition is a standard partition. Full FIX_DISK will be run on this partition even if you specified the –FAST option because fast FIX_DISK may not fully correct the situation on a standard partition. See Table 7-1 for recommendations concerning running FIX_DISK.

```
Disk Read/Write Error.  Record = XX
Track = YY Head = ZZ Sector = SS.
```

An error occurred during the reading or writing of record XX. If you specify the –FIX and –TRUNCATE options, the file is truncated on read errors and this badspot record is added to the BADSPT file. If you do not specify the –TRUNCATE option, a zero record is added.

```
Disk record address 'XX is illegal!
```

A record points to address XX (octal), which is outside the current partition. If you specify the –FIX option, the address is corrected.

```
Disk uses direction sectoring.
Disk uses direction sectoring with -DBS mode.
```

If this is a Rev. 20.0 disk, FIX_DISK indicates only the sectoring *direction*, either forward or reverse. If this is a Rev. 21.0 or later disk that is capable of Dynamic Badspot Handling, FIX_DISK also indicates the *mode*, either ON or OFF.

```
The DSKRAT header has wrong length.  (RAT_CK).
The file structure of DSKRAT is bad.  (RAT_CK).
The file system is larger than the disk.  (RAT_CK)
The length of UFD header is incorrect.
The number of heads is different.  It should be YY is XX.
The physical record size is different.  (RAT_CK).
```

You may see one or more of the above five error messages. They mean that the information contained in the DSKRAT header does not correspond to the information computed from the physical device number. Either the pdev is incorrect or the DSKRAT header contains incorrect information. You may not have created the partition by using MAKE or you may need to remake the partition.

If you do not specify the –INTERACTIVE and –FIX options, FIX_DISK aborts. Otherwise, FIX_DISK prompts

```
Fix DSKRAT header?
```

A NO response causes FIX_DISK to abort. A YES response initiates a dialog that results in fixing the DSKRAT. FIX_DISK first prompts

```
Physical Disk?
```

Enter the physical device number (pdev) of the partition containing the DSKRAT to be fixed. FIX_DISK computes the number of records in the partition from the pdev and the model of disk by displaying a list of models and then prompting for the model:

```
Valid disk model names:
          SMD              80 or 300 MB removable
          CMD              Cartridge module device
          68MB              68 MB fixed media
          158MB            158 MB fixed media
          160MB            160 MB fixed media
          600MB            600 MB fixed media
          MODEL_4475       300 MB fixed media
          MODEL_4714        84 MB fixed media
          MODEL_4711        60 MB fixed media
          MODEL_4715       120 MB fixed media
          MODEL_4735       496 MB fixed media
          MODEL_4719       258 MB fixed media
          MODEL_4845       770 MB fixed media
          MODEL_4721       328 MB fixed media
          MODEL_4860       817 MB fixed media
          MODEL_4729       673 MB fixed media
          MODEL_4730       213 MB fixed media
          MODEL_4731       421 MB fixed media
          MODEL_4832       1.34 GB fixed media

Please enter model name of disk...
```

If you enter an invalid model, FIX_DISK reprompts for a model name. FIX_DISK then prompts

```
Split disk?
```

If part of the partition is to be used for paging, then enter YES; otherwise, enter NO. If you answer YES, FIX_DISK then prompts

```
Paging Records (Decimal)?
```

Enter the number of records to be used for paging. If you specify too many paging records, you will not leave enough room for system overhead, such as the DBS file if this is a head zero partition of a Rev. 21.0 or later disk that supports Dynamic Badspot Handling (see Chapter 5), or a minimum amount (3 records) at other disk revisions, FIX_DISK displays the following message and reprompts for the number of paging records:

```
Too many paging records.
```

After you enter a reasonable number for paging records, FIX_DISK then displays the partition pdev, file records, and paging records in the form

```
DISK  FILE-RECORDS  PAGING-RECORDS (DECIMAL).
pdev  file_recs     paging_recs
```

and prompts

```
Parameters OK?
```

If the numbers are incorrect, enter NO. FIX_DISK attempts to recompute the parameters. If the numbers are correct, enter YES. You then either see the message `This must be a rev 20, 21, 22. or 22.1 partition.` based on features this partition supports (such as variable geometry) and are prompted

```
Is this a rev 22.1 partition?
Is this a rev 22 partition?
Is this a rev 21 partition?
Is this a rev 20 partition?
```

or you are prompted

```
Is this a rev 19 partition?
```

Answer these questions appropriately depending on whether you are repairing a Rev. 22.1, a Rev. 22.0, a Rev. 21.0, a Rev. 20.0, or a Rev. 19.0 partition. FIX_DISK then repairs the DSKRAT from the information you have supplied. If this is a Rev. 22.0 or later partition, FIX_DISK then prompts to determine if this is a robust partition:

```
Is this a robust partition?
```

If this is a Rev. 22.1, a Rev. 22.0, a Rev. 21.0, or a Rev. 20.0 standard partition or you are converting to either Rev. 21.0 or Rev. 20.0, you are then prompted for the record allocation scheme you want, in this manner:

```
Forward Sectoring ?
Reverse Sectoring ?
```

These two lines are displayed one at a time and one will have the word `(default)` after it, depending on the CPU and disk controller in your system (see Table 10-2). If this is a Rev. 22.1 robust partition, you are not prompted for the record allocation scheme; it is set to forward allocation.

After you answer the record allocation prompt, FIX_DISK prompts for the disk controller mode in a similar manner:

```
Does this disk use -DBS ON
Does this disk use -DBS OFF
```

The default mode is indicated by the word `(default)` depending on whether the controller is an IDC1 or not.

`DSKRAT MISMATCH!`

The record allocation information in the DSKRAT disagrees with the record allocation information generated by FIX_DISK during processing. You did not specify –FIX so FIX_DISK does not correct the DSKRAT file. If the DSKRAT is all right, the message `DSKRAT OK` is displayed.

`DSKRAT UPDATED!`

The record allocation information in the DSKRAT disagrees with the record allocation information generated by FIX_DISK during processing, and you specified the –FIX option. FIX_DISK corrects the DSKRAT file. If the DSKRAT is all right, the message `DSKRAT OK` is displayed.

`Entry at word 'XX at hash index 'XX is in a circular list.`

The hash links are chained in a circular fashion. If you specify the –FIX option, the hash chain is terminated at the previous entry to correct the circularity.

`Entry at word XX is at hash index YY, it should be at ZZ.`

FIX_DISK computed the hash index for an entry in the directory and found that the entry is not where the index indicates. Use the –FIX option to correct the entry.

`Entry at word XX, hash index YY is not valid.`
`Removed from hash table.`

The entry found by FIX_DISK is not a hashed entry but may have pointed to an entry that does not have a name, such as an ACL or a deleted or free entry. Use the –FIX option to repair the entry.

`Entry at word XX not in free list.`

A free, or vacant, entry is found that is not in the list of free entries. The file system maintains a list of free, or vacant, directory entries. These are entries that have been deleted from the directory and thus are freed for use. To save space in the partition, you can remove these free entries by using the –UFD_COMPRESSION (–CMPR) option to compress the directory. This message indicates that the directory is damaged. Use the –FIX option to repair the directory and to put the entry in the free list.

`Entry at word XX not in hash table.`

The entry for a file is not in the directory's hash table. Use the –FIX option to put it in the hash table.

`Entry at word XX was already in hash table.`

An entry is found in a directory's hash table more than once. Use the –FIX option to remove duplicate entries, leaving only one entry for this file.

`This entry is deleted.`

If an error is encountered while FIX_DISK is checking the badspot file, the entry for that badspot is deleted from BADSPT if you specify –FIX.

`EOF occurs in the middle of entry at record 'YY, word XX`

A directory ends in the middle of the last directory entry. If you specify the –FIX option, the entry is deleted.

`Equivalence Section Begins.`
`The equivalence section has dd entries.`

FIX_DISK is processing the equivalence section of the badspot file. If processing is successful, the number of entries, *dd*, is displayed. Otherwise, another error message is displayed.

`Error reading first record of subsequent extent (uncorrectable).`

An uncorrectable error is encountered in checking a CAM file and attempting to read an extent subsequent to the first extent in the file. FIX_DISK truncates the file beginning at the bad extent. You can restore the file from the most recent backup.

`Error occurs during validation.`

FIX_DISK was unable to read the MFD record on a file system partition. You may need to remake the partition.

`Extent map full, unable to remap badspot record.`

This message refers to Contiguous Access Method (CAM) files. The index, or extent map, can contain 340 entries for pre-Rev. 22.1 partitions or 16381 entries for a Rev. 22.1 or later partition. The extent map is full and thus no room is available to remap a badspot because a new extent cannot be created. If you specify the –FIX option, the record is truncated from the file. You can restore the directory from a backup.

`Extent map is invalid.`
`The Beginning Record Address (BRA) pointer (XX) is bad, it points to a record that belongs to another file.`

This error refers to a CAM file on a robust partition. The BRA does not match the entry for the file in the directory. If you specify the –FIX option, FIX_DISK deletes the file. You can recover the file from a backup copy.

`Fatal error during remapping: record not found in dam index`

A record is found to be bad, but FIX_DISK is unable to remap it to a new location. The record is deleted.

`Fatal error has occurred in the file structure of a DAM Segment Directory!`

FIX_DISK is unable to fix the pointers in a DAM segment directory. The directory is truncated or deleted. You can restore the directory from a backup.

`The father pointer is bad.`
`It should be` *`XX`* `is` *`YY`*`.`
`Bad record address =` *`XX`* `BRA =` *`YY`* `Father =` *`ZZ`* `Type =` *`N`*

The Beginning Record Address (BRA) pointer of the first record of a file does not point to the BRA of the directory or SEGDIR in which this file is entered (its father). If you specify the –FIX option, the file is deleted. You can restore the directory from a backup.

`The` *`filename`* `file cannot be read, ignored!`

While FIX_DISK is processing existing badspots, FIX_DISK encounters an error while attempting to read either the DBS file or the BADSPT file. The badspot information is lost. If you have a list of the existing badspot, such as a copy of the –DUMP_DBS display, you can reenter the badspots by using the –ADDBADS option.

`File` *`filename`* `does not reference an ACL or Access Category!`

The ACL pointer of a file does not point to a valid ACL or access category. If you specify the –FIX option, the file reverts to default protection.

`File entry in password ufd has non-zero acl pointer!`

FIX_DISK has found a file in a password directory that has an ACL set on it. If you specify the –FIX option, the pointer is corrected.

`File` *`filename`* `is not pointed at by ACL it points to!`

The ACL pointer of a file does not point to a valid ACL or access category. If you specify –FIX, the file reverts to default protection.

`File` *`filename`* `points outside the directory!`

The ACL pointer of a file points to something in a different directory. If you specify the –FIX option, the ACL pointer is changed to the default type.

`The file structure of DSKRAT is bad.  (RAT_CK)`

The DSKRAT file contains bad record pointers or contains inconsistent information. If you specify both the –INTERACTIVE and the –FIX options, you are prompted

`Fix it?`

If you answer YES, FIX_DISK attempts to reconstruct the DSKRAT file structure. Otherwise, FIX_DISK aborts. (See the message `The DSKRAT header has wrong length.  (RAT_CK)`.)

`The file system is larger than the disk.  (RAT_CK)`

The partition size, as computed from the information in the RAT built by FIX_DISK, is larger than the size specified by the pdev you entered with the –DISK option on the command line. If you specify the –INTERACTIVE and –FIX options, you are prompted `Fix DSKRAT header?` If your response is YES, you are further prompted for information to repair the DSKRAT. If your response is NO or if you did not specify the –INTERACTIVE and –FIX options, FIX_DISK aborts. Use the –FIX option with the –INTERACTIVE option to repair the DSKRAT. (See the message `The DSKRAT header has wrong length.  (RAT_CK)`.)

```
file type mismatch!
it should be 0 instead of XX.
```

An error is found in the DAM index level; FIX_DISK corrects the error.

```
File type mismatch!
It should be XX instead of YY.
FRA = MMMM BRA = NNNN.
```

```
File type mismatch!
It should be XX instead of YY.
FRA = MMMM BRA = NNNN.
Bad record address = XX BRA = YY Father = ZZ Type = N>
```

The file type in the first record of the file does not match the file type in the directory entry. If you specify the –FIX option, the file is deleted. You can restore the file by using a backup procedure.

```
First partition (pdev) is not a Revision 21 or higher partition.
Must convert the head zero partition to Revision 21 before any other
partitions
```

The partition *pdev* is the head zero partition on this physical disk and must be converted before any other partitions on this disk, because the DBS file and the RMA for the entire physical disk must be on the head zero partition. If you intend to convert partitions, convert the head zero partition (*pdev*) first.

```
First partition must be in the same mode as the conversion
on this partition
```

In converting a partition that is not the head zero partition, you have specified either –DBS ON or –DBS OFF mode and the head zero partition of this spindle is not in the same mode. Either select the same mode for this partition as for the head zero partition or change the mode of the head zero partition.

```
The first partition needs to be assigned for this
conversion
```

To select the controller mode of a Rev. 21.0 or later partition that is not the head zero partition or to convert to a Rev. 21.0 format partition, you must assign the head zero partition so that FIX_DISK has access to the DBS.

`FIX_DISK is unable to read badspot 'XX to remap the record.`

In order to remap a data record, FIX_DISK must read the record first. In this case, FIX_DISK encounters an uncorrectable read-error and cannot read the data. If this is a Rev. 21.0 conversion, the message `Resetting badspot file!` is also displayed. If you used the –FAST option, FIX_DISK advises you to run full FIX_DISK (without –FAST) and add the badspot. If you did not initially use –FAST, FIX_DISK advises you to rerun full FIX_DISK and add the badspot. The data in the record that has become a badspot is lost in these cases; restore the file from backups.

`Fix DSKRAT header?`

You have specified the –INT and –FIX options. See the discussion under `The DSKRAT header has wrong length.` (RAT_CK)

```
The -FIX option is required if any of these options are specified:
   -CMPR, -ADD_BADSPOT, -CONVERT_19, -CONVERT_20, -CONVERT_21, -CONVERT_22.1,
   -DBS (-IC/-AC), -SECTOR (-ODI/-RDI), -MIN_EXTENT_SIZE, -MAX_EXTENT_SIZE.
```

In order to compress directories, convert partitions to another revision, select controller modes, add badspot entries to BADSPT, select record allocation mode, or change the maximum or minimum extent size of CAM files, you must specify the –FIX option. If you do not, FIX_DISK aborts.

```
For a Rev 21 conversion, the -disk_type model MUST be
specified.
```

When you use the –CONVERT_21 option to convert a partition to Rev. 21.0 format, you must also use the –DISK_TYPE option. FIX_DISK builds the DBS file and the RMA on a Rev. 21.0-format partition and must know the type of disk, because the size of the DBS file and RMA depend on the disk geometry.

```
The following CAM file is in the DBS/RMA area and will prevent
conversion of this disk to revision 21.
```
pathname

After this message appears, you see the following message.

```
CAM file(s) in DBS/RMA area prevent conversion to rev 21 disk.
Please save & delete these files, and restore following the conversion.
```

When converting a partition to Rev. 21.0 with –CONVERT_21, a CAM file, *pathname*, is found in the area of the head zero partition where it is necessary for FIX_DISK to create a DBS file and the RMA. FIX_DISK aborts. If you want to convert this partition to Rev. 21.0, save the offending files to tape, delete them from the disk, convert the disk, and then restore the CAM files you saved and deleted.

```
The forward pointer XX is bad, it is not in the range of
the current partition.
   Bad record address = XX BRA = YY Father = ZZ Type = N
```

The address that the forward pointer points to is not between zero and the maximum record address of this partition. If you specify the –FIX option, the file is truncated. You can restore the file by using a backup procedure.

```
The forward pointer XX is bad, it points to a record that
belongs to another file.
Bad record address = XX BRA = YY Father = ZZ Type = N
```

The record that the forward pointer points to belongs to another file. This error may occur if the current DSKRAT is bad or if the BADSPT file was changed since the last time FIX_DISK was run. If you specify the –FIX option, the file is truncated. You can restore the file by using a backup procedure.

```
The forward pointer of the previous record does not point
to the pre-remap record!
```

The forward pointer of the previous record points to another file after having been remapped. This error may occur if the current DSKRAT is bad or if the BADSPT file was changed since the last time FIX_DISK was run. If you specify the –FIX option, the file is truncated. You can restore the file by using a backup procedure.

```
The forward pointer of the top level index record of DAM
file is not zero.
```

The top-level index must be only one record long; therefore, the forward pointer of this record must be zero. If you specify the –FIX option, the pointer is set to zero.

```
Free list at word XX contains a non free entry.
```

The directory entry found in the free list is an entry that is in use. The file system maintains a list of free, or vacant, directory entries. These are entries that have been deleted from the directory and thus freed for use. To save space in the partition, you can remove these free entries by using the –UFD_COMPRESSION (–CMPR) option to compress the directory. Use the –FIX option to remove that entry from the free list.

```
Free list at word XX contains a too small free entry.
```

An invalid entry is found in the free list. The file system maintains a list of free, or vacant, directory entries. These are entries that have been deleted from the directory and thus freed for use. To save space in the partition, you can remove these free entries by using the –UFD_COMPRESSION (–CMPR) option to compress the directory. Use the –FIX option to remove that entry from the free list.

`Free list is circular at word XX.`

An entry is found in the free list that points to the entry that points back to it. The file system maintains a list of free, or vacant, directory entries. These are entries that have been deleted from the directory and thus freed for use. To save space in the partition, you can remove these free entries by using the –UFD_COMPRESSION (–CMPR) option to compress the directory. Use the –FIX option to correct the pointers.

`Handling soft disk error recovery for 'XX`

FIX_DISK has found a correctable error at record number *XX* (octal) and will add it to the badspot file.

`The hash table size is incorrect.`

The size of the hash table for a directory is either less than the minimum or greater than the maximum. Use the –FIX option to correct the inconsistency.

`Inconsistent entry.  Record = XX, Word = YY`
`Entry is deleted!`

The information in an entry within a directory is inconsistent and cannot be reconciled. If you specify the –FIX and –DUFE options, the entry of this file is deleted. You can restore the file by using a backup procedure.

`Incorrect hash function version number; is XX,`
`should be YY.`

The algorithm used to compute entries for the hash table is invalid. Other messages will be displayed concerning entries in the hash table. Use the –FIX option to correct the hashing function.

`Incorrect logical end of file (correctable).`

A CAM file contains more records than are indicated in the extent map of the file. Use the –FIX option to fix the inconsistency.

`Index entry NN is bad, should be XX is YY.`

The list of records at *NN* contains a record address that is not part of this directory. Use the –FIX option to correct the entry.

`Index entry NN is bad, should be 0 is XX.`

The list of records at *NN* contains a record address that is not part of this directory. Use the –FIX option to correct the entry.

`The index level of this DAM file is incorrect.`
`It should be YY instead of XX!`

The index level word of this record is incorrect. It should be zero for SAM files or one less than the previous level for DAM files. If you specify the –FIX option, the index level word is fixed.

```
The index level of this record is incorrect.
It should be XX is YY.
Bad record address = XX BRA = YY Father = ZZ Type = N
```

A file that is not a DAM file is found to have an index level other than 0. It is fixed by making the index level 0.

```
The index level of the top index record of a DAM file is
incorrect.
It is XX, it should be > 0 & < 4.
```

The index level of the top index record of a DAM file is incorrect. It should be between 0 and 4. If you specify the –FIX option, the index record is corrected.

```
The index of this DAM file is too short.  The DAM index may
be rebuilt!
```

A DAM file contains more data records than its index indicates. If you specify the –FIX option, the index is fixed if the extra index words will fit into the index record; if the words will not fit, the next message is displayed.

```
The index of this DAM file is too short to represent the
data records.
```

This message is displayed if the condition noted in the previous message is not reparable. A DAM file contains more data records than its index indicates. If you specify the –FIX option, the inconsistency between the index and the number of data records is resolved, and the file is truncated. You can restore the file by using a backup procedure.

```
Insufficient disk space available for conversion.  Please
free up dd records on partition pdev.
```

When you use the –CONVERT_21 option on the head zero partition, FIX_DISK must move records from the DBS area and create the DYNBSP file. Before doing this, FIX_DISK must ensure that there is enough space to hold the DBS file, the RMA records, and the DYNBSP file. FIX_DISK finds that there is not enough space and informs you that you must delete *dd* records from the partition.

```
Internal error - UFD overflow!
```

The number of files and the length of their filenames are greater than the amount of space allotted for them in the FIX_DISK database. This error is unlikely to occur unless you have a partition with an extremely large number of small files with long (32-character) filenames and many deeply nested directories.

`Invalid -DBS argument.  Only ON or OFF allowed.`

You specified an argument with the –DBS option that is not one of the two acceptable arguments; perhaps you mistyped the argument. Reenter the command line and the –DBS option with either the ON or the OFF argument.

`Invalid disk model name; Valid disk types are:`

Either you did not enter a valid disk type with the –DISK_TYPE option or you did not enter any disk type. FIX_DISK displays a list of types; choose the correct one from the list and reenter the command line. (The list is shown in the discussion of the message `The DSKRAT header has wrong length.`)

`Invalid max nested level, set to max of 3855.`

The maximum level of nested directories specified with the –MAX_NESTED_LEVEL option is invalid and out of the range of 0 through 3855. It is set to 3855.

`Invalid -SECTOR argument.  Only FORward or REVerse allowed.`

You specified an invalid –SECTOR argument; perhaps you mistyped the argument. Reenter your command line with the correct argument.

`The length of UFD header is incorrect.`

See the message `The DSKRAT header has wrong length.`

`The MFD is full, unable to add` *`filename`* `file.`

The partition has insufficient space to enable FIX_DISK to write or add to either the BADSPT file or the DYNBSP file. FIX_DISK continues, but badspot handling is disabled. Clean up the partition by having users delete unnecessary files and directories.

```
MINIMUM_EXTENT_SIZE cannot be greater than
MAXIMUM_EXTENT_SIZE.  (rat_ck)
Current MINIMUM and MAXIMUM extent values are: min, max>
```

You specified the minimum extent size of a CAM file to be larger than the maximum extent size. The current maximum and minimum extent size are displayed and FIX_DISK aborts. If you want to change the maximum and minimum extent sizes, reenter the command line with proper values.

```
MINIMUM_EXTENT_SIZE cannot be greater than
MAXIMUM_EXTENT_SIZE.  (cl_par)
Please respecify MINIMUM and/or MAXIMUM extent values.
```

You specified the CAM file minimum extent size to be larger than the maximum extent size. Reenter the *size* values on a new command line.

`Missing badspot number. (cl_par).`

You specified the –ADD_BADSPOTS option, but did not enter any badspot records on the command line.

`Missing maximum extent size (cl_par)>`
`Missing minimum extent size (cl_par)`

You specified either the –MAXSIZ or the –MINSIZ option but did not specify the corresponding CAM file extent *size*. Reenter the command line, specifying the appropriate *size*.

`Moving DBS file system records.`

In Nondynamic Badspot Handling (–DBS OFF) mode, any data stored in the RMA must be moved to the partition containing the file associated with that data.

`Multiple CONVERT_ options can not be specified at the same time.`

Only one of the –CONVERT_19, –CONVERT_20, –CONVERT_21 or –CONVERT_22.1 options can be specified on a command line.

`The next extent map record address of the previous map does not point to the pre-remap record!`

The address of the next extent map in the previous extent map does not point to either the beginning record address or the remapped beginning record address of the next extent map. If you specified the –FIX option, the file is truncated. You can restore the file from a recent backup.

`The next index does not match the forward pointer of the current data record!`

The pointers of the index section and the data section do not agree. If you specify the –FIX option, the following actions are taken: the backward pointer of the record that is pointed to by the DAM index and the backward pointer of the record that is pointed to by the forward pointer of the current data record are examined; the record whose backward pointer points to the previous data record is chosen; if neither backward pointer points to the previous record or both backward pointers point to the previous record, the file is truncated.

`Non zero data count(s) past logical eof (correctable).`

FIX_DISK has found an indication of more records in a file going beyond the logical end-of-file (EOF). If you specify the –FIX option, the data counts of all records beyond the logical EOF are set to zero.

`The number of heads is different. It should be` *`YY`* `is` *`XX`*`.`

See the message `The DSKRAT header has wrong length.`

`Only one of -DBS, -IC, or -AC may be used.`

You cannot specify more than one of the controller mode selection options (–DBS, –AC, or –IC) on the same command line. Reenter the command line with the one selection you want.

`Paging Records (Decimal)?`

You are interactively repairing the DSKRAT. See the discussion under `The DSKRAT header has wrong length.`

`Parent of a file is not a UFD or SEGDIR`

The pointer to a directory or to a segment directory identifies a file header. FIX_DISK aborts.

`The partition cannot be handled by this version of Fix_disk.`

FIX_DISK found a discrepancy in the DSKRAT header or in the pdev indicating that the partition on which you are running FIX_DISK and the version of FIX_DISK you are using are incompatible. FIX_DISK aborts unless you specify the –FIX and the –INTERACTIVE options. If you do specify the –FIX and the –INTERACTIVE options, you are prompted

```
Do you wish to change the disk revision?
```

A NO response causes FIX_DISK to abort. A YES response initiates the dialog discussed under the message `The DSKRAT header has wrong length.`

`Partition not shutdown properly during the previous session; please run fix_disk!`

The partition was not shut down under PRIMOS with the SHUTDN command; for example, the system halted or the disk drive was spun down. This message results if you run FIX_DISK with the –CHECK option; it is the only message you will see indicating problems when you use the –CHECK option. To check for other problems, run FIX_DISK but do not specify either –CHECK or –FIX. Use –FIX to correct any problems.

`Physical Device Number is missing.   (CL_PAR)`

The physical device number is not specified in the command line with the –DISK option. FIX_DISK aborts. Reenter the command line and specify the correct pdev.

`Please answer "YES" or "NO"?`

The previous prompt requires a yes or no answer. Acceptable answers, in either uppercase or lowercase, are YES, YE, Y, NO, N.

`The physical record size is different.  (RAT_CK).`

See the message `The DSKRAT header has wrong length.`

`Pre Rev 22 disks may not be robust.`

In checking the DSKRAT, FIX_DISK found the partition to be labeled as robust but of a disk revision prior to Rev. 22.0. FIX_DISK changes the DSKRAT to indicate that the partition is not robust.

`The prev. extent map record address of the next extent map record does not point back to the pre-remap record!`

The record address of the previous extent map does not point to either the beginning record address or the remapped beginning record address for the next extent. If you specified the –FIX option, the file is truncated. You can restore the file from a backup.

`Processing add badspot request for` *`XX`*

You used the –ADD_BADSPOT option and the badspot (record address *XX*) is being remapped to the RMA.

`The Quota system may be incorrect.`

This message is issued if the partition was changed under PRIMOS II or if the partition was not shut down correctly. Because PRIMOS II does not support quotas, some directories on this partition may contain incorrect quota information.

`Record address` *`XX`* `is out of range` *`YY`*`.`

The next record address XX is out of range of the current record address YY in a segment directory. If you specify the –FIX option, the file is deleted. You can restore the file by using a backup procedure.

`The record:` *`XX`* `has not been initialized`
`The record will be zeroed out.`

`The record:` *`XX`* `has not been initialized`
`The file will be truncated.`

During verification of the integrity of the header of a record on a robust partition, the beginning record address for the record is found to be incorrect or the record's identification is found not to be unique. If you used the –TRUNCATE option, the file is truncated; otherwise, the record is replaced with zeros.

`REMAP: Track =` *`NN`* `Head =` *`NN`* `Sector =` *`NN`* `Record = '`*`XX`*

You specified the –LIST_BADSPOTS option and the badspots are listed.

`Re-run FIX_DISK without -FAST option to add this badspot.`

See the message `FIX_DISK is unable to read badspot 'XX to remap the record.`

`Resetting badspot file!`

See the message `FIX_DISK is unable to read badspot 'XX to remap the record.`

`The -SECTOR option requires an argument (FORward or REVerse).`

You must specify an argument with the –SECTOR option, either FORWARD or REVERSE. Reenter your command line.

`Segdir has more than 64K entries!`

The maximum size of a segment directory (SEGDIR) is 64K halfwords. If a SEGDIR exceeds this limit, it is truncated if you specify the –FIX option. You can restore the directory by using a backup procedure.

`Single zero record added starting at word XX. chars length`

A zero record has been created in place of an unreadable record in a SAM file.

`Single zero record added starting at word XX. index length`

A zero record has been created in place of an unreadable record in a DAM file and the index and index length are displayed.

`Split disk?`

You are interactively repairing the DSKRAT. See the discussion under `The DSKRAT header has wrong length.`

`Structure error for a DAM segdir. (SEGDR_CK)`

The header information for a DAM segment directory is inconsistent with the file information. If you specify the –FIX option, the inconsistency is corrected.

`System console command only (fix_disk).`

You must be at the supervisor terminal to use the FIX_DISK –COMDEV command.

`System file is bad - ignored!`

An error that would normally cause deletion of a file has been found in one of the MFD's special files (BOOT, MFD, DSKRAT). The file will not be deleted. To correct this condition, you may need to remake the partition by using MAKE.

`This is a remapping record of record address 'XX.`

A read error occurred while FIX_DISK was attempting to remap a record. The record is not remapped, but FIX_DISK continues.

`This is a revision rev robust partition.`
`This is a revision rev partition.`

FIX_DISK determines the type (robust or standard) and the revision of the partition that is being repaired.

`Too many paging records.`

You are interactively repairing the DSKRAT. See the discussion under `The DSKRAT header has wrong length.`

`The Tree Used count is bad. It should be YY instead of XX`

The tree-used count of this directory does not match the tree-used count that is calculated by FIX_DISK. If you specify the –FIX option, the tree-used count is fixed.

`Treename is not available.`

FIX_DISK found an error on its second pass in this record. Information on what file the record belongs to is missing. This error should occur only if the disk is failing.

`2 files point to the same record!`
`bad record address = XX`
`BRA = YY Father = ZZ`

`2 files point to the same record!`
`Bad record address = XX BRA = YY Father = ZZ Type = N`

Two files each have a record with the same address; that is, they point to the same record. If you specify the –FIX option, all entries for the second and subsequent files pointing to that record are truncated. You can restore the files by using a backup procedure.

`The UFD header is missing.`

The header of the MFD is missing or the wrong type. If you specify the –FIX option, the directory file is deleted. Try to repair the header by using the –INTERACTIVE and –FIX options.

`The UFD header length is incorrect.`

The length of the directory header is wrong. If you specify –FIX, it is corrected.

`The UFD header of the MFD is missing.`

In checking to see if the partition is a valid PRIMOS file system, FIX_DISK does not find the MFD header entry. FIX_DISK aborts. Reformat the partition by using MAKE.

`Ufd nesting exceeds maximum specified.`

FIX_DISK cannot follow the directory tree because directories are nested to more than *n* levels deep (default *n* = 99). FIX_DISK aborts unless you specify the –AUTO_TRUNCATION option, in which case directories that are nested too deeply are truncated. You can restore the directory from a backup.

`UFD truncated.`

If an error occurs in a directory while FIX_DISK is checking the directory (the name of which is displayed in the BEGIN message), the directory is truncated if you specify the –FIX option. You can restore the directory from a backup.

`Unable to add_badspot 'XX.   (fix_disk)`

FIX_DISK is unable to remap a badspot to another record because the partition is full. Clean up the partition and rerun FIX_DISK.

`Unable to add zero record(s).`

FIX_DISK found an unreadable record in a file, but is unable to add a record of zeros because the partition is full. FIX_DISK aborts.

`Unable to add zero record(s).   (fix_disk)`

FIX_DISK found an unreadable record in a SAM file, but is unable to add a record of zeros because the partition is full. The file is truncated. You can restore the file by using a backup procedure.

```
Unable to correctly handle password origin directories.
Origin was: origin_path
Please use the -set option to the ORIGIN command to set the
origin of the console.   (fix_disk)
```

This message is displayed only when the –COMDEV option is used and only when the supervisor terminal origin directory is a password (not an ACL) directory. Use the command ORIGIN –SET *origin_path* to set the origin directory to origin_path. The supervisor terminal origin directory is normally CMDNC0.

`Unable to create a new extent map.`

Just previous to this message, FIX_DISK displays the message Disk is full!; thus, FIX_DISK cannot create the new extent map that is necessary in order to remap a badspot. If you specify –FIX, the file is truncated. You can restore the file from a backup.

`Unable to move record, rewriting to attempt to fix it up.`

FIX_DISK has encountered a badspot in one of the first 16 records of the partition and this record cannot be moved. The first 16 records on a partition cannot be defective. FIX_DISK attempts to write to the record again and if the write attempt is unsuccessful, FIX_DISK aborts. If FIX_DISK aborts, call PrimeService.

`Unable to read extent map.`

FIX_DISK is unable to find the extent map of a CAM file possibly because of a badspot. If you specify the –FIX option, the file is deleted. You can restore the file by using a backup procedure.

`Unable to read next extent map.`

FIX_DISK cannot read a subsequent extent map after the first extent map, possibly because of a badspot. If you specified the –FIX option, the file is truncated at this point. You can restore the file from a recent backup.

`Unable to register geometry information with PRIMOS.`
`(RAT_CK)`

You are using the wrong version of FIX_DISK. Rev. 20 FIX_DISK attempts to determine the number of sectors per track on the disk, since disks may have different geometries at Rev. 20. FIX_DISK aborts.

`Unable to reset origin directory at this revision of PRIMOS.`
`Origin was:` *`origin_path`*

You used the –COMDEV option and an older version of PRIMOS or FIX_DISK is unable to reset the origin. Cold start the system to set the origin directory of the supervisor terminal.

`Unable to skip second badspot in this file.  (fix_disk)`
`BRA =` *`XX`*`, FRA =` *`YY`*`.`

FIX_DISK has found more than one bad record in one file, and is unable to determine how many bad records have been found. If you specify the –FIX option, the file is truncated. You can restore the file by using a backup procedure.

`Unexpected error` *`NN`* `from hash_uid.  (fix_disk)`

While attempting to compute a hash index, the hashing algorithm returns an error. The error is ignored. If you specify the –FIX option and this message is displayed, there may be other problems with the disk. Call PrimeService.

`Unknown entry type (`*`XX`*`).  Record =` *`YY`* `Word =` *`ZZ`*

FIX_DISK does not recognize the entry type of the current record. If you specify the –FIX option, the file to which the record belongs is truncated. You can restore the file by using a backup procedure.

`Unknown file type XX encountered.`
`Record = YY Index = ZZ`

The file type *XX* in the file entry is unknown. It is either an illegal file type or a new file type that is not recognized by this version of FIX_DISK. If you specify the –DUFE and –FIX options, this file entry is deleted. If you omit these options, the file entry is left untouched, and no compression is performed for the directory in which this file entry resides.

`Unknown logical file type.`

FIX_DISK does not recognize the logical file type. If you specify the –FIX option on the command line, FIX_DISK changes the file type to the default type and issues the message: `Changed to default type.`

`Unknown number of records removed:`
`Two zero records added starting at word XX.`

FIX_DISK has found more than one contiguous bad record. Two records of zeros are added and any additional good records follow the zero records.

`Unrecoverable read error. (fix_disk)`

FIX_DISK cannot read the DSKRAT file while trying to reconstruct a truncated SAM file by adding zero records. The SAM file is left truncated, but other errors will probably occur.

`Unrecoverable read error. (VFY_RAT)`
`Unrecoverable read error while reading the DSKRAT file. (RAT_CK)`
`Unrecoverable read error while reading DSKRAT.`
`(repair_rat_file_header)`

These messages may be issued if the DSKRAT file contains any bad record pointers or contains inconsistent information. If you specify both the –INTERACTIVE and –FIX options on the command line, FIX_DISK attempts to reconstruct the DSKRAT file structure. Otherwise, FIX_DISK aborts. In the latter case, rerun FIX_DISK with the –FIX and –INTERACTIVE options.

`Unrecoverable read error reading first partition's DSKRAT file (BADSPT)`
`Unrecoverable error reading first partition's DSKRAT file. (RAT_CK)`

Be sure that the head zero partition is assigned to you. If it is not, it may be damaged; run FIX_DISK on the head zero partition in that case.

`Unrecoverable read error while reading`
`UFD.(update_dir_count)`

This is an extremely unlikely error. A CAM file was remapped and FIX_DISK is unable to do the directory quota update. This does not cause a problem and the quota count will be fixed the next time you run FIX_DISK.

`Vacant entry on free list but not a main entry at XX.`

A list of free, or vacant, entries in a directory is maintained. These are entries that have been deleted and thus freed for use. Adjacent free entries are merged into one entry. In this case, the free list points within a merged entry rather than to the start of the entry. Use the –FIX option to correct the list.

`Warning: Illegal reference to record zero ignored.`

A pointer has been found pointing to the BOOT file from a record unrelated to the BOOT file. This message indicates corruption of a directory entry. If you specify the –FIX option, the entry is deleted.

`WARNING: The 3rd file entry of the MFD file is not BOOT.`
`The partition may need to be remade!`

FIX_DISK checks that the third entry in the MFD is BOOT. If this entry is missing, FIX_DISK does not abort. If you specified the –UFD_COMPRESSION option on the command line, the message `Compression has been disabled!` is also displayed.

`The word count of record XX is bad.`
`The word count of this record XX is bad.`
`Bad record address = XX BRA = YY Father = ZZ Type = N`

The data halfword count of a record is not appropriate. For every record except the last record, the data halfword count should equal the record data size. The data halfword count of the last record should be between zero and the record data size. If you specify the –FIX option, the halfword count is set to the appropriate value.

`Writing DBS file`

FIX_DISK is creating or updating a blank DBS file. The intelligent disk controller fills in the badspot locations.

`You may not change the default sectoring on a robust partition.`
`-SECTOR argument option ignored.`

Do not use either the –SECTOR FORWARD (–ODI) or –SECTOR REVERSE (–RDI) option with robust partitions because the allocation direction is always forward with an interleave factor of three on robust partitions. The *argument* you specified, either FORWARD or REVERSE, is ignored and FIX_DISK continues.

`You must use the -auto_truncation flag to process this disk`

If directory nesting exceeds the maximum specified, and you did not specify the –AUTO_TRUNCATION option, this message is displayed and FIX_DISK aborts. Either specify more levels of nesting with the –MAX_NESTED_LEVEL option or use the –AUTO_TRUNCATION option to truncate the directory tree.

`Zero record will be added to file.`

FIX_DISK has encountered an unreadable record. A record of zeros will be added to mark the location of the unreadable record.

Mirroring Messages

C

This appendix lists the informational and error messages associated with the mirroring process and the mirroring commands. The messages are presented in alphabetical order with a brief explanation of each message. If an article, such as *The* or *This*, is the first word in the message, the second word is used for alphabetizing. Variable names in the messages are italicized, for example, *pdev*.

For a discussion of mirroring and the mirroring commands, see Chapter 9.

`A catch-up copy of primary device` *`pdev1`* `to secondary device` *`pdev2`* `will be started.`
`Are you sure you want to continue?`

You specified the MIRROR_ON command and the two partitions to be mirrored are not identical either based on their shutdown stamps, or because they were not started or stopped simultaneously. If you want to mirror these partitions and have the catch-up copy start, answer YES. Be sure you know which partition has the most recent data. (See the warning in the Catch-up Copy section of Chapter 9.)

`Cannot mirror on COMDEV pdev` *`pdev`* `- incorrect Rev, mode, or model.`
`Cannot mirror COMDEV pdev` *`pdev`* `- incorrect Rev. or mode.`
`Cannot mirror on COMDVM pdev` *`pdev`* `- incorrect Rev., mode, or model.`
`Cannot mirror on PAGINM pdev` *`pdev`* `- incorrect Rev, mode, or model.`
`Cannot mirror PAGING pdev` *`pdev`* `- incorrect Rev. or mode.`

These messages are displayed at cold start. The *pdev* specified in the configuration file with either the COMDVM, the PAGINM, or the PAGING directive is that of a partition that is not a Rev. 21.0 or later partition or that is not in Dynamic Badspot Handling (–DBS ON or –IC) mode.

`Cannot mirror on PAGINM pdev` *`pdev`* `- mirror table full.`

More than 128 partitions have been specified as mirrored pairs and another partition cannot be started as a mirror.

`Cannot shut down both - devices` *`pdev1`* `and` *`pdev2`* `are being used for paging.  (mirror_off)`

You cannot use the MIRROR_OFF –SHUT_BOTH command on paging partitions. One of the paging partitions must be running while the system is running. You can shut down one of the pair of mirrored paging partitions.

The other paging partition can be shut down only at system shutdown with the SHUTDN ALL command.

`Cannot shut down primary device` *`pdev1`* `- catch-up copy did not complete successfully and secondary device` *`pdev2`* `is not up to date. (mirror_off)`

The copy server logged out before completing the catch-up copy to the secondary partition of the mirrored pair. This may have happened either because an uncorrectable read-error on the primary partition occurred or because an uncorrectable write-error occurred or because the catch-up copy server may have been logged out. Look at the output on the supervisor terminal to see if there was an uncorrectable write error on the secondary partition. If so, run FIX_DISK with the –ADD_BADSPOT option on the secondary partition to add that badspot. Then use the MIRROR_ON command to start the mirror and the copy server. If there was no uncorrectable error, assume the copy server was logged out and use the MIRROR_ON command to start it.

`Cannot shut down primary device` *`pdev1`* `- catch-up copy is in progress to bring secondary device` *`pdev2`* `up to date. (mirror_off)`

You cannot shut down the primary mirrored partition before the catch-up copy to the secondary partition is finished because the secondary partition is not up-to-date and cannot be used until the catch-up copy is complete.

`Cannot shut down primary device` *`pdev1`* `- primary is only survivor. (mirror_off)`

You can shut down the primary partition of the mirrored pair with the –SHUT_PRIMARY option only if the mirror is on or if the primary partition is damaged and the secondary partition is functioning.

`Cannot shut down secondary device` *`pdev2`* `- secondary is only survivor. (mirror_off)`

You can shut down the secondary partition of the mirrored pair with the –SHUT_SECONDARY option only if the mirror is on or if the secondary partition is damaged and the primary partition is functioning.

`Cannot spawn Copy Server for Ldev` *`ldev`* `- no right. (START_COPYSR)`

The copy server phantom can be started only from the supervisor terminal by the User 1 process.

`Cannot spawn Copy Server for Ldev` *`ldev`*`. Unable to attach to dirname. (START_COPYSR)`

In attempting to make a catch-up copy for logical device *ldev*, the copy server cannot be started because an attach to the directory *dirname* (usually SYSTEM) was not possible. This may be because the partition or the directory does not exist or because user SYSTEM does not have proper access rights to the directory. Check the ACLs on *dirname*; SYSTEM should have ALL access.

`Cannot spawn Copy Server for Ldev` *`ldev`*`, status code (START_COPYSR)`

The copy server cannot be started to make a catch-up copy for logical device *ldev*. The status *code* returned is one of the following: e$nrit (Insufficient access rights.); e$bver (Incorrect version number.); e$bpar (Bad parameter.); e$exst (Already exists.); or e$nval (Validation error.) Take appropriate action based on the status.

`Catch-up copy in progress. Copy Server must be logged out before the mirror is turned off. (mirror_off)`

A catch-up copy must not be in progress on the pair of partitions for which mirroring is being turned off with the MIRROR_OFF command using either the –SHUT_BOTH or the –SHUT_SECONDARY option. Ordinarily, the MIRROR_OFF command logs out the copy server and you do not see this message. If it is necessary for you to stop the catch-up copy facility, first log out the copy server process (copy_server). Then use the MIRROR_OFF command to stop the mirroring process.

`COMDVM must follow COMDEV directive.`
`COMDVM directive ignored.`

The COMDVM directive must follow the COMDEV directive in the configuration file. If you want the command device to be mirrored at cold start, change the configuration file so that it is correct.

`Copy Server for Ldev` *`ldev`* `aborted by raising of hardware condition at pointer Fault type was` *`nn`*

The copy server aborted due to hardware errors.

`Copy Server for Ldev` *`ldev`* `completed catch-up copy.`

The copy server successfully updated the two mirrored partitions, the primary being logical device *ldev*, so that they are identical.

`Copy Server for Ldev` *`ldev`* `logging out due to forced logout.`

The copy server has been forcibly logged out. To restart the copy server, first use the MIRROR_OFF command and then use the MIRROR_ON command.

`Copy Server for Ldev` *`ldev`* `logging out due to error or shutdown.`

The copy process failed in attempting to update the mirrored pair of partitions, the primary being logical device *ldev*, due either to a disk read-error or write-error or to a shutdown by the MIRROR_OFF command. Use the MIRROR_OFF command and run FIX_DISK on the mirrored partitions. Then you can restart the mirror with the MIRROR_ON command. This will also cause the catch-up copy to start. (See the section Catch-up Copy Failure in Chapter 9.)

`Copy Server for Ldev` *`ldev`* `logging out - mirror was inactive.`

A catch-up copy for mirroring logical device *ldev* started but the mirroring process aborted. This is an unlikely occurrence.

`Copy Server for Ldev` *`ldev`* `starting catch-up copy.`

The catch-up copy for the mirror of logical device *ldev* is starting.

`Could not CLOSE PDEV` *`pdev1`* `due to controller or device errors. (mirror_on)`
`Could not CLOSE PDEV` *`pdev2`* `due to controller or device errors. (mirror_on)`
`Could not OPEN PDEV` *`pdev2`* `due to controller or device errors. (mirror_on)`

A controller or other hardware error occurred when you attempted to start the mirroring process. Check to be sure that the disk drive is powered on, that it is on line, and that it is not write-protected. Similar messages may also appear when you use ADDISK.

If you attempt a warm start on your system and it is successful, you may encounter an error subsequent to the warm start when adding or mirroring disks that are associated with an intelligent disk controller. In this case, you may see the last message above or a similar message from the ADDISK command. If this message does appear, wait until one of the the following messages appears at the supervisor terminal.

`DLL and init ICOP complete (`*`cntrllr_addrss`*`) - (disk_init)`
`DLL of 7210 controller complete. (`*`cntrllr_addrss`*`) - (disk_init)`

The intelligent disk controller is now downloaded again and you can now add or mirror the disks that you previously attempted to add or mirror.

This problem occurs only on disk drives associated with intelligent disk controllers and only on those disk drives where all the logical disks (partitions) on a physical disk were shut down prior to the warm start.Could not shut down secondary device *pdev2*. You answered something other than Y, YE, or YES at the prompt asking if you want the catch-up copy to continue (see the first message in this section), causing PRIMOS to attempt a shutdown of the secondary partition. PRIMOS encountered an error and the secondary partition is now not accessible. Use the MIRROR_OFF command with the –SHUT_SECONDARY option. Then run FIX_DISK on the secondary partition.Device *pdev* is not mirrored. (mirror_off) The *pdev* specified with the MIRROR_OFF command is not part of a mirrored pair. Check the pdev for the partition you want to shut down or remove from mirroring.

`Could not shut down secondary device` *`pdev2`*`.`

You answered something other than Y, YE, or YES at the prompt asking if you want the catch-up copy to continue (see the first message in this section), causing PRIMOS to attempt a shutdown of the secondary partition. PRIMOS encountered an error and the secondary partition is now not accessible. Use the MIRROR_OFF command with the –SHUT_SECONDARY option. Then run FIX_DISK on the secondary partition.

`Device` *`pdev`* `is not mirrored.  (mirror_off)`

The *pdev* specified with the MIRROR_OFF command is not part of a mirrored pair. Check the pdev for the partition you want to shut down or remove from mirroring.

`*** Disk` *`pdev`* `- A dynamic badspotting partition cannot be started on this controller.  Run FIX_DISK to convert to an -All_Controller format partition.`

Both the primary partition and the secondary partition must be associated with disk controllers that can perform dynamic badspot handling. You have attempted to start mirroring with a Dynamic Badspot Handling (–DBS ON or –IC) mode partition that is on a nonintelligent disk controller. Run FIX_DISK with the –DBS OFF (–AC) option on this partition. This message is also displayed if you attempt to add a Dynamic Badspot Handling (–DBS ON) mode partition on a disk drive associated with a nonintelligent disk controller.

`Disk Read Error: Mirror Broken.`
`Unable to shutdown disk properly, please run FIX_DISK`

A disk read error occurred when shutting down the mirrored disks with the MIRROR_OFF command and the mirror is broken. You should run FIX_DISK on the partition. A catch-up copy will start when you restart this mirror.

`*** Disk write errors detected, Run Fix_Disk.`

PRIMOS detected file system errors when you shut down the partition with the MIRROR_OFF command. You should run FIX_DISK on the partition.

`Duplicate pdev` *`pdev`* `is ignored by COMDVM directive.`
`Duplicate PDEV` *`pdev`* `is ignored by PAGINM directive.`

These messages are displayed at cold start. The same *pdev* is entered more than once as an argument to either the COMDVM directive, the PAGINM directive, or another directive in the configuration file. Mirroring will not take place on either the command device or the paging device indicated. Check the configuration file to see what you intended.

`Error while shutting down disk. Please run Fix_Disk`

A file system error is detected while shutting down a mirrored pair of disks with the MIRROR_OFF command. You should run FIX_DISK on the partitions.

`Force option can only be used with Shut_Both option. (mirror_off)`

You can use the –FORCE option only with MIRROR_OFF –SHUT_BOTH.

`I/O still pending on device` *`pdev`*`. Unable to shut down disk. Please retry. (mirror_off)`

Read requests or write requests are pending either on the primary partition five seconds after you issued the –SHUT_PRIMARY option or on the secondary partition five seconds after you issued the –SHUT_SECONDARY option. Reissue the command option.

`Invalid COMDVM argument` *`pdev`* `is ignored by COMDVM directive.`
`Invalid PAGINM argument` *`pdev`* `is ignored by PAGINM directive.`

These messages are displayed at cold start. A *pdev* argument for the COMDVM directive to mirror the command device or for the PAGINM directive to mirror a paging device is an invalid pdev and is ignored. If you want to mirror the command device or a paging device at startup, correct the pdev in the configuration file.

`Maximum number of mirrored partitions already in use. (mirror_on)`

No more than 128 partitions (64 pairs) may be set up as mirrored pairs. You cannot start another mirror with the MIRROR_ON command if there are already 64 pairs of mirrored partitions.

`The mirror of primary device` *`pdev1`* `and secondary device` *`pdev2`* `will be broken.`
`Are you sure you want to continue?`

This message applies when you specify the –SHUT_PRIMARY or the –SHUT_SECONDARY option of the MIRROR_OFF command. Answer YES if that is what you intend. If you answer NO, nothing is done and you are returned to PRIMOS command level. If you want to turn the mirror off and want the two partitions to have identical shutdown stamps, use the –SHUT_BOTH option.

Note While PRIMOS is awaiting your response, users cannot log in and the system may appear to be hung. When you use the MIRROR_ON command, you should promptly answer the above prompt.

`Mirrored disks out of sync.  (mirror_off)`

This is a warning. Even though both partitions of the mirrored pair are being shut down, the partitions are known not to be identical because the mirror is already broken or the catch-up copy is not complete. Their shutdown stamps are not updated. Use the MIRROR_ON command to start a catch-up copy and make the two partitions identical.

`Mirroring is not enabled.  (mirror_on)`
`Mirroring is not enabled.  (mirror_off)`

The MIRROR_ON command is valid only if one or more of the MIRROR, COMDVM, or PAGINM directives was in the configuration file at cold start. If mirroring is not enabled, the MIRROR_OFF command is ineffective.

`Must specify a Shut option.  (mirror_off)`

One of the shut options (–SHUT_BOTH, –SHUT_PRIMARY, or –SHUT_SECONDARY) must be specified with the MIRROR_OFF command. Reenter the command with the appropriate option.

`Must specify two devices.  (mirror_off)`
`Must specify two devices.  (mirror_on)`

You must specify two pdevs in the MIRROR commands to start or stop the mirroring process. Reenter the command with the correct pdevs in the proper order.

`Must use Mirror_OFF to shut down mirrored disk` *`pdev`*`.`
`(shutdn)`

You used the SHUTDN *pdev* command in an attempt to shut down a partition that is part of a mirrored pair of partitions. You must use the MIRROR_OFF command with the –SHUT_BOTH option. However, if you use the SHUTDN ALL command, both partitions of the mirrored pair are shut down but you are not warned if the partitions are not identical.

`Only one Shut option is allowed.  (mirror_off)`

You must specify only one of the shut options (–SHUT_BOTH, –SHUT_PRIMARY, or –SHUT_SECONDARY). You may not specify more than one.

`PAGINM args must correspond 1-1 to PAGING args.`
`PAGINM directive ignored.`

This message is displayed at cold start. The PAGINM and the PAGING directives in the configuration file must both have the same number of *pdev* arguments. If you want the paging devices to be mirrored at cold start, change the configuration file so that it is correct.

`PAGINM must follow PAGING directive.`
`PAGINM directive ignored.`

This message is displayed at cold start. The PAGINM directive must follow the PAGING directive in the configuration file. If you want the paging devices to be mirrored at cold start, change the configuration file so that it is correct.

`PDEV` *`pdev`* `conflicts with assigned or paging or mirrored device. (addisk)`
`PDEV` *`pdev`* `conflicts with existing LDEV ldev. (addisk)`

Using the ADDISK command, you have attempted to start up a partition that may already be part of a mirrored pair of partitions and thus may already be started.

`pdev` `is not a valid PDEV. (mirror_on)`
`pdev` `is not a valid PDEV. (mirror_off)`

The *pdev* you specified in the MIRROR_ON command or the MIRROR_OFF command is not a valid pdev. Check the pdevs of the partitions that you used.

`PDEVs are identical. (mirror_on)`

The pdevs you specified in the MIRROR_ON command are identical and indicate the same partition. Although the basic pdevs of a mirrored pair must be the same, the pdevs used in the command line must be adjusted for disk drive unit number and disk controller address and thus will be different. Check the pdevs you used.

`PDEVs are not on same position on disk. (mirror_on)`

Both partitions of the mirrored pair must have the same number of surfaces and the same starting surface number such that the pdevs are similar and differ only by the disk drive unit number and possibly by the controller number.

`Primary` *`pdev1`* `and secondary` *`pdev2`* `are not on same disk model. (mirror_on)`

Both partitions of the mirrored pair must be on physical disks of the same model, or type.

`Primary device` *`pdev1`* `and secondary device` *`pdev2`* `are not a mirrored pair. (mirror_off)`

The partitions you specified by the pdevs in the MIRROR_OFF command are not presently mirrored. Check the pdevs you used and reenter the command with the correct pdevs.

`Primary device` *pdev1* `cannot be added. (mirror_on)`

You may be trying to add too many partitions to the system; there can be only 23710 (3558) partitions added to a system. Use STATUS DISKS to see how many are on your system. If there is some other reason for this message, a previous message should indicate the reason.

`Primary device` *pdev1* `conflicts with` *diskname* `disk` *pdev*`.`
`(mirror_on)`

When you attempt to start a mirrored pair, the MIRROR_ON command detects that the primary *pdev1* overlaps with disk *diskname pdev*, or the disk is assigned, already mirrored, or is a crash dump disk.

`Primary device` *pdev1* `conflicts with assigned device.`
`(mirror_on).`

The primary partition that you are attempting to mirror is an assigned partition and therefore cannot be started with the MIRROR_ON command. Unassign the partition and then start the mirror.

`Primary device` *pdev1* `disk Rev. or mode is not mirrorable.`
`(mirror_on)`

pdev1 cannot be mirrored either because it is not a Rev. 21.0 or later partition or it is not in Dynamic Badspot Handling (–IC) mode and dynamic badspot handling cannot thus be enabled.

`Primary device` *pdev1* `is not a file system or paging device.`
`(mirror_off)`

The primary partition, *pdev1*, is neither an added, or started, partition nor is it a paging partition that is being mirrored. Use the STATUS DISKS command at the supervisor terminal to check that the pdev you used is on the system as either an added partition or a paging partition.

`Primary device` *pdev1* `is on a controller that does not support mirroring.`
`(mirror_on)`
`Secondary device` *pdev2* `is on a controller that does not support mirroring.`
`(mirror_on)`

In order to be mirrored, a partition must be Rev. 21.0, or subsequent, on a disk drive associated with an intelligent disk controller, and have dynamic badspot handling enabled. Check that the controller for the partition you have specified as *pdev1* or *pdev2* is an intelligent disk controller (IDC) and that you specified a correct pdev that itself specified the correct controller address.

`Secondary device` *pdev2* `conflicts with` *diskname* `disk` *pdev*`.`
`(mirror_on)`

When you attempt to start a mirrored pair, the MIRROR_ON command detects that the secondary *pdev2* overlaps with disk *diskname pdev*, or the disk is assigned, already mirrored, or is a crash dump disk.

`Secondary device` *`pdev2`* `disk Rev. or mode not mirrorable. (mirror_on)`

pdev2 cannot be mirrored either because it is not a Rev. 21.0 or later partition or it is not in Dynamic Badspot Handling mode and dynamic badspot handling cannot thus be enabled.

`Secondary device` *`pdev2`* `is in use. (mirror_on)`

The partition you have designated as the secondary partition of a mirrored pair is already either added, assigned, or being used for paging. Specify another partition.

`Secondary device` *`pdev2`* `is not a file system or paging device. (mirror_off)`

The secondary partition, *pdev2*, is not an added, or started, partition and is not a paging partition. Use the STATUS DISKS command at the supervisor terminal to check that the pdev you used is on the system as either an added partition or a paging partition that is mirrored.

`Secondary device` *`pdev2`* `is on a controller that does not support mirroring. (mirror_on)`

In order to be mirrored, a partition must be on a disk drive associated with an intelligent disk controller and have dynamic badspot handling enabled. Check the controller for the partition you have specified as *pdev2*.

`Server for Ldev` *`ldev`* `aborted by raising of` *`v`* `condition at` *`p`*
`Copy Server for Ldev` *`ldev`* `aborted by raising of hardware condition at` *`p`*
`Fault type was` *`nn`*

The copy server aborted due to hardware errors.

`Starting Copy Server for Ldev` *`ldev`*`.`

The catch-up copy for a mirrored pair of partitions that is not identical is starting.

`System Error: Unable to purge locate. (mir_on_cmd$)`

PRIMOS has failed a consistency check. You should shut the system down, take a tape dump, and reboot. Then run FIX_DISK on all disks.

`There are disks/portals which are subordinate to this disk. These disks/portals must be shutdn/removed before the MIRROR_OFF command can be issued. Command aborted.`

You must shut down any disks that are subordinate in the file system name space to the mount point pdev specified in the MIRROR_OFF command before you can turn off the mirror.

`This is not the copy servers user type.`

A process other than the copy server attempted to do the catch-up copy. This is an unlikely error but means that someone has written a routine to access the copy server process.

`Unable to shutdn disk`

You used the MIRROR_OFF command but the command is unable to shut down the disk for the reason indicated by the returned PRIMOS error code.

`Unstartable PDEV` *`pdev`* `is ignored by PAGINM directive.`

This message is displayed at cold start. The partition *pdev* specified with the PAGINM directive in the configuration file cannot be started. Check that the pdev in the configuration file is a valid pdev.

```
WARNING: The primary device may have broken the mirror previously.
A catch-up copy may overwrite valid data on the secondary device!
```

```
A catch-up copy of primary device pdev1 to secondary device pdev2 will
be started.
Are you sure you want to continue?
```

You either used the MIRROR_ON command or the COMDVM configuration directive in the PRIMOS.COMI file in attempting to start the mirroring process. The mirror of primary partition *pdev1* and secondary partition *pdev2* was broken and the primary partition was improperly shut down. The data on the secondary partition is likely more current. Thus, the secondary partition should be the primary partition. If you wish to change the positions of the pdevs in the command or in the configuration directive, answer NO to this prompt.

D

Disk Errors

Appendix D describes the meaning of disk-error messages sent to the supervisor terminal. In the messages, all record address, device, and status numbers are in octal. The format and meaning of these messages depend on the type of disk controller that generated the error condition.

There are two general classes of disk controllers for disks supported by Prime (see Table 3-1 for a list of these disks):

- Nonintelligent disk controllers
- Intelligent disk controllers

The Model 6580 SMD intelligent disk controller operates in either of two modes, Nondynamic Badspot Handling (–DBS OFF) mode or Dynamic Badspot Handling (–DBS ON) mode. The Model 7210 SCSI Disk/Tape is downloaded with ICOP+ software for enhanced error correction and performance. Disk errors on disks connected to the Model 7210 controller are handled transparently by the disk subsystem.

Note The Model 6580 intelligent disk controllers are programmable and are downline loaded at cold start with Intelligent Channel Order Protocol (ICOP) software for enhanced error correction. ICOP software enables the intelligent disk controller to operate either in Dynamic Badspot Handling (–DBS ON or –IC) mode or in Nondynamic Badspot Handling (–DBS OFF or –AC) mode. In either mode, the controller acts as an intelligent disk controller. If the controller is not downline loaded with ICOP software for enhanced error correction, it functions as a nonintelligent disk controller.

Nonintelligent Disk Controllers

The sections that follow describe error detection and correction on Storage Module Disks (SMDs), Cartridge Module Devices (CMDs), and Fixed-Media Disks (FMDs) associated either with nonintelligent disk controllers or with intelligent disk controllers that have not been downline loaded with ICOP software and thus function as nonintelligent disk controllers.

Error Message Format

Under PRIMOS, disk read-errors and disk write-errors on supported disks generate error messages with the following format. The parts of the error message are explained below.

```
*** Message from product LOG DISK, user USER on SYSTEM
    (Severity information, logged at day month year time)
DISK XXXX ERROR, DEVICE NUMBER = pdev (OCT)
(ctrlr CTRLR dd UNIT dd)
CRA = ooooooooooo (OCT) RCRA = ooooooooooo (OCT)
CYLINDER = dd HEAD = dd RECORD = dd (dd SECTORS PER TRACK)
STATUS (ACCUM) = oooooo (OCT) STATUS (LAST) = oooooo (OCT) RETRIES = dd (err)
```

XXXX

READ for a read error and WRITE for a write error.

dd

A decimal number.

oooooo

An octal number.

pdev

The octal physical device number of the disk, or partition, on which the error occurred.

ctrlr

A number designating the disk controller. These numbers are as follows:

4004	Nonintelligent disk controller
4005	Nonintelligent disk controller
2047	2250 CPU disk controller
2382	SCSI disk/tape controller
10019	Intelligent disk controller (IDC1 Model 6580)
7210	SCSI disk/tape controller (Model 7210)

CRA

The desired record number within the partition (one 32-bit word).

RCRA

The actual record number read (one 32-bit word). This number is displayed only if the CRA does not equal RCRA. On read operations, RCRA should match the CRA. On write operations RCRA has no validity.

STATUS

The reason for the error. Status is indicated by the sum of two or more 16-bit halfwords provided in Table D-1. (ACCUM) is the accumulated bit-wise OR of all the status words obtained at each retry and (LAST) is the last status word obtained.

RETRIES

The number of times the read or write operation was attempted unsuccessfully before the error message was displayed. Maximum is 10.

err

The disposition of the error, UNCORRECTED or RECOVERED. If the error is RECOVERED, PRIMOS displays CORRECTED WORD # = *oooooo* where the octal number is the record address.

Table D-1. Nonintelligent Controller Status Words

Status Word	*Meaning*
177777	Bad record identifier
177776	Device not ready
177773	Controller hung or warm start occurred
177772	Seek failure
100000	Always set
040000	DMX overrun
020000	Write-protect violation
010000	Check error, uncorrectable read on controller 2382
004000	Checksum error – controller parity error
002000	Header check failure
001000	Alternate record found
000400	Always reset
000200	Degraded mode
000100	Error encountered on last read but retries performed and data recovered
000040	Dual port only – busy
000020	Always reset
000010	Disk drive seeking
000004	Invalid seek
000002	Select error
000001	Not available, not ready, faulted, unformatted, or remap failed

Error Correction

The nonintelligent disk controller writes a 32-bit fullword correcting code checksum on each record. An error detection and correction scheme using an Error Correction Code (ECC) is implemented for supported disks.

The code, together with the correction logic in the controller, is capable of detecting any of the following:

- Two error bursts whose combined length does not exceed 22 bits
- A single error burst whose length does not exceed 32 bits
- Any odd number of errors

In addition, the ECC algorithm is capable of correcting any single error burst whose length does not exceed 11 bits. Error correction is attempted only after 10 attempts to read a record have failed.

The message displayed in the *err* field when error correction was attempted and failed is

```
UNCORRECTED.
```

The message displayed in the *err* field when error correction succeeds is

```
RECOVERED
```

An example of a disk error message follows.

```
*** Message from Prime product LOG DISK, user GEORGE on ENPUB2
    (Severity information, occurred at 29 NOV 91 14:27:08
DISK WRITE ERROR
DEVICE NUMBER = 003460 (OCT) (4005 CTRLR 0 UNIT 0)
CRA = 00000010356 (OCT) RCRA = 00000000000 (OCT)
CYLINDER = 34 HEAD = 5 RECORD = 5 (9 SECTORS PER TRACK)
STATUS (ACCUM) = 120011 (OCT) STATUS (LAST) = 120011 (OCT) RETRIES = 10
(UNCORRECTED)
```

The error message is interpreted as follows: On physical device 3460, the STATUS (120011) indicates that there is a write-protect violation, the drive was seeking, and the drive is not available; the write-protect switch was on.

Note If DSM is running on your system, DSM controls the display of all disk-error messages at the supervisor terminal through the DSM CONSOLE log selection. (See the *DSM User's Guide*.) The display is similar to the examples shown in this appendix.

Model 6580 Disk Controller

The sections that follow describe error detection and correction on Storage Module Disks (SMDs), Cartridge Module Devices (CMDs), and Fixed-Media Devices (FMDs) associated with the downline loaded Model 6580 disk controller (intelligent disk controller – IDC1) in either Dynamic Badspot Handling (–DBS ON or –IC) or Nondynamic Badspot Handling (–DBS OFF or –AC) mode.

Error Detection

Under PRIMOS, disk read-errors and disk write-errors on fully supported disks associated with the Model 6580 SMD disk controller generate error messages with the following format. The parts of the error message are explained below.

```
*** Message from Prime product LOG DISK, user USER on SYSTEM
    (Severity information, occurred at day month year time)
DISK ERROR IN ICOP MODE:
OPCODE = oooooo (OCT) XXXX DEVICE NUMBER = pdev (OCT)
(ctrlr CONTROLLER dd UNIT dd)
CRA = ooooooooooo (OCT) RCRA = ooooooooooo (OCT)
CYLINDER = dd HEAD = dd RECORD = dd (dd SECTORS PER TRACK)
LSW = oooooo oooooo (OCT) PSW = oooooo oooooo (OCT) ECC RESIDUE = oooooo oooooo
(INTERRUPT TYPE = oooooo (OCT) REQUEST ID = oooooo (OCT)
```

dd

A decimal number.

oooooo

An octal number.

OPCODE

An octal number code for the current operation plus a descriptor (*XXXX*) for the information. See Table D-2 for a description of this code.

pdev

The octal physical device number of the disk, or partition, on which the error occurred.

ctrlr

A number identifying the disk controller. These numbers are as follows:

4004	Nonintelligent disk controller
4005	Nonintelligent disk controller
2047	2250 CPU disk controller

2382	SCSI disk/tape controller
10019	Intelligent disk controller (IDC1 Model 6580)
7210	SCSI disk/tape controller (Model 7210)

CRA

The desired record number within the partition (one 32-bit word).

RCRA

The actual record number read (one 32-bit word). On read operations RCRA should match the CRA. On nonread operations RCRA has no validity.

CYLINDER
HEAD
RECORD
SECTORS PER TRACK

The location on the disk of the bad record. CYLINDER is the track number, HEAD is the surface number, RECORD is the sector number, and SECTORS PER TRACK refers to the geometry of the physical disk, or the number of records on a track. These values are valid only for operations that are specific to a record. On format, only CYLINDER and HEAD are valid. On open and close operations, none of the values are valid.

LSW

Two 16-bit halfwords indicating the logical status of the controller or the reason for an error. The meanings of the first logical status halfword (LSW1) are shown in Table D-3 and the meanings of the second logical status halfword (LSW2) are shown in Table D-4. LSW 1 is a general description of the problem. You should always look at LSW 2 and the physical status halfwords for additional information.

PSW

Two 16-bit halfwords indicating the physical status of the controller or the reason for an error. The error may be indicated by the sum of the halfwords. The meanings of the first physical status halfword (PSW1) are shown in Table D-5 and the meanings of the second physical status halfword (PSW2) are shown in Table D-6.

INTERRUPT TYPE

An octal number indicating operation completed (4) or unrecoverable I/O error detected (5).

REQUEST ID

An octal number representing the address of a queue block and used by a representative from PrimeService for debugging purposes.

The following example shows an ICOP disk-error message.

```
*** Message from product LOG DISK, user WALKER on PRIME1
    (Severity information, occurred at 29 NOV 91 21:31:41
DISK ERROR IN ICOP MODE:
OPCODE = 000014 (OCT) DEVICE NUMBER = 006166(OCT)
(10019 CONTROLLER 2 UNIT 3)
CRA = 00000023401 (OCT) RCRA = 00000000000 (OCT)
CYLINDER = 29 HEAD = 17 RECORD = 3 (14 SECTORS PER TRACK)
LSW = 101400 000004 (OCT) PSW = 000000 020021 (OCT)
(INTERRUPT TYPE = 000005 (OCT) REQUEST ID = 150354 (OCT)
```

The OPCODE indicates that a write was attempted on pdev 6166 associated with intelligent disk controller 2 and disk drive 3. The address and location of the bad record are indicated. LSW1 indicates command completion was unsuccessful and LSW2 indicates four retries of the write were attempted. PSW2 indicates that the disk is write-protected, thus faulted and not ready. The last retry operation was attempted for 14 (168) milliseconds and an unrecoverable I/O error was detected.

Table D-2. ICOP Command Operation Code (OPCODE)

OPCODE (octal)	*Meaning*
000000	(RESPONSE CHECK) Response check for controller functionality
000001	(SEEK) Disk seeking
000002	(SELECT) Selecting drive for next I/O operation
000003	(GET INFO) Get controller information
000005	(OPEN SPINDLE) Open spindle; logically activate a disk for use
000006	(CLOSE SPINDLE) Close spindle; logically deactivate a disk
000007	(INQUIRY) Obtain controller and drive configuration and revision information
000009	(ABORT REQUEST)
000010	(RETURN TO TRACK ZERO)
000011	(READ) Read a record
000012	(WRITE Write a record)
000014	(WRITE VERIFY) Write a record and verify the write
000015	(FORMAT) Format a disk
000017	(DUMP) Dump controller microcode
000018	(SPIN UP)
000019	(SPIN DOWN)
000020	(ADD BADSPOT) Add a badspot
000021	(PAUSE CONTROLLER)
000022	(UNPAUSE CONTROLLER)
000100	(ASYNC WRITE) Asynchronously write a record
000101	(ENTER INTELLIGENT MODE) Entering ICOP, ICOP+, or 7210 DLL mode

Note The leftmost two octal numbers (4 bits) in the above codes have no meaning and can be ignored.

Table D-3. First Logical Status Halfword
IDC1 LSW 1 Model 6580 Intelligent Controller

Logical Status Halfword 1	*Meaning*
140000	Drive error, information in physical halfwords
110000	Badspot handling invoked, information in other status words
101400	Command completion unsuccessful
100400	Command completed successfully with multiple retries
100200	Invalid parameter passed with command block
100100	Protocol error
100040	Undefined OTA function code
100020	Disk open, operation requested must be done on closed disk
100010	Disk closed, has not been added or assigned
100004	Bad operation code or operation code length
000000	Command completed successfully

Table D-4. Second Logical Status Halfword
IDC1 LSW 2 Model 6580 Intelligent Controller

Logical Status Halfword 2	*Meaning*
100000	Record has been remapped
040000	ICOP controller in DBS mode on ADDISK or ASSIGN
020000	Requested a remap of an already remapped record
010000	Disk number nonexistent or out-of-range
004000	Attempt to access RMA badspot
002000	Drive error attempting badspot I/O; record not remapped, DBS mode exited
001000	Data error attempting badspot I/O; record not remapped, DBS mode exited
000400	Error occurred reformatting record; bits 6 and 7 (1000_8 and 2000_8) denote type; DBS mode exited
000200	Error occurred accessing DBS record; bits 6 and 7 (1000_8 and 2000_8) denote type; DBS mode exited
000XXX	Retry count. On read operations, ranges to 104_8 (bits 14–16). 10_8 through 100_8 indicate that the controller is physically adjusting on the track in attempting the read, as follows:

Table D-4. Second Logical Status Halfword
IDC1 LSW 2 Model 6580 Intelligent Controller

Logical Status Halfword 2	*Meaning*	
000XXX	000000	Normal read
	000010	Servo plus
	000020	Servo plus and data strobe early
	000030	Servo plus and data strobe late
	000040	Data strobe early
	000050	Data strobe late
	000060	Servo minus
	000070	Servo minus and data strobe early
	000100	Servo minus and data strobe late
	1_8 through 3_8 indicate number of retries. 4_8 indicates error correction (ECC) used. The controller sequences through the above nine methods and retries the read up to three times for each method.	
	On write operations, 1_8 through 4_8 indicate number of retries.	

Table D-5. First Physical Status Halfword
IDC1 LSW 2 Model 6580 Intelligent Controller

Physical Status Halfword 1	*Meaning*
004000	Parity error on data transfer to or from CPU
000010	Cannot remap when disk is write-protected
000004	Unable to select disk on ADDISK or ASSIGN
000002	Encountered reformatted header (see Note)
000001	Data transfer timeout

Note If the controller encounters reformatted headers, either the controller went out of DBS mode for this drive or the DBS file does not match reformatted headers on this disk. Look at previous errors in the system event log for LSW2 of 000400 and 000200 as the cause of the present error.

To correct these errors, try shutting down all partitions on the drive where the error occurs and then adding them back to the system. If the problem persists, back up all partitions on the drive and then run MAKE on all the partitions with the -FORMAT and -NEW_DISK options. These errors may also indicate that the physical disk is failing.

Table D-6. Second Physical Status Halfword
IDC1 PSW 2 Model 6580 Intelligent Controller

Physical Status Halfword 2	*Meaning*
040000	Data transfer overrun
020000	Disk write-protected (see drive manual)
010000	Read check, see LSW 2
004000	Data parity error
002000	Header not found
001000	Underflow on read or write, controller memory bandwidth insufficient
000400	No servo clock detected (on ADDISK or ASSIGN)
000200	No response from disk, issued seek, seek not complete
000100	Dual port; other port has disk (see drive manual)
000040	Sequencer detected error
000020	Disk faulted (see drive manual)
000010	Disk seeking, not on cylinder (see drive manual)
000004	Seek error (see drive manual)
000002	Disk selection invalid (see drive manual)
000001	Disk not ready (see drive manual)

Error Correction

The Model 6580 Intelligent disk controller performs extensive error correction in addition to the error detection and correction scheme of the nonintelligent controller. Multiple attempts are made to read a record after the normal read process fails. If the attempted reads are successful, that is, the error is a recoverable read-error, the data is remapped after it is recovered if the controller is in –DBS ON mode, and the physical record is marked as a badspot by the controller so that the file system cannot use that record again. If the error is unrecoverable or the controller is in –DBS OFF mode, a message is displayed informing you of the error and the bad record address is included. You should then add that bad record to the badspot file by using FIX_DISK.

In the case of an uncorrectable disk write-error, the disk controller in –DBS ON mode remaps the record to a good location on the disk and adds the bad location to the dynamic badspot file.

In addition to the error messages and status words, the Model 6580 controller has LEDs that indicate status. The LEDs are in two separate banks: the first bank contains a green and a red LED and the second bank contains four yellow LEDs arranged in the order shown below.

green red yellow-4 yellow-3 yellow-2 yellow-1

When lit, the Model 6580 Disk Controller LEDs have the following meanings:

green	ICOP functioning correctly
red	ICOP error, see yellow-2, -3, or -4 setting
yellow-4	Invalid interrupt occurred
yellow-3	Excessive ECC errors occurred
yellow-2	Communication with PRIMOS failed
yellow-1	ICOP run lamp; flashes two times per second when controller is downloaded

Model 6580 (IDC1) Disk Controller Messages

The following messages are associated with the Model 6580 intelligent disk controller. All but one generally indicate serious problems associated with activating the intelligent disk controller. One message indicates successful operation of the controller.

`Could not attach to DOWN_LINE_LOAD* - (disk_init)`

An attempt to attach to DOWN_LINE_LOAD* failed. Check to be sure the top-level directory DOWN_LINE_LOAD* exists. The disk controller will operate as a nonintelligent disk controller.

`Could not OPEN PDEV` *`pdev`* `due to controller or device errors. (assign_disk)`
`Could not OPEN PDEV` *`pdev`* `due to controller or device errors. (addisk)}`

The disk controller could not access disk *pdev*. Among the reasons are that the disk drive is offline or not ready, or a cable is loose or not connected. If those are not the reasons, the downline-load file may not be the correct revision for the controller, or may be in error, or the controller may have malfunctioned.

`Disk download file (`*`filename`*`) not found - (disk_init)`

The downline-load file *filename* was not found in the DOWN_LINE_LOAD* directory. The controller will operate as a nonintelligent disk controller. You should install the downline-load file.

`Disk I/O error (assign_disk)`

An error is encountered during an attempt to read from disk. Check to be sure that the disk drive is online and ready.

`DLL and init ICOP mode complete (`*`controller_address`*`) - (disk_init)`

Controller *controller_address* is successfully downline loaded and initialized as an intelligent controller with Intelligent Channel Order Protocol.

`Failure to enter ICOP mode (controller_address) - (disk_init)`

Disk controller *controller_address* failed to be initialized as an intelligent controller. This message indicates a hardware problem, for example, a download to controller revision mismatch or a controller memory problem. This message is entered in the system event log and the controller functions as a nonintelligent controller.

`I/O Errors while processing a disk DLL file (filename) - (disk_init)`

Errors were encountered during a read of the disk controller downline-load file *filename*. The downline load file may be corrupted. The controller operates as a nonintelligent disk controller.

Model 7210 SCSI Disk/Tape Controller

The sections that follow describe the Model 7210 SCSI disk/tape controller error detection and correction.

Error Detection

Under PRIMOS, disk read-errors and disk write-errors on the Model 7210 SCSI controller downloaded with ICOP+ generate error messages with the following format. The parts of the error message are explained below.

```
*** Message from Prime product LOG_DISK, user USER on SYSTEM
    (Severity information, occurred at day month year time)
DISK ERROR IN ICOP+ MODE,
OPCODE = ooo (COMMAND) DEVICE NUMBER = pdev (OCT)
(ctrlr CONTROLLER dd UNIT dd)
CRA = ooooooooooo (OCT) RCRA = ooooooooooo
LBA = dd
EXTENDED STATUS = (LSW1) (LSW2) (LSW3) (LSW4) (LSW5) (LSW6) (LAST)
```

OPCODE

An octal number indicating the code and a descriptor of the operation code. See Table D-2 for a description.

dd

A decimal number.

pdev

oooooo

An octal number.

The octal physical device number of the disk, or partition, on which the error occurred.

ctrlr

A number identifying the disk controller. These numbers are as follows:

4004	Nonintelligent disk controller
4005	Nonintelligent disk controller
2047	2250 CPU disk controller
2382	SCSI disk/tape controller
10019	Intelligent disk controller (IDC1 Model 6580)
7210	SCSI disk/tape controller (Model 7210)

CRA

The desired record number within the partition (one 32-bit word).

RCRA

The actual record number read (one 32-bit word). On read operations RCRA should match the CRA. On nonread operations RCRA has no validity.

LBA

The Logical Block Address location on the disk of the bad record. The LBA is the absolute block number, in decimal, considering the entire disk and non-user addressable blocks.

LSW

Six 16-bit halfwords indicating the logical status of the controller or the reason for an error. The meanings of each logical status halfword are shown in Tables D-7 through D-11. LSW 6 is always zero.

YYYY

CORRECTED, UNCORRECTED, or RECOVERED disk read or write error status.

Table D-7. First Logical Status Halfword
LSW 1 Model 7210 Disk/Tape Controller

Logical Status Halfword 1	*Meaning*
100000	Unsuccessful (unable to process command)
040000	Corrected error (recoverable error occurred)
010000	Unsolicited message
004000	Log message (log and display all status halfwords)
002000	Disk is in DBS mode
001000	Disk data error (see LSW 3)
000400	Disk drive error (see LSW 4
000200	PRIMOS communication error (see LSW 2)
000040	DBS error (see LSW 5)
000020	DBS mode exited
000010	Controller detected hardware error

Table D-8. Second Logical Status Halfword
LSW 2 Model 7210 Disk/Tape Controller

Logical Status Halfword 2	*Meaning*	
100000	Parity error	
040000	Timeout error	
020000	End of range error	
010000	Invalid OTA function code	
004000	Maximum control blocks exceeded	
002000	Invalid parameter in command block	
001000	Status block error	
000400	Redundant command	
000200	Incomplete transfer during DMA	
000110	Drive not reserved	
000XX0	Miscellaneous status defined below:	
	000010	Spindle open
	000020	Spindle closed
	000030	Bad spindle number
	000040	Bad opcode in command block
	000050	Static memory parity error
	000060	Excessive dynamic memory correctable ECC errors
	000070	Uncorrectable dynamic memory ECC error
	000100	Excessive backplane parity errors

Table D-8. Second Logical Status Halfword
LSW 2 Model 7210 Disk/Tape Controller

Logical Status Halfword 2	*Meaning*	
00000X	DMX type and direction defined below:	
	000002	DMT to PRIMOS
	000003	DMT from PRIMOS
	000004	DMA to PRIMOS
	000005	DMA from PRIMOS
	000006	DMQ to PRIMOS
	000007	DMQ from PRIMOS

Table D-9. Third Logical Status Halfword
LSW 3 Model 7210 Disk/Tape Controller

Logical Status Halfword 3	*Meaning*
100000	Retries attempted
040000	Header check (could not find word)
020000	Synchronous transfer error (transfer did not complete)
010000	Unrecoverable read error
004000	Compare error
002000	Parity error on SCSI bus

Table D-10. Fourth Logical Status Halfword
LSW 4 Model 7210 Disk/Tape Controller

Logical Status Halfword 4	*Meaning*
100000	Disk has been shut down
040000	Invalid SCSI command
020000	Disk is write protected
010000	Undefined
004000	Select error
002000	Disk is not formatted
001000	Invalid logical block address (LBA)
000400	Undefined
000040	Reservation conflict
000020	Disk faulted
000010	Duplicate drive response
000004	Seek error
000002	SCSI bus hung
000001	Disk not ready

Table D-11. Fifth Logical Status Halfword
LSW 5 Model 7210 Disk/Tape Controller

Logical Status Halfword 5	*Meaning*
100000	Remap attempted
040000	No more records available for mapping
020000	Remapping discontinued on the device
010000	Unsuccessful remap
000002	Not in DBS mode

Error Correction

In addition to the Model 7210 SCSI disk/tape controller error messages and status words, the controller itself has LEDs that indicate status. When lit, the LEDs, which consist of green, red, and yellow LED indicators, have the following meanings:

green	Functioning correctly
red	Controller failed
red and green	Self test, power up, or reset
yellow	Degraded mode, call PrimeService

Note The Model 7210 SCSI disk/tape controller has enhanced error correction when downline loaded. Write errors are remapped automatically by the controller.

Model 7210 SCSI Disk/Tape Controller Messages

The following controller messages are associated with the Model 7210 disk/tape controller. These messages appear at the supervisor terminal when a Model 7210 controller is used as a tape controller.

`Directory DOWN_LINE_LOAD* not found or inaccessible. (MTINIT)`

An attempt to attach to DOWN_LINE_LOAD* failed. Check to be sure the top-level directory DOWN_LINE_LOAD* exists. The disk controller will operate as a nonintelligent disk controller.

`DLL file SDTC_`*`xxxx`*`_`*`y`*`.DL not found. (MTINIT)`

xxxx is the model number of the tape drive attached to the Model 7210 controller. *y* is the number of the port on the Model 7210 controller to which the tape drive is attached. Check the DOWN_LINE_LOAD* directory for the existence of the downline-load file.

The following messages will be logged to DSM when a Model 7210 controller is used as a tape controller.

```
Attempt to down-line load tape controller FAILED.  (MTINIT)
Controller address = xxxxxx (OCT) Error Code = File in use.
```

If this message appears without one of the first two messages, then this is an indication that the downline-load file is corrupt or the Model 7210 controller is not working properly. Otherwise, the error code will be a file system error similar to the example above or an octal error code. An octal error code (*xxxxxx*) indicates a problem with the Model 7210 controller or a bad file.

```
Tape controller successfully Down-line loaded.  (MTINIT)
Controller address = xxxxxx (OCT)
```

The tape controller was successfully downline loaded.

```
Tape drive error.  Controller address = xxxxxx (OCT)
Controller status = xxxxxx xxxxxx xxxxxx (OCT)
```

An error was encountered during an attempt to use the tape drive. Check to ensure the tape drive is online and ready. The controller address argument helps identify which controller received the error. The valid octal addresses are 128, 138, and 148. The three controller status words give specific error information that explains reasons for failure while using a specific drive. Refer to Table 7-8 (statv(2) and statv(4) Hardware Status Words) in the *Subroutines Reference IV: Libraries and I/O*.

The following messages can be displayed as a result of an attempted downline load of the Model 7210 disk controller.

```
Failure to enter 7210 DLL mode.  (controller_address) -
(disk_init)
```

The disk controller failed to be initialized as an intelligent controller. This message indicates a hardware problem and is subsequently entered in the system event log and the controller functions as a nonintelligent controller.

```
DLL of 7210 controller complete.  (controller_address) -
(disk_init)
```

The Model 7210 controller is successfully downline loaded and initialized as an intelligent controller with enhanced error correction.

```
Intelligent disk controller successfully Down-line loaded.
Attempt to enter Intelligent mode FAILED.
Controller address = xxxxxx (OCT)
```

The disk controller failed to be initialized as an intelligent controller. Either the downline-load file is corrupted or the controller is failing. A system warm start or a cold start will redownline load the controller. If this message persists, call PrimeService.

```
Intelligent disk controller successfully Down-line loaded.
Attempt to enter Intelligent mode SUCCESSFUL.
Controller address = xxxxxx (OCT)}
```

The disk controller was successfully initialized as an intelligent controller.

```
Intelligent disk controller was reset due to WARMSTART.
Controller address = xxxxxx (OCT)
```

The disk controller was reset due to the initiation of a system warm start. Since the controller was reset, it is no longer running intelligent mode. The next time this controller is accessed by PRIMOS while attempting to obtain data, the controller will be downline loaded. This message should be followed by one of the above messages indicating that it was downline loaded.

```
Intelligent disk controller was reset due to TIMEOUT.
Controller address = xxxxxx (OCT)
```

The disk controller was reset due to a system timeout. Since the controller was reset, it is no longer running intelligent mode. The next time this controller is accessed by PRIMOS while attempting to obtain data, the controller will be downline loaded. This message should be followed by one of the above messages indicating that it was downline loaded.

```
Disk download file (filename) not found - (disk_init)
```

The downline-load file *filename* was not found in the DOWN_LINE_LOAD* directory. The controller will operate as a nonintelligent disk controller. You should install the downline-load file.

Determining Downline-load File Characteristics

To determine the type and revision of your intelligent disk controller downline-load file, resume the DLINFO program. For example:

```
OK, R DOWN_LINE_LOAD*>DLINFO
[DLINFO Rev.  23.3 Copyright (c) Prime Computer, Inc.  1992]
Usage: DLINFO dl_file

ER! R DOWN_LINE_LOAD*>DLINFO DOWN_LINE_LOAD*>IDC1.DL
[DLINFO Rev.  23.3 Copyright (c) Prime Computer, Inc.  1992]
DL File Type = 0
Project Name:
Project Identifier:
 hardware revision: 9
 software revision: 105
 Primos revision : 22.1
Date: THU, OCT 27 1988
Time: 13:12:37
Medium: DMT - with checksum
Max Packet Size = 2048 bytes
Total Program Size = 26062 bytes
Total Number of Packets = 50
OK,
```

MAKE_ROBUST Messages

E

Special messages indicate that MAKE_ROBUST is unable to convert the partition according to the options specified on the command line or according to other input. This appendix lists the messages from MAKE_ROBUST in alphabetical order and presents a brief explanation of each message. If *The* or *This* is the first word in the message, the second word is used for alphabetizing. Variable names in messages are italicized, for example, *pdev*.

There are three groups of related messages among those listed here (Read and write errors, Errors associated with changes in the MFD, and Errors associated with MFD entry types). These three groups consist of a single explanation followed by a listing of the error messages.

For a discussion of MAKE_ROBUST and its command-line options, see Chapter 7. For a summary of the MAKE messages, see Appendix A and for a summary of the FIX_DISK messages, see Appendix B.

Command-line Errors

```
Bad parameter.  Maximum extent size must be between 1 and
32767 (MAKE_ROBUST)
```

You specified a number for the maximum extent size outside of the range of 1 through 32767. Reenter the command line with the proper value.

```
Bad parameter.  Maximum extent size must be larger than the
minimum (MAKE_ROBUST)
```

Either you entered a value for minimum extent size that was larger than that of the maximum extent size or you did not enter a maximum extent size value. Reenter the values at the appropriate prompts.

```
Bad parameter.  Minimum extent size must be between 1 and
32767 (MAKE_ROBUST)
```

You specified a number for the minimum extent size outside of the range of 1 through 32767. Reenter the command line with the proper value.

`Bad parameter.  Reading command line (MAKE_ROBUST)`

MAKE_ROBUST encountered an unexpected character or token when parsing your command line. Reenter the command line.

`Bad parameter.  You must supply a pdev.  (MAKE_ROBUST)`

You must specify the partition that you want to convert by entering the physical device number (pdev) of the partition either with the –DISKS option or without it.

`Bad parameter.  You must supply only one pdev (MAKE_ROBUST)`

You can convert only one partition at a time. Reenter the command line with only one pdev.

Read and Write Errors

In the following group of messages, MAKE_ROBUST encountered an error in attempting to read or write a system directory or file. *error_code* is a standard PRIMOS error message such as `No room`, `Device not assigned.`, `Bad number of words.`, or `Bad parameter.` Check that the disk is running, the partition is assigned to you, and the disk is not write-protected.

`error_code.  Reading CMDNC0.  (MAKE_ROBUST)`

`error_code.  Reading DOS.  (MAKE_ROBUST)`

`error_code.  Reading the CMDNC0 directory (MAKE_ROBUST)`

`error_code.  Reading the disk rat (MAKE_ROBUST)`

`error_code.  Reading the DOS directory (MAKE_ROBUST)`

`error_code.  Reading the MFD (MAKE_ROBUST)`

`error_code.  Rewriting the CMDNC0 directory (MAKE_ROBUST)`

`error_code.  Rewriting the DOS directory (MAKE_ROBUST)`

`error_code.  Rewriting the MFD record.  (MAKE_ROBUST)`

`error_code.  Updating the disk rat.  (MAKE_ROBUST)`

Partition Errors

`The partition must be a rev 22.1 standard partition. (MAKE_ROBUST)`

The partition you want to convert to a robust partition must be created as a Rev. 22.1-format partition with Rev. 22.1 or later MAKE prior to conversion.

`The partition is already robust! (MAKE_ROBUST)`

This partition was converted previously. MAKE_ROBUST exits.

Errors Associated With Changes in the MFD

In the following group of messages, the partition you want to be converted to a robust partition must be created with Rev. 22.1 or later MAKE as a Rev. 22.1-format partition and must not be altered or added to in any way. Use MAKE to recreate the partition and then do the conversion with MAKE_ROBUST. Since you have already created the partition once and it has a badspot file, you can use the –BADLEV 0 option, but be sure that you do not use the –NEWDSK or –FORMAT options.

```
Sorry. A vacant entry was found in the MFD record
Partition can't be converted - It has been used!
(MAKE_ROBUST)
```

```
Sorry. An access category was found. Partition can't be
converted - It has been used! (MAKE_ROBUST)
```

Sorry. An unexpected non-special file was found: *filename*
Partition can't be converted - It has been used!
(MAKE_ROBUST)

```
Sorry. CMDNC0 directory has more than 1 record. Partition
can't be converted - It has been used! (MAKE_ROBUST)
```

```
Sorry. CMDNC0 must be an ACL directory. Partition can't
be converted - It has been used! (MAKE_ROBUST)
```

```
Sorry. CMDNC0 not found. Partition can't be converted -
It has been used! (MAKE_ROBUST)
```

```
Sorry. DOS directory has more than 1 record. Partition
can't be converted - It has been used! (MAKE_ROBUST)
```

```
Sorry. DOS must be an ACL directory. Partition can't be
converted - It has been used! (MAKE_ROBUST)
```

```
Sorry.  DOS not found.  Partition can't be converted - It
has been used! (MAKE_ROBUST)

Sorry.  MFD has more than 1 record.  Partition can't be
converted - It has been used! (MAKE_ROBUST)

Sorry.  Non-special BADSPT found.  Partition can't be
converted - It has been used! (MAKE_ROBUST)

Sorry.  Non-special BOOT found.  Partition can't be
converted - It has been used! (MAKE_ROBUST)

Sorry.  Non-special DYNBSP found.  Partition can't be
converted - It has been used! (MAKE_ROBUST)

Sorry.  The MFD acl has been modified.  Partition can't be
converted - It has been used! (MAKE_ROBUST)
```

Errors Associated With MFD Entry Types

For the following group of messages, in checking the directory headers in the MFD, MAKE_ROBUST found an entry that should not be there or that is not the proper type or did not find an entry that should exist. Use MAKE to recreate the partition and then do the conversion with MAKE_ROBUST. Since you have already created the partition once and if it has a badspot file, you can use the –BADLEV 0 option, but be sure that you do not use the –NEWDSK or –FORMAT options.

```
ERROR: Another directory header found in MFD record.  I
can't make sense of this partition! (MAKE_ROBUST)

ERROR: BOOT file not found.  I can't make sense of this
partition! (MAKE_ROBUST)

ERROR: Disk rat not found.  I can't make sense of this
partition! (MAKE_ROBUST)

ERROR: First entry in MFD record is not a directory header.
I can't make sense of this partition! (MAKE_ROBUST)

ERROR: MFD acl not found.  I can't make sense of this
partition! (MAKE_ROBUST)

ERROR: MFD directory index block not found.  I can't make
sense of this partition! (MAKE_ROBUST)
```

ERROR: MFD entry BRA does not point to MFD record. I can't make sense of this partition! (MAKE_ROBUST)

ERROR: MFD entry not found. I can't make sense of this partition! (MAKE_ROBUST)

ERROR: Two BADSPT files found. I can't make sense of this partition! (MAKE_ROBUST)

ERROR: Two BOOT files found. I can't make sense of this partition! (MAKE_ROBUST)

ERROR: Two CMDNC0 files found. I can't make sense of this partition! (MAKE_ROBUST)

ERROR: Two directory index blocks were found in the MFD record I can't make sense of this partition! (MAKE_ROBUST)

ERROR: Two disk rat files found. I can't make sense of this partition! (MAKE_ROBUST)

ERROR: Two DOS files found. I can't make sense of this partition! (MAKE_ROBUST)

ERROR: Two DYNBSP files found. I can't make sense of this partition! (MAKE_ROBUST)

ERROR: Two MFD acl entries found. I can't make sense of this partition! (MAKE_ROBUST)

ERROR: Two MFD files found. I can't make sense of this partition! (MAKE_ROBUST)

ERROR: Unexpected end of MFD record found. I can't make sense of this partition! (MAKE_ROBUST)

ERROR: Unknown entry found in MFD record. I can't make sense of this partition! (MAKE_ROBUST)

Summary of Command Syntax for MAKE and FIX_DISK

F

This appendix summarizes the MAKE and FIX_DISK command-line options and is provided for your convenience. For more detailed descriptions, see Chapters 5 and 6. See Chapter 4 for a description of what to do before and after running MAKE and FIX_DISK.

Summary of MAKE Command Syntax

When you boot MAKE from disk or from magnetic tape or when you use MAKE at the PRIMOS command level, the following are the available options you use with MAKE. When you use MAKE at PRIMOS command level, enter the options on this command line:

MAKE [*options*]

For a full explanation of the use of these options and for cautions on their use, see the full text in Chapter 5.

MAKE Option

–BADSPOT_LEVEL *level*
–BADLEV, –LEV

Specifies the *level* of badspot checking that MAKE should do. *level* is an integer ranging from 0 through 4. See the –DISK_TYPE option for the default levels. If you use this option but do not include *level* or include an invalid *level* on the command line, MAKE prompts for a valid level.

–BAUD_RATE *baud*
–BAUD, –BAU

Specifies the initial supervisor terminal baud rate if it is to be different from the default baud rate of 300 bps. If you use this option but do not include *baud* or include an invalid *baud* on the command line, MAKE prompts for a valid baud rate.

MAKE Option

–COPY_BADSPOTS [*pdev* or *diskname*]
–CPYBAD, –CPY

Specifies the *pdev* of the assigned partition or the *diskname* of the added partition containing the badspot file, BADSPT, that MAKE is to copy. If you do not include *pdev* or *diskname* with this option or include invalid arguments, MAKE prompts for the information. You can specify up to four devices with this option.

–DBS OFF

Specifies that the partition should be created to be compatible with a nonintelligent disk controller (4005) and an intelligent controller (IDC1) in nonintelligent mode. The partition is not capable of Dynamic Badspot Handling or mirroring and cannot be used as a crash dump disk.

MAKE ignores this option on SCSI disks associated with the Model 7210 disk controller downloaded with ICOP+ software.

–DBS ON

Specifies that a partition should be created to be compatible with intelligent disk controllers capable of Dynamic Badspot Handling and mirroring. Create the head zero partition of the disk first if you are creating Rev. 21.0 or later partitions that are capable of having Dynamic Badspot Handling occurring on them. You cannot use the –DBS ON when running MAKE standalone.

–DISK *pdev*
–DSK

Specifies *pdev* of the partition you are creating. If you do not include –DISK or *pdev*, MAKE prompts for the necessary information.

–DISK_REVISION [*rev*]
–DSKREV, –REV

Specifies the revision of the partition that you want MAKE to create. *rev* must be 18, 19, 20, 21, 22, or 22.1. If you do not include *rev* or if you include an invalid *rev*, MAKE prompts for a valid rev. If you do not include this option, MAKE creates a Rev. 22.1 format partition.

–DISK_TYPE [*type*]
–DTP, –DT

Specifies the particular type of disk to be processed by MAKE. This is a required option, and if you do not specify –DISK_TYPE or if you do not include *type*, MAKE prompts for *type* and lists the valid types if you request a list. Use the correct *type*. Designations and definitions of available type are as follows from the MAKE display:

-DISK_TYPE	Description	Default -BADLEV	No. of heads	Records per head	
SMD	80MB or 300MB removable	1	5 or 19	7407	
CMD	Cartridge module device	1	21	7407	
68MB	68 megabyte fixed media	4	3	10071	
158MB	158 megabyte fixed media	4	7	10071	
160MB	160 megabyte fixed media	4	10	7389	
600MB	600 megabyte fixed media	4	40	7569	
MODEL_4475	300 megabyte fixed media	4	19	7407	
MODEL_4714	84 megabyte fixed media	2	5	8120	
MODEL_4711	60 megabyte fixed media	2	4	7140	
MODEL_4715	120 megabyte fixed media	2	8	7140	
MODEL_4735	496 megabyte fixed media	4	24	9954	
MODEL_4719	258 megabyte fixed media	2	17	7320	
MODEL_4845	770 megabyte fixed media	4	23	16112	
MODEL_4721	328 megabyte fixed media				
	- on model 2382 controller	2	12	13128	
	- on model 7210 controller	0	31	5080	
MODEL_4860	817 megabyte fixed media	4	15	26201	
MODEL_4729	673 MB SCSI fixed media	0	31	10414	[4]
MODEL_4730	213 MB SCSI fixed media	0	31	3302	[4]
MODEL_4731	421 MB SCSI fixed media	0	31	6350	[22]
MODEL_4732	1.34 GB SCSI fixed media	0	31	20574	[21]

Where "[n]" is shown, add 254 to Records/head for the first <n> heads.

MAKE Option

–FORMAT
–FMT

Specifies that every track is to be formatted. It is needed when

- The disk has never been used on a Prime system before.
- The DBS file on a head zero partition needs to be rebuilt or removed.
- You are making any non head zero partition on the same spindle after rebuilding or removing the DBS file.

–FORMAT_OK
–FOK, –FMTOK

Specifies that MAKE should format the partition only if necessary. –FORMAT_OK is useful when the DBS file on a head zero partition needs to be rebuilt or removed. Use when running MAKE as a phantom.

MAKE Option

–HELP
–HEL

Specifies that you want a help display similar to this listing of MAKE options and arguments.

–INIT
–INI

Specifies that MAKE should initialize the file system portion of the partition by writing a record address into every record header. Recommended that you do not use this option to save processing time. –NO_INIT is the default.

–LIST_BADSPOTS
–LSTBAD, –LST, –LBS

List all the badspots on the spindle. MAKE lists the badspots sorted by partition on the spindle and marks the badspots that are on the current partition. MAKE list the badspots by record number and by Track, Head, and Sector values.

–MAP_UNCORR
–UNCORR, –UNC

Specifies that MAKE should not consider records with correctable errors as badspots. Prime recommends that you do not use this option.

–MAX_EXTENT_SIZE [*size*]
–MAXSIZ, –MAX

Specifies the maximum extent *size* for CAM files at Rev. 22.0 and later. If you do not specify the maximum extent size, MAKE defaults to 32 records for standard partitions.

–MIN_EXTENT_SIZE [*size*]
–MINSIZ, –MIN

Specifies the minimum extent size for CAM files at Rev. 22.0 and later. If you do not specify the minimum extent size, MAKE defaults to 16 records for standard partitions.

–NEW_DISK
–NEWDSK, –NEW

Specifies that this is the first time the partition is being created or that the partition is corrupt and that MAKE should not expect to find a valid badspot file but should create a new badspot file if necessary. If MAKE finds a valid badspot file, MAKE asks permission to read it.

MAKE Option

–NEW_DISK (continued)

Do not use –NEW_DISK on a disk type that supports Dynamic Badspot Handling unless you are certain there is no data of importance on any partition on the spindle.

–NO_FLAW_MAP
–NOFLMP, –NFL

Specifies that MAKE should not process a flaw map but should use the default level of badspot checking for the disk type involved. Using the flaw map is the default.

–NO_INIT
–NIN

Specifies that MAKE is not to initialize the file system portion of the partition. This is the default.

–NO_QUERY
–NQY, –NQ

Use when running MAKE as a phantom. Provided command line options leave no doubt about how to proceed, –NO_QUERY tells MAKE not to pause for the confirmation normally requested in the following situations:

- MAKE has reason to question the specified disk type.
- When making a head zero partition, MAKE requires or recommends that –FORMAT be enabled in order to rebuild or remove the DBS file.
- When making a non-head zero partition, the head zero partition is not assigned or MAKE recommends that –FORMAT be enabled.

WARNING When making a head zero partition, the combination of –NO_QUERY with –FORMAT or –FORMAT_OK allows MAKE to proceed with actions that risk loss of data on other partitions on the spindle. To determine if such a risk exists, you should first run MAKE without –FORMAT or –FORMAT_OK, and observe any warnings that MAKE displays.

–PARTITION [*name*]
–PART, –PAR

Specifies a valid *name* of up to six-characters for the partition you are creating. If you do not include this option on the command line, MAKE prompts for a name.

MAKE Option

–QUERY_BADSPOTS
–QRYBAD, –QRY, –QBADS

Specifies that you want to enter known badspots manually, either from a list or from a hard-copy flaw map provided by the manufacturer.

–REPORT
–RPT

Displays the progress of formatting, badspot checking, and disk initialization at 5 percent intervals. –REPORT also causes MAKE to report the number of badspots added from the flaw map, BADSPT, and DBS files.

–SECTOR FORWARD or **REVERSE**
–SEC FOR or **REV**

Specifies the method of file record allocation to be either forward with an interleave factor of 3 or reverse with an interleave factor of 1. If you do not specify –SECTOR, MAKE sets the method based on the CPU and disk controller in your system. If you run MAKE standalone, MAKE prompts for the method of file record allocation.

MAKE ignores this option on SCSI disks associated with the Model 7210 disk controller downloaded with ICOP+.

–SPLIT [*recs* or **MAXIMUM**]
–SPL [**MAX**]

Specifies that the disk is to be a split partition to be used for paging or a crash dump disk, with *recs* records used as paging or crash dump records and the rest of the records used for file system data. If you omit *recs*, MAKE displays the maximum number of reserved records and prompts for the number of paging or crash dump records. You may also specify the argument as MAXIMUM or MAX to tell MAKE to minimize the file system and make the reserved space the maximum size possible.

–USAGE
–USA

Provides a list of all the MAKE options.

Summary of FIX_DISK Command Syntax

When you use FIX_DISK at PRIMOS command level or when you resume FIX_DISK.SAVE from magnetic tape, the following are the available options you use. When you use FIX_DISK at PRIMOS command level, enter the options on this command line:

FIX_DISK –DISK *pdev* [*options*]

For a full explanation of the use of these options and for cautions on their use, see the full text in Chapter 6. To cause FIX_DISK to effect any file system changes, you must use the –FIX option; otherwise FIX_DISK only reports the status of the file system.

FIX_DISK Option

–DISK *pdev*

Specifies the *pdev* of the partition you are repairing. –DISK must be used and must be the first option on the command line. (Not needed with –HELP.) *pdev* must be included with –DISK.

–ADD_BADSPOT *rec_num-1* [. . . *rec_num-16*]
–ADBADS

Specifies as many as 16 record numbers to be added to the badspot file. FIX_DISK attempts to map the bad records to new locations. Record numbers (rec_num-1 ... rec_num-16) are specified in octal, offset from the beginning of the partition.

–AUTO_TRUNCATION
–AT

Truncates or deletes directories nested too deeply in a directory tree. If you omit this option, FIX_DISK aborts if the maximum depth is reached. Maximum depth is set with the –MAX_NESTED_LEVEL option (default is 99). You must use the –FIX option with –AT.

–CHECK

Checks to see whether a partition was shut down properly. If the partition was not shut down properly, FIX_DISK should be run with the –FIX option, and you may wish to include the –FAST option if the partition is robust.

–COMMAND_DEVICE [*pdev*]
–COMDEV

Indicates that FIX_DISK is to operate on the command device; this option shuts the command device down, assigns it, repairs it (if you use the –FIX option), unassigns it, and starts it up again.

FIX_DISK Option

–COMDEV (continued)

There are two reasons for using the optional *pdev* argument with the –COMDEV option:

1. If all three of the following are true:
   - You are converting a partition to Dynamic Badspot Handling (–DBS ON) mode or Nondynamic Badspot Handling (–DBS OFF) mode or you are converting a partition from a pre-Rev. 21.0 format to Rev. 21.0.
   - The partition being created is not the head zero partition.
   - The head zero partition of this spindle is the command device.
2. The first partition is the command device and you want to display the DBS file

When you use this option, FIX_DISK must be invoked from the supervisor terminal. Warn all users, then log out all users, and shut down all servers. You must restart servers when FIX_DISK finishes, particularly the Login server and DSM or cold start the system.

–CONVERT_19

Converts a partition to a Rev. 19.0 format partition from an earlier revision. Converts the BADSPT file to Rev. 19.0 format; initializes quota information; disables the display of warning/error messages related to quotas; and creates a new revision stamp. You must use the –FIX option with –CONVERT_19.

–CONVERT_20

Converts a partition to Rev. 20.0 format from an earlier revision. Directories created *after* the conversion are hashed. Existing directories are not hashed; they retain their current formats. In converting from pre-Rev. 20.0, –CONVERT_20 converts the BADSPT file to Rev. 20.0 format; initializes quota information; disables the display of warning/error messages related to quotas; and creates a new revision stamp. The MFD is not hashed. You must use –FIX with –CONVERT_20.

–CONVERT_21

Converts a partition to a Rev. 21.0 format partition from an earlier revision. If a badspot file exists, it is read and the data is written to the DBS file. The DBS file and the RMA are created and initialized on the first partition of the physical disk. The revision stamp is updated. You must use the –FIX and the –DISK_TYPE options with –CONVERT_21.

FIX_DISK Option

–CONVERT_22.1

Converts a Rev. 22.0 partition to a Rev. 22.1 standard partition having unlimited CAM file extents by updating the partition's DSKRAT. You must use the –FIX option with –CONVERT_22.1.

–DBS OFF

Selects Nondynamic Badspot Handling mode for this Rev. 21.0 or later partition. –DBS is recommended when converting to Rev. 21.0 with the –CONVERT_21 option. All partitions on the spindle must be of the same mode. You must use the –FIX option with –DBS.

–DBS ON

Selects Dynamic Badspot Handling mode for this Rev. 21.0 or later partition and spindle allowing Dynamic Badspot Handling and mirroring to take place. –DBS is recommended when converting to Rev. 21.0 with the –CONVERT_21 option. All partitions on the spindle must be of the same mode. Use –DISK_TYPE and –FIX with –DBS ON.

–DISK_TYPE [*type*]
–DT

Specify the *type* of disk to partition. Required with –CONVERT_21 since the size of the DBS file and the RMA depend on disk type.

–DUMP_DBS
–DDBS

Displays the dynamic badspot file (DBS) on those partitions that support Dynamic Badspot Handling. If the partition is in Dynamic Badspot Handling (–DBS ON) mode, the DBS file is up-to-date; if in Nondynamic Badspot Handling (–DBS OFF) mode, the DBS file may not be up-to-date. Use with –DISK *pdev* to specify the head zero partition where the DBS file is located or with –COMDEV *pdev* if the head zero partition is the command device.

–DUFE

Eliminates all inconsistent file entries or entries of unknown type. –DUFE is the default. If you use –DUFE or do not specify –SUFE, unknown file entries are eliminated and the DSKRAT is altered to indicate which records are actually in use. You may also wish to use –CMPR to compress directories containing unknown file entries. Use –SUFE to avoid the accidental deletion of valid file entries caused by running the wrong version of FIX_DISK. You cannot use both –DUFE and –SUFE on the same command line.

FIX_DISK Option

–FAST

Rapidly checks the condition of a robust partition or rapidly repairs a robust partition. You must include the –FIX option to repair the partition. You can also use this option with a standard partition where the only file system inconsistency is a damaged quota system. If there are other problems on the standard partition, FIX_DISK defaults to ignoring the –FAST option and full FIX_DISK is run.

–FIX

Corrects quota information, truncates or deletes defective files, generates a corrected DSKRAT if the current one is bad, and maps the badspot records to the BADSPT file and to the DBS file. If you omit the –FIX option, no disk modifications are performed.

Use –FIX whenever repair or conversion operations are to be performed. However, if you suspect that the disk drive itself is faulty, do not use –FIX. Must be used with these options:

–ADD_BADSPOT	–DBS (–IC, –AC)
–CONVERT_19	–MAX_EXTENT_SIZE
–CONVERT_20	–MIN_EXTENT_SIZE
–CONVERT_21	–SECTOR (–ODI, –RDI)
–CONVERT_22.1	–INTERACTIVE
–UFD_COMPRESSION	

It is important to run FIX_DISK once without using the –FIX option. FIX_DISK then reports inconsistencies, but does not attempt to repair them.

–HELP

Displays a list of FIX_DISK options and brief explanations of their operation. Can be used alone and from any terminal.

–INTERACTIVE
–INT

Asks questions leading to construction of a consistent DSKRAT if the current DSKRAT is defective or missing. If you omit –INT and the current DSKRAT is bad or missing, FIX_DISK aborts. You must use the –FIX option with –INTERACTIVE

Use this option if FIX_DISK has previously aborted and generated an error message.

–LEVEL [*n*]

Sets *n* (decimal) as the lowest level in the tree structure to be displayed. When this option is omitted, the default value is level 1, the first level in the MFD.

FIX_DISK Option

–LIST_BADSPOTS

Displays octal record numbers of all badspots and, for equivalence blocks, displays record numbers of the badspot and the remapped record. In addition to the record number, the track, head, and sector numbers are also listed for badspots and equivalence blocks.

–LIST_FILE
–LF

Displays filenames in all directories down to the directory level displayed according to the –LEVEL *n* in effect at the time.

–MAX_EXTENT_SIZE ***size***
–MAXSIZ

Specifies that you wish to set the maximum extent size for CAM files at Rev. 22.0 and later. You must use the –FIX option with –MAXSIZ.

–MAX_NESTED_LEVEL [*n*]
–MAX

Sets *n* (decimal) as the maximum depth to which directories can be nested. The default maximum depth is 99 levels.

–MIN_EXTENT_SIZE ***size***
–MINSIZ

Specifies that you wish to set the minimum extent size for CAM files at Rev. 22.0 and later. You must use the –FIX option with –MINSIZ.

–NO_QUOTA
–NQ

Assumes the partition is not a quota partition; disables quota checking. You must use this option on all pre-Rev. 19.0 disks because these disks cannot be quota disks.

–NUMBER_OF_RETRIES ***number***
–NUMRTY

Specifies the number of times FIX_DISK is to try to read a failing record. Each retry includes nine different ways of reading a record. The default is two retries.

–SECTOR FORWARD
–SEC FOR

Sets the file record allocation direction to forward and the interleave factor to 3. Valid only with Rev. 20.0 and later standard partitions.

FIX_DISK Option

–SECTOR REVERSE
–SEC REV

Sets the file record allocation direction to reverse and the interleave factor to 1. Valid only with Rev. 20.0 and later standard partitions.

–SUFE

Saves all inconsistent file entries or entries of unknown type. If you omit –SUFE, the default is –DUFE and all unknown file entries are eliminated, directories containing unknown file entries are compressed, and the DSKRAT is altered only to indicate which records are actually in use. Use this option to avoid the accidental deletion of valid file entries caused by running the wrong version of FIX_DISK. You cannot use both –DUFE and –SUFE on the same command line.

–TRUNCATE
–TRU

Truncates files when an uncorrectable error or an uninitialized record in a CAM file on a robust partition is found. When a file is truncated, the part of the file that is located at or beyond the file pointer is eliminated from the file.

If the file pointer is at the beginning of the file, all the information in the file is removed, but the filename remains in the file directory. Normally when FIX_DISK encounters an uncorrectable error or an uninitialized record, it creates a null record on a good portion of the partition and appends the remaining records of the original file to the null record.

–UFD_COMPRESSION
–CMPR

Compresses directories by eliminating all entries for file system objects flagged as being deleted. Use of this option results in a decrease in the search time

G

Glossary

The following terms are used in this book and are defined here for your convenience.

access control list (ACL)

A list of users and their access rights to file system objects as produced by the LIST_ACCESS command.

added partition

A PRIMOS file system partition that is added to the system, or started, by the ADDISK command for user input.

assignable disks

Disks listed in the Assignable Disks Table by pdev and which may be assigned by a single user.

Assignable Disks Table

A table kept by PRIMOS that lists the pdevs of disks that may be assigned by a user. You add and remove entries in this table with the DISKS and DISKS NOT commands. Display the contents of the table with the STATUS DEVICE command.

assigned partition

A partition that has been assigned to one user for that user's exclusive use and is unavailable as a file system partition.

badspot

A physical defect in the disk medium that prevents data from being correctly read from or written to the disk. It is identified by either a record address within the partition or by a combination of head, cylinder, and sector number.

BADSPT file

The Nondynamic Badspot Handling (–DBS OFF or –AC) mode file listing badspots on a partition. There is one MFD>BADSPT file per partition if there are badspots on the partition. This file contains the physical address of each badspot on the partition.

BRA

Beginning record address; the address of the first record in a file system object.

breaking the mirror

Disabling of the mirroring process between two partitions by shutting down one of the mirrored pair.

buffer

To temporarily store records as an intelligent disk controller does when reading records.

cache

See buffer.

CAM

See Contiguous Access Method.

catch-up copy

The copy initiated when PRIMOS determines that two partitions that are to be mirrored are not identical either because their date and time of shutdown (DTS) stamps are not the same or because the two partitions were not started at the same time.

CMD

Cartridge module device; a type of physical disk for file system storage that has a removable portion and a fixed portion.

COMDEV

The PRIMOS system command device; the logical disk where PRIMOS and related files exist; logical device 0 (ldev 0).

Contiguous Access Method (CAM)

A method of allocating and storing records in a file. The file records are stored contiguously in extents. PRIMOS accesses the records by reading an extent map.

correctable error

A disk data-error that PRIMOS or the disk controller can correct by using an error correction code or by physically adjusting the read head on the surface of the disk.

crash dump disk

A portion of a spindle set aside to receive a crash dump after a system halt. Crash dump disks must be on a spindle that is connected to an intelligent disk controller and must be created as split partitions.

cylinder

The intersection of individual tracks on all the surfaces of a disk; for example, if a geometrical solid was passed through track 123 of each surface of a physical disk, it would form a cylinder. *See also* disk formatting.

DAM

See Direct Access Method.

DBS

The dynamic badspot file. This file contains addresses of all the known badspots for an entire physical disk. It also contains a list of all of the available remapping records. All badspots are matched to a remapping record. Additional remapping records are available for new, or dynamically occurring, badspots. *See also* dynamic badspot.

Direct Access Method (DAM)

A method of allocating and storing records in a file. PRIMOS accesses the records by reading an index to them.

disk

Generally used to refer to a partition but also used as a term for a disk drive and disk pack.

disk drive

The peripheral device that contains the physical disks and the hardware and electronic circuitry to accomplish reading and writing on the physical disk surfaces. The disk drive may be external to the system or it may be internal (as in the case of 2455 systems). Also referred to as a disk storage device and drive unit.

disk formatting

See formatting.

disk geometry

The physical attributes of a physical disk such as the number of cylinders, or tracks, per surface; the numbers of sectors, or records, per track; and the number of surfaces. These attributes are defined by a physical device number for disk partitions.

Disk Information Table

A table of logical device numbers (ldevs) consisting of arrays of words each of which includes the following information for each partition.

- pdev for each ldev
- Sectors per track
- Total number of records
- Number of words per record

disk mirroring

The creation of two logically equivalent partitions that store the same data such that, if either partition fails, the other can be used in its place.

disk pack

The physical disks that are removable from a disk drive as in the case of the 80MB and 300MB SMDs. Also used to refer to physical disks in general. *See also* Winchester disk; disk drive.

DSKRAT

The Disk Record Availability Table. The DSKRAT contains disk geometry information for each partition and bit settings for each record on the disk telling whether the record is in use or available. These bits are either set (=1) indicating that a record is available for file system use or are reset (=0) indicating that a record is in use. *See also* RAT.

dynamic badspot handling

The process whereby a disk controller, upon detecting a badspot, remaps the record containing the badspot to another good record on the partition.

Dynamic Badspot Handling (–DBS ON, -IC) mode

A state of a disk that allows intelligent disk controllers to handle badspots and to allow mirroring on these partitions. This disk mode is not compatible with nonintelligent controllers.

equivalence blocks

Describes where one record is actually stored on the partition. There is one equivalence block for every remapped record on a partition. These equivalence blocks are stored in the BADSPT file of the target partition. They are created by COPY_DISK and PHYRST in order to indicate that badspot handling has taken place for the partition to which data were copied (the target partition). Until the equivalence blocks are deleted by FIX_DISK, the partition must not be used for any purpose.

extent

Groups of contiguous records in CAM files.

extent map

An index of the extents in a CAM file used by PRIMOS to locate and retrieve CAM file records.

fast FIX_DISK

FIX_DISK with the -FAST option. Fast FIX_DISK should be used only on robust partitions. The use of the -FAST option causes FIX_DISK to check only directory entries, including CAM file extent maps, the DSKRAT, and the quota system on robust partitions.

file system disk

A logical disk, or partition, used by PRIMOS to store system and user files.

first partition

See head zero partition.

fixed-media Disk

See FMD.

flaw

A badspot; an area of the physical disk that cannot store data.

flaw map

A list of flaws provided by the disk manufacturer and written on an unused cylinder of the disk. The flaw map is available for MAKE to read. Also sometimes refers to a list of badspots written on paper and affixed to the physical disk by the disk manufacturer. The Operator can then enter these badspots manually by using the appropriate MAKE or FIX_DISK options.

FMD

Fixed-Media Disk; a type of physical disk for file system storage that includes the sealed storage media and the disk drive. Sometimes referred to as a Winchester disk.

formatting

Preparing the disk medium for use by PRIMOS with MAKE. MAKE writes physical record headers onto the partition that are recognizable to PRIMOS. Disk controllers store the location and any auxiliary information in the disk header for each sector on the disk. Cylinder, head (or surface), and sector values are stored with each sector. Cylinders are numbered from the outermost to the innermost. Heads are numbered from the top surface to the bottom surface on the disk. Sectors within a track are currently numbered in a clockwise, or forward, order from sector 0 to the maximum number of sectors per track minus one.

forward sectoring

A method of file record allocation used by PRIMOS in which the next record to be allocated is three sectors forward of the last record; the interleave factor is 3. *See also* interleaving; interleave factor; reverse sectoring.

full FIX_DISK

FIX_DISK without the -FAST option. Full FIX_DISK checks (and repairs if you use the -FIX option) the entire file system.

hashing

The use of an algorithm by PRIMOS to rapidly access data or records within a partition. Directories on robust partitions and non-ACL directories are not hashed; only ACL-protected directories are hashed.

head

The physical device that reads data from the disk surfaces; newer disks may contain more than one read head per disk surface. This term is also used synonymously with *surface* when referring to the number of surfaces in a partition.

head zero partition

The partition of a physical disk that contains the first surface (starting surface 0) of the spindle; thus, the first four bits of its pdev are 0. On a spindle partitioned by MAKE at Rev. 21.0 and later, the head zero partition contains the dynamic badspot file (DBS) and the remapped area (RMA) for all the partitions on that spindle if the spindle supports Dynamic Badspot Handling.

intelligent disk controller

A microprocessor-based disk controller that is capable of buffering data, of using algorithms to perform the read and write operations on a disk, and of dynamically remapping badspots that occur on the disk. A nonintelligent disk controller does not have these capabilities. An intelligent disk controller must be used for disk mirroring because it provides dynamic badspot remapping.

interleave factor

The sector gap between consecutively allocated records. It is 3 for forward sectoring and 1 for reverse sectoring. *See also* interleaving; forward sectoring; reverse sectoring.

interleaving

The order of writing records to disk so as to maximize the potential for the sequentially next record of a file to be under the read head of the disk after processing of the current record is complete. *See also* interleave factor.

ldev

An octal number between 0 and 355_8 (0 through 237, decimal) that is assigned to a partition when the partition is started by the ADDISK or the STARTUP command. It also indicates the location of the pdev of the added PRIMOS file system partition in the Disk Information Table.

logical device number

See ldev.

logical disk

Synonymous with partition or logical device. A logical division of a physical disk used for file storage or for paging.

logical file type

What a subroutine or utility creating a file sets the file type to be as opposed to how the file is physically arranged on the storage medium (the *physical* file type). For example, all user files on a robust partition are physically arranged as CAM files but the software creating the file may set the file type to SAM or DAM; thus, the files are logically created as SAM or DAM files.

logical save

Saving of records as logical entities such as files as opposed to a physical save. The MAGSAV and MAGRST utilities save and restore records logically. *See also* physical save.

Master File Directory (MFD)

The highest level directory on a partition; each partition contains one MFD. The MFD contains a file that is an index to each top-level directory and file in the partition, or MFD. Also refers to the partition itself.

Nondynamic Badspot Handling (–DBS OFF or –AC) mode

The state of a disk that allows all disk controllers (intelligent and nonintelligent controllers) to access the disk. The badspot handling process consists of adding badspots to the file BADSPT automatically by MAKE; adding badspots manually by using the appropriate MAKE or FIX_DISK options; copying badspots from other partitions by using the appropriate MAKE or FIX_DISK options; and adding badspots from vendor flaw maps.

nonintelligent disk controller

A disk controller that is incapable of buffering data and that can execute only one command at a time such as 4005 disk controllers. *See also* intelligent disk controller.

paging

Moving programs or parts of programs that are not currently in use out of main memory to a designated area of disk storage referred to as the paging device. This technique makes it appear as if the system has more memory than it actually does and is the basis for virtual memory.

paging partition

A logical disk where paging records reside; used by PRIMOS for paging as part of the virtual memory space. Paging partitions should always be split partitions. Also referred to as *paging disks*. A Rev. 21.0 or later system can have a maximum of eight paging partitions.

partition

A logical grouping of physical disk surfaces that provides a logical range of disk record addresses. This area is defined by the DSKRAT and by the physical device number (pdev) of the partition. A partition is a self-contained file system that is added to PRIMOS with the ADDISK command and removed with the SHUTDN command.

pdev

A 16-bit octal number that defines to the file system a range of surfaces as a logical partition of a physical disk and that specifies the disk controller address and a disk drive unit number. The location and size of a partition are described by starting surface (surface offset), number of surfaces, drive unit number, and controller address.

physical device number

See pdev.

physical disk

An entire multi-surface disk (SMD, CMD, or FMD) containing 1 through *n* partitions. *See also* spindle.

physical file type

How the file is physically organized on the disk as opposed to what the logical file type is set to by the routine creating the file. When a command such as LD lists file type, the type listed is the *logical* file type.

physical save

Saving of records in the order that they are stored on the disk without consideration for what file they belong to. The utilities PHYSAV and COPY_DISK use a physical save. *See also* logical save.

primary partition

The main partition of a mirrored pair of partitions; the partition from which a catch-up copy is made. *See also* secondary partition.

RAT

The Record Availability Table, which contains a header that describes the partition and a bit map that indicates which records are available for use and which records are in use. Synonymous with DSKRAT.

remapped area (RMA)

An area of the head zero partition on a spindle that is set aside to contain records that would be written into badspots but that are instead written to the RMA by an intelligent disk controller. This area of the disk is normally accessed only by the intelligent controller but is also accessed by FIX_DISK when converting to Nondynamic Badspot Handling (-DBS OFF or –AC) mode, in which case these records are read directly before their pointers are restrung into their parent file. The RMA records are marked as in-use in the DSKRAT and are never directly accessed by PRIMOS. The RMA records are full disk records that contain parts of various files that the file system initially attempted to write to badspots.

reverse sectoring

A method of file record allocation used by PRIMOS in which the next record to be allocated is one sector behind the last record such that logically contiguous records are adjacent to one another. The interleave factor in this case is 1. *See also* interleaving; interleave factor; forward sectoring.

robust partition

A PRIMOS file system partition that contains CAM files only and that is designed to be less subject to disk errors resulting from system halts and that can generally be rapidly repaired by using fast FIX_DISK.

SAM

See Sequential Access Method.

secondary partition

The alternate partition of a mirrored pair of partitions and the partition that the primary partition is copied during the catch-up copy process. *See also* primary partition.

sector

A portion of a track on the surface of a disk. A sector contains one record, or block, of data and, on PRIMOS disks, contains 2048 bytes of user data and 32 bytes of housekeeping data.

segment directory (SEGDIR)

Contains entries referenced by file numbers from 0 through 65535 rather than by file names. File are referred to as subfiles. Generally used by programs rather than by users.

Sequential Access Method (SAM)

A method of allocating and storing records in a file. The file records are accessed sequentially such that to get to a record in a file, all previous records in the file must be read by PRIMOS.

SMD

Storage module disk; a type of physical disk for file system storage that can be removed from the disk drive.

spindle

An entire physical disk consisting of all heads, or surfaces on the disk. A spindle can be partitioned into logical disks, or partitions. Synonymous with physical disk.

split partition

A partition that has part of its storage space reserved for file system use and part reserved for paging or crash dump storage use. *See also* paging partition.

splitting

Dividing a partition into a file system area and an area for paging or for crash dump use.

standalone

Refers to a program that can be booted to run by itself without the services of PRIMOS. An example is MAKE.SAVE.

standard partition

A nonrobust partition. The type of PRIMOS file system partition always created prior to Rev. 22.1. Full FIX_DISK must be used to repair standard partitions.

static badspot

A badspot that is present on the disk surface and that is detected by MAKE when the partition is first created.

Storage Module Disk

See SMD.

surface

The magnetic area of a disk where data is actually stored (written to) and retrieved (read from). Groups of surfaces constitute partitions and all partitions on a physical disk except the last must contain an even number of surfaces.

survivor

Describes the most up-to-date and usable partition of a mirrored pair when the mirror breaks.

top-level directory

The directories at the highest level in the file system tree structure immediately inferior to the Master File Directory (MFD). These directories contain files and other directories referred to as subdirectories. CMDNC0 is a top-level directory on the command device.

user disks

Disks or partitions used for the storage and retrieval of user files. These disks are prepared for use by the MAKE utility.

virtual memory

Disk storage memory that is used by PRIMOS in the paging process with the result that the system appears to have considerably more physical memory than actually exists. Use of virtual memory provides each user with 512MB of virtual address space. *See also* paging.

volume

Term used synonymously with disks and partitions. *Volume* may also refer to an entire physical disk as one logical disk.

Winchester disk

A sealed disk subsystem in which the physical disks and their associated disk drive and circuitry are all contained. A fixed-media disk (FMD).

Index

Index

E

F

N

O

P