AMOS

system commands reference manual

alpha micro

DWM-00100-49

SOFTWARE MANUAL

AMOS

SYSTEM COMMANDS REFERENCE MANUAL

DWM-00100-49
REV. A02

alpha micro

NOTE: This printing of the manual contains the contents of Change Page Packet #1 for the "AMOS System Commands Reference Manual", (DWM-00100-62), and Change Page Packet #2 for the "AMOS System Commands Reference Manual", (DSS-10000-09), which may be ordered separately from Alpha Micro.

First printing: 1 October 1979
Second printing: 1 May 1980
Third printing: 30 April 1981

This document reflects AMOS Versions 4.5 and later

ALPHA MICROSYSTEMS
17881 Sky Park North
Irvine, CA 92714

30 April 1981
DSS-10000-09

CHANGE PAGE PACKET #2 FOR THE AMOS SYSTEM COMMANDS REFERENCE MANUAL

This set of documents is the second update package for the AMOS System Commands Reference Manual, (DWM-00100-49). Once you incorporate these pages into your copy of the manual, your manual will contain information that reflects AMOS Versions 4.5 and later.

NOTE: Look at the title page of your current manual. If it says "Revision A01," the manual already contains the contents of the first change page packet for the AMOS System Commands Reference Manual; if it does not show a revision level, you will want to order the first change page packet (part number DWM-00100-62) so that you can bring your manual completely up to date.

This change page packet contains:

1. Updating instructions.
2. A new title page for the manual (indicating the current revision level of the manual, Revision A02).
3. The manual pages we have changed.

Each page that was changed contains a legend at the bottom of the page that reads: (Changed 30 April 1981). We have marked with change bars (vertical black lines in the left margin) those portions of each page that have changed.

Each page that is new for this revision of the manual contains the legend (30 April 1981).

1.0 UPDATING INSTRUCTIONS

To make the update process easier, we suggest that you put your AMOS System Commands Reference Manual in a three-ring binder notebook. If the pages of the manual are secured with a staple, remove the staple.

First, remove the title page from this change page packet and exchange it for the title page in the AMOS System Commands Reference Manual.

Now insert and replace sheets as described in the next section, "List of Change Pages."

2.0 LIST OF CHANGE PAGES

Below is the list of change pages in this packet. Remove the original pages listed below and replace them with the revised pages. Those pages for which no originals exist are to be inserted into the manual in proper alphabetic order; these are new reference sheets for AMOS Version 4.5.

Replace:

Original Page		Revised Pages
Title page/ii	with	Title page/ii
Page iii	with	Page iii
Page v	with	Page v
Pages 1-1 through 1-5	with	Pages 1-1 through 1-5
Page 2-1/2-2	with	Pages 2-1 through 2-4
Pages 6-1 through 6-5	with	Pages 6-1 through 6-5
APPEND reference sheet	with	APPEND reference sheet
	(new)	BATCH reference sheet
BITMAP reference sheet	with	BITMAP reference sheet
	(new)	CAL120 reference sheet
COM reference sheet	with	COM reference sheet
COMPIL reference sheet	with	COMPIL reference sheet
COPY reference sheet	with	COPY reference sheet
CREATE reference sheet	with	CREATE reference sheet
DATE reference sheet	with	DATE reference sheet
DDT reference sheet	with	DDT reference sheet
DO reference sheet	with	DO reference sheet
DSKANA reference sheet	with	DSKANA reference sheet
DSKCPY reference sheet	with	DSKCPY reference sheet
	(new)	EMAIL reference sheet
ERASE reference sheet	with	ERASE reference sheet
FILDMP reference sheet	with	FILDMP reference sheet
	(new)	FILTAP reference sheet
FIX reference sheet	with	FIX reference sheet
FIXDVR reference sheet	with	FIXDVR reference sheet
FMT200 reference sheet	with	FMT200 reference sheet
FMT210 reference sheet	with	FMT210 reference sheet
FMT400 reference sheet	with	FMT400 reference sheet
FMT500 reference sheet	with	FMT500 reference sheet
HASHER reference sheet	with	HASHER reference sheet
	(new)	LIB reference sheet
LINK reference sheet	with	LINK reference sheet
LOAD reference sheet	with	LOAD reference sheet
MACRO reference sheet	with	MACRO reference sheet
MONTST reference sheet	with	MONTST reference sheet

	(new)	NEWTRM reference sheet
	(new)	OPR reference sheet
	(new)	PARITY reference sheet
*** Delete PASCAL reference sheet ***		
	(new)	PC reference sheet
	(new)	PCL reference sheet
	(new)	PCU reference sheet
	(new)	PL reference sheet
PRINT reference sheet	with	PRINT reference sheet
	(new)	PRUN reference sheet
	(new)	PU reference sheet
SET reference sheet	with	SET reference sheet
SYMBOL reference sheet	with	SYMBOL reference sheet
SYSMEM reference sheet	with	SYSMEM reference sheet
SYSTEM reference sheet	with	SYSTEM reference sheet
	(new)	TAPDIR reference sheet
	(new)	TAPFIL reference sheet
TIME reference sheet	with	TIME reference sheet
TXTFMT reference sheet	with	TXTFMT reference sheet

For your quick reference:

The new reference sheets are:

BATCH	PC
CAL120	PCL
EMAIL	PCU
FILTAP	PL
LIB	PRUN
NEWTRM	PU
OPR	TAPDIR
PARITY	TAPFIL

The revised reference sheets are:

APPEND	ERASE	LOAD
BITMAP	FILDMP	MACRO
COM	FIX	MONTST
COMPIL	FIXDVR	PRINT
COPY	FMT200	SET
CREATE	FMT210	SYMBOL
DATE	FMT400	SYSMEM
DDT	FMT500	SYSTEM
DO	HASHER	TIME
DSKANA	LINK	TXTFMT
DSKCPY		

(For a list of the reference sheets grouped by function, see Section 6.3, "Functional Summary of Commands," of the AMOS System Commands Reference Manual.)

1 May 1980
DWM-00100-62

CHANGE PAGE PACKET #1 FOR THE AMOS SYSTEM COMMAND REFERENCE MANUAL

1.0 INTRODUCTION

This set of documents is the first update package for the AMOS System Commands Reference Manual, (DWM-00100-49). Once you incorporate these pages into your copy of the manual, your manual will contain information that reflects AMOS Versions 4.4 and later.

This change page packet contains:

1. Updating instructions.
2. A new title page for the manual (indicating the current revision level of the manual).
3. The manual pages we have changed.

Each page that was changed contains a legend at the bottom of the page that reads: (Changed 1 May 1980). We have marked with change bars (vertical black lines in the left margin) those portions of each page that have changed.

Each page that is new for this revision of the manual contains the legend (1 May 1980).

2.0 UPDATING INSTRUCTIONS

To make the update process easier, we suggest that you put your AMOS System Commands Reference Manual in a three-ring binder notebook. If the pages of the manual are secured with a staple, remove the staple.

First, remove the title page from this change page packet and exchange it for the title page in the AMOS System Commands Reference Manual. (Notice that we have not included a new table of contents; the changes we made do not affect your current table of contents.)

Now insert and replace sheets as described in the next section, "List of Change Pages."

3.0 LIST OF CHANGE PAGES

Below is the list of change pages in this packet. Remove the original pages listed below and replace them with the revised pages. Those pages for which no originals exist are to be inserted into the manual in proper alphabetical order; these are new reference sheets for AMOS Version 4.4.

Replace:

Original Page		Revised Pages
Title page	with	Title page
Page 1-1/1-2	with	Page 1-1/1-2
Pages 6-1 through 6-5	with	Pages 6-1 through 6-5
ATTACH Reference Sheet	with	ATTACH Reference Sheet
BADBLK Reference Sheet	with	BADBLK Reference Sheet
BITMAP Reference Sheet	with	BITMAP Reference Sheet
	(new)	CDC210 Reference Sheet
	(new)	COM Reference Sheet
	(new)	CONT Reference Sheet
CPY410 Reference Sheet	with	CPY410 Reference Sheet
	(new)	CREATE Reference Sheet
CRT410 Reference Sheet	with	CRT410 Reference Sheet
DEVTBL Reference Sheet	with	DEVTBL Reference Sheet
DO Reference Sheet	with	DO Reference Sheet
DSKANA Reference Sheet	with	DSKANA Reference Sheet
DSKCPY Reference Sheet	with	DSKCPY Reference Sheet
DUMP Reference Sheet	with	DUMP Reference Sheet
	(new)	EXIT Reference Sheet
	(new)	FIX Reference Sheet
FIXDVR Reference Sheet	with	FIXDVR Reference Sheet
	(new)	GLOBAL Reference Sheet
	(new)	GOTO Reference Sheet
	(new)	HASHER Reference Sheet
	(new)	ISMFIX Reference Sheet
	(new)	LABEL Reference Sheet
LOAD Reference Sheet	with	LOAD Reference Sheet
LOG Reference Sheet	with	LOG Reference Sheet
	(new)	LOOKUP Reference Sheet
MACRO Reference Sheet	with	MACRO Reference Sheet
	(new)	MEMERR Reference Sheet
MOUNT Reference Sheet	with	MOUNT Reference Sheet
	(new)	PAUSE Reference Sheet
SRCCOM Reference Sheet	with	SRCCOM Reference Sheet
SYSACT Reference Sheet	with	SYSACT Reference Sheet
SYSMEM Reference Sheet	with	SYSMEM Reference Sheet
SYSTAT Reference Sheet	with	SYSTAT Reference Sheet
	(new)	TRACE Reference Sheet
TRMDEF Reference Sheet	with	TRMDEF Reference Sheet
VUE Reference Sheet	with	VUE Reference Sheet
	(new)	WNG210 Reference Sheet

For your quick reference:

The new reference sheets are:

CDC210	HASHER
COM	ISMFIX
CONT	LABEL
CREATE	LOOKUP
EXIT	MEMERR
FIX	PAUSE
GLOBAL	TRACE
GOTO	WNG210

The revised reference sheets are:

ATTACH	LOAD
BADBLK	LOG
BITMAP	MACRO
CPY410	MOUNT
CRT410	SRCCOM
DEVTBL	SYSACT
DO	SYSMEM
DSKANA	SYSTAT
DSKCPY	TRMDEF
DUMP	VUE
FIXDVR	

(For a list of the reference sheets grouped by function, see Section 6.3, "Functional Summary of Commands," of the AMOS System Commands Reference Manual.)

NOTE: The ISMFIX and MEMERR reference sheets were issued in an informal update package with the AMOS Version 4.3 Release Notes. If your manual contains these two reference sheets, discard them from this change page packet; they have not been revised since that time.

IMPORTANT NOTE

This manual is a reference manual for the experienced user of the AMOS system. Before you use the system for the first time, read the Introduction to AMOS, (DWM-00100-65), the AMOS User's Guide, (DWM-00100-35), and the documents in the AMOS Software Update Documentation Packet.

You may also be interested in reading Introduction to AMOS, (DWM-00100-65), which contains an introduction to computers in general and the AMOS system in particular.

(Changed 30 April 1981)

Table of Contents

CHAPTER 1 INTRODUCTION TO THE MANUAL

1.1 PREFACE 1-1
1.2 CONVENTIONS USED IN THIS MANUAL 1-2
1.3 CONCEPTS 1-3

CHAPTER 2 INTRODUCTION TO AMOS COMMANDS

2.1 COMMANDS TO BE USED WITH CAUTION 2-2
2.1.1 Commands that May Only Be Used From the First Memory Partition 2-2
2.1.2 Commands that Destroy Disk Contents ... 2-2
2.1.3 Commands Only For the Use of the System Operator 2-3
2.1.4 Access Limitations 2-3
2.1.5 Commands for Experienced Users 2-4
2.2 PRIVILEGED COMMANDS 2-4
2.4 WILDCARD FILE COMMANDS 2-4

CHAPTER 3 FILE SPECIFICATIONS

3.1 INTRODUCTION 3-1

CHAPTER 4 FILE SPECIFICATION DEFAULTS AND WILDCARD SYMBOLS

4.1 DEFAULTS 4-1
4.1.1 Standard System Defaults 4-1
4.2 WILDCARD SYMBOLS 4-2
4.2.1 Standard System Wildcard Symbols 4-2

CHAPTER 5 REFERENCE SHEET FORMAT

5.1 INTRODUCTION 5-1
5.2 REFERENCE SHEET SUBHEADINGS 5-1

CHAPTER 6 THE AMOS SYSTEM COMMANDS REFERENCE SHEETS

6.1 INTRODUCTION 6-1
6.2 ALPHABETIC SUMMARY OF AMOS COMMANDS 6-1
6.3 FUNCTIONAL SUMMARY OF COMMANDS 6-2
6.4 THE COMMAND REFERENCE SHEETS 6-5

APPENDIX A THE ASCII CHARACTER SET A-1

(Changed 30 April 1981)

CHAPTER 1

INTRODUCTION TO THE MANUAL

1.1 PREFACE

The purpose of this manual is to help the experienced AMOS user gain quick access to information on every command on the system. This manual does not give you the information you need to operate the system. It does serve as a quick reference manual for those occasions when you need to jog your memory by glancing at the format of a specific command.

In writing these reference sheets, we've assumed that you are already quite familiar with the AMOS commands. If this is not the case, do NOT attempt to use the system without reading the AMOS User's Guide and the AMOS Software Update Documentation Packet.

If you find yourself confused about the use of a particular command, refer to the AMOS User's Guide and the AMOS Software Update Documentation Packet for examples of command use. These documents also contain general system information. For detailed information on the system language- and text-processors, refer to the specific manuals for those processors.

The new AMOS user may want to refer to the manual Introduction to AMOS, which contains general background information on the AMOS system and on computer terms and concepts.

(Changed 30 April 1981)

1.2 CONVENTIONS USED IN THIS MANUAL

To make our examples concise and easy to understand, we've adopted a number of graphics conventions throughout our manuals:

PPN	A Project-programmer number. This number identifies a user account (e.g., [100,27]). We also represent an account number as [p,pn].
Devn:	A device specification. This symbol represents a logical unit of a physical device. Such a specification usually refers to a disk, but can represent any valid system device (e.g., a magnetic tape drive or a printer for which a special driver program is required).
Filespec	A file specification. Such a specification identifies a file. It usually has these elements: Devn:Filename.Extension[p,pn]
default	Information assumed by the system if you omit necessary data. For example, if you omit an account specification from a file specification, most AMOS commands assume that you want to access a file in the account you are logged into. (In this case, your own account is the default.)
{}	Optional elements of a command line. When these symbols appear in a sample command line, they designate elements that you may omit from the command line.
____	Underlined characters indicate those characters that AMOS prints on your terminal display. For example, throughout this document you see an underlined dot, .., which indicates the prompt symbol that the operating system prints on your terminal when you are at AMOS command level.
(RET) or ↵	Carriage return symbol. The RET symbol or curly arrow marks the place in your keyboard entry to type a RETURN (i.e., hit the key labeled RETURN). For example: ".LOGOFF (RET)" tells you "After an AMOS prompt, type LOGOFF and a RETURN."
^	Indicates a Control-character. As you enter characters from the keyboard directly to AMOS, the system usually displays these characters on your terminal. If you type a Control-C, you see a ^C on your terminal display. (Refer to the AMOS User's Guide, (DWM-00100-35), for more information on Control-characters.)

(Changed 30 April 1981)

1.3 CONCEPTS

Below we define several of the terms that appear frequently in the command reference sheets that follow. For more information on system concepts, read "Part I - Getting Started" in the AMOS User's Guide, (DWM-00100-35).

1. AMOS COMMAND LEVEL - When you are at AMOS command level, you are communicating directly with AMOS (the Alpha Micro Operating System) and not with a program (e.g., BASIC or VUE) that AMOS is executing.

2. AMOS PROMPT - When you are at AMOS command level you see the AMOS prompt symbol, ., which tells you that the operating system is ready for you to enter a command.

3. COMMAND LINE - Whenever you enter a command to AMOS, you include the name of the command optionally followed by file specifications and option switches. The entire input line up to a RETURN is called a command line.

4. FILE SPECIFICATION - Data on a disk is organized into logically-related groups called files. Whenever you want to identify a file to an AMOS command, enter that file's specification.

5. DEFAULTS - When you omit information from a command line, AMOS has a set of information that it substitutes for the missing items. For example, if you do not tell AMOS what account a file belongs to, it usually assumes that the file resides in the account you are currently logged into. In this case, the default account is your own.

   NOTE: Defaults vary among commands. Check with the reference sheet for a specific command to see what defaults it uses. In particular, the special commands called wildcard file commands handle defaults differently than other commands on the system. (See Chapter 9, "The Wildcard File Commands," in the AMOS User's Guide for information on these commands.)

6. WILDCARD - A wildcard is a special symbol that appears in a file specification. Wildcards enable a file specification to represent more than one file. For example, the wildcard symbol * in this file specification:

   ```
   F*.TXT
   ```

   creates a specification that selects all .TXT files whose names begin with F, regardless of the rest of the file name.

NOTE: The use of wildcard symbols varies among commands. Some commands do not recognize wildcards; others (the wildcard file commands) handle wildcards differently than do the rest of the commands on the system. Refer to the reference sheet for a particular command to see how it handles wildcards.

7. SWITCH (OR OPTION) - Several AMOS commands and programs allow you to select among several options by including switches on a command line. A switch is a slash (/) followed by one or more characters. You can sometimes include several switches on one command line.

   The specific form that switches take varies depending on the particular command. Some commands expect every single character after a slash to represent a different switch (e.g., .MAP/FSR); others require that each switch begin with a new slash (e.g., .PRINT NET.BAS/COPIES:2/BANNER/HEADER). Refer to the reference sheet for a particular command to see the switches for that command.

8. WILDCARD FILE COMMAND SWITCHES - Wildcard file commands make a distinction between two types of switches: file switches and operation switches. If a file switch is placed directly after a file specification, it affects only that file. For example:

   .ERASE MNTDVR.MAC,MTNDVR.PRG/QUERY,MTNDVR.OBJ (RET)

   tells ERASE to ask for confirmation before erasing the file MTNDVR.PRG. It erases the other two files without asking for confirmation, however.

   An operation switch affects all files on the command line, no matter where it is placed. For example, the DIR/WIDE option affects the directory display for all specified files, no matter where it appears on the command line.

   NOTE: Wildcard file commands allow you to set the default switch by placing the switch in front of a file specification. For example:

   .ERASE/QUERY MTNDVR.MAC,MTNDVR.PRG,MTNDVR.OBJ/NOQUERY,SRCFIL.BAS (RET)

   tells ERASE to ask for confirmation before erasing the first, second, and fourth files specified on the command line.

   See Chapter 9, "The Wildcard File Commands" in the AMOS User's Guide for more information on wildcard file command switches and default switches.

9. COMMAND FILE - A command file is an ASCII text file that contains valid AMOS system commands and file specifications. It can contain any commands and data that you can enter at AMOS command level (including the name of another command file). As AMOS processes a

command file, it performs the functions called for by each line of the file. Command files can also contain several special symbols that affect the way the file is displayed on the terminal screen as it is processed, and that allow the file to ask for input from the user of the command file. A special kind of command file, called a DO file, also allows the user of the file to specify text arguments which AMOS then substitutes into the DO file in the place of special parameter symbols.

CHAPTER 2

INTRODUCTION TO AMOS COMMANDS

Each reference sheet in this manual gives you detailed information on the use of a specific command. Before getting into the use of particular commands, however, it's a good idea to discuss exactly what we mean by the term "command."

A command is simply a specification that selects a file on the disk that has been loaded into memory (a memory module). AMOS responds to the command by trying to locate the memory module and executing it. If the module doesn't exist, AMOS finds the file on the disk, loads it into memory, and then executes it. The file selected by a command must be either a machine language program (usually identified by a .PRG file extension) or a command file (identified by a .CMD or .DO extension.)

In other words, when you type a command, AMOS loads into memory the file specified by that command and executes it. Because the programs specified by commands are not actually part of the operating system, but are simply files on the disk, you can add to the commands that AMOS recognizes by creating your own machine language programs and command files.

When you enter a command at AMOS command level, AMOS performs a thorough search procedure as it looks for the program or command file specified by the command. It looks in various accounts on the System Disk and on the device you are logged into. For example, if you enter:

```
.RECALL RET
```

AMOS looks first for a memory module by that name in system memory or in your memory partition. If such a module is not there, AMOS next looks for a disk file DSK0:RECALL.PRG[1,4]. If that file is not there, AMOS looks for it in your account. If the file is not there either, AMOS assumes that the file must be a command file, and looks for it in the System Command File Library account, DSK0:[2,2], as DSK0:RECALL.CMD[2,2]. Next it looks for RECALL.CMD in your own account. These are but a few of the steps in the command search procedure that AMOS follows. For an exact outline of the

AMOS command search procedure, refer to Appendix B, "AMOS Command Processing" in the AMOS User's Guide, (DWM-00100-35). At the completion of its search, if it still has not found the file specified by the command, AMOS echoes the command back to you enclosed in question marks. Then it displays the AMOS prompt to indicate that it is ready for a new command. For example:

```
?RECALL?
.
```

2.1 COMMANDS TO BE USED WITH CAUTION

This section is a quick reference to those commands which you should use with caution. Some of these commands can destroy the contents of your disk. Others can bring other users on your system to a halt or must only be used under certain conditions. Refer to the reference sheet for a particular command for more information on that command.

2.1.1 Commands that May Only Be Used From the First Memory Partition

The job that uses these commands must be running in the first memory partition on the system (Bank Zero for bank switched systems):

CDC210	HWKLOD	ICMLOD	MONTST
PERLOD	SMDLOD	TRILOD	T80LOD
WNGLOD	WNG210		

2.1.2 Commands that Destroy Disk Contents

CPY410	CPY500	CRT410
DSKCPY	RAZA	SYSCPY
SYSACT (Initialize option)		

NOTE: The disk diagnostic commands DIAG2, RNDRED, and REDALL do not harm the data on your disk.

2.1.3 Commands Only For the Use of the System Operator

You must be logged into the System Operator's account, [1,2], to run these commands:

DSKANA	SYSACT

(Changed 30 April 1981)

2.1.4 Access Limitations

These commands may only be used when no other job is running on the system:

CPY500	FMT200	FMT210
FMT400	FMT500	
DSKCPY (in fast Hawk mode)		

These commands may only be used when no other user is accessing the affected disk controller:

CPY410	CRT410

These commands may only be used when no other user is accessing the affected disk(s):

DIRSEQ	MOUNT	DSKPAK
COPY/P	SYSACT	SYSCPY
DSKCPY (in all modes but Hawk fast copy)		

Note also that never more than one user may access the same file at the same time.

2.1.5 Commands for Experienced Users

In addition to the commands above, the commands below are especially dangerous if used by inexperienced users:

ATTACH	DSKDDT	FORCE
JOBMEM	JOBPRI	MEMORY
QDT	SLEEP	SUSPND

2.2 PRIVILEGED COMMANDS

This manual contains a reference sheet for every command on the system. It is important to emphasize that not all commands can or should be used by the general user of the system. You must be logged into the system as System Operator (i.e., in account DSK0:[1,2]) in order to use several of the commands discussed in this manual. That account is usually protected with a password.

Many of the commands (e.g., a disk formatting program) can be very destructive to disk files if used recklessly. Therefore, the System Operator may want to transfer to DSK0:[1,2] those programs that he wants to reserve for the use of privileged personnel (e.g., JOBPRI, JOBMEM, FMT500, etc.). It is the responsibility of the System Operator to restrict the use of dangerous commands to those users knowledgeable enough to handle them wisely.

2.3 WILDCARD FILE COMMANDS

The reference sheets for several of the AMOS commands mention that those commands are "wildcard file commands." These commands are a special group of commands that, even though they perform vastly different functions, all handle file specification wildcards and defaults in the same way. Because they have much more powerful abilities to process file specifications than other commands on the system, it's important that you be aware of how they handle file specifications before you begin to use them extensively. They do not follow the standard system rules for specification defaults and wildcards. If you are not yet familiar with wildcard file commands, read Section 9.1, "Introduction to Wildcard File Commands," in the AMOS User's Guide.

CHAPTER 3

FILE SPECIFICATIONS

3.1 INTRODUCTION

Most AMOS commands require that you supply one or more file specifications on a command line. The file specification identifies a file to AMOS, and takes this form:

```
Devn:Filename.Extension[p,pn]
```

where:

Devn: Three letters and a number that select a logical unit of a physical device. This specification tells AMOS the device the specified file resides on. The device specification and the filename are separated by a colon.

Devn: usually identifies a disk (e.g., DSK0:, the System Disk), but can identify a magnetic tape unit (e.g., MTU7:) or a special device (such as system memory, RES:).

Several commands also recognize a different type of device called an "ersatz device." Ersatz devices identify specific accounts on the System Disk. For example, the ersatz device BAS: identifies the System BASIC Language Library account, DSK0:[7,6]. For more information on ersatz devices, refer to Section 9.1.5, "Ersatz Devices" in the AMOS User's Guide, (DWM-00100-35). For information on special devices, turn to Section 6.1.1.1, "Special Devices," of that manual.

Filename — A one- to six-character file name. AMOS usually considers upper and lower case letters in a filename to be the same, but some programs require that you enter filenames as all upper case. WARNING: If you enter more than six characters, AMOS may not (depending on the command you are using) process any extension or PPN that follows the name.

Extension — Zero to three characters that follow the filename and give information to the command about the contents of the file. The filename and extension are separated by a dot. For information on the various extensions recognized by the system, refer to Chapter 6, "Identifying Files to AMOS" in the AMOS User's Guide.

[p,pn] — Project-programmer number. Identifies the account in which the file resides. When it is part of a file specification, always enclose the PPN within square brackets. For information on PPNs, refer to Chapter 5, "Identifying Yourself to AMOS" in the AMOS User's Guide.

An example of a typical file specification:

```
SMD5:PROJCT.TXT[100,2]
```

where: the device specification, SMD5:, tells AMOS that the file resides on logical unit SMD5 of the physical device SMD; the filename PROJCT identifies the file; the extension .TXT further identifies the file (and tells AMOS that it contains text data); and the PPN [100,2] specifies the account on SMD5: where AMOS can find the file.

CHAPTER 4

FILE SPECIFICATION DEFAULTS AND WILDCARD SYMBOLS

4.1 DEFAULTS

If you omit elements of a file specification, most commands can fill in some of the missing information for you. For example, if you omit the device specification, most commands assume that you want to access a file on the device you are currently logged into.

The assumptions that commands make about missing file specification elements are called defaults. The defaults that a command uses depends on the specific command. Each reference sheet in this manual lists the defaults used by the command if they differ from the standard system defaults.

4.1.1 Standard System Defaults

All AMOS commands use the standard file specification defaults below. Several other commands (the wildcard file commands) also assume additional default information.

1. If you omit a device and unit number from a specification (e.g., WORKER.BAS[100,4]), AMOS assumes the device and unit number that you are currently logged into.

2. If you omit the device unit number (e.g., DSK:WDOBJ.PRG[23,4]), AMOS uses the default unit number zero. The specification above, then, selects file DSK0:WDOBJ.PRG[23,4].

3. If you omit the PPN, AMOS uses as the default the account you are currently logged into.

4. If you omit a file extension, the default extension depends on the command you are communicating with. For example, TXTFMT assumes a .TXT extension; ERASE assumes an empty extension.

IMPORTANT NOTE: The wildcard file commands handle file specification defaults a little differently than do the rest of the commands on the system. Refer to Section 9.1, "Introduction to Wildcard File Commands," in the AMOS User's Guide for information on these commands.

4.2 WILDCARD SYMBOLS

This section discusses the special symbols that can appear in file specifications.

The basic file specification selects only one file. For example, the specification:

```
DSK1:CRLF.MAC[300,2]
```

selects the file CRLF.MAC on device DSK1: in account [300,2]. Wildcard symbols allow one file specification to select several files. For example:

```
*.TXT
```

selects all files in the account and device you are logged into that have .TXT extensions, regardless of name. Not all AMOS commands recognize wildcard symbols. All of the commands that are able to process wildcards recognize the standard system wildcards. In addition, wildcard file commands have an advanced wildcarding ability not shared by the rest of the commands on the system.

4.2.1 Standard System Wildcard Symbols

Below are the standard system wildcard symbols:

* Matches any symbol or group of symbols in a filename or extension. BOTANY.* selects all files in your account that have the name BOTANY, regardless of extension.

  You may precede the * with one or more symbols (e.g., F1*.MAC), but within that name or extension, no symbols may follow the *.

? Matches any one symbol in a filename or extension. ???DSK.MAC selects PACDSK.MAC, DIRDSK.MAC, and ARTDSK.MAC.

You may place characters before or after ?s. If ?s appear at the end of a filename or extension, that many or fewer characters can match the ? symbols; otherwise, the number of characters that matches these wildcard symbols exactly equals the number of ?s.

For more information on wildcards, refer to Section 6.2, "Wildcard Symbols," in the AMOS User's Guide.

IMPORTANT NOTE: The wildcard file commands handle wildcards differently than do the rest of the commands on the system. In addition to the standard system wildcards (above) they also recognize several other symbols. For example, these commands allow you to use the wildcard symbol * in PPNs as well as in filenames and extensions. Wildcard file commands also recognize the wildcard PPN symbol, [], in file specifications. (The [] symbol, equivalent to [*,*], selects ALL accounts.) For more information on how wildcard file commands process file specifications containing wildcard symbols, refer to Section 9.1.1., "Wildcard Symbols," in the AMOS User's Guide.

CHAPTER 5

REFERENCE SHEET FORMAT

5.1 INTRODUCTION

The reference sheets that follow are designed to provide you with the information you need to use the commands of the AMOS system. The sheets are each in the same format and contain the same subheadings. The purpose of this discussion is to guide you through the architecture of the sheets while you gain familiarity with them so that you can quickly access the information they contain.

We discuss each subheading individually. Some of the subheadings do not appear in each reference sheet, but are only used occasionally when needed. These exceptions are labeled "(as needed)." Most of the subheadings appear on every sheet, however, so you can glance at any sheet and quickly locate the topic you need.

5.2 REFERENCE SHEET SUBHEADINGS

Below we discuss the reference sheet subheadings. The form this discussion takes is similar to that of an actual reference sheet.

FUNCTION:

Provides a brief statement of the purpose, use, and features of the command. It reports the action that takes place when you use the command correctly.

HINTS/RESTRICTIONS:

This narrative portion describes the action of the command, how the command makes decisions, and the consequences of the command. It also suggests special uses of the command, helps you determine what to expect when using the command, and reminds you of other commands you must use first.

This section refers you to other Alpha Micro documentation for more information on the use and purpose of the command. It also helps you interpret the results of the command and suggests further steps to you.

Within this section you can also see paragraphs marked NOTE: or IMPORTANT NOTE:. These subsections contain special warnings. They also highlight important information, such as definitions of terms used in the text. An IMPORTANT NOTE: can advise you of potentially dangerous situations that can result from improper use of the command. For example, some commands may not be used while other users are accessing the same disk. Important warnings usually also appear under the CHARACTERISTICS section near the end of the reference sheet.

DEFAULTS (as needed):

Every command has a set of file specificaton defaults. The particular defaults used by a command depend upon that command. If the defaults differ from the standard system defaults, we list them here.

OPTIONS (as needed):

Options are available with most commands; you may select the options by including switches on the command line. This section describes the options available with the command.

FORMAT:

The first line of this section is always a sample command line. For example:

.COMMAND {Filespec}{/Switch{/Switch}}↵

COMMAND is the AMOS command being discussed. In the example above, Filespec is the specification of the file you want the command to act upon and /Switch is a switch that selects a command option.

The FORMAT section shows the syntax of the command; i.e., how to enter the command line correctly. Notice that all command lines begin with the AMOS prompt, ., which is the indication that you are at AMOS command level. The {} symbols indicate that the

enclosed command line elements are optional. These optional elements may be nested several deep (as in the case of the /Switches, above).

A curly arrow indicates the place in the command line where you must type a RETURN. The brief narrative following the sample command line identifies the elements of the command line and gives you information on using the command or the options and switches.

OPERATION:

This section details the step-by-step function of the command. It discusses all action that occurs between you and AMOS while you use the command, including what happens when you use the available options.

COMMAND SUMMARY (as needed):

A few reference sheets contain summaries of the special symbols you can enter to the program invoked by the command.

ERRORS:

This section contains a list of all important error messages you can see while using the command. The error messages are produced either by the operating system or by the command itself. Most operating system messages usually begin with a question mark and the word "Cannot." For example:

```
?Cannot OPEN Devn: - disk is not mounted
```

The second word in this type of message is capitalized because it designates the particular monitor call that failed. (For a list of all operating system error messages, refer to Appendix A, "AMOS System Error Messages" in the AMOS User's Guide.)

The command itself generates the second type of error message. These messages take different forms, depending on the command, but usually begin with a special symbol such as a question mark, a square bracket, a percent sign, etc.

A short paragraph follows each error message. These sentences discuss the reasons for the appearance of the error message and give suggestions for recovering from the error.

CHARACTERISTICS:

Summarizes in several brief statements the unique features of the command.

States which accounts or devices are affected by the command.

Any warnings that appeared in HINTS/RESTRICTIONS are restated here.

Tells you where your terminal is returned to after use of the command. For example, this section may say: "Returns your terminal to AMOS command level."

EXAMPLES (as needed):

For clarity, we may include the subheading EXAMPLES if further elaboration on command use seems necessary. The section contains a series of definitions followed by examples and occasional comments.

CHAPTER 6

THE AMOS SYSTEM COMMANDS REFERENCE SHEETS

6.1 INTRODUCTION

The rest of this manual consists of reference sheets that briefly summarize the use of each command on the system. We have organized these sheets alphabetically to help you access them quickly. Below is an alphabetically ordered list of all AMOS commands. We also provide a functional summary of the AMOS commands, so that if you are not familiar with the name of a specific command, but know its function, you will be able to find it.

6.2 ALPHABETIC SUMMARY OF AMOS COMMANDS

Below is an alphabetic list of all AMOS commands.

APPEND	ASCDMP	ATTACH	BADBLK	BASIC
BATCH	BAUD	BITMAP	BMVR	CAL120
CDC210	CLKFRQ	COM	COMPIL	CONT
COPY	CPMCPY	CPMDIR	CPY410	CPY500
CREATE	CRT410	DATE	DDT	DEL
DEVTBL	DIAG2	DIAG3	DIAG4	DING
DIR	DIRSEQ	DO	DSKANA	DSKCPY
DSKDDT	DSKDMP	DSKFIL	DSKPAK	DUMP
DYSTAT	EDIT	EMAIL	ERASE	EXIT
FILCOM	FILDMP	FILTAP	FIX	FIXDVR
FIXMTM	FMT200	FMT210	FMT400	FMT500
FORCE	GLOBAL	GOTO	HASHER	HEDLOD
HELP	HWKLOD	IBMCPY	IBMDIR	ICMLOD
ISMBLD	ISMCOM	ISMDMP	ISMFIX	JOBMEM
JOBPRI	JOBS	KILL	LIB	LABEL
LINK	LISP	LOAD	LOG	LOGOFF
LOOKUP	MACRO	MAKE	MAP	MEMDEF

(Changed 30 April 1981)

MEMERR	MEMORY	MONGEN	MONTST	MOUNT
NEWTRM	OPR	PARITY	PASS	PAUSE
PC	PCL	PCU	PDLFMT	PERLOD
PPN	PRINT	PL	PRUN	PU
QDT	QUEUE	RAZA	REDALL	RENAME
RES	REVIVE	REWIND	RNDRED	RUN
SAVE	SEND	SET	SIZE	SKIP
SLEEP	SMDLOD	SORT	SRCCOM	SUSPND
SYMBOL	SYSACT	SYSCPY	SYSMEM	SYSTAT
SYSTEM	TAPE	TAPFIL	TAPDIR	TIME
TRACE	TRIDDT	TRIINI	TRILOD	TRISET
TRMDEF	TXTFMT	TYPE	T80INI	T80LOD
U	VUE	WAIT	WNGLOD	WNG210
XY				

6.3 FUNCTIONAL SUMMARY OF COMMANDS

Below is a functional summary of all AMOS system commands. NOTE: Those commands that perform several functions appear under more than one heading.

1. Disk Directory and Account Commands:

CPMDIR	DIR	DIRSEQ
IBMDIR	LOG	LOGOFF
PASS	PPN	SYSACT

2. File Commands:

APPEND	ASCDMP	COPY
COM	CREATE	DIR
DSKFIL	DSKDMP	DUMP
ERASE	FILCOM	FILDMP
MAKE	PRINT	RENAME
SIZE	SORT	SRCCOM
TYPE		

3. Wildcard File Commands:

COPY	DIR	ERASE
PRINT	RENAME	

4. Disk and File Copy Commands:

COPY	CPMCPY	CPY410
CPY500	DSKCPY	IBMCPY
SYSCPY		

5. Special Commands:

BMVR	DSKPAK	DO
EMAIL	FIXDVR	FIXMTM
MONGEN	MONTST	MOUNT
NEWTRM	OPR	

6. Command File Commands:

BATCH	COM	CONT
EXIT	GOTO	LOOKUP
PAUSE	TRACE	

7. Text Processing Commands:

EDIT	PDLFMT	TXTFMT
VUE		

8. Language Processer Commands:

BASIC	COMPIL	GLOBAL
LIB	LINK	LISP
MACRO	PC	PCL
PCU	PL	PRUN
PU	RUN	SYMBOL

9. Job and Terminal Handling Commands:

ATTACH	BAUD	DING
FORCE	JOBMEM	JOBPRI
JOBS	KILL	LOG
LOGOFF	REVIVE	SEND
SET	SLEEP	SUSPND
WAIT	XY	

(Changed 30 April 1981)

10. Memory Partition Commands:

DEL	JOBMEM	LOAD
MAP	MEMORY	RES
SAVE	SYSMEM	

11. Disk Analysis and Certification Commands:

BADBLK	CRT410	DIAG2
DSKANA	HASHER	LABEL
RAZA	REDALL	RNDRED

12. Memory Diagnostic Commands:

DIAG3	DIAG4

13. Disk Formatting Programs:

FMT200	FMT210	FMT400
FMT500		

14. Magnetic Tape Unit Commands:

FILTAP	REWIND	SET
SKIP	TAPE	TAPDIR
TAPFIL		

15. System Initialization Commands:

BITMAP	CLKFRQ	DEVTBL
HEDLOD	JOBS	KILL
MEMERR	PARITY	QUEUE
SYSTEM	TRMDEF	

16. System Information Commands:

ATTACH	BITMAP	CLKFRQ
DATE	DEVTBL	DYSTAT
HEDLOD	HELP	JOBMEM
JOBPRI	JOBS	MEMORY
PPN	QUEUE	SET
SYSMEM	SYSTAT	SYSTEM
TIME	TRMDEF	

(Changed 30 April 1981)

17. ISAM Commands:

ISMBLD	ISMCOM	ISMDMP
ISMFIX		

18. Debugging Commands:

CAL120	DDT	DSKDDT
QDT	QDT	TRIDDT

19. Bootstrap Loader Commands:

CDC210		
HWKLOD	ICMLOD	PERLOD
SMDLOD	TRILOD	T80LOD
WNGLOD		

20. Trident Hard Disk Initialization Commands:

TRIINI	TRISET	T80INI

6.4 THE COMMAND REFERENCE SHEETS

The next section of the manual consists of the command reference sheets themselves, one for each command on the system. For information on the format of these sheets, refer to the previous chapter, "Reference Sheet Format."

append

FUNCTION:

APPEND takes the contents of one or more source files and places the combined contents into a single destination file.

HINTS/RESTRICTIONS:

You may use APPEND to combine sequential files of any type. You can append data onto the end of a file by specifying the same specification for the new file as one of the old files you are combining (e.g., APPEND BOOK.TXT=BOOK.TXT,BIBLIO.TXT). APPEND does not change the contents of the old files (unless, of course, the new file has the same name and extension as one of the old files).

If you do not specify the extension of the new file specification, APPEND assumes a null extension; that is, an extension with no characters. If you do not specify the extension of the first old file specification, APPEND assumes that the first old file specification has the same extension the new file specification has. Similarly, if you do not specify the extension of any subsequent old file specification, APPEND assumes that that old file specification has the same extension as the old file specification immediately previous to it.

FORMAT:

.APPEND Newfilespec=Oldfilespec1{,Oldfilespec2,...OldfilespecN} RET

where Newfilespec specifies the new file that will hold the contents of the combined files, and Oldfilespec1,....OldfilespecN lists the one or more files that you want to merge.

DEFAULTS:

APPEND assumes the account and device you are currently logged into.

APPEND assumes a null file extension (i.e., a no-character extension) for the new file specification.

APPEND assumes the extension of the new file specification for the first old file specification, and then assumes for each subsequent old file specification the actual or assumed extension of the previous old file specification.

(Changed 30 April 1981)

OPERATION:

1. Type APPEND followed by the specification of the new file, an equal sign, and one or more specifications (separated by commas) of the files whose contents you want to merge. Now type a RETURN. For example:

   .APPEND ASMBLR.MAC=PARSE.MAC,SCAN.MAC,TABLE.MAC,CNVRT.MAC (RET)

   When APPEND has written the combined contents of the old files into the new file, you see the AMOS prompt symbol.

2. If you type:

   .APPEND ASMBLR.MAC=PARSE,SCAN,TABLE,CNVRT (RET)

   APPEND will assume the default extension of .MAC for each of the files PARSE, SCAN, TABLE and CNVRT because that is the extension of the new file ASMBLR.

3. If you type:

   .APPEND ASMBLR=PARSE,SCAN,TABLE,CNVRT (RET)

   APPEND will assume that the extension of the file ASMBLER is a null extension (ASMBLR.), or an extension having no characters. Then APPEND will also assume, by default, that the extensions of PARSE, SCAN, TABLE and CNVRT will also be null extensions.

4. If you type:

   .APPEND EXAMPL.TXT=INTRO,SAMPL1.BAS,SAMPL2,CLOSE.TXT (RET)

   APPEND will create the new file EXAMPL.TXT to include INTRO.TXT (the extension is assumed to be the same as that of the new file specification), SAMPLE1.BAS, SAMPL2.BAS (the extension is assumed to be the same as that of the previous old file specification, SAMPL1.BAS), and CLOSE.TXT.

ERRORS:

?Command error

APPEND did not recognize the characters on the command line as being in valid command format. For example:

.APPEND VIEW.BAS= (RET)

Try again, making sure that you have remembered to type an equal sign and new and old file specifications.

(Changed 30 April 1981)

?File specification error

You did not enter a file specification in proper form. For example, you see this message if you enter a null file specification:

.APPEND (RET)

Any of the standard system error messages may occur. For example:

?Cannot OPEN Filespec - file not found

AMOS cannot find the file you specified. Make sure that you entered the correct account and device specification, or that the extension you allowed by default is that of an existing file.

?Cannot OPEN Filespec - protection violation

You tried to create a file in an account not in the project you are currently logged into. For example, you are logged into DSK0:[100,1], but tried to create a file in DSK0:[40,1]. Make sure that you are logged into the same project as the account in which you are trying to create the new file.

CHARACTERISTICS:

Assumes null file extensions.

Does not create a file in an account outside of the project you are currently logged into.

Assumes the extension of the newfilespec for the first oldfilespec. Assumes for each subsequent oldfilespec the actual or assumed extension of the previous oldfilespec.

ascdmp

FUNCTION:

ASCDMP displays the data in physical disk records on your terminal display in ASCII form.

HINTS/RESTRICTIONS:

ASCDMP tries to display all of the data in a record in ASCII form; if you've asked it to look at data that is not ASCII (e.g., a machine language program), the display won't make sense.

Enter the number of the disk record you want to display in the same number base that the system is using for your numeric displays (usually octal). (You can change this system display base to hexadecimal by using the SET HEX command.)

FORMAT:

.ASCDMP Devn:nnnn ↵

where Devn: specifies the logical unit on which the record occurs and nnnn is the number of the physical record on the disk that you want to display.

OPERATION:

1. Type ASCDMP followed by a device specification and physical record number. Then type a RETURN. For example:

   .ASCDMP DSK0:200 ↵

2. ASCDMP tells you what record it is accessing ([RECORD nnnn]), and then displays the contents of the physical disk record on your terminal in ASCII form. (To find out what records are contained in a specific file, use the DSKFIL command.)

ERRORS:

ASCDMP generates no error messages of its own, but can display some of the standard system error messages. For example:

?Cannot INIT Devn: - device does not exist

The system cannot find the logical unit you specified. Check your spelling and try again. (For example, did you enter DKS1: instead of DSK1: ?)

(1 October 1979)

?Cannot READ Devn: - disk not mounted

You are trying to access a logical unit that is not mounted. Use the MOUNT command to mount the disk and try again.

CHARACTERISTICS:

Only accepts record numbers in the base that the system is using for your numeric displays (usually octal).

Returns your terminal to AMOS command level.

attach

FUNCTION:

Attaches (that is, links) a job to a terminal or tells you what jobs and terminals are attached on the system.

HINTS/RESTRICTIONS:

When the system is reset or powered up, it automatically attaches the first job and the first terminal defined in your SYSTEM.INI. Except for that special case, however, the system does not automatically attach any jobs and terminals. If you want a job to be able to use a terminal for input and output, you must explicitly attach the job and the terminal by using the ATTACH command.

If you want to attach a job and a terminal that are already linked to other units, the ATTACH command will go ahead and detach the job and the terminal from their previous attachments. Then it will attach the freed job and terminal to each other. This is the only way that you can detach jobs and terminals (i.e., by attaching them to something else).

If you want to attach a terminal to a job that is logged into an account, ATTACH asks you for the password of that account, if one exists, before it attaches the specified job and terminal.

You can attach your terminal to a job, but be careful that the job has some memory allocated to it or you will not be able to ask for most AMOS system commands.

A typical use for the ATTACH command might be to attach a printer to a job as a terminal so that you can get hardcopy of program output.

FORMAT:

.ATTACH RET

or:

.ATTACH Job RET

or:

.ATTACH Terminal,Job RET

Using ATTACH in the first format tells you what jobs and terminals are attached to each other.

Using ATTACH in the second format attaches the specified job to your terminal.

(Changed 1 May 1980)

Using ATTACH in the third format attaches the specified terminal and job.

OPERATION:

1. To find out what jobs and terminals are attached to each other, type ATTACH followed by a RETURN:

```
.ATTACH (RET)
TERM1   ATTACHED  TO  JOBA
TERM2   ATTACHED  TO  JOBC
DUKE    ATTACHED  TO  SPOOL
```

2. To attach your terminal to a specific job, type ATTACH followed by a job name. Type a RETURN. (The job name is set by the JOBS command in the SYSTEM.INI. To see the names of all of the jobs on the system, type SYSTAT followed by a RETURN. The first column on the left of the SYSTAT display lists the jobs.)

```
.ATTACH DUKE (RET)
```

3. To attach a specific job to a specific terminal, type ATTACH followed by the terminal name, a comma, and the job name. Type a RETURN.

```
.ATTACH TERM1,JOB4 (RET)
```

(The terminal names are set by the TRMDEF commands in the SYSTEM.INI. To see the names of all of the terminals on the system, type TRMDEF followed by a RETURN. The first column on the left of the display lists all of the terminals.)

4. If you try to attach a terminal to a job, and that job is logged into an account that has a password, ATTACH asks you for the password before attaching the job and the terminal. For example:

```
.ATTACH MULTRM,JOB3 (RET)
PASSWORD:
.
```

ERRORS:

?Nonexistent terminal
You tried to attach a job to a terminal that is not defined in a TRMDEF command in your SYSTEM.INI. You can type TRMDEF followed by a RETURN to see a complete list of all of the terminals defined on the system.

(Changed 1 May 1980)

?Nonexistent job
You tried to attach a terminal to a job that is not defined in the JOBS command in your SYSTEM.INI. Use the SYSTAT command to see a list of the jobs defined on the system.

?Account not found on login disk for specified job
You attempted to attach a terminal to a job that is logged into an account which no longer exists on the device. This can happen if, while the job is logged into the account, the disk that the account is on is removed from the device or the account is deleted from the disk using SYSACT.

?Bad password
You did not specify the correct password.

CHARACTERISTICS:

Allows you to detach a terminal from a job by attaching it to some other job.

Returns your terminal to AMOS command level.

badblk

FUNCTION:

BADBLK allows you to see the contents of the file BADBLK.SYS[1,2].

HINTS/RESTRICTIONS:

A disk certification program analyzes a disk and identifies bad disk blocks or tracks. Such a program creates account [1,2] on the disk to be certified and writes a list of the bad blocks or tracks into the file BADBLK.SYS[1,2]. (See the CRT410 reference sheet for an example of a certification program.) The output from BADBLK will vary slightly depending on whether the particular BADBLK.SYS in question is block- or track-oriented.

BADBLK allows you to see the contents of BADBLK.SYS[1,2]. BADBLK also verifies the BADBLK.SYS[1,2] hash total. (A hash total is a computed value based on the contents of a file. It serves as a validity check on the data in the file.)

FORMAT:

.BADBLK Devn: RET

where Devn: specifies the logical unit that contains the BADBLK.SYS[1,2] file you want to display.

OPERATION:

1. Type BADBLK followed by the specification of the logical unit whose BADBLK.SYS file you want to see. Then type a RETURN. For example:

   .BADBLK SMD5: RET

2. BADBLK searches for the BADBLK.SYS file in account DSK0: on the specified device. If it finds it, it tells you so. For example:

   .BADBLK SMD4: RET

   SMD4: BADBLK.SYS[1,2]

3. Next it tells you the serial number associated with that device and the number of blocks or tracks marked as bad on that disk. For example:

(Changed 1 May 1980)

```
Serial number: INVENTORY2
Number of bad blocks: 3
```

or:

```
Serial number: BACKUPA
Number of bad tracks: 3
```

4. If there are any blocks or tracks listed in the BADBLK.SYS file, BADBLK lists them for you. For example:

```
Number of bad blocks: 20

1035  1036  1041  1042  1043  1045  1046  1047
1052  1053  1056  1057  1060  1062  1063  1064
1067  1070  1073  1074
```

or:

```
Number of bad tracks: 2

15  19
```

5. After displaying the numbers of the bad disk blocks or tracks, BADBLK exits and returns you to AMOS command level:

```
EXIT
.
```

NOTE: Block numbers are octal; track numbers are decimal.

ERRORS:

You can see the following BADBLK error messages:

?File not found: Devn:BADBLK.SYS
BADBLK could not find the BADBLK.SYS file for the device you specified. Make sure that the device you specified has been certified by a disk certification program.

CAUTION: HASH TOTAL DID NOT VERIFY
The BADBLK.SYS file contains a bad hash total. This indicates that the data in the file is invalid. Use COPY to copy all files off the logical unit containing the bad BADBLK.SYS file. Then re-certify the disk.

You can also see several system error messages if you enter an invalid device specification. For example:

(Changed 1 May 1980)

?Cannot INIT Devn: - device does not exist

The system did not recognize the device specification you supplied. Check your spelling and try again. You can type DEVTBL followed by a RETURN to see a list of valid system devices.

?Cannot READ Filespec - disk not mounted

The sytem is unable to read the device you specified on the BADBLK command line because it is not mounted. Use the MOUNT command to mount the logical unit and try using BADBLK again.

CHARACTERISTICS:

For use on disks that have been processed by a disk certification program that creates a BADBLK.SYS[1,2] file.

Returns your terminal to AMOS command level.

(Changed 1 May 1980)

FUNCTION:

Allows you to use the BASIC language processor in interactive mode.

HINTS/RESTRICTIONS:

BASIC is an easy-to-learn programming language. This command invokes both the compiler and runtime package portions of the BASIC language processor. Use this command when you want to use BASIC in interactive mode. Once "in" BASIC, you may use the BASIC LOAD and SAVE commands to load and save BASIC programs.

For information on writing BASIC programs or on using either the compiler (COMPIL.PRG) or the runtime package (RUN.PRG) portions of BASIC separately, refer to the AlphaBASIC User's Manual, (DWM-00100-01).

BASIC.PRG is reentrant; the System Operator may include it in system memory via the SYSTEM command in the system initialization command file. However, because of the size of BASIC.PRG, this is not usually done unless most users on the system will be doing extensive BASIC program development.

NOTE: To exit from BASIC, enter BYE followed by a RETURN. To interrupt the execution of a program, type a Control-C.

FORMAT:

.BASIC ↵

OPERATION:

1. Type BASIC followed by a RETURN:

   .BASIC ↵

   You now see the BASIC prompt:

   READY

   You can either load in a BASIC program:

   LOAD PAYROL.BAS ↵

or begin to create a new program:

```
10 PRINT "This program computes interest rates."
20 INPUT "Enter balance: $",BALANCE
                .
                .
                .
```

2. To run the program that is in memory, type RUN followed by a RETURN:

```
RUN ↵
COMPILE
Compile time was 0.05 seconds
This program computes interest rates.
Enter balance: $ 2000 ↵
        .
        .
        .
Runtime was 1.71 seconds

READY
```

3. To exit from BASIC, use the BYE command:

```
BYE ↵
.
```

ERRORS:

You may see any of the standard BASIC error messages. For a list of the BASIC messages, refer to the back of the AlphaBASIC Users's Manual.

CHARACTERISTICS:

Invokes the compiler and runtime package portions of the BASIC language processor.

batch

FUNCTION:

Loads commands frequently used by command files into your memory partition.

HINTS/RESTRICTIONS:

A command file executes faster if the programs that it accesses are already loaded into memory. BATCH loads into memory: GOTO, LOOKUP, END, TRACE, PAUSE, and LOAD. Using BATCH speeds up the execution of your command file if that file uses those programs. Include BATCH at the front of the command file. (NOTE: GOTO, END, LOOKUP, TRACE, and PAUSE are re-entrant; the System Operator may place them into system memory where they can be accessed by all users on the system.)

BATCH takes up about 1K of your memory partition. (You may want to include a DEL* command at the end of a command file that contains the BATCH command, so that the programs loaded into your memory partition will be deleted when you exit the command file.) You may use BATCH at AMOS command level. You will probably find it especially useful within a command file or DO file. BATCH is a command file in the System Command File Library Account, DSK0:[2,2].

FORMAT:

.BATCH (RET)

OPERATION:

1. Use BATCH at AMOS command level by typing BATCH and a RETURN:

   .BATCH (RET)

2. Use BATCH from within a command file by placing it at the front of the file.

ERRORS:

BATCH displays no error messages unless the programs it tries to load have been deleted from the System Disk, in which case you see the standard AMOS error message: file not found.

CHARACTERISTICS:

BATCH is a command file in DSK0:[2,2].

Accepts no arguments or switches.

(30 April 1981)

baud

FUNCTION:

Changes the baud rate (the data-transfer rate) that the system uses to communicate with your terminal.

HINTS/RESTRICTIONS:

Works only with the AM-300 and AM-310 serial I/O boards.

Useful when changing from a CRT-type terminal to a hard-copy terminal.

Make sure that your terminal is able to communicate at the speed you specify to the BAUD command.

The baud rates that the system supports are:

50	baud	75	baud	110	baud	134.5	baud
150	baud	200	baud	300	baud	600	baud
1200	baud	1800	baud	2400	baud	3600	baud
4800	baud	7200	baud	9600	baud	19200	baud

FORMAT:

.BAUD Baud-rate-specification ↵

where Baud-rate-specification is the baud rate at which you want your terminal to run (e.g., 1200 baud).

OPERATION:

1. Type BAUD followed by a legal baud rate; then type a RETURN. For example:

.BAUD 300 ↵

ERRORS:

Baud rate nnnnn is not a legal baud rate
You have specified an invalid baud rate; try again making sure that the baud rate is correct.

CHARACTERISTICS:

Works only with the AM-300 and AM-310 I/O boards.

Returns your terminal to AMOS command level.

(1 October 1979)

bitmap

FUNCTION:

As part of the system initialization command file, defines the disk bitmap areas used by the operating system. At AMOS command level, BITMAP tells you what memory locations are used by these bitmap areas.

HINTS/RESTRICTIONS:

A bitmap is a storage allocation map that tells the system which disk records are free and which are in use.

You can only use BITMAP to allocate disk bitmaps from within the system initialization command file.

You must have one BITMAP command in the SYSTEM.INI for each type of disk device defined in the system device table. (See the reference sheet for DEVTBL.) Every time you add a new type of disk device to the system, you must add a new BITMAP command to the SYSTEM.INI file.

All logical units of one physical device may share the same bitmap area or each may have its own. Place all BITMAP commands before any SYSTEM commands.

The size of the bitmap area depends on the device. For example, the 10-megabyte Hawk (a hard disk) needs a bitmap size of 606 words to keep track of 9696 disk records. A single-density floppy disk in AMS format needs a much smaller bitmap (39), since such a disk contains only 616 disk records. The FIXDVR program will tell you the bitmap size you need for any floppy disk device. To find out the bitmap size required for a hard disk drive, refer to the documentation accompanying that device.

You may also define the bitmap to be in switchable system memory via the /S option. By placing the bitmap in switchable system memory, you reduce the total resident monitor size. To use the /S option, your system initialization command file, SYSTEM.INI, must define switchable system memory via the SYSMEM command. (See the SYSMEM reference sheet.)

FORMAT:

.BITMAP RET

to find out what areas of memory are used by the disk bitmaps, or:

BITMAP Dev,Size,Unit1{,Unit2,...UnitN}{/S}

(Changed 30 April 1981)

to define the disk bitmap areas from within the system initialization command file, where "Dev" specifies the device type (for example, DSK, AMS, HWK, etc.); "size" is a decimal number that defines the size (in words) of the bitmap area you want to reserve; and, "unit" specifies the number of the logical unit that will be using the bitmap area. (More than one logical unit of the same device may share the same bitmap area.)

OPERATION:

Using BITMAP as a user command at AMOS command level:

1. Type BITMAP followed by a RETURN:

```
.BITMAP (RET)
```

The display you see tells you what memory locations have been assigned to the disk bitmaps that were defined at the time of system initialization. For example:

```
.BITMAP (RET)
DSK0:   3:31350-40450
DSK1:   3:31350-40450
DSK2:   3:31350-40450
DSK3:   3:31350-40450
DSK4:   3:31350-40450
DSK5:   3:31350-40450
HWK0:   3:40476-43006
HWK1:   3:40476-43006
STD0:   3:43034-43150
STD1:   3:43034-43150
```

2. Each line of the BITMAP display gives you the following information:

```
Devn:   Bank:StartAddress-EndAddress
```

where:

a. Devn: is the device for which the bitmap is allocated.

b. Bank: is the memory bank the bitmap is in, if the bitmap is in switchable memory. (If it is not in switchable memory, this item is omitted.)

c. StartAddress is the first memory location used by the bitmap area.

d. EndAddress is the last memory location used by the bitmap area.

(Changed 30 April 1981)

If bitmap areas are shared between units, those devices will have the same StartAddress and EndAddress, because the devices share the same area of memory for their bitmaps. (Note that DSK0:-DSK5: in the sample display above share the same bitmap area, as do HWK0:-HWK1: and STD0:-STD1:.)

Using BITMAP as a system initialization command:

1. Type BITMAP followed by the three-character name of the device for which you are defining a bitmap area, the size of the bitmap area, and the one or more logical units that will be sharing that bitmap area. For example:

```
BITMAP AMS,39,0,1
```

   The command above defines a bitmap area of 39 words for device AMS. The two logical units that will be sharing this bitmap are drives zero and one (that is, AMS0: and AMS1:).

2. Each logical unit of a physical device may have its own bitmap area. For example, you could split the command above into:

```
BITMAP AMS,39,0
BITMAP AMS,39,1
```

3. You must have one BITMAP command in your system intialization command file for each type of disk device defined in the DEVTBL commands.

ERRORS:

?System memory not allocated - monitor memory will be used

You tried to place a bitmap in switchable system memory, but BITMAP was not able to find any such memory. It therefore placed the bitmap in the area of memory reserved for the monitor, thus increasing the size of your monitor.

CHARACTERISTICS:

Acts both as a user command and as a system initialization command.

bmvr

FUNCTION:

Programs 2708-type EPROMs on a CROMEMCO BYTESAVER PROM programmer board. (EPROMs are Erasable, Programmable Read-Only Memories.)

HINTS/RESTRICTIONS:

Programs only one PROM at a time. Program must not be greater than 1K in size or it will not fit in the PROM.

Enter the PROM address in the number base that the system is using for your numeric displays (usually octal). (Type SET followed by a RETURN to see if the system is using octal or hexadecimal for your numeric displays.)

FORMAT:

```
.BMVR Filespec ↵
PROM ADDRESS: PROM-address ↵
```

where Filespec is the specification of the file containing the program that you want to burn into the PROM. PROM-address is the octal address of the PROM that is to be programmed.

DEFAULTS:

BMVR assumes a file extension of .PRG and it assumes the device and PPN that you are logged into.

OPERATION:

1. Type BMVR followed by a file specification; then type a RETURN.

```
.BMVR ↵
```

2. BMVR asks for the address of the PROM. For example:

```
.BMVR PERLOD ↵
PROM ADDRESS: 160000 ↵
```

3. BMVR now programs the PROM by writing the specified file onto it 64 times.

(1 October 1979)

4. BMVR verifies the PROM by comparing the program and the contents of the PROM. If the PROM looks OK, BMVR tells you so and exits:

```
.BMVR PERLOD ↵
PROM ADDRESS: 160000
PROM VERIFIED OK
.
```

ERRORS:

THIS PROM NEEDS TO BE ERASED

BMVR has detected some data on the PROM. (If a PROM is erased, all bits are set.)

nnnn VERIFICATION ERRORS

When BMVR compared the original program and the contents of the PROM, it found some differences. (The number of verification errors is nnnn.)

PROGRAM WILL NOT FIT IN PROM

The program was larger than 1K and so would not fit in the PROM.

CHARACTERISTICS:

Programs 2708-type EPROMS.

The program you want to place in the PROM must not be larger than 1K.

Returns your terminal to AMOS command level.

cal120

FUNCTION:

Allows you to calibrate the AM-120 time-of-day clock oscillator.

HINTS/RESTRICTIONS:

This program is only for use by qualified maintenance personnel.

The AM-120 board is shipped from the factory fully calibrated. Normally, you will not need to re-calibrate the time-of-day clock. You will only need to use this program if the time-of-day clock oscillator is out of calibration when you initially install the AM-120 Auxiliary I/O Controller board in the system, or during subsequent maintenance.

Requires the use of a high-precision counter for the calibration procedure.

CAL120 calibrates the first AM-120 board it finds in the system.

After you have completed the calibration procedure, you should reset the AM-120 clock/calendar by using the TIME and DATE commands. (See the DATE and TIME reference sheets for information on using those programs with the AM-120.)

CAL120 is both re-entrant and re-usable.

For further information on calibrating the AM-120, see the document Installation Instructions AM-120, (PDI-00120-XX).

FORMAT:

.CAL120 RET

OPERATION:

1. After you have installed the AM-120 Auxiliary I/O Controller board in your system, you may calibrate it by entering CAL120 followed by a RETURN:

   .CAL120 RET

2. Now, adjust the trimmer capacitor to exactly 1024 Hz at the calibration test point.

3. When you have completed the calibration procedure, type a Control-C to exit CAL120.

(30 April 1981)

ERRORS:

You may see the following error message when using CAL120:

?Unable to locate AM-120 board
There is no AM-120 Auxiliary I/O Controller board in the system or the board's addressing jumpers are incorrectly set. (For information on how to physically address the AM-120 board, see the document Installation Instructions AM-120, (PDI-00120-XX).)

CHARACTERISTICS:

Allows you to calibrate the time-of-day clock oscillator on the first AM-120 Auxiliary I/O Controller board in the system.

CAL120 is re-entrant and re-usable.

cdc210

FUNCTION:

Bootstrap loader program for a system that uses the CDC floppy disk as the System Device running under the AM-210 floppy disk controller.

HINTS/RESTRICTIONS:

The CDC210 program when contained on a 2716 PROM allows the system to boot off a System Disk on a CDC floppy disk when a hardware reset occurs (that is, when you hit the RESET button).

The program is also in account DSK0:[1,4] of the System Disk.

You may use CDC210 at AMOS command level to reset the system if your System Device is a CDC floppy disk drive. The memory partition of the job that uses the CDC210 command MUST be in Bank Zero if your system bank switches memory. (For information on bank-switched systems, refer to the document Memory Management Option, (DWM-00100-10) in the AM-100 documentation packet.)

You may use CDC210 to boot from double- or single-sided diskettes that are in single- or double-density STD format or double-density AMS format.

FORMAT:

.CDC210 RET

OPERATION:

1. Type CDC210 followed by a RETURN:

   .CDC210 RET

   The system now resets itself by reading a copy of the CDC210 bootstrap program into system memory and executing it.

2. Once invoked, the CDC210 program reads the operating system skeleton monitor, DSK0:SYSTEM.MON[1,4], into memory. SYSTEM.MON then brings up the system under the control of your system initialization command file, SYSTEM.INI.

3. Once the system is up and running, you see the AMOS prompt.

ERRORS:

CDC210 generates no error messages. However, if it does not find SYSTEM.MON[1,4] and SYSTEM.INI[1,4], the start-up procedure fails.

(1 May 1980)

CHARACTERISTICS:

Boots the system from an AMS- or STD-format CDC floppy disk if the CDC disk drive is the System Device and runs under control of the AM-210 floppy disk controller.

Returns your terminal to AMOS command level if the system resets successfully.

FUNCTION:

When you use it within the system initialization command file, CLKFRQ tells the system the frequency being applied to the CPU real-time clock. When you use it at AMOS command level, CLKFRQ tells you the frequency defined by the CLKFRQ command in the SYSTEM.INI or resets that frequency.

HINTS/RESTRICTIONS:

As part of the system initialization command file, SYSTEM.INI, CLKFRQ tells the system the frequency (in Hz) that is being applied to the clock input line. This gives the programs that access the CPU real-time clock a value to use in converting clock ticks to actual time in seconds. The usual frequency in the United States is 60 Hz; overseas it is 50 Hz. Place CLKFRQ after all SYSTEM commands in the SYSTEM.INI.

Use CLKFRQ at AMOS command level to find out what frequency was defined by the CLKFRQ command in the system initializaton command file. You can also change the frequency set in the SYSTEM.INI by using CLKFRQ at AMOS command level.

NOTE: You must include the CLKFRQ command in the SYSTEM.INI. If you do not, the system stores a zero in the monitor location reserved for the clock frequency and those programs that refer to the real-time clock (e.g., BASIC) will not be able to calculate time correctly. Note that CLKFRQ in no way affects the speed at which your system operates.

FORMAT:

.CLKFRQ ↵

or:

.CLKFRQ n ↵

where n is the frequency in Hz that is being applied to the input line of the CPU real-time clock.

OPERATION:

1. At AMOS command level, to find out what value is being used as the clock frequency, type CLKFRQ followed by a RETURN:

```
.CLKFRQ ↵
CURRENT CLOCK FREQUENCY VALUE SET FOR 60 HZ
```

(1 October 1979)

2. To set the frequency used by the real-time clock, use one of the system text editors and edit the SYSTEM.INI file. Enter the CLKFRQ command after all SYSTEM commands. Type CLKFRQ followed by the frequency (in Hz) being applied to the CPU real-time clock line. For example:

```
CLKFRQ 60
```

3. After the system is up and running, you can use CLKFRQ to change the frequency previously set in the SYSTEM.INI. Enter CLKFRQ followed by the number that selects the frequency being applied to the clock line. Type a RETURN. For example:

```
.CLKFRQ 50 ↵
.
```

ERRORS:

CLKFRQ generates no error messages.

CHARACTERISTICS:

You must include CLKFRQ in the SYSTEM.INI file.

CLKFRQ is both a user command and a system initialization command.

Returns your terminal to AMOS command level.

COM

FUNCTION:

Processes a file based on its extension.

HINTS/RESTRICTIONS:

COM is a DO file in the system Command File Library Account, DSK0:[2,2]. It searches for a specified file, and then invokes the appropriate processor. For example, if the specified file is a .BAS file (a BASIC program), COM invokes COMPIL to compile the file. You may find COM especially useful as a command file element.

FORMAT:

.COM Filespec (RET)

where Filespec specifies the file you want to process. You may not specify an extension, and the file must be in the account you are logged into. (The file may appear on a different device, however, in which case you would include a device specification.)

DEFAULTS:

If you omit the device specification, COM assumes the device you are logged into. COM begins to search for the file in the following order:

1. Is it a .MAC file? If yes, assemble it with MACRO.
2. Is it a .BAS file? If yes, compile it with COMPIL.
3. Is it a .PAS file? If yes, compile it with PRUN CMPILR.
4. Is it a .TXT file? If yes, format it with TXTFMT.

OPERATION:

1. At AMOS command level, type COM followed by the specification of the file you want to process; then type a RETURN. For example:

   .COM SPRAD (RET)

   COM searches for the file in the order listed above in the DEFAULTS section. If the file is found, COM processes it with the appropriate processer based on the file's extension.

(Changed 30 April 1981)

ERRORS:

If COM cannot find the file or if the file does not have one of the extensions listed above in the DEFAULTS section, COM displays the message:

?Filename is not a compilable file

where Filename is the file you specified on the COM command line. If you use COM in a command file, if COM cannot find the file you specified, the command file aborts operation.

CHARACTERISTICS:

Processes a file with the proper processor; the selection of the processor is based on the file's extension.

COM is a DO file in DSK0:[2,2].

When specifying the file to be processed, you may not specify an extension. The file must appear in the account you are logged into. If you omit the device specification, COM assumes the device you are logged into.

compil

FUNCTION:

Compiles BASIC programs.

HINTS/RESTRICTIONS:

Invokes the compiler portion of the BASIC language processor. The program you compile is a program you have previously saved while using BASIC in interactive mode or is a program you have created using one of the text editors. For information on COMPIL, writing BASIC programs, using the BASIC runtime package (RUN.PRG) to execute programs, or using BASIC in interactive mode (BASIC.PRG), refer to the AlphaBASIC User's Manual, (DWM-00100-01).

COMPIL is re-entrant; the System Operator may include it in system memory (thus saving space in individual users' memory partitions) via the SYSTEM command in the system initialization command file.

If COMPIL finds errors, it does not produce a .RUN file.

NOTE: COMPIL does not require that your BASIC program contain line numbers.

FORMAT:

.COMPIL Filespec{/Switches} RET

where Filespec specifies the file that contains the program you want to compile and /Switches selects one or both of the COMPIL operation switches.

DEFAULTS:

COMPIL assumes an extension of .BAS.

OPTIONS:

You may select the following options by including them at the end of the COMPIL command line after a slash, /.

/T This switch is designed primarily as a debugging tool. It tells COMPIL to display each line of your source program as COMPIL scans the line. If an error occurs during compilation, the use of this switch helps you to pinpoint the program line where the error occurred.

(Changed 30 April 1981)

/O The /O switch tells COMPIL to omit line number references from the compiled code. This reduces the total size of your compiled program, but prevents compile or run-time error messages from reporting the number of the program line where the error occurred.

/M The /M switch causes COMPIL to verify that each variable in your program is mapped. If COMPIL encounters an unmapped variable, an error message is displayed.

OPERATION:

1. Type COMPIL followed by the specification of the file you want to compile. Then type RETURN. For example:

```
.COMPIL PROJCT.BAS (RET)
```

2. COMPIL now displays messages following the command line that indicate the status of the program compilation. For example:

```
Phase 1 - Initial work memory is 2310 bytes
Phase 2 - Adjust object file and process errors
Illegal MAP level - 350 MAP FILL'7,S,2
Syntax error - 980 BALANCE = BALANCE ; INTEREST
Memory usage:
        Total work space - 4712 bytes
        Label symbol tree - 322 bytes
        Variable symbol tree - 1186 bytes
        Data statement pool - 0 bytes
        Variable indexing area - 274 bytes
        Compiler work stack - 140 bytes
        Excess available memory - 11918 bytes
End of compilation
.
```

3. When COMPIL finishes, and if it found no errors (there were two in the example above), it produces a file containing a compiled program. This file has the same name as the file you specified on the COMPIL command line, but it has a .RUN extension. To execute this file, use the BASIC runtime package via the RUN command.

ERRORS:

You may see the standard BASIC error messages. For a list of the BASIC messages, refer to the back of the AlphaBASIC User's Manual (DWM-00100-01).

(Changed 30 April 1981)

CHARACTERISTICS:

Assumes an input file extension of .BAS. Produces a file with a .RUN extension.

(Changed 30 April 1981)

cont

FUNCTION:

Allows you to resume execution of a command file that was previously interrupted by a PAUSE command.

HINTS/RESTRICTIONS:

When you are using a command file, and its execution is interrupted because of a PAUSE command in that file, you are then returned to AMOS command level. At that point, you can use other command files, run programs, use the text editor, etc. When you are ready to resume execution of the command file, you may use the CONT command.

For more information on interrupting command file execution, and on the PAUSE and CONT commands, see New Features of Command Files and DO Files, (DWM-00100-63), in the "User's Information" section of the AM-100 documentation packet. Also, see the PAUSE reference sheet in this manual.

When command file execution is interrupted because of a PAUSE command, PAUSE saves the remainder of the unexecuted command file in a disk file named CNT.CMD in the account the user of the command file is logged into. CONT loads the CNT.CMD file into memory from the account you are logged into and executes it. Therefore, do not erase CNT.CMD from your account.

NOTE: If you do not type CONT after a PAUSE'd command file has returned you to AMOS command level, and you then invoke another command file that is also interrupted because of a PAUSE command, the contents of the second command file are written into CNT.CMD, replacing the remainder of the first command file. This does no harm, except that you may not now resume execution of the first command file.

FORMAT:

.CONT RET

OPERATION:

1. To resume execution of a command file that has previously been interrupted via the PAUSE command, type CONT at AMOS command level; then type a RETURN:

   .CONT RET

   After CONT loads CNT.CMD into memory and processes it, CONT erases the CNT.CMD file from the disk.

(1 May 1980)

2. You may also include CONT within another command file, which then resumes execution of the first command file.

ERRORS:

If CNT.CMD does not exist in the account you are logged into, CONT simply returns you to AMOS command level, since it is unable to resume execution of a command file; it then displays the message:

?Can't continue

Make sure that you are logged into the account where the command file you want to continue was interrupted (i.e., that you are in an account that contains a CNT.CMD file).

CHARACTERISTICS:

Resumes execution of a command file that has previously been interrupted by the PAUSE command.

Accepts no arguments or switches.

FUNCTION:

Copies one or more files: within accounts, between accounts, and between disks.

HINTS/RESTRICTIONS:

You may not copy to an account if it is not in the project you are logged into unless you are logged in as System Operator. You may copy files into your account from any other account, regardless of project number.

You may use COPY to pack the disk if you use the /PACK switch. However, do NOT pack the disk in this way while other users are accessing the disk.

.COPY TRM:=Filespecs	performs same function as: .TYPE Filespecs
.COPY MEM:=Filespecs	performs same function as: .LOAD Filespecs
.COPY =MEM:Filespecs	performs same function as: .SAVE Filespecs

COPY understands the ersatz devices.

NOTE ON DISK BACKUP:

If you are not logged in as System Operator, you may back up files in accounts that are within the same project. For example:

.COPY DSK2:[]=DSK1:[100,*] RET

copies all files on DSK1: in Project 100 accounts over to the same accounts on DSK2:. (The wildcard PPN symbol, [], tells the system to copy the files over to their corresponding accounts on DSK2:, instead of just the account you are logged into.) If you are logged in as System Operator, you may back up all accounts, regardless of project number:

.COPY DSK2:=DSK1:[] RET

(If you are logged into the System Operator's account, [1,2], the COPY command uses the [] symbol as the PPN default on the left side of the equal sign.)

NOTE: The file BADBLK.SYS[1,2] is created by a disk certification program for certain kinds of disk devices. It contains a list of the bad blocks or tracks on those devices. You must never destroy or alter the contents of a BADBLK.SYS[1,2] file. To protect the integrity of a device's certification data, COPY will not overwrite

the file BADBLK.SYS[1,2] on that device. (However, note that ERASE will erase the BADBLK file.)

COPY is a wildcard file command. Refer to Section 9.5, "Copying Files (COPY)," in the AMOS User's Guide, (DWM-00100-35), for more information on the use of COPY.

FORMAT:

.COPY {Newfilespec}={Oldfilespec1{,...OldfilespecN}}{/Switches} (RET)

where Newfilespec is the specification of the file you want to create, Oldfilespec is the one or more files you want to copy, and Switch is an option request.

DEFAULTS:

The initial default Oldfilespec is *.* and the account and device you are logged into. The default Newfilespec is *.* and the account and device you are logged into unless you are logged into [1,2], in which case it is *.*[] and the device you are logged into.

The default switches are: /DELETE/NOQUERY/NOPACK.

OPTIONS:

Use the switches below to select COPY options. (Precede each switch with a slash. Remember that the placement of the switch on the command line modifies its effect.)

/QUERY	or /Q	Ask user for confirmation before copying files (file switch).
/NOQUERY	or /NOQ	Don't ask for confirmation (default, file switch).
/DELETE	or /D	Copy over to an existing file, thus deleting it (default, file switch).
/NODELETE	or /NOD	Don't copy over to any existing files (file switch).
/PACK	or /P	Allow COPY to copy files over to themselves, thus packing disk (operation switch).
/NOPACK	or /NOP	Don't allow copying files over to themselves (default, operation switch).

OPERATION:

1. Type COPY followed by the specification you want to assign to the new file. Now type an equal sign followed by the specifications that select the files you want to copy. For example:

```
.COPY SORT.TXT=WRK1.TXT (RET)
WRK1.TXT to SORT.TXT
Total 1 file transferred
```

2. Remember that you can use wildcard symbols and that COPY assumes certain Filespec defaults. For example:

```
.COPY DSK1:*.OLD=*.NEW (RET)
```

copies all files with a .NEW extension from the account and device you are logged into over to the same account on DSK1:. The new files have the same names, but have extensions of .OLD.

3. When you use the /Q switch, COPY asks for confirmation of each transfer. (Remember, however, that the placement of the switch on the command line can affect which files it affects.) When COPY prompts you for confirmation, answer with a Y for Yes or an N for No. Do not type a RETURN after your answer. For example:

```
.COPY SRCFIL[110,2]=WRKFIL/QUERY (RET)
WRKFIL.BAS to SRCFIL.BAS[110,2]?Y
WRKFIL.LST to SRCFIL.LST[110,2]?Y
WRKFIL.RUN to SRCFIL.RUN[110,2]?Y
WRKFIL.SEQ to SRCFIL.SEQ[110,2]?N
Total of 3 files transferred
```

You may enter a Control-C at any time to prevent further transfers.

ERRORS:

?Cannot find DSK0:SCNWLD.SYS[1,4] or MEM:SCNWLD.SYS

The COPY program needs this file to be able to process wildcard symbols in your file specifications. This message can indicate that SCNWLD.SYS doesn't exist, or that you do not have enough memory to load the file into your memory partition.

?Specification error ^

Your command line is not in proper format. The ^ symbol points to the location in the command line that COPY does not understand.

?Cannot READ Devn - device does not exist
?Cannot READ Devn - device is not mounted

You tried to copy to or from a device that is not listed in the DEVTBL command in your SYSTEM.INI, does not have a driver in area [1,6] of the System Disk, is not file-structured, or is not mounted. ("Devn:" is the device you specified.)

(Changed 30 April 1981)

%Account does not exist - [p,pn]
The indicated PPN does not exist; to create it, you must be logged in under [1,2].

%No file-oriented device corresponding to Devn: is mounted
You specified a device, but left off the unit number. COPY cannot find a logical unit that matches your specification. Try mounting the device.

?Missing output specification
You omitted the equal sign in your COPY command line; COPY couldn't tell which information was your Newfilespec and which your Oldfilespec.

?More than one output specification
You may not supply more than one Newfilespec.

?Files may not be transferred to RES:
You may only add programs to system memory by using the SYSTEM command within your system initializaton command file, SYSTEM.INI.

%Random files can not be transferred to MEM:
You may only put sequential files in your memory partition.

%Not copied - Destination file already exists
You tried to copy to an existing file while the /NODELETE option was in effect.

?You are not logged in under [1,2], can't create [p,pn]
You cannot copy from an account to a nonexistent account unless you are logged in under [1,2]. If you copy to a nonexistent account while logged under [1,2], COPY will create the account with the same password as the account copied from.

?Output MFD is full
The Master File Directory only has room for 64 entries. The transfer in progress would have created a new account, but there is no room in the MFD.

?Device full
There is no more room on the disk.

%Bypassing BADBLK.SYS[1,2]
% BADBLK.SYS exists to prevent bad blocks
% on a device from being allocated, and
% should never be directly accessed.
%No files transferred
You tried to copy the file BADBLK.SYS[1,2], which must never be altered, moved or destroyed.

CHARACTERISTICS:

Because it is a wildcard file command, COPY has advanced wildcarding abilities.

You can use COPY to back up or pack entire disks if you are logged in as System Operator.

(Changed 30 April 1981)

cpmcpy

FUNCTION:

Transfers a copy of a file from a floppy diskette created under the C/PM operating system to an AMOS file-structured device.

HINTS/RESTRICTIONS:

CPMCPY assumes that the data in the C/PM file is in ASCII or binary form; it does no translation of the data. CPMCPY assumes that the C/PM diskette is mounted on C/PM device B: (i.e., AMOS device IMG1:). (C/PM device A: is AMOS device IMG0:.) You must have a copy of the IMG.DVR in area [1,6] of your System Disk. You must have the IMG device defined in your system device table (see the DEVTBL reference sheet).

If you are using an AMS device, you should not copy between floppy devices that are attached to the same floppy controller.

The extension you provide in your C/PM file specification must be three characters. This means that if the C/PM extension is only two characters, you must enter a space as the third character. If the extension is a null extension, you must enter three spaces for the extension.

FORMAT:

```
_.CPMCPY AMOSfilespec=CPMfilespec{/B} ↵
```

where AMOSfilespec is the Filespec you want to assign to the file copy (e.g., DSK1:CBIOS.CPM[110,3]) and CPMfilespec gives the C/PM specification of the file you want to copy (e.g., A:CBIOS.ASM). /B is an optional switch that tells CPMCPY that the data in the file is in binary form, not ASCII.

DEFAULTS:

If you do not specify an extension for the new file, CPMCPY uses the .CPM extension. CPMCPY assumes that the file you are copying contains ASCII data.

OPTIONS:

If you want to copy a file that contains binary data (rather than data in ASCII form), use the /B switch on the command line. For example:

```
_.CPMCPY DSK0:CRDR.PRG[1,4]=A:CRDR.ASM/B ↵
```

(1 October 1979)

OPERATION:

1. Type CPMCPY followed by the AMOS specification of the new file you want to create. Then type an equal sign followed by the C/PM specification of the file on the C/PM diskette. Type a RETURN. For example:

```
.CPMCPY AMS1:FORMS.CPM=A:FORMS.ASM
```

2. You now see a message that looks something like this:

```
Begin transfer....
83 Records read
20 Records written
Transfer completed
.
```

ERRORS:

You may see any of the standard system error messages that can result from invalid device specifications. In addition, if CPMCPY cannot find the specified C/PM file, you see:

```
Begin transfer
CPM file not found in directory search.
.
```

CHARACTERISTICS:

Transfers data in either binary or ASCII form.

To use CPMCPY, you must have IMG.DVR in DSK0:[1,6] and the IMG device must be defined in your system device table.

Returns your terminal to AMOS command level.

cpmdir

FUNCTION:

Displays the directory of a C/PM diskette.

HINTS/RESTRICTIONS:

CPMDIR uses the IMG.DVR driver to access the data on the C/PM diskette; make sure that you have a valid copy of that program in area [1,6] of your System Disk. The IMG device must be defined in your system device table. CPMDIR assumes that the diskette whose directory you want to see is mounted on C/PM device B: (i.e., AMOS device IMG1:). C/PM device A: is AMOS device IMG0:.

FORMAT:

.CPMDIR Devn: ↵

where Devn: is the specification of the diskette (in C/PM format) whose directory you want to see.

OPERATION:

1. Type CPMDIR followed by the specification (in C/PM format) of the diskette you want to read. Then hit RETURN. For example (if the C/PM specification of the diskette is A:), enter:

   .CPMDIR A: ↵

2. You now see a display that looks something like this:

```
[Filename]    [Ext]  [Ex]
  ..VUE       ...COM ...0
  ..EDIT      ...COM ...0
  ..APOLLO    ...ASC ...0
  ........
  End of directory.
.
```

ERRORS:

You may see these error messages when using CPMDIR:

?Cannot READ IMG3: - device does not exist

Make sure that IMG is defined as a valid device on your system and that the IMG.DVR program exists in DSK0:[1,6].

(1 October 1979)

?Cannot READ IMG1: - device not mounted

You have tried to access a valid disk device, but the diskette is not mounted. Use the MOUNT command to mount the diskette.

CHARACTERISTICS:

Requires the IMG.DVR in area [1,6] of your System Disk. Also requires that IMG device is defined in your system device table. (See the reference sheet on DEVTBL.)

Returns your terminal to AMOS command level.

cpy410

FUNCTION:

Copies an AMOS Version 4.4 System Disk update cartridge onto the first fixed platter of a CDC Phoenix hard disk drive.

HINTS/RESTRICTIONS:

CPY410 is a command file that certifies DSK1: of a Phoenix drive, and then copies the contents of DSK0: onto DSK1:. It destroys all data on DSK1:, so make sure you make a back up of everything on DSK1: you want to save before you use CPY410. You usually use it to update your System Disk on the fixed disk by copying the contents of a new System Disk cartridge down onto it.

NOTE: You may only use CPY410 to copy an AMOS Version 4.4 or later System Disk. (The SYSTEM command will tell you which version of AMOS you are running under.)

The disk off of which the system is running (i.e., the System Disk) is ALWAYS called DSK0:. When you reset or turn on the system, if the cartridge on the system contains account [1,4] and the files SYSTEM.MON[1,4] and SYSTEM.INI[1,4], then the system uses the cartridge as the System Disk. (In this case, the cartridge is DSK0: and the first fixed disk becomes DSK1:.) If these files do not appear on the cartridge, the system tries to boot up off the first fixed disk; if it is successful, THAT disk becomes DSK0: and the cartridge becomes DSK5:.

If the Phoenix disk drive contains a System Disk cartridge, when you reset or turn on the system, the system boots off that cartridge, regardless of whether or not the fixed disk is also a System Disk. In this case, CPY410 copies the contents of the cartridge down onto the first fixed disk, DSK1:. Remember, when you mount the System Disk cartridge and reset the system, the cartridge becomes DSK0: and the fixed disk becomes DSK1:.

IMPORTANT NOTE:

If you bring up the system with a non-System Disk cartridge mounted and then mount a System Disk cartridge without resetting the system, it continues to run off the fixed disk. In this case, if you use CPY410, it copies everything from fixed disk DSK0: to fixed disk DSK1:, writing over all data on DSK1:. Unless that is precisely what you want to do, make very sure that you are running off the cartridge before using CPY410.

(Changed 1 May 1980)

If you are not sure whether or not you are running off the cartridge or the fixed disk, type SYSTEM followed by a RETURN. If you are running off the cartridge (that is, if the cartridge is DSK0:), SYSTEM tells you so:

```
.SYSTEM (RET)

*** SYSTEM IS RUNNING FROM CARTRIDGE DISK ***
```

FORMAT:

```
.CPY410 (RET)
```

OPERATION:

1. Place the System Disk update cartridge in the Phoenix drive. Reset the system by pushing the RESET button.

2. Type SYSTEM followed by a RETURN. If you do not see:

   ```
   *** SYSTEM IS RUNNING FROM CARTRIDGE DISK ***
   ```

   you are running off the fixed disk; CPY410 will copy from the fixed disk DSK0: to fixed disk DSK1:. If are running off the cartridge, type CPY410 followed by a RETURN:

   ```
   .CPY410 (RET)
   ```

3. You now see the following message. (Type a RETURN when you are sure that the cartridge is DSK0: and that all information that you want to save on DSK1: is backed up.)

   ```
   .LOG DSK0:[1,2]
   This command file will certify DSK1: and copy a system onto
   it.  The certification process destroys the contents of DSK1:.
   When you are certain that everything is ready, type a return.
   ```

4. Now CPY410 certifies DSK1: by using the CRT410 command, specifying a maximum of 15 bad tracks.

5. After the certification, CPY410 displays this message:

   ```
   Starting COPY process
   ```

6. Now CPY410 uses the DSKCPY program to copy all files in all accounts from DSK0: to DSK1:.

7. When the copy process is done, CPY410 returns you to AMOS command level.

(Changed 1 May 1980)

ERRORS:

?Device has exceeded maximum number of errors
The certification program, CRT410, found more than 15 bad tracks on DSK1:. CRT410 stops the certification process.

You can also see any of the error messages generated by the DSKCPY command. (See the DSKCPY reference sheet.)

CHARACTERISTICS:

CPY410 is a command file in the System Command File library account, DSK0:[2,2].

Destroys all data on DSK1: in the process of certification.

Uses the DSKCPY command to copy all files from logical unit DSK0: to logical unit DSK1:. Verifies the copy.

cpy500

FUNCTION:

Copies a System Disk cartridge onto the fixed platter of a Hawk hard disk drive.

HINTS/RESTRICTIONS:

CPY500 is a command file that formats and initializes DSK1:. Then it uses the COPY command to copy the contents of logical unit DSK0: to DSK1:. It destroys all data on DSK1: before it copies the new data over, so before you use CPY500 make sure you make a backup copy of all data on DSK1: that you want to keep. Because CPY500 uses the COPY command (instead of DSKCPY) it is much faster than SYSCPY. However, unlike SYSCPY, CPY500 does NOT verify the disk copy.

The disk off of which the system is running is ALWAYS called DSK0:. (DSK0: is usually the fixed platter.) When you reset the system or turn it on for the first time, if the cartridge that is on the system contains account [1,4] and the files SYSTEM.MON[1,4] and SYSTEM.INI[1,4], the system recognizes that cartridge as a System Disk (and therefore as DSK0:).

NOTE:

> If the Hawk disk drive contains a System Disk cartridge when you reset or turn on the system, the system boots off that cartridge regardless of whether the fixed platter is also a System Disk. When the system boots off a System Disk cartridge, that cartridge is logical unit DSK0:.

Because the CPY500 command copies the contents of DSK0: to DSK1:, you can use it to update your fixed disk with a new software release by copying from the System Disk update cartridge to the fixed platter (your old System Disk). Remember, when you mount the System Disk cartridge and reset the system, the cartridge becomes DSK0: and the fixed platter becomes DSK1:. If you have any doubts about whether you are running off the cartridge or the fixed disk, type SYSTEM followed by a RETURN. If you are running off the cartridge, the system tells you so:

*** SYSTEM IS RUNNING FROM CARTRIDGE DISK ***

(If you bring up the system with a data cartridge mounted and then mount the System Disk update cartridge without resetting the system, it continues to run off the fixed disk. In this case, if you use CPY500, it copies everything from the fixed platter to the cartridge, wiping out everything on the System Disk update cartridge. Be careful.)

(1 October 1979)

FORMAT:

.CPY500 ↵

OPERATION:

1. Place the System Disk update cartridge in the Hawk drive. Reset the system by pushing the RESET button. Type SYSTEM followed by a RETURN. If you do not see:

   *** SYSTEM IS RUNNING FROM CARTRIDGE DISK ***

   you are running off the fixed disk. If you are running off the cartridge, type CPY500 followed by a RETURN:

   .CPY500 ↵

2. Now when you see the following message, type a RETURN:

   This command reformats DSK1: then copies all of DSK0: onto it.
   It is normally used to transfer the system pack.

   Enter CR when ready to execute: ↵

3. CPY500 now mounts DSK1: (the fixed disk) and formats it via the FMT500 program. CPY500 initializes DSK1: and copies every file on DSK0: over to DSK1:.

4. When CPY500 has finished copying the disk, it signals you by ringing your terminal bell five times. Then you see the message:

   System pack transfer is complete

ERRORS:

If the system encounters a hard disk-error, you can see the standard AM-500 error messages. For example:

?Cannot READ DSK0:MOUNT.PRG[1,4] - device error

Check with the System Operator for help.

CHARACTERISTICS:

Use to update fixed disk on a Hawk hard disk drive by copying all files from a System Disk update cartridge.

(You may also use CPY500 to copy all files from the fixed disk over to a cartridge if the system is running off the fixed disk rather than the cartridge.)

Returns your terminal to AMOS command level.

create

FUNCTION:

Creates a random file of specified size.

HINTS/RESTRICTIONS:

A random file (also known as a contiguous file) is one in which the blocks that make up the file physically adjoin on the disk (as opposed to a sequential file, in which the disk blocks that make up the file may be scattered across the disk). (See Chapter 5, "Introduction to Files," in Introduction to AMOS, (DWM-00100-65), for a discussion of random and sequential files.)

Since random files may not be expanded once they are allocated on the disk, programs that increase the size of files (such as text editors) work only with sequential files. You will be most likely to use random files for applications such as BASIC data files, which never need to expand in size.

FORMAT:

.CREATE Filespec,Size RET

where Filespec selects the file you want to create and Size is the number of disk blocks you wish to allocate to that file.

DEFAULTS:

If you omit portions of the file specification, CREATE assumes the device and account you are logged into and a .DAT extension.

OPERATION:

1. Type CREATE followed by the specification of the file you want to create and the number of blocks to allocate to that file. Then type a RETURN. For example:

   .CREATE CUSIDX.NEW,10 RET

   The command above creates the file CUSIDX.NEW in the account and device you are logged into. The file is ten disk blocks long.

2. When CREATE is finished creating the file, it tells you so. For example:

(Changed 30 April 1981)

```
.CREATE DSK3:MTNDEW[110,11,5 RET
MTNDEW.DAT created
.
```

ERRORS:

If you do not enter the CREATE command line in the proper format (e.g., if you type CREATE followed by a RETURN), you see:

```
?Command error
```

You can also see standard AMOS error messages. For example:

```
?Cannot OPEN Filespec - device full
```

AMOS wasn't able to find enough contiguous disk blocks to allocate your file. Even though the total number of disk blocks free on the disk may be greater than the number of blocks you want to assign to your random file, AMOS cannot create your file unless it finds enough blocks that physically adjoin on the disk. Ask the System Operator to pack the disk to consolidate the free disk blocks into a contiguous group on the disk.

CHARACTERISTICS:

Creates random files of specified size.

If you omit portions of a file specification, CREATE assumes the device and account you are logged into and a .DAT extension.

(Changed 30 April 1981)

crt410

FUNCTION:

Certifies the disk media of a device that runs under control of the AM-410 Hard Disk Controller. Also takes the place of the FMT410 command.

HINTS/RESTRICTIONS:

Because of the high data density of the disks that run with the AM-410 (e.g., the CDC Phoenix), media flaws are a more likely possibility than on more conventional drives (e.g., the CDC Hawk). Therefore, you must run CRT410 on each logical unit of every device that runs under control of the AM-410.

CRT410 certifies a logical unit by writing and verifying data in every block of the device, so make sure that you back up any data you need on that device before you certify it. Because CRT410 destroys any data on the disk it is certifying, CRT410 requires that you be logged into the System Operator's account, [1,2], before certifying a disk.

NOTE: CRT410 takes the place of the disk formatting program, FMT410, because it formats the disk as well as certifying it.

CRT410 creates account [1,2] on the disk you are certifying. Then it creates a file BADBLK.SYS[1,2] that contains a list of all bad disk tracks on the certified device. CRT410 assigns alternate tracks for these bad tracks, thereby making the fact that some tracks are bad completely "transparent" to the user of the device. For information on the program that you can use to display the BADBLK.SYS file, see the BADBLK reference sheet.

You do not have to mount a logical unit before certifying it.

IMPORTANT NOTE:

You may use this certification program ONLY on devices that run under control of the AM-410. CRT410 communicates directly with the AM-410 without going through the Phoenix driver program. Therefore, you MUST NOT run CRT410 at the same time as other programs that access devices that run under control of the AM-410.

FORMAT:

.CRT410 Devn: RET

where Devn: is the specification of the device you want to certify.

(Changed 1 May 1980)

OPERATION:

1. Type CRT410 followed by the specification of the device you want to certify; then type a RETURN. For example:

   .CRT410 SMD5: (RET)

2. CRT410 now warns you:

   CAUTION: This program writes to all blocks.

   In other words, make sure that you have backed up any data on the disk that you may need; CRT410 destroys all data on the disk. You can interrupt the program at this point by typing a Control-C.

3. CRT410 now creates a file named BADBLK.SYS in account [1,2] of the disk you are certifying. This file will hold a list of all bad disk blocks that CRT410 finds.

4. Now CRT410 asks you several questions:

   a. Enter maximum acceptable number of bad tracks:
      Give the number of bad tracks that you will accept on the disk. When CRT410 finds more than this number, it tells you so and returns you to AMOS command level.

   b. Display current track? (Y or N):
      If you want CRT410 to tell you as it certifies each track, enter Y; otherwise, enter an N. NOTE: Asking CRT410 to display the number of the track that it is verifying greatly increases the time it takes to certify a disk.

   c. Enter serial number (10 char. max):
      You may optionally give CRT410 a ten-character alphnumeric I.D. for the logical unit it is certifying. CRT410 writes this identifier into the BADBLK.SYS file.

5. After you answer the questions above, CRT410 begins to certify the disk. You see this message:

   Begin certification of Devn:

   where Devn: is the specification of the device you are certifying.

6. If you asked CRT410 to tell you its current track position, you see a list of messages that take this form:

   Current track is: 12

(Changed 1 May 1980)

7. When CRT410 encounters a track it cannot verify, it tells you so. For example:

```
?Track 79 did not verify
```

8. When it finishes certifying the disk, CRT410 tells you how many bad tracks it found. For example:

```
?2 bad tracks detected
Certification complete
.
```

9. CRT410 writes a list of all bad tracks into BADBLK.SYS[1,2] and computes and stores in the file a hash total for BADBLK.SYS. (Other programs can check this hash total against the contents of the file to make sure that BADBLK.SYS contains undamaged data.)

ERRORS:

You can see the following error messages when using CRT410:

?You must be logged into PPN [1,2] to run CRT410.
Because it writes data into every block on the disk, CRT410 is a rather dangerous command to use. You must be logged in as System Operator to use it.

?15 bad tracks is maximum
You tried to give a number larger than 15 as the number of bad tracks you will accept on the disk you are certifying.

?Track 0 did not verify. (First track must verify.)
CRT410 could not verify track #0. Because the first track MUST verify for the certification to continue, CRT410 now stops the certification and returns you to AMOS command level.

?Track n did not verify
CRT410 marked track #n as a bad track in the BADBLK.SYS file.

?Device has exceeded maximum number of errors
CRT410 found more bad tracks than the value you specified as the maximum number of bad tracks you will accept. CRT410 now stops the certification and returns you to AMOS command level.

?S100 data transfer error
An error occurred with the AM-410 controller. CRT410 stops the certification and returns you to AMOS command level. A number of these errors can indicate hardware problems.

(Changed 1 May 1980)

?Nonexistent device

Your device specification on the CRT410 command line is invalid. The system believes that the device does not exist. Check your spelling and try again.

?Certification incomplete

You typed a Control-C or some other event happened to interrupt the certification. CRT410 now intentionally writes a bad hash total to the BADBLK.SYS file so that other programs know that the data in the file is incomplete and not to be trusted.

CHARACTERISTICS:

Communicates directly with the AM-410, so do not run at the same time as other programs which access devices that run under control of the AM-410. Use only on devices that run with the AM-410.

You must be logged in as System Operator to certify a disk.

Creates account [1,2]; then creates a file BADBLK.SYS[1,2] that contains a list of all bad tracks on the disk.

Returns your terminal to AMOS command level.

date

FUNCTION:

Sets or displays the system date in American or European format.

If your system contains an AM-120 Auxiliary I/O Controller board, DATE also reads and sets the clock/calendar on the AM-120.

If used from within the system initialization command file, sets the system date from the AM-120 Auxiliary I/O Controller board and optionally selects European date format and/or automatic date rollover at midnight.

HINTS/RESTRICTIONS:

To set the system date and to reset the date on the AM-120 board, you must be logged into the System Operator's account, [1,2], or must use DATE in response to a :K command that appears before the final SYSTEM command in the system initialization command file.

You must use DATE from within the system initialization command file to set the system date from the AM-120 board.

You must be logged into the System Operator's account or use DATE from within the system initialization command file to request date rollover at midnight (available only if your system contains an AM-120 board) or to select European date format.

DATE allows you to set and display the system date in either American or European format. (A date in American format looks like this: January 1, 1982; a date in European format looks like this: 1 January 1982.)

Certain functions (e.g., day of the week, automatic date rollover) require that your system contain an AM-120 Auxiliary I/O Controller board.

The optional feature that tells DATE to automatically rollover the date at midnight requires approximately 350 extra bytes of system memory.

Sets the date for the AlphaAccounting package. DATE is not re-entrant, but is re-usable.

(Changed 30 April 1981)

FORMAT:

```
     _.DATE (RET)
or:
     _.DATE MM/DD/YY (RET)
or:
     _.DATE DD/MM/YY{/E} (RET)
or:
     DATE{/E{/R}}
```

The first format displays the current system date (if it has been previously set). You may use DATE in this format when you are logged into any account.

The second and third formats set the system date, where MM is month, DD is day, and YY is year, and /E is an optional switch that tells DATE to use European format. (If your system contains an AM-120 Auxiliary I/O Controller board, these formats also allow you to set the AM-120 calendar.) You must be logged into [1,2] to use DATE in these formats or must be using the DATE command as response to a :K command which appears before the final SYSTEM command in the system initialization command file.

You may use the fourth format from within your system initialization command file to set the system date from the AM-120 board, and may optionally specify switches that select European format for date display and automatic date rollover at midnight.

DEFAULTS:

If the system initialization command file has not specified European date format, the default display mode is American format. You may use the /E switch to change the display mode to European format if you are logged into [1,2].

OPTIONS:

You may select the following options by specifying the appropriate switches:

/E Tells DATE to display the date in European format, and that all subsequent date entries will be in European format. Must be logged into [1,2] to use /E.

/R Automatic rollover. Tells DATE to automatically increment the date and day of the week at midnight. For example, at 24:00:01, October 5th becomes October 6th, and Tuesday becomes Wednesday. Uses an extra 350 bytes of system memory for a module named DATROL.SYS. You may only use the /R switch when you use DATE from within the system initialization command file before the final SYSTEM command.

(Changed 30 April 1981)

OPERATION:

At AMOS command level when logged into any account:

1. To display the date, type DATE followed by a RETURN. For example:

```
.DATE (RET)
April 4, 1945
```

or:

```
.DATE (RET)
4 April 1945
```

If your system contains an AM-120 board, DATE also displays the day of the week. For example:

```
Wednesday, November 26, 1980
```

At AMOS command level from account [1,2]:

1. To set the day of the year, type DATE and the date in the form given in FORMAT, above. Type a RETURN. For example:

```
.DATE 10/21/84 (RET)
```

or:

```
.DATE 31/06/82 (RET)
```

(if European format mode has been set).

2. If you want to specify European format, use the /E switch:

```
.DATE 31/05/83/E (RET)
```

3. If your system contains an AM-120 board, after you have set the system date, DATE asks:

```
Do you wish to reset the AM-120 board also?
```

Enter Y or N followed by a RETURN. If you enter N, DATE sets the system date without changing the date stored by the AM-120 board. If you answer Y, DATE asks for the day of the week:

```
Enter day of week (Mon=1, Tue=2, etc.):
```

Enter the appropriate single digit that selects the current day of the week. Then type a RETURN.

Within the SYSTEM.INI file:

1. To reset the system date from the AM-120 board every time the system is rebooted, include the DATE command within your

system initialization command file before the final SYSTEM command. For example:

```
SYSTEM TRM.DVR
DATE
SYSTEM
```

2. If you wish DATE to display and accept the date in European format, include the /E switch. If you wish DATE to automatically update the date and day of the week at midnight, use the /R switch. For example:

```
SYSTEM TRM.DVR
DATE/E/R
SYSTEM
```

3. Note that the /R option is only available when you are using the DATE command from within the SYSTEM.INI file.

ERRORS:

You may see the following error messages when using DATE:

?System date has not been set
You may not display the date until an initial date has been entered using the DATE command.

?Use format MM/DD/YY
You did not enter the date in the proper format; refer to the FORMAT section, above. DATE is using American format for date entry and display, so enter the date as month/day/year-- for example:

.DATE 12/31/85 RET

?Use format DD/MM/YY
You did not enter the date in the proper format; refer to the FORMAT section, above. DATE is using European format for date entry and display, so enter the date as day/month/year-- for example:

.DATE 31/11/82 RET

?Day of week must be 1-7
You did not enter a number between 1 and 7 inclusive when DATE asked for the day of the week.

?Please enter Y or N
You may only enter Y or N to the question "Do you wish to reset the AM-120 board also?".

?You must be logged into [1,2] to reset the date

(Changed 30 April 1981)

You must be logged into the System Operator's account, [1,2], or must use DATE in response to a :K command that appears before the final SYSTEM command in the system initialization command file.

?The R option is valid only within the SYSTEM.INI file prior
? to the final SYSTEM command

You may not use the /R switch when you use DATE at AMOS command level.

?Error occurred during GETMEM in roll-over setup

There was not enough memory available to be able to allocate the date rollover module in system memory. Therefore, no date rollover at midnight will occur. Check with the System Operator.

CHARACTERISTICS:

Sets and displays the system date in American or European format. The default format is American.

If your system contains an AM-120 Auxiliary I/O Controller board, DATE makes available some extra features, such as automatic date rollover at midnight, day of the week, and so on.

DATE is not re-entrant, but is re-usable.

(Changed 30 April 1981)

ddt

FUNCTION:

Symbolic debugger. Allows you to examine and modify your assembly language program in memory as it executes.

HINTS/RESTRICTIONS:

NOTE: DDT echoes Escapes as dollar signs. In the discussions that follow, we represent an Escape with a $ symbol. For example, when you see the characters:

```
$B
```

you know that you are supposed to type an Escape followed by a B. DDT does not use RETURNs as command delimiters.

DDT automatically expands the program you are examining to accommodate patches.

Because DDT runs in terminal image mode, the standard AMOS line editing commands do not work when you are in DDT. (For example, a RUBOUT does not erase the last character; instead, a RUBOUT cancels the entire current command and echoes as an XXX followed by a tab.)

Both input and output expressions may be either in symbolic or numeric form. All numeric input is ALWAYS in octal. Use the SET HEX command at AMOS command level if you want to see data displayed in hexadecimal form.

You may reference local symbols. First open the location of the non-local symbol that occurs before the local symbol you want to reference. Then reference the local symbol.

DDT allows you to single-step through an assembly language program you have previously assembled using MACRO. If you want to enter symbolic input (e.g., labels), you must create a symbol table file for your assembled program using the SYMBOL program before you use DDT. (See the MACRO and SYMBOL reference sheets.)

For information on using DDT, MACRO, SYMBOL, LIB, GLOBAL and LINK, refer to the AMOS Assembly Language Programmer's Reference Manual, (DWM-00100-43). (Chapter 13 of that manual contains DDT operating instructions.)

(Changed 30 April 1981)

FORMAT:

.DDT Filespec (RET)

where Filespec selects the file that contains the program you want to debug.

DEFAULTS:

If you do not specify a file extension, DDT assumes that the file is a .PRG file. The default DDT mode is program-relative mode.

OPERATION:

1. Type DDT followed by the specification of the file you want to debug. For example:

   .DDT DSK0:CREATE[100,2] (RET)

   DDT loads the file into your memory partition, but first it checks to see if the file is already in memory. If it is, DDT deletes it and replaces it with a fresh copy.

2. Now DDT looks in the account in which your file resides for a symbol table file for that program. If such a file exists, DDT loads it into memory (enabling you to reference memory addresses in symbolic rather than numeric form).

3. Next DDT displays the base memory address and the size in bytes of the program. For example:

   ```
   PROGRAM BASE IS 56670
   PROGRAM SIZE IS 2346
   ```

4. Now you can begin to enter DDT commands. Most entries to DDT consist of a numeric or symbolic argument followed by a DDT command. These commands are usually one or two symbols such as a slash, /, or an Escape followed by a character (e.g., $B). To modify the contents of a location, you must first use a command that "opens" it.

   DDT operates in several modes: 1. program-relative mode; 2. absolute mode; and, 3. register mode.

   The default mode is program-relative mode. This mode assumes that addresses are relative to the base address of the program you are debugging.

(Changed 30 April 1981)

Absolute mode assumes that all addresses are absolute memory addresses. Enter absolute mode by typing a TAB; leave this mode by use the $R command.

Register mode assumes that expressions refer to the registers instead of memory locations. Enter register mode by using any of the special symbols that represent the registers: R0-R5, SP, or PC.

Any of these symbols followed by a command that opens a location enter you into register mode. Leave register mode by using the $R command.

5. To exit, type a Control-C. DDT now returns you to AMOS command level:

```
^C
.
```

Your modified program is still in memory. To save the debugged program, use the SAVE command. For example:

```
.SAVE DSK0:CREATE.PRG (RET)
```

NEVER save a program that contains breakpoints. Running such a program could result in severe problems.

COMMAND SUMMARY:

/	Open location or register (can take numeric or symbolic argument; e.g., n/, tag/, or reg/).
RETURN	Close a location (can take numeric or symbolic argument specifying data to place in location; e.g., nRETURN, tagRETURN).
Line-Feed	Open next location.
=	Display value in octal (can take symbolic or numeric argument).
^	Open previous location.
@	Open location indirectly.
TAB	Open absolute location indirectly. (Enter absolute mode.)
$G	Start program at relative address zero. DDT waits for one line of input after the $G cnmmand, which it passes to the program and then it executes the program.
$B	Set or list breakpoints. Can take numeric or symbolic argument (e.g., $B, nB, tagB, tag$nB).
$C	Clear breakpoints. Can take symbolic or numeric argument (e.g., $C, nC, tagC, tag$nC).
$P	Proceed from a breakpoint.
$X or \	Execute single instructions. Valid only after a breakpoint has been reached.

(Changed 30 April 1981)

$R	Enter program-relative mode.
$D	Display data in decimal. Takes either argument specifying how many locations to translate or an argument that gives the expression to translate (e.g., $xD or exp$D).
$=	Display data in octal. Uses same format as $D command.
$H	Display data in hex. Uses same format as $D command.
$*	Display data in current location in unpacked RAD50.
$"	Display data in current location as two ASCII characters.
$#	Display data in current location as two eight-bit bytes.
$A	Display a string of ASCII characters in current location or at location of symbolic argument. (String ends with a null byte.)
:	Define new symbol. Value attached to symbol is last location examined (e.g., tag:).
%	Examine register contents. Takes register argument followed by equal sign. (e.g., %R0=).
^C	Exit DDT.
RUBOUT	Cancel current line of input.

ERRORS:

If DDT does not understand a command, it displays a ? symbol. Compare your command with the instructions in the AMOS Assembly Language Programmer's Reference Manual, (DWM-00100-43), Section 9.4, to see if the format you used was correct.

You can also see these error messages:

Filespec NOT FOUND

DDT wasn't able to find the file you specified. Check your spelling and make sure you specified the correct device and account.

CAN'T SINGLE STEP THROUGH SVC

You cannot use the $X command to single-step through a supervisor call. You must skip over the call by placing a breakpoint after the call and its arguments. Then use the $P command to skip to that location. At that point you can continue single-stepping.

?DDT INTERNAL BUSERR

A bus error occurred within the DDT program itself. This error was not caused by your program.

?BUSERR AT MONITOR PC nnnn

A bus error occurred, but was not caused by DDT. Your program is probably at fault. The number that appears in the message tells you what memory address was loaded into the Program Counter when the error occurred.

(Changed 30 April 1981)

CHARACTERISTICS:

Allows you to set breakpoints, modify and examine contents of registers, program locations and absolute memory addresses.

(Changed 30 April 1981)

del

FUNCTION:

Erases modules from your memory partition.

HINTS/RESTRICTIONS:

When you load a file into memory from the disk (e.g., a program), that copy in memory is called a module. DEL erases such modules from your memory partition (NOT from the disk). You may use the wildcard symbols * and ? in your file specifications.

FORMAT:

.DEL Filespec1{,Filespec2,.....FilespecN} ↵

where Filespec1,Filespec2,... is a list of valid file specifications that select the modules you want to erase from your memory partition.

DEFAULTS:

DEL assumes a file extension of *.

OPERATION:

1. Type DEL followed by one or more file specifications; then type a RETURN. For example:

```
.DEL CREATE ↵
CREATE.PRG
CREATE.SYM
```

   DEL erases from your memory partition the modules selected by your specifications; then it lists them.

2. You can use standard wildcard symbols in your file specifications. For example:

```
.DEL NEWPR* ↵
NEWPR1.SBR
NEWPRT.PRG
```

ERRORS:

DEL displays no error messages. You'll know that it wasn't able to find the modules you specified if it does not report back with a list of modules that it erased.

CHARACTERISTICS:

When it erases modules, it shifts any remaining modules up in memory if they have memory addresses greater than the deleted modules.

Returns your terminal to AMOS command level.

devtbl

FUNCTION:

As part of the system initialization command file, defines the devices used on the system. At AMOS command level, DEVTBL tells you what devices are defined in the system initialization command file.

HINTS/RESTRICTIONS:

When used in the system initialization command file, SYSTEM.INI, DEVTBL tells the system which device driver programs to include in the system monitor. As the system processes the DEVTBL command line, it builds a device table in memory. The file system consults the device table for device assignments. The DEVTBL command line thus defines the devices that you can use on the system. Every time you add a new device to the system (e.g., a new type of disk), you must add it to the system device table. Remember to also define bitmaps in the SYSTEM.INI for every new disk device (see the BITMAP reference sheet).

NOTE: Several system commands require that you have both MEM and RES defined in your SYSTEM.INI as devices.

As a user command at AMOS command level, DEVTBL lists the devices defined on the system, tells whether the devices are sharable or nonsharable, and tells you if alternate track tables are assigned for those devices.

FORMAT:

.DEVTBL RET

To find out what devices are defined on the system, or:

DEVTBL Devn1{,Devn2,.....DevnN}

to define the system device table in the system initialization command file, where Devn is the one or more devices you want to incorporate device drivers for in the system monitor.

OPERATION:

Using DEVTBL as a user command at AMOS command level:

1. Type DEVTBL followed by a RETURN:

.DEVTBL RET

(Changed 1 May 1980)

The display you see tells you what devices have been defined on the system. For example:

```
DSK0    (sharable)  (alternate)
DSK1    (sharable)  (alternate)
DSK2    (sharable)  (alternate)
DSK3    (sharable)  (alternate)
DSK4    (sharable)  (alternate)
DSK5    (sharable)  (alternate)
HWK0    (sharable)
HWK1    (sharable)
MEM0    (sharable)
RES0    (sharable)
TRM0    (sharable)
MTM0
SND0    Assigned to JOB3
```

where:

(sharable) — This means that the device was defined as sharable (e.g., the device can be accessed by more than one user at a time; for example, a disk). (Nonsharable devices are those devices that only one user can access at a time, such as a printer.)

(alternate) — This means that an alternate track table has been assigned. Certain disk devices, such as the Phoenix, use an alternate track table to handle media flaws.

assigned to — This means that a non-sharable device is currently being used by the specified job.

The devices defined in the display above are: Logical devices zero through five of the System Device (DSK0, DSK1, DSK2, DSK3, DSK4 and DSK5) where the System Device is a Phoenix disk drive; logical devices HWK0 and HWK1, where a HAWK drive is on the system; the MEM device that allows you to use your memory partition much as if it were an ordinary device (MEM0); the RES device (RES0) that allows you to refer to system memory (e.g., .DIR RES:); the TRM driver (TRM0) that allows you to use terminals as devices (e.g., .COPY TRM:=TEXT.LST); a printer (MTM0) that requires its own special device driver program; and SND0, representing a printer or any other nonsharable device that is currently being used by a particular job.

Using DEVTBL as a system initialization command:

1. To define a system device table or to add items to the device table, edit the system initialization command file, SYSTEM.INI, with one of the text editors.

2. Place the DEVTBL command line directly after the terminal definition (TRMDEF) commands.

3. The system already knows about device DSK0: (the device off of which the system booted), so do not include DSK0: in the system device table. Every other device for which the system uses a device driver program (and each logical unit of every device) must have an entry on the DEVTBL command line.

4. Enter DEVTBL followed by the list of devices you want to define in the system device table. Separate the entries with commas. For example:

```
DEVTBL DSK1,AMS0,AMS1,TRM,MEM,RES,/MTM
```

   The first three letters of each entry identifies the device driver program the system will use to access that device. The optional fourth character of each entry identifies the unit number of that device.

   The command line above identifies a device table for a System Device that contains two logical units, DSK0: and DSK1:. (Remember that the system already knows about DSK0:.) We also define a two-drive (AMS0: and AMS1:) floppy device that uses the AMS driver. We also tell the system that we are going to want to use the MEM driver (allowing us to use our memory partition as another device), the TRM driver (allowing us to use a terminal as a device), and the RES driver (allowing us to reference system memory as a device). All of these devices are sharable (that is, more than one user can use them at the same time). If you want to define any devices that are NOT sharable (e.g., a printer or a paper tape punch), the entries for such devices must appear at the end of the DEVTBL command line, after a single slash. (For example, the example above defines a Multiterm printer, MTM, as a nonsharable device. If you had more nonsharable devices, you would enter them after the MTM entry.

5. If you have more device table entries than will fit on one line, you may include several DEVTBL command lines in your SYSTEM.INI. These DEVTBL command lines must follow one another directly, with no intervening commands.

(Changed 1 May 1980)

ERRORS:

?No device table allocated
You tried to use DEVTBL at AMOS command level, but the system initialization command file for your system does not contain a DEVTBL command line, so the system does not have a device table.

%Device driver xxx.DVR not found
The device driver DSK0:xxx.DVR[1,6] was not found. This is a warning message intended to inform you that the device driver is not in the default library. If you plan to load the device driver into your own memory partition or system memory, then you can disregard this message. If not, perhaps you misspelled the device name in the DEVTBL command line.

CHARACTERISTICS:

Acts as both a user command and a system initialization command.

Returns your terminal to AMOS command level.

diag2

FUNCTION:

Tests floppy disks by performing read/write tests.

HINTS/RESTRICTIONS:

This program tests a floppy disk device, the disk controller board, and the diskette in the drive. For this test to be effective, you must know for certain that two of these pieces of equipment are good. For example: to test a drive, make sure that the controller and the diskette you use are good. DIAG2 does NOT verify write operations and it does not destroy the data on your diskette.

If you use the SET DSKERR command before running DIAG2, DIAG2 reports all soft errors that occur; otherwise, it reports only hard errors. (For information on hard and soft errors, and on hard and soft error messages, see Section 4.0, "Disk Diagnostic Tests," in Disk Maintenance Procedures for the System Operator, (DWM-00100-40), in the System Operator's Information section of the AM-100 documentation packet.)

FORMAT:

.DIAG2 Devn:↵

where Devn: is the specification of the device you are testing.

OPERATION:

1. Type DIAG2 followed by a device specification; then type a RETURN. For example, to test a disk in Drive Zero of a floppy device that handles disks in AMS format, enter:

   .DIAG2 AMS0:↵

2. DIAG2 waits until you tell it that you are ready. Then it performs four different tests and lists any errors:

```
.DIAG2 STD1:↵
Hit return when ready
Test 1 - track 0 read/write
Test 2 - track 76 read/write
Test 3 - random seek-verify 500 times
Test 4 - speed seek tracks 0 and 76 10 times
EXIT
.
```

ERRORS:

(1 October 1979)

DIAG2 displays no error messages of its own, but you may see some of the standard system error messages if you try to test a device that does not exist or is not mounted.

DIAG2 reports hard errors (and soft errors if you have used the SET DSKERR command). A hard error message takes the form:

```
?Cannot READ Filespec - device error
```

A soft error message may look something like this:

```
CRC Error - AMS1: Record 145
CRC Error - AMS1: Record 145
CRC Error - AMS1: Record 145
CRC Error - AMS1: Record 145
CRC Error - AMS1: Record 145
CRC Error - AMS1: Record 145
```

CHARACTERISTICS:

Assumes that two of the three components it is testing (i.e., the drive, the controller, and the diskette) are good.

Returns your terminal to AMOS command level.

diag3

FUNCTION:

Tests memory by writing patterns of ones and zeros into memory and reading them back, verifying the pattern as it does so. DIAG3 also performs a leak-down test.

HINTS/RESTRICTIONS:

Allows you to specify which memory locations to test from above DIAG3 itself to the last contiguous memory address in the block. You may restart the test by typing a Control-C. You may only end the test by resetting the computer.

DIAG3 performs a 10-second leak-down test on dynamic memories. That is, it writes a block of ones into memory, waits a while, then returns to that block to see if the ones are still there.

FORMAT:

.DIAG3 ↵

DEFAULTS:

If you do not supply a starting address, DIAG3 begins at the first free memory location above itself; if you do not supply an ending address, DIAG3 ends at the last contiguous memory address.

OPERATION:

1. Type DIAG3 followed by a RETURN:

.DIAG3 ↵

DIAG3 now asks you for the starting and ending addresses of the test area. Then it repeats them back to you (or, if you replied to its questions with carriage returns, it displays the default addresses it is going to use).

```
.DIAG3 ↵
Starting address?2000 ↵
Ending address?3000 ↵
Starting address:2000
Ending address:3000
```

At the end of each pass, DIAG3 prints the number of the pass and the number of errors that have occurred. For example

END OF PASS 1 - ERROR COUNT IS 0

2. To interrupt the test and start it over, type a Control-C (hold down the CONTROL key while you type a C). To end the test, reset the machine.

ERRORS:

DIAG3 lists the memory addresses where it finds errors. It also tells you the data that is in those addresses and the data that was supposed to be there. For example:

```
END OF PASS 3 - ERROR COUNT IS 1
[ERROR AT ADDRESS 137776 - WAS 005420 SHOULD BE 005421]
```

CHARACTERISTICS:

You may only end DIAG3 by resetting the system.

DIAG3 can test memory locations outside of your memory partition from the location of DIAG3 itself up to the last contiguous memory address.

FUNCTION:

Performs a thorough test of your memory partition by writing data patterns into memory and then reading and verifying those patterns.

HINTS/RESTRICTIONS:

In addition to the normal memory testing procedure, DIAG4 performs a leak-down test on dynamic memories. (That is, it writes a block of ones into memory, waits for a while, and then returns to that block to see if the ones are still there. This checks to make sure that the memory locations have held their charge.)

FORMAT:

.DIAG4 ↵

DEFAULTS:

If you do not specify beginning and ending memory addresses, DIAG4 uses the smallest and largest addresses in your memory partition.

OPERATION:

1. Type DIAG4 followed by a RETURN:

```
.DIAG4 ↵
```

2. DIAG4 asks you for the starting and ending memory address of the block of memory you want to test. DIAG4 then repeats these addresses back to you:

```
.DIAG4 ↵
What starting address? 140000 ↵
What ending address? 177376 ↵
Starting address: 140000
Ending address: 177376

END OF PASS 1 - ERROR COUNT IS 0
END OF PASS 2 - ERROR COUNT IS 0
^C
.
```

3. If you do not specify a starting or ending address (that is, you type a RETURN after each question), DIAG4 assumes that you want to test your entire memory partition, and it uses the minimum and maximum memory addresses in your memory partition:

```
.DIAG4
What starting address?
What ending address?
Starting address: 55034
Ending address: 177376
^C
.
```

4. Each time DIAG4 cycles through memory it lets you know that it has completed a pass:

END OF PASS 34 - ERROR COUNT IS 0

5. To end the test, type a Control-C.

ERRORS:

BAD RANGE OF ADDRESSES

You did not supply proper memory addresses; make sure that the addresses you give are within your memory partition and that the starting address is smaller than the ending address.

END OF PASS n - ERROR COUNT IS x

At the end of every pass through memory, DIAG4 tells you the number of the pass and the number of errors it found.

ERROR AT ADDRESS nnnnnn - WAS Data1 SHOULD BE Data2

When DIAG4 finds an error, it tells you the memory location that contained the error (nnnnnn). Then it tells you the data that it found in that location (Data1), and the data that should have been there (Data2); this gives you an idea of what memory chip is bad.

CHARACTERISTICS:

In addition to usual memory testing, it performs a leak-down test on dynamic memory.

Tests only the memory partition of the job running DIAG4.

Returns your terminal to AMOS command level if you type a Control-C.

ding

FUNCTION:

Rings the terminal bell.

HINTS/RESTRICTIONS:

DING is useful in a command file for telling you that an input is expected or that a process is finished.

FORMAT:

.DING n ↵

where n is the number of times you want to sound the terminal bell.

OPERATION:

1. To ring your terminal bell, type DING; then give the number of times you want the terminal bell to ring. Type a RETURN. For example:

```
.DING 5 ↵
```

2. You will most often use the DING command in command files to let the user of the command file know that an event is finished or that he or she is expected to enter input. For example:

```
:T
:<Backup Project 100 accounts on DSK1: over to DSK0:>
;
COPY DSK0:[]=DSK1:[100,*]
;
:<FINISHED...  Remove backup disk.>
DING 5
```

ERRORS:

DING generates no error messages.

CHARACTERISTICS:

Used within command file to get the attention of the user of the file.

Returns your terminal to AMOS command level.

(1 October 1979)

FUNCTION:

Produces a directory listing for specific files or accounts.

HINTS/RESTRICTIONS:

You can use DIR to: display the list of all files in a specific account, tell you what accounts and devices a particular file appears in, find out the complete specification of a file, and place a copy of a directory listing into a file.

```
.DIR MEM:Filespec  performs same function as:  .MAP Filespec
.DIR RES:          performs same function as:  .SYSTEM
```

(To use the two formats above, the devices MEM and RES must be defined in your system device table.)

To search for a particular file on all mounted devices and in all accounts, use .DIR ALL:[]Filespec.

If you use DIR to create a file that holds a directory listing (and if you use the /DATA option) then BASIC can read the file specifications in your file and you can use the file with the OPEN statement.

To send a directory listing directly to a printer, use TRM:xxx as your Listfilespec, where xxx is the terminal name associated with the printer.

Use the /WIDE option to fit a long directory listing on your terminal display.

DIR is a wildcard file command. For more information on the use of DIR, refer to Section 9.2, "Finding Out What Files Are on the Disk (DIR)," in the AMOS User's Guide, (DWM-00100-35).

FORMAT:

```
.DIR {Listfilespec=}{Filespec1{,Filespec2,...FilespecN}}{/Switches}
```

where Filespec is one or more file specifications that select the file or files for which you want a directory listing. If you don't want to display the directory listing on your terminal, but instead want to write it into a file, include a Listfilespec. You may include one or more option requests (/Switches).

DEFAULTS:

If you don't include a Filespec on the DIR command line, DIR assumes that you want a directory listing for all of the files in the account you are currently logged into. If you include filenames, but omit device specifications and PPNs, DIR assumes the device and PPN you are currently logged into. The default switch is /WIDE:1/NOBASE. The default file specification is *.*.

The default Listfilespec (e.g., if you say DIR =Filespec) is DIRECT.LST in the account and device you are logged into.

OPTIONS:

You may request the following options by including a slash (/) and the appropriate switch code in your DIR command line:

/DATA or /D	Just list complete Filespecs, one per line. (Operation switch.)
/KILL or /K	Delete and replace existing Listfile if it has same specifications as your Listfilespec. (Operation switch.)
/WIDE or /W	Generate directory listing in four columns. (Operation switch.)
/WIDE:n or /W:n	Generate directory listing in n columns. (Operation switch.)
/HASH or /H	Displays a hashmark for each file (computed value based on characteristics of the file); serves to help you distinguish between file versions. (File switch.)
/CONTIGUOUS or /C	Displays a "C" next to the extension of contiguous files. The directory display thus can tell you which files are sequential and which are random (contiguous). (Operation switch.)
/BASE or /B	Displays the base disk address of the file (or base memory address, if you say DIR/B MEM:). (A file switch.)
/NOBASE or /NOB	Turns off /B switch. (A file switch.)
/FULL or /F	Same as saying: /HASH/BASE/CONTIGUOUS.

OPERATION:

1. Type DIR optionally followed by a Listfilespec and an equal sign (if you want the directory listing in a file) and one or more file specifications. You may include one or more switches. Type a RETURN. For example:

```
.DIR VRTUAL.PRG[110,9],DPL.PRG[] ↵
```

2. DIR displays directory listings for the files you specified. (See EXAMPLES, below.) It gives you the following information

FUNCTION:

Alphabetizes the entries in all of the directories on a given logical unit.

HINTS/RESTRICTIONS:

Alphabetizes ALL directories on a disk. (NOTE: By alphabetizing, we mean that DIRSEQ arranges the directory entries in order based on their ASCII values. This means that filenames that begin with numbers come after filenames that begin with letters.)

IMPORTANT NOTE: Do not use DIRSEQ while other jobs are accessing the disk.

FORMAT:

.DIRSEQ Devn:↵

where Devn: specifies the logical unit containing the directories you want to alphabetize.

DEFAULTS:

If you do not supply a device, DIRSEQ uses the device you are currently logged into.

OPERATION:

1. Type DIRSEQ (optionally followed by a device specification); type a RETURN:

   .DIRSEQ SMD1:↵

2. DIRSEQ orders the entries in the directories on the disk. When it is finished, it displays an AMOS prompt:

   .DIRSEQ↵
   .

   Use the DIR command to see a display of your alphabetically ordered account directory.

ERRORS:

DIRSEQ generates no error messages of its own, but you may see some of the standard system error messages. For example:

?Cannot INIT Devn: - device does not exist

The system cannot find the specified device. Check your spelling. To see a list of the valid system devices, type DEVTBL followed by a RETURN.

?Cannot READ Devn: - disk is not mounted

The system cannot read the specified device because it is not mounted. Use the MOUNT command to mount that logical unit.

CHARACTERISTICS:

Alphabetizes all directories on a specified disk.

DO NOT use DIRSEQ on a disk while other jobs are accessing that disk.

Returns your terminal to AMOS command level.

FUNCTION:

Executes DO files.

HINTS/RESTRICTIONS:

A DO file is a special, extended type of command file. (For information on command and DO files, see the AMOS User's Guide, (DWM-00100-35), Chapter 8, "Command Files and Do Files."

In summary, a command file is a text file that contains system commands. You can invoke such a file by entering its specification at AMOS command level. A DO file can contain all of the elements of a command file. In addition, a DO file allows you to pass arguments to the file by including parameter symbols in that file. You can then specify the arguments to be substituted for those symbols when you invoke the DO file. Build a DO file by creating a text file with one of the system text editors.

NOTE: If a DO file has a .DO extension, you may invoke that file simply by entering the specification of that file at AMOS command level, providing that no .PRG or .CMD files of the same name exist in your account or the system library accounts. The system then calls the DO command for you. If the file does not have a .DO extension, you must use the DO command.

(When processing DO files, AMOS uses the program DSK0:MDO.PRG[1,4]; do not erase that file from the System Disk.)

FORMAT:

.DO Filespec {Arg1 Arg2 ... ArgN} RET

where Filespec selects the DO file you want to invoke and Arg1 through ArgN is an optional list of one or more text items (separated by spaces) that you wish to substitute for parameter symbols in the DO file.

DEFAULTS:

DO assumes a file extension of .DO.

OPERATION:

1. Type DO followed by the specification of the DO file (and, optionally, a list of arguments required by that file). (If the file does not have a .DO extension, you must specify the extension.) Type a RETURN. For example:

   .DO WRITE.TXT RECRD1 RECRD2 (RET)

   The command above tells the system to execute the file WRITE.TXT as a DO file. The system substitutes the two text arguments RECRD1 and RECRD2 for parameter symbols in the DO file.

SYMBOL SUMMARY:

Besides all valid system commands, command file special symbols, and file specifications, DO files can contain the following symbols:

$n where n is a number between 0 and 9, inclusive. These symbols are the parameter symbols into which DO substitutes the arguments you specify on the DO command line.

$D The $D symbol identifies the default parameter list. $D must begin in the first character-position on the first line of the DO file. Follow it with the list of default text items you want DO to substitute for the parameter symbols if no arguments appear on the DO command line. Separate the items with spaces.

$ Null parameter symbol. Useful in argument list on DO command line for specifying which parameter symbols associate with which arguments and in default parameter list.

$: Current device symbol. Represents device that user of the DO file is currently logged into.

$P Current PPN symbol. Represents account that user of the DO file is currently logged into.

ERRORS:

?Cannot locate Filespec
DO could not find the file you specified. Check the extension of the file you want to use. For example, if DO says:

?Cannot locate REMAKE.DO

does your DO file, REMAKE, have a .DO extension? If not, you must specify the extension on the DO command line. Make sure that you are logged into the correct account.

CHARACTERISTICS:

Assumes a file extension of .DO.

dskana

FUNCTION:

Analyzes the data on a disk, re-creating the disk bitmap and tracking down lost disk blocks. Reports file errors, inconsistent block counts, and bad bitmap hash totals.

HINTS/RESTRICTIONS:

You must be logged into account [1,2] to run DSKANA.

DSKANA reads every block on the disk. It keeps track of what the bitmap should be, and compares that to what it actually is; then it rewrites the bitmap. Should be run on a regular basis; reclaims temporarily allocated disk blocks and cleans up bitmap. Tells you if more than one file claims the same disk block, if there are illegal block links in a file, and if your bitmap has a bad hash total.

IMPORTANT NOTE: DO NOT run DSKANA when other users are accessing the specified disk.

To see a summary of the DSKANA switches and modes, enter DSKANA followed by a RETURN.

NOTE TO PHOENIX DRIVE USERS: DSKANA checks the specified device for the file BADBLK.SYS[1,2]. (This file is created by CRT410, the program that certifies disks in devices that run under control of the AM-410; the file contains a list of all bad blocks or tracks on the certified disk.) If BADBLK.SYS was written by a 4.4 Release or later version of CRT410, DSKANA ignores the BADBLK.SYS data.

If BADBLK.SYS was written by a pre-4.4 version of CRT410, DSKANA checks the BADBLK.SYS file before proceeding with the disk analysis. If the hash total for BADBLK.SYS[1,2] is bad, DSKANA tells you so:

[BADBLK.SYS contains a bad hash code]

You then know that the original certification was not allowed to finish or that the data in BADBLK.SYS has become damaged. Although DSKANA continues with the disk analysis if BADBLK.SYS has a bad hash total, you should copy all files off the disk and then re-certify it, since DSKANA is using information in BADBLK.SYS that is of doubtful integrity. For more information on disk certification, see the reference sheets for BADBLK and CRT410.

(Changed 30 April 1981)

If you are using the /L switch, and if the the BADBLK.SYS hash total is OK, DSKANA now prints the numbers of any bad blocks. For example:

```
[bad disk blocks]
        2035   2036    2041    2042
```

FORMAT:

.DSKANA {Filespec}=Devn:{/switch} (RET)

where the optional Filespec selects a DSKANA output file. Devn: selects the device that contains the disk you want to analyze; the optional /switch selects either the /E (errors only) or /L (full listing) options.

OPTIONS:

The /L switch:
The /L switch tells DSKANA to display a full listing of all: 1. PPNs; 2. account directories; and, 3. files; along with the disk addresses of the blocks that they occupy. If you omit the /L switch from the DSKANA command line, you do not see this information or any specific file error messages, but you still see the regular DSKANA messages.

The /E switch:
The /E switch tells DSKANA to list only the PPNs of the accounts on the disk and those disk blocks in which errors occurred. For each error you see the following information: 1. number of the block at which the error occurred; 2. the appropriate file error message; 3. the file, account, and device in which the error occurred.

OPERATION:

Using DSKANA without the /L or /E switches:

1. Type DSKANA followed by the specification of the device whose contents you want to analyze; type a RETURN. DSKANA now tells you that it is beginning to analyze the disk. For example:

```
.DSKANA DSK3: (RET)
[Begin analysis of DSK1]
```

2. You see nothing more for some moments until DSKANA finishes reading the disk, except the numbers of the PPNs as DSKANA progresses through the disk accounts. For example:

```
[1,2]
[20,0]
[20,1]
```

(Changed 30 April 1981)

3. When DSKANA finishes reading the disk, it displays the following information:

[The following blocks were marked in use but not in a file]

You may next see a list of block numbers. This is not necessarily anything to be alarmed about; the system occasionally temporarily allocates a disk block for some purpose, and then fails to reclaim that block when finished with it. Running DSKANA "frees up" the blocks listed under this message by marking them free for use.

[The following blocks were in a file but not marked in use]

You may next see a list of block numbers (if you do not, then there is nothing to worry about). A list of block numbers following this message is an indication of something wrong-- for example, the linking structure of the disk has gone astray; you must run DSKANA again, using the /L or /E switches. The second time you run DSKANA, look for the file error messages listed in ERRORS, below, so that you can figure out what is wrong with the disk.

4. The last pieces of information that DSKANA displays on your terminal are concerned with the bitmap and with file errors.

To perform a check on the validity of the disk bitmap, every time the system updates the bitmap it computes a hash total and compares it with the previously stored value. DSKANA also computes a bitmap hash total based on the blocks that it has processed; if this value does not match the official bitmap hash total, you see this message:

[BITMAP on disk had a bad hash total]

which lets you know that something was not quite right. Whether the bitmap hash total is correct or not, DSKANA always rewrites the bitmap and you see:

[Rewriting BITMAP]

The last line of data in the terminal display tells you how many file errors were detected. If no file errors were seen by DSKANA as it read the disk, you see:

No file errors

Otherwise, DSKANA tells you how many file errors were detected:

```
5 file errors detected
```

(The messages listed in ERRORS, below, tell you what kinds of file errors occurred. Frequent file errors can indicate hardware problems. To see these error messages, you must run DSKANA with the /L or /E switches.)

If you want to see more information about how DSKANA is progressing as it analyzes your disk (or if DSKANA has reported file errors on your disk and you want to find out where the errors occurred), use the /L or /E switches:

1. To use the /L switch, enter DSKANA followed by the specification of the device that contains the disk you want to analyze. Then enter /L followed by a RETURN. For example:

```
.DSKANA HWK3:/L (RET)
```

You now see all of the information discussed in the paragraphs above, plus you also see a list of all PPNs, account directories, filenames and extensions, and disk block numbers for the specified disk. As DSKANA progresses through the disk, you might see a display that looks something like this:

```
[Begin analysis of DSK5:]
[1,4]
Directory              143    354    712    1126    4010
AMSORT  OLD            144    145    146    147     150
APPEND  PRG            151
                        .
                        .
                        .
```

If DSKANA finds an error, you see where on the disk it occurred. For example:

```
Block  1703 - block creates endless loop in file DSK5:MDO.PRG[1,4]
```

2. The /E switch tells DSKANA not to display the account directories, filenames and extensions, and block numbers of the disk as the analysis proceeds unless an error occurs. You do see a list of the PPNs as DSKANA reads through the disk. For example:

```
.DSKANA DSK3:/E (RET)
[Begin analysis of DSK3]
[20,1]
Block 0 - block reserved for system use only in DSK3:READ.PRG[1,4]
[40,1]
```

(Changed 30 April 1981)

To place the DSKANA output into a file:

1. Whether or not you use the /L or /E switches, you can tell DSKANA to create a file containing the information that DSKANA usually displays on your terminal screen. Include a file specification followed by an equal sign on the DSKANA command line. For example:

```
.DSKANA DSK0:ERROR.TXT=DSK1:/L (RET)
```

   If the specified file already exists, DSKANA deletes it before beginning the disk analysis. The first line of the file includes the date of the analysis. (For example: Disk Analysis list file on 4/17/80.)

   When DSKANA is finished, you may display the file by using the TYPE command or you may print it via the PRINT command.

ERRORS:

There are a number of file error messages that you can see if something is wrong on the disk. You only see these messages if you use the /L or /E switches:

Block used in previous file
   Last block in the file where this message appears also exists in a previous file. The system made an error when allocating blocks to the files.

Block marked as bad
   A block marked as bad in the BADBLK.SYS[1,2] file has somehow been allocated to a file.

Device error on Devn:
   This block contains a hard error that the system could not recover from.

[unable to locate BITMAP for rewrite]
   DSKANA couldn't find the bitmap area in memory for the device. This means that the bitmap in memory may be invalid.

BITMAP rewrite error code XXXXX
   For some reason, the bitmap could not be written back out to the disk. The number you see is the error code that indicates what the problem was. For a list of these error codes, see Chapter 6, "The File Service System," in the AMOS Monitor Calls Manual, (DWM-00100-42).

This file has a bum block count
   The actual block count for the file where this message appears does not match the block count assumed by the file itself. Once again, an error has been made in allocating blocks to this file.

(Changed 30 April 1981)

Illegal block link
A link in the file where this message appears points to an invalid block address (e.g., to a block that does not exist).

Block reserved for system use only
A link in the file where this message appears points to a block that cannot be allocated to a file. Once again, an error has been made in allocating blocks to this file.

Block creates endless loop in file
The linking structure of this file is such that eventually the blocks point back to themselves. That is, Block-A points to Block-B, which points back to Block-A. An error was made in block allocation.

If you see any of these messages in the listing of the blocks processed by DSKANA, you have serious problems with the data on your disk. Your best course is to delete the files affected, run DSKANA again, and restore the deleted files from your backup disk. (NOTE: You do not see the messages above if you have not used the /L or /E option.)

Aside from the block allocation and file errors that DSKANA itself reports, you can also see the standard system error messages dealing with invalid device specifications. In addition, you can see:

?Must be logged into PPN [1,2] to run DSKANA
You must log into the System Operator's account, [1,2], to run DSKANA.

?Cannot output to device being analyzed
If you want DSKANA to write to an output file, you may not specify that the file appear on the disk being analyzed.

?Device write protected - Please unprotect for retry
type RETURN when ready
Write-enable your disk and type a RETURN. DSKANA must be able to write on your disk to rewrite the disk bitmap.

[BADBLK.SYS has a bad hash code]
The information in BADBLK.SYS is not complete. If you want to see if anything else is wrong with the disk, let the analysis continue; otherwise, type a Control-C. Copy all files off the disk and re-certify it.

CHARACTERISTICS:

Performs a thorough analysis of the block links of the specified device. You should run DSKANA regularly to reclaim lost blocks and to check the bitmap.

DO NOT run while other users are accessing the specified disk. You must be logged into [1,2] to run DSKANA.

(Changed 30 April 1981)

dskcpy

FUNCTION:

Creates a backup disk by making a literal image of one disk onto another. Optionally generates a hash total for the backup disk.

HINTS/RESTRICTIONS:

You may use DSKCPY on any kind of disk device. However, the three-character device specification must be the same for both devices you are copying between. For example, if you want to make a copy of the disk in device HWK1:, you may copy it to HWK2: or HWK0:, but you may not copy it to AMS0: or STD3:.

Remember that DSKCPY makes a literal image of one disk onto another. This means that any data on the disk you are copying to will be irretrievably lost after a disk copy is done. As of AMOS Version 4.4, you may use DSKCPY on devices that run under the control of the AM-410 hard disk controller (e.g., the Phoenix drive).

NOTE TO HAWK HARD DISK USERS: DSKCPY uses a special "fast copy" mode for Hawk hard disk devices. If you do not inhibit the Hawk fast copy mode, no one else may run on the system while you perform the Hawk disk backup. This is because this mode causes DSKCPY to communicate directly with the AM-500 controller instead of going through AMOS. See OPTIONS, below, for information on the /0 switch, which tells DSKCPY to use the old disk copy method for copying between Hawk devices.

DSKCPY optionally generates a hash total for the backup disk when it has finished the disk copy. This feature is especially useful when you are making multiple copies of a disk-- the hash total displayed at the end of the disk copy should be the same for each disk copied. The hash total gives you an extra way to verify that the copies made are identical to the master disk, since two disks will only have the same hash total if their contents are identical. (For information on generating a hash total for a disk without using DSKCPY, see the HASHER reference sheet in this manual.)

FORMAT:

.DSKCPY{/Switches} (RET)

where Switches select optional DSKCPY features.

DEFAULTS:

If you enter just the unit numbers of the disks you want to copy between instead of a full device specification, DSKCPY asumes that you

(Changed 30 April 1981)

are copying between DSK devices. (For example, if you enter 1 and 0 as the input and output drives, DSKCPY assumes that you want to copy from DSK1: to DSK0:.)

If you are copying between Hawk hard disks, and do not use the /O switch, DSKCPY automatically uses the Hawk fast copy mode. No other user may run on the system while DSKCPY is using the fast copy mode to backup Hawk disks.

OPTIONS:

You may select the following options by specifying the appropriate switch codes:

/H Generates a hash total for the copied disk. (If you use the Hawk "fast copy" mode, DSKCPY gives you a different hash total for the disk than if you use the old disk copy mode. See below for information on inhibiting the fast copy mode.)

/O Tells DSKCPY to use the old disk copy method for copying between Hawk hard disks. Even though this mode is much slower than the default fast copy mode, you may want to use this switch if your system has more than one set of disk drives, since the fast copy mode prevents any other user from running on the system while the disk copy is taking place.

This switch has no effect if you are copying between other disks.

OPERATION:

To copy between floppy disks or Phoenix hard disks:

1. Enter DSKCPY followed by a RETURN:

```
.DSKCPY (RET)
```

If your disk drive permits, you should now write-protect the drive you are copying from; this ensures that you won't accidentally copy the backup disk onto your source disk.

DSKCPY asks you for the Input drive. Enter the specification of the device you are copying from; type a RETURN. Now DSKCPY asks you for the Output drive. Enter the specification of the device you are copying to; type a RETURN. For example:

```
.DSKCPY (RET)
Input drive: AMS0: (RET)
Output drive: AMS1: (RET)
```

(Changed 30 April 1981)

Now DSKCPY makes a literal image of the disk in device AMS0: onto the disk in device AMS1:. DSKCPY tells you when it is finished. As it copies and verifies, it tells you how many blocks it is copying. (DSKCPY copies every block on the disk, even if some blocks contain no data.)

```
[Copying 500 blocks]
[Duplication and verification completed]
```

The amount of time it takes to perform this disk copy depends on the device. An AMS-format diskette takes about two minutes to copy and verify; a Phoenix hard disk takes about 28 minutes.

2. If you want DSKCPY to generate a hash total for the disk copied to, use the /H switch. For example:

```
.DSKCPY/H (RET)
Input drive: STD0: (RET)
Output drive: STD1: (RET)
[Copying 500 blocks]
[Duplication and verification completed]

Hash is 672
```

DSKCPY displays the hash total for disk after it has finished the disk copy. You can now check this number against the hash total for the source disk to make sure that a perfect copy was done. To generate a hash total for the source disk, use the HASHER command. (See the HASHER reference sheet in this manual.)

To copy between Hawk hard disks:

1. Enter DSKCPY followed by a RETURN:

```
.DSKCPY (RET)
```

Now DSKCPY asks you for the Input drive and the Output drive. Enter the specifications of the devices that contain the disk you want to copy from and the disk you want to copy to. (You should write-protect the disk you are copying from so that you don't accidentally write to it.)

When you are copying between Hawk disks, DSKCPY automatically uses the special fast copy mode. Because this mode causes DSKCPY to communicate directly with the AM-500, no other users on the system may run while you are performing the disk copy. DSKCPY warns you that it is suspending all users while the disk copy takes place:

```
%All other users will be suspended while HAWK copy is running.
Hit return to continue or control-C to abort:
```

If any users are running on the system, type a Control-C. When you are sure that no one else is running, you can use the DSKCPY command again. Remember that if the device specification of a Hawk device is DSK, you can just enter the unit numbers of the devices you want to copy between. For example:

```
.DSKCPY (RET)
%All other users will be suspended while HAWK copy is running.
Hit return to continue or control-C to abort: (RET)
Input drive: 0 (RET)
Output drive: 1 (RET)
[Copying 9696 blocks]
[Duplication and verification completed]
```

The fast copy takes about six minutes.

2. If other users are running on the system, you may want to use the old mode of disk copying. This type of disk copy will take longer than the fast copy, but you can perform it while others are using the system. Type DSKCPY followed by a /O; then type a RETURN:

```
.DSKCPY/O (RET)
Input drive: HWK2: (RET)
Output drive: HWK3: (RET)
[Copying 9696 blocks]
[Duplication and verification completed]
```

The old Hawk disk copy mode takes about 13 minutes.

3. To generate a hash total for the disk copied to, use the /H switch. For example:

```
.DSKCPY/H (RET)
```

or:

```
.DSKCPY/O/H (RET)
```

ERRORS:

You may see the following errors when using DSKCPY:

?Invalid switch, please use one or more of the following:

/H	generate a hash code for the copied disk
/O	use old (slow) copy method for the AM-500

You specified a switch on the command line other than /O or /H. Enter the command line again.

?Driver not found

DSKCPY couldn't find the device driver program for the specified non-DSK device. That means that the driver was not in

system memory, user memory, or DSK0:[1,6]. Check with the System Operator.

?Input and output devices must be the same

You entered device specifications in which the three-character device code was different. (For example, you tried to copy HWK1: to AMS1:.) Remember that you may only copy between devices that use the same device driver program; that is, devices that have the same device codes.

?Disk size not defined in table

DSKCPY doesn't know the number of disk blocks per disk for the devices you are trying to copy between. This means that you are copying between devices that DSKCPY doesn't know about. Check with the System Operator for help.

?Verification error at block nnn

After DSKCPY copied the data in the specified disk block, it was not able to verify the data on the output disk. This means that the data changed between the time it was read on the source disk and the time it was read on the output disk. Try using DSKCPY again. If you see this message frequently, you may have hardware problems.

There are also several AM-500 hard-disk controller "hard error" messages that it is possible but extremely unlikely that you would ever see. (If any of these do occur, please contact Alpha Micro.)

?Could not find DEVTBL entry for disk mount

This error message reflects an almost impossible set of circumstances affecting the device table. Please submit to Alpha Micro, on a standard Software Performance Report, the details of your use of DSKCPY if you ever see this message.

?Sector not found during disk copy - drive N block X

where drive N is a decimal number and block X is an octal or hexadecimal number. This can only happen when you are using the fast copy mode of DSKCPY or HASHER on a Hawk hard disk system. The contents of a disk sector have become unformatted. Verify if there is a problem by using REDALL to diagnose the disk and report any read errors. (See the REDALL reference sheet for more information about disk diagnostic tests.)

?CRC error during disk copy - drive N block X

where drive N is a decimal number and block X is an octal or hexadecimal number. This can only happen when you are using the fast copy mode of DSKCPY or HASHER on a Hawk hard disk system. The Cyclic Redundancy Check device on the AM-500 board has detected a problem in data transmission. First verify if there is a problem by using REDALL to diagnose the disk and report any read errors. (See the REDALL reference sheet for more information about disk diagnostic tests.) The problem can be fixed using DSKDDT, but the

data in the block may be lost. (For more information on DSKDDT, see the DSKDDT reference sheet.)

?Sentinel field error during disk copy - drive N block X

where drive N is a decimal number and block X is an octal or hexadecimal number. This can only happen when you are using the fast copy mode of DSKCPY or HASHER on a Hawk hard disk system. The contents of a disk block have become unformatted. Verify if there is a problem by using REDALL to diagnose the disk and report any read errors. (See the REDALL reference sheet for more information about disk diagnostic tests.)

?Undefined error during disk copy - drive N block X

where drive N is a decimal number and block X is an octal or hexadecimal number. This can only happen when you are using the fast copy mode of DSKCPY or HASHER on a Hawk hard disk system. A hard error occurred which was not definable as any of the foregoing AM-500 hard-disk controller errors. Verify if there is a problem by using REDALL to diagnose the disk and report any read errors. (See the REDALL reference sheet for more information about disk diagnostic tests.)

See the documentation that accompanies the AM-500 board for more information on these errors and the conditions they report.

CHARACTERISTICS:

Makes a literal image of one disk onto another. Optionally generates a hash total for the disk copied to.

May NOT be run while anyone is accessing the disks being copied between. Assumes a device code of DSK.

If used on Hawk hard disks, DSKCPY may be used either in fast copy mode (with no other users running on the system) or in a slower copy mode (which allows other users to run on the system). Other devices (floppy disk drives and Phoenix hard disk drives) may not use the Hawk fast copy mode.

(Changed 30 April 1981)

FUNCTION:

Allows you to examine and change data directly on the disk.

HINTS/RESTRICTIONS:

Useful for restructuring files and correcting ERROR 10 conditions.

The record number you enter to DSKDDT must be in the number base the system is currently using for your numeric displays (usually octal). (See the SET reference sheet for information on setting the display base.) The numbers you give to DSKDDT to tell it which disk locations you want to examine and the replacement data for those locations must be in octal regardless of the number base the system is using for numeric displays.

FORMAT:

.DSKDDT Devn:Record ↵

where Devn: is the device containing the record you want to see, and Record is the number of the disk record you want to examine and change.

OPERATION:

1. Type DSKDDT followed by a device specification and a record number; then type a RETURN. For example:

   .DSKDDT DSK1:147 ↵

2. DSKDDT reads the record specified, lists any errors it finds, and is then ready for you to modify the record.

3. Use the DSKDDT commands listed below. To see the contents of the first location, type a slash, /. To see the contents of the next location, type a line-feed. DSKDDT displays two bytes of data at a time. The DSKDDT display might look something like this:

   .DSKDDT DSK1:202 ↵

   ```
   2/     00452
   4/     44510
   6/     52040
   ```

   The number on the left of the slash is the RELATIVE disk location within the record that you want to examine. That is, given that you are examining record 202, the display above

tells you that the first two bytes in record #202 are 00452, the second two bytes are 44510, and the the third two bytes are 52040.

4. After you finish using the DSKDDT commands, type E to exit DSKDDT. DSKDDT then rewrites the record and takes your terminal back to AMOS command level.

COMMAND SUMMARY:

nnn/	Where nnn is the disk location you want to examine. (This number is the relative position of the location from the front of the record. For example, 6/ displays the contents of the sixth and seventh bytes in the record.)
nnn/NNN	where nnn is the relative disk location you want to examine (entered in octal), and NNN is the octal data (two bytes) with which you want to replace the contents of nnn.
Line-feed	Display the next two bytes of data in the record.
^	Display the previous location.
/	Display location zero in the record (that is, display the first two bytes of data in the record).
RETURN	No operation.
RUBOUT	Cancel current command line and display XXX followed by a TAB.
E	Rewrite the modified record and exit.
^C	Exit without updating record.

ERRORS:

The only DSKDDT error message is a backspace, question mark, and a tab. This means that DSKDDT didn't understand your command.

You may also see some of the standard system error messages that result from an invalid device specification. For example:

?Cannot READ Devn: - device does not exist

Check your spelling. AMOS does not recognize the device you specified. For example, did you enter DKS1: instead of DSK1:?

?Cannot READ Devn: - disk is not mounted

DSKDDT can't read the device because it is not mounted. Use the MOUNT command to mount the disk.

CHARACTERISTICS:

Allows you to examine and modify disk records.

Fixes most disk errors by rewriting the record and so recomputing the Cyclic Redundancy Check. (The CRC is a computed value based on the numeric data in the record; it is used to verify the contents of the record.)

Returns your terminal to the AMOS command level.

dskdmp

FUNCTION:

Displays physical disk records on your terminal in numeric form.

HINTS/RESTRICTIONS:

The system internally represents all data in numeric form. The DSKDMP display shows you the data in the number base that the system is currently using for your numeric displays; enter record numbers in that same base. (See the SET reference sheet for information on changing the display base.)

NOTE: You may freeze the display by typing a Control-S and resume it by typing a Control-Q. To interrupt the display, type a Control-C.

FORMAT:

.DSKDMP Devn:Record ↵

where Devn: is the logical unit that contains the data you want to display; Record is the number of the physical disk record you want to see.

DEFAULTS:

If you do not supply a device specification, DSKDMP uses the disk you are currently logged into.

OPERATION:

1. Type DSKDMP followed by a device specification and a disk record number; then type a RETURN. For example:

   .DSKDMP DSK1:202 ↵

2. DSKDMP displays the entire record in the number base that the system is currently using for your displays (usually octal, base-8), with a message that tells you which record you are seeing. Each line of the display groups data into eight groups of 16 bits:

```
.DSKDMP 3063 ↵
[RECORD 3063]
003101  052057  046124  042040  045523  046504  020120  041450
067157  023564  024544  005015  066057  033040  005015  052506
```

ERRORS:

?Cannot READ - illegal block number
You gave DSKDMP a record number that does not exist. For example:

```
.DSKDMP 3333333
[RECORD 1333333]
?Cannot READ - illegal block number
```

You may also see some of the standard system error messages if you supply invalid device specifications. For example:

.?Cannot READ Devn: - device does not exist
Check your spelling. The system does not recognize the device you specified.

?Cannot READ Devn: - disk is not mounted
Use the MOUNT command to mount the disk you want to access.

CHARACTERISTICS:

Displays physical disk records in numeric form on your terminal display.

Returns your terminal to AMOS command level.

dskfil

FUNCTION:

Tells you what disk records are used by the specified file.

HINTS/RESTRICTIONS:

The numbers that you see are octal (or hexadecimal, if you have previously used the SET HEX command), and give the physical disk addresses of the records in the file.

FORMAT:

.DSKFIL Filespec ↵

where Filespec selects the file whose record numbers you want to see.

DEFAULTS:

Default file extension is .PRG

OPERATION:

1. Enter DSKFIL and the specification of the file whose disk addresses you want to see; then type a RETURN. For example:

   .DSKFIL SWITCH.TXT ↵

2. Now you see a list of octal numbers. These numbers tell you what physical records on the disk are being used by your file:

```
.DSKFIL BYTE.MAC↵
1164   1165   1166   2033
```

ERRORS:

You may see the standard system error messages when using this command, the most common being:

?File specification error
DSKFIL did not understand your command line (e.g., you typed DSKFIL and a RETURN). Retype the line.

Filespec NOT FOUND
DSKFIL was not able to find the file you specified. Check your spelling, and make sure that you supplied the proper account and device specification.

CHARACTERISTICS:

Lists disk records in use by the specified file.

The default file extension is .PRG.

Returns your terminal to AMOS command level.

dskpak

FUNCTION:

Allows you to consolidate or "pack" the contiguous file area of the disk by moving contiguous files toward the end of the disk.

HINTS/RESTRICTIONS:

Use the DSKPAK program to avoid fragmentation of open space on the disk. DSKPAK consolidates the contiguous files on the disk and thus reduces the number of small free areas on the disk. This creates more room on the disk for new contiguous files. (For more information on file allocation, refer to the document New Method of Allocating Contiguous Files, (DWM-00100-23), in the System Programmer's Information section of the AM-100 documentation packet.)

NOTE: DSKPAK only packs contiguous files. To pack sequential files, use the COPY command with the /PACK option. (See the COPY reference sheet.)

FORMAT:

.DSKPAK Devn:↵

where Devn: is the specification of the logical unit you want to pack.

OPERATION:

1. Type DSKPAK followed by the device specification of the disk you want to pack. Type a RETURN. For example:

   .DSKPAK DSK1:↵

2. DSKPAK tells you that it is finished by returning you to AMOS command level.

ERRORS:

If you supply an invalid device specification, you can see several system error messages. For example:

?Cannot ACCESS Devn: - device does not exist
AMOS does not recognize the device you specified. Check your spelling and retype the command line.

?Cannot ACCESS Devn: - disk is not mounted
DSKPAK cannot read the disk because it is not mounted. Use the MOUNT command.

(1 October 1979)

?File specification error
You probably forgot to include the device specification on the DSKPAK command line. Retype the command line.

CHARACTERISTICS:

Packs the contiguous files on a disk together, reducing fragmentation of open space on the disk.

Returns your terminal to AMOS command level.

dump

FUNCTION:

Displays on your terminal the contents of memory, random and sequential disk files, Master File Directories, disk bitmaps, user directories, and disk blocks.

HINTS/RESTRICTIONS:

DUMP accepts different kinds of arguments, depending upon the kind of data you want to display. Give DUMP numbers in the same number base (usually octal) that the system is currently using for your displays. (See the SET reference sheet for information on changing the system display base.)

You may abbreviate the DUMP keywords (e.g., DI instead of DIRECTORY).

DUMP is re-entrant, and may be loaded into system memory by the System Operator. For more information on DUMP, see The DUMP Program, (DWM-00100-24), in the "User's Information" section of the AM-100 documentation packet.

FORMAT:

There are six different formats for the DUMP command:

1. To display memory -

   .DUMP Address1 Address2 (RET)

   where Address1 is the first address of the memory block you want to display and Address2 is the last.

2. To display a file -

   .DUMP Filespec (RET)

   where Filespec is a valid file specification of a random or a sequential file.

3. To display a disk block -

   .DUMP BLOCK Block-number1 {Block-number2} {Devn:} (RET)

   where Block-number1 is the number of the block you want to see, and Devn: is the specification of the logical unit that contains the block. If you omit the device specification, DUMP assumes the device you are logged into.

(Changed 1 May 1980)

You may optionally supply a second block number. If you give two block numbers, DUMP displays the data from the first to the second block, inclusive.

4. To display a disk bitmap -

.DUMP BITMAP {Devn:} RET

where Devn: is the specification of the logical unit whose bitmap you want to see. If you omit the device specification, DUMP assumes the device you are logged into.

5. To display the disk Master File Directory -

.DUMP MFD {Devn:} RET

where Devn: is the specification of the logical unit whose MFD you want to see. If you omit the device specification, DUMP assumes the device you are logged into.

6. To display a user's disk directory -

.DUMP DIRECTORY [p,pn] {Devn:} RET

or:

.DUMP DIRECTORY Block-number {Devn:} RET

where-- Devn: is the specification of the logical unit that contains the user's directory you want to see; [p,pn] is the PPN associated with the user's directory; and, Block-number is the starting block number of the user's directory. If you omit the device specification, DUMP assumes the device you are logged into.

DEFAULTS:

DUMP assumes the device and PPN you are currently logged into. It also assumes a file extension of .PRG.

OPERATION:

1. Type DUMP followed by the keyword and arguments required by your particular application of the DUMP command. (See FORMAT, above.) Then type a RETURN. For example:

.DUMP BLOCK 16407 DSK0: RET

2. To freeze a DUMP display, type a Control-S; to resume it, type a Control-Q. To interrupt a display, type a Control-C.

(Changed 1 May 1980)

NOTE: DUMP uses several different formats for its displays, depending on the kind of data it is displaying. The bitmap, MFD and User's File Directory displays have their own format. All other displays show the data in both numeric and ASCII form. The most common type of display takes the form of the example below, where the numeric form of the data appears on the left of the display and the character form of the data (i.e., as ASCII characters) appears on the right.

```
.DUMP DSK1:DUMP.TXT (RET)

Block number 12033 of DSK1:DUMP.TXT, next block link is 6562
005620:006562 020056 052040 062550 066400 071557 020164 067543 rm.  The most co
005640:066555 067157 072040 070171 020145 063157 062040 071551 mmon type of dis
005660:066160 074541 072040 065541 071545 072040 062550 063040 play takes the f
005700:071157 006555 067412 020146 064164 020145 074145 066541 orm..of the exam
^C
.
```

ERRORS:

DUMP displays two error messages:

?Illegal user code
: You used the DUMP DIRECTORY [p,pn] command. DUMP was not able to find that account on the specified device. Make sure that the account exists on that device.

?Command format error
: You probably made a typing error. Make sure that you have followed the proper DUMP format for your application of the command.

You can also see standard AMOS error messages if you supply invalid device or file specifications. For example:

?Cannot OPEN Filespec - file not found
: The system couldn't find the file you specified. Make sure that your device and account specifications are correct.

?Cannot OPEN Filespec - invalid filename
: The system did not recognize the file specification you gave. (For example, you gave the device specification, but did not include the filename.) Check your spelling and try again.

?Cannot OPEN Filespec - disk not mounted
: You must use the MOUNT command to mount the device you want to access.

(Changed 1 May 1980)

?Cannot OPEN Filespec - device does not exist

The system does not understand your device specification. Check your spelling. Did you type DKS1: instead of DSK1:, for example?

?File specification error

You did not supply a proper file specification on the DUMP command line. For example, you see this message if you type DUMP followed by a RETURN. Check the FORMAT section above for information on the valid formats of the DUMP command.

CHARACTERISTICS:

Displays on your terminal the contents of memory, sequential and random disk files, disk blocks, disk bitmaps, disk Master File Directories, and disk user file directories.

Returns your terminal to AMOS command level.

dystat

FUNCTION:

Video monitor real-time display that gives information about system status.

HINTS/RESTRICTIONS:

To use DYSTAT, your system must have a memory-mapped video display board.

Both DSK0:TODCNV.PRG[1,4] and DYSTAT must be in system memory. Change the SYSTEM.INI file to include:

```
SYSTEM TODCNV
SYSTEM DYSTAT
```

FORMAT:

.DYSTAT {Base-mem-addrs I/O-port}↵

where Base-mem-addrss is the beginning address of the RAM associated with the video board and I/O-port is the I/O port into which the video board is connected.

DEFAULTS:

The default Base-mem-addrss is: 174000 (octal)
The default I/O port is: 177710 (octal)

OPERATION:

1. Type DYSTAT (optionally followed by the memory addresses and the I/O port associated with the video board); then type RETURN:

   .DYSTAT↵

2. Now you see a display on the DYSTAT video monitor that gives you real-time information about the status of the system. You see one line of data for each job on the system; this line contains the following information about the job: 1. name of the job; 2. the account the job is logged in under; 3. the job's priority level; 4. the name of the last program run by the job; and, 5. the current status of the job. (The symbols that give you information about job status are described in the AMOS User's Guide, Chapter 12, "System Information Commands.")

(1 October 1979)

3. If the symbols indicating current job status are enclosed in square brackets, that job is in a suspended state (caused either by the user typing a Control-S or by using the SUSPND command). The display also shows an arrow pointing to the job that is currently active.

4. Included in the left-hand corner of the display is the current time in seconds.

5. You may run DYSTAT at any time to reinitialize the display or to change the memory address or I/O port associated with the video board. DYSTAT remembers the last set of addresses entered.

ERRORS:

If the program DSK0:TODCNV.PRG[1,4] is not in system memory when you try to run DYSTAT, you see:

?TODCNV must be resident in system memory - change SYSTEM.INI

CHARACTERISTICS:

Requires that your system be equipped with a memory-mapped video board.

Requires that TODCNV.PRG and DYSTAT itself be in system memory.

Allows you to select the memory address and I/O port used by the video board.

Returns your terminal to AMOS command level.

FUNCTION:

Allows you to create and edit text files.

HINTS/RESTRICTIONS:

EDIT is a character-oriented text editor. Use various editing commands to move a pointer in your text. Other commands allow you to delete, insert, and change characters at the location of the text pointer. Most EDIT commands are one- or two-characters. EDIT uses an Escape (rather than a RETURN) as a command delimiter, so you can enter RETURNs as part of your commands. (EDIT echoes Escapes as dollar signs, $.) To exit, enter an E and type two Escapes:

*E$$

EDIT brings into memory a copy of the file you want to edit. After making your editing changes and insertions, EDIT renames your original file to a .BAK extension (to create a backup file) and writes your edited file out to the disk under the same name and extension as the original file.

NOTE: Before you can edit a new file, you must first use the MAKE command to create that file. (See the MAKE reference sheet.)

For information on using EDIT and on the EDIT commands, refer to the document EDIT - A Character-Oriented Text Editor, (DWM-00100-39, Rev A01) in the AM-100 documentation packet.

FORMAT:

.EDIT Filespec ↵

where Filespec selects the file you want to edit.

DEFAULTS:

EDIT assumes a file extension of .MAC.

OPERATION:

1. Type EDIT followed by the specification of the file you want to edit. Then type a RETURN. For example:

   .EDIT MASTER.MAC ↵

   After a moment you see the EDIT prompt, a *.

2. Now you can begin to enter EDIT commands. (See the summary, below.) Commands may be either in upper or lower case. To tell EDIT to execute a line of commands, end that line with two Escapes.

3. To exit EDIT, enter the E command (or EQ, if you want to exit without updating your original file) and two Escapes:

*E$$

COMMAND SUMMARY:

The list below briefly summarizes most of the EDIT commands. For an explanation of the commands, see the document EDIT - A Character-oriented Text Editor, (DWM-00100-39, Rev A01), in the AM-100 documentation packet. To repeat a command, enclose the command in angle brackets preceded by the number of times you want to repeat it. For example: 30<10C-2D>$$.

A - Append file records.
F - Print free memory.
I - Insert text.
nI - Insert special character (n is its ASCII value.
S - Search.
FS - Search and replace.
N - Whole file search.
HK - Kill entire buffer.
HD - Delete entire buffer.
EQ - Exit without updating.

Linefeed - Line advance and type.
0J - Jump pointer to front of buffer.
ZJ - Jump pointer to end of buffer.
nJ - Jump pointer to nth character in buffer.
Gx - Get auxiliary buffer.
Vx - Verify auxiliary buffer.
Xx - Save auxiliary buffer.
E - Exit.
EG - Exit and go.

The commands below take a numerical argument. The argument selects which text the command will affect. For example, for command "Y," the command may take the following forms:

Y - (No argument.) Affects the first character after the text pointer or from the first character to the end of the line (depending on the command).

nY - Affects next n characters or lines (depending on the command).

-Y - Affects character (or line) just behind pointer.

-nY - Affects previous n characters (or lines) just behind the text pointer (depending on command).

0Y - Affects from beginning of line up to text pointer or moves pointer back to start of current line (depending on command).

The commands:

C - Character advance	D - Delete characters
K - Kill lines	L - Line advance
R - Reverse character advance	T - Type lines
	X - Save lines in auxiliary buffer

ERRORS:

You can see the usual system messages if you supply an invalid file specification. For example:

?Cannot OPEN Filespec - file not found

You did not specify a valid file. Check your spelling. Remember, if you want to create a new file, you have to "start" it first by using the MAKE command.

?File specification error

The system does not understand your EDIT command line. For example, you entered EDIT followed by a RETURN (i.e., you did not include a file specification on the EDIT command line).

You can also see several EDIT messages once you begin to edit your file. For example:

[SEARCH FAILED]

You asked EDIT to search for a group of characters that does not exist in your file (or, at least, not in the portion of your file that is in memory).

X?

If you enter a command that EDIT does not recognize, EDIT echoes the command back to you with a question mark. For example:

```
*W$$
W?
```

CHARACTERISTICS:

Character-oriented text editor.

Creates a backup (.BAK) file; assumes an input file extension of .MAC.

Recognizes an Escape (which it echoes as a dollar sign) as a command delimiter.

Returns your terminal to AMOS command level.

FUNCTION:

Allows you to access AlphaMAIL, the Alpha Micro electronic mail system. You may use EMAIL to exchange messages with other AlphaMAIL users on your computer system. You may also create messages, maintain your incoming and outgoing mail list, and create an Alpha Micro Software Performance Report (SPR).

HINTS/RESTRICTIONS:

You may only use EMAIL if you are an authorized user of AlphaMAIL. The AlphaMAIL Operator must assign you a user-ID and a password.

NOTE: To run EMAIL, the terminal you are using must be attached to a job that has at least 32K of memory.

Make sure that your disk is not write-protected.

For more information on EMAIL, see the AlphaMAIL User's Manual, (DSS-10000-06). For information on the program used by the AlphaMAIL Operator to manage and maintain the AlphaMAIL system, see the OPR reference sheet.

FORMAT:

.EMAIL (RET)

OPERATION:

1. Log into the account you have set aside for using AlphaMAIL. Next, type EMAIL followed by a RETURN:

   .EMAIL (RET)

2. Now the screen clears and EMAIL asks you for your password:

   EMAIL Version 1.0

   ENTER YOUR PASSWORD:

   Enter your password (exactly five characters). You must enter it exactly as it was given to you by the AlphaMAIL Operator.

3. When you have correctly entered your password, the screen clears and you see the EMAIL main menu:

```
EMAIL Version 1.0                                [new messages received]

COMMAND CODES:
?                   Display command menu
M                   Send mail and receive mail
X                   Return to AMOS

Incoming Mail:
L                   List all messages
T n                 Read message #n
P n                 Print message #n
F n                 Forward message #n to another user
D n                 Delete message #n from incoming mail list

Outgoing Mail:
B                   List all messages
R n                 Read message #n
W n                 Print message #n
C n                 Delete message #n from outgoing mail list
S                   Create new message
I                   Create Software Performance Report
E n                 Edit message #n

EML>
```

Choose one of the EMAIL commands by entering the appropriate command code after the EMAIL prompt symbol, EML>. For example, to create a message:

```
EML>S (RET)
```

4. If you forget what commands are available, enter a question mark followed by a RETURN after the EMAIL prompt:

```
EML>? (RET)
```

and EMAIL will display its command menu again.

5. To exit, enter an X followed by a RETURN after the EMAIL prompt:

```
EML>X (RET)
```

and EMAIL returns you to AMOS command level.

ERRORS:

You may see the following error messages when using EMAIL:

?You must enter exactly five characters

When entering your EMAIL password, you entered more or less than five characters. If you don't remember your password, consult the AlphaMAIL Operator on your system.

?No user-ID defined for you, please contact your Operator

EMAIL did not recognize your password. Make sure that you are entering the password with the proper spacing and punctuation, and try again. If you still do not succeed, check with the AlphaMAIL Operator on your system, who will look up your password for you.

?"V"? - Type ? for help

where V is the command you entered. EMAIL does not recognize this command. Type a ? followed by a RETURN to see a menu of the commands you may use.

?Invalid message number

You tried to delete or look at a message that does not exist. Make sure that you did not specify a message on the outgoing mail list when you wanted to look at a message on the incoming mail list, or vice versa.

?Sorry - FILENAME is a random file
?Sorry - FILENAME is a binary file

where FILENAME is the name of the file you specified. You tried to look at a random or binary file. These files conventionally do not contain printable, ASCII characters, and you may not look at them. This file is probably a machine language program or a data file.

?"USERNAME" - not in the User Directory

You tried to send a message to someone, but EMAIL does not recognize the user-ID you addressed the message to. Check with the AlphaMAIL Operator for an up to date list of all AlphaMAIL users.

?"FILENAME" not found

An error occurred when EMAIL transferred a message. Check with the AlphaMAIL Operator.

CHARACTERISTICS:

EMAIL is the interface to the AlphaMAIL system. Your memory partition must be at least 32K to run EMAIL, and the disk must not be write-protected.

EMAIL is re-entrant and re-usable.

To exit to AMOS level from EMAIL, use the X command. To see the EMAIL main menu again, use the ? command.

(30 April 1981)

FUNCTION:

Deletes one or more files from the disk.

HINTS/RESTRICTIONS:

You may use the standard wildcard symbols in your file specifications. (ERASE is a wildcard file command. Refer to Section 9.4, "Erasing Files (ERASE)," in the AMOS User's Guide, (DWM-00100-35), for more information on using ERASE.

NOTE: ERASE recognizes the ersatz devices (for example, .ERASE BAS:BADGER.SBR) and the special device MEM:. (MEM: must be defined in your system device table. See the DEVTBL reference sheet.)

Examples of special uses:

.ERASE MEM:Filespecs performs the same function as .DEL Filespecs.

.ERASE *.BAK=*.MAC erases all .BAK (backup) files in your account if a .MAC file of the same name exists in that account.

If you are logged in under [1,2], .ERASE ALL:[]*.BAK erases all backup files from all accounts on all file-structured devices that are mounted.

NOTE: It is very important that you do not erase the file BADBLK.SYS[1,2] if it exists on a device, since it contains important certification data for that device.

FORMAT:

.ERASE {Outfilespec=}Filespec1{,Filespec2,....FilespecN}{/Switch} RET

where Filespecs are one or more files to be erased. The optional Outfilespec is a file specification that acts as a qualifier on the files that the Filespecs select. /Switch is an option request. (See OPTIONS.)

DEFAULTS:

ERASE uses the device and PPN you are logged into as default file specifications. A null extension is the default extension (i.e., an extension zero characters long). The default switch is /NOQUERY.

(Changed 30 April 1981)

OPTIONS:

You may ask for the options below if you place a slash (/) and the appropriate option code (switch) on the ERASE command line:

/QUERY	or /Q	Request confirmation to erase
/NOQUERY	or /NOQ	Erase without asking for confirmation

OPERATION:

1. Type ERASE (optionally followed by an Outfilespec and an equal sign). Now type the one or more file specifications that select the files you want to delete. Type a RETURN.

   ERASE tells you which files it deleted and how many disk blocks were freed.

   ```
   .ERASE *.BAK,*.TXT (RET)
   ACCNT.BAK
   OE.TXT
   Total of 2 files deleted, 45 disk blocks freed
   ```

2. If you use the /QUERY option, ERASE asks you to confirm each deletion:

   ```
   .ERASE *.BAK/QUERY (RET)
   MATH.BAK?Y
   GEOM.BAK?Y
   Total of 2 files deleted, 56 disk blocks freed
   ```

   In response to the question mark, type a y for YES or n for NO. Do not type a RETURN after your answer. You may type a Control-C at any time to prevent further deletions; you then return to AMOS command level. The placement of the switch on the command line affects the operation of ERASE. (.ERASE *.BAK/Q,*.TXT only asks you to confirm the .BAK file deletions; .ERASE/Q *.BAK,*.TXT asks for confirmation of all deletions.)

ERRORS:

?Cannot find DSK0:SCNWLD.SYS[1,4] or MEM:SCNWLD.SYS
ERASE needs this file to process wildcard symbols in file specifications. The system cannot find the file in DSK0:[1,4] or memory, or your memory partition does not have enough room to load in the file.

?Specification error ^
Your command line is not in proper format; the ^ symbol points to the location on the line that ERASE does not understand.

(Changed 30 April 1981)

%No file-oriented device corresponding to Dev: is mounted.

You specified a device, but did not include a unit number. ERASE was not able to find a mounted logical unit matching that specification.

?Cannot READ Devn: - device does not exist
?Cannot READ Devn: - device is not mounted

You tried to erase from a device that is not listed in the DEVTBL command in your SYSTEM.INI, does not have a driver in the area [1,6] of the System Disk, is not file-structured, or is not mounted. (Devn: is the device you tried to erase from.)

%Account does not exist - [p,pn]

The indicated PPN does not exist; to create it, you must be logged under [1,2]. (See the SYSACT reference sheet.)

?More than one output specification

You cannot specify more than one output specification. For example, .ERASE *.BAK,*.TXT=*.MAC is an illegal command.

%No files deleted

ERASE could find no files that had the names and extensions you specified.

CHARACTERISTICS:

A wildcard file command. Understands ersatz and special devices.

Default switch is /NOQUERY. Default file specification is null extension and account and device you are logged into.

(Changed 30 April 1981)

exit

FUNCTION:

Terminates a command file and (optionally) displays a message to the user of the command file.

HINTS/RESTRICTIONS:

May appear only in a command or DO file. You will find EXIT most useful in the case of conditional transfers within a command file, where EXIT can be used to prevent user from "falling into" a segment of the command file branched to by a GOTO statement. (See GOTO reference sheet.)

For more information on EXIT, see New Features in Command Files and DO Files, (DWM-00100-63), in the "User's Information Section" of the AM-100 documentation packet. For information on command files, see Chapter 8, "Command Files and DO Files," in the AMOS User's Guide, (DWM-00100-35).

FORMAT:

EXIT {message}

You may include an optional message on the EXIT command line, which is displayed to the user of the command file when EXIT returns him or her to AMOS command level.

OPERATION:

1. Place the EXIT statement in the command file where you want to terminate operation of that command file. (NOTE: The sample below contains GOTO, PAUSE, and LOOKUP commands. If you are not familiar with GOTO, PAUSE, and LOOKUP, you may want to refer to the reference sheets for those commands for additional information.)

```
; Command file to compile GLIDX.BAS
;
LOOKUP GLIDX.BAS/?File does not exist.
GOTO NOFILE; Oops. File does not exist. Skip to NOFILE.
;
; File does exist, go ahead and compile it.
COMPIL GLIDX.BAS
RUN GLIDX.BAS
;
; EXIT keeps user from dropping into NOFILE routine.
EXIT ***Returning you to AMOS command level.***
;
;NOFILE
; If file doesn't exist, send user to text editor to create it.
PAUSE To create file, hit RETURN; otherwise, type anything else.
VUE GLIDX.BAS
Y
```

ERRORS:

EXIT displays no error messages.

CHARACTERISTICS:

May be used only in a command file or DO file.

Terminates the processing of a command file and then (optionally) displays a message to the user of the command file.

Returns the command file user to AMOS command level.

FUNCTION:

Allows you to compare the contents of two files.

HINTS/RESTRICTIONS:

All numeric displays are in octal (or hexadecimal, if you have previously used the SET HEX command).

FORMAT:

.FILCOM Filespec1,Filespec2 N ↵

where Filespec1 and Filespec2 specify the two files you want to compare. FILCOM stops the comparison when it reaches the Nth difference between the two files.

DEFAULTS:

The default file extension is .PRG.

If you do not specify N, FILCOM compares every byte in the two files.

OPERATION:

1. Type FILCOM followed by the specifications of the two files you want to compare. (Separate the two Filespecs with a comma). Now enter the number of differences (a decimal number) at which you want the comparison to stop. Hit a RETURN:

```
.FILCOM DSKRED[1,4],DSKRED[100,3] 5 ↵
```

2. Now you see a display that lists one line for each data mismatch between the two files. Each line of the display gives the following information: 1. the address of the mismatch (in number of bytes from the front of the file); 2. the specification of the first file; 3. the data in the first file at that position; 4. the specification of the second file; and, 5. the data in the second file at that position. For example:

```
.FILCOM CRLF.MAC,WRKFIL.MAC 3 ↵
0       CRLF.MAC 000000          WRKFIL.MAC 000204
2       CRLF.MAC 133700          WRKFIL.MAC 010527
6       CRLF.MAC 000116          WRKFIL.MAC 000054
```

The first line of the display above gives the relative address

of the first data mismatch (the first byte of the files; byte number zero); the specification of the first file (CRLF.MAC); the data in that file at byte number zero (000000); the specification of the second file (WRKFIL.MAC); and the data in that file at byte number zero (000204).

ERRORS:

You may see the usual system error messages when using this command; in particular:

?File specification error

FILCOM doesn't understand your command line. For example:

```
.FILCOM SUBJCT.OBJ,↵
?File specification error
```

Check your command line against the FORMAT section above.

Filespec NOT FOUND

FILCOM couldn't find one or both of the files you specified. Check your spelling; next check your account specifications (use the DIR Filespec[] command to find the location of the files).

CHARACTERISTICS:

Compares two files and displays the data mismatches between them.

Returns your terminal to AMOS command level.

Displays data in octal form (or hexadecimal if SET HEX is in effect).

fildmp

FUNCTION:

Displays the contents of a sequential file in numeric form.

HINTS/RESTRICTIONS:

Displays the data in the file in octal form (or hexadecimal, if the SET HEX command is in effect), 16 bytes per screen line.

FORMAT:

.FILDMP Filespec (RET)

where Filespec specifies the file whose contents you want to see.

DEFAULTS:

The default file specification is the account and device you are currently logged into and a file extension of .OBJ.

OPERATION:

1. Type FILDMP followed by the specification of the file whose data you want to see. Hit a RETURN:

   .FILDMP INIT.PRG[100,3] (RET)

2. Now you see a numeric display of the data in the file, 16 bytes per line (3 digits per byte). For example:

   ```
   057 124 124 114 040 106 111 114 104 115 120 040 050 103 157 156
   164 047 144 051 015 012 057 114 040 066 015 012 106 125 116 103
   ^C
   .
   ```

   To freeze the display, type a Control-S; to resume it, type a Control-Q. To interrupt the display, type a Control-C.

ERRORS:

You may see several of the common system error messages when using FILDMP. For example:

?Cannot OPEN Filespec - file not found
FILDMP cannot find the file you specified. Check your spelling and the account and device specification.

(Changed 30 April 1981)

?Cannot OPEN Filespec - file type mismatch
You tried to dump a random file. To dump a random file, use the DUMP command.

?File specification error
FILDMP did not understand the format of your command line (e.g., .FILDMP).

CHARACTERISTICS:

Gives a numeric display of the data in the specified sequential file.

(Changed 30 April 1981)

filtap

FUNCTION:

Writes copies of disk files to magnetic tape. A file-oriented disk backup program.

HINTS AND RESTRICTIONS:

Used for file-oriented disk backup. Writes files to tape along with their device and account specifications. Also writes date and time of backup. Used in combination with TAPFIL (to transfer files from tape back to disk) and TAPDIR (to see list of files on a tape). For more information on FILTAP, TAPDIR, and TAPFIL, see The Magnetic Tape File Backup Programs, in the "System Operator's Information" section of the AMOS Software Update Documentation Packet.

Allows you to perform one backup on multiple tape reels.

A wildcard file command. (See Chapter 9, "Wildcard File Commands," of the AMOS User's Guide (DWM-00100-35), for information on wildcard specifications.)

You may back up files from any disk account onto tape whether or not the account is within the project you are logged into. Although FILTAP writes the disk specification of the file to the magnetic tape along with the file, it does not transfer any password that may be associated with that disk account.

Always use the /NOAPPEND switch when you are writing to a blank (e.g., new) tape. (/NOAPPEND tells FILTAP to start writing files at the beginning of the tape; the /APPEND switch tells FILTAP to start writing files at the end of any existing files on the tape.)

Not for transferring data between Alpha Micro and non-Alpha Micro computers-- use TAPE for that purpose. (See the TAPE reference sheet.)

Because FILTAP writes the date and time of backup to the tape, before you use FILTAP you should remember to use the DATE and TIME programs to make sure that the system date and time are set.

FORMAT:

.FILTAP Filespec1{/switches}{,Filespec2{/switches}{,...}} RET

where Filespec specifies the files you want to back up onto tape, and /switches are option requests.

(30 April 1981)

DEFAULTS:

The default file specification is *.* and the account and device you are logged into. The default switches are /NOQUERY and /APPEND. The default magnetic tape drive device specification is MTU.

OPTIONS:

Use the switches below to select FILTAP options. Each switch must begin with a slash; remember that the placement of the switch on the command line modifies its effect.

/QUERY or /Q	Ask user for confirmation before copying files (file switch).
/NOQUERY or /NOQ	Don't ask for confirmation before copying files (default, file switch).
/APPEND or /A	Write files to tape at the end of existing files (default, operation switch).
/NOAPPEND or /NOA	Write files starting with beginning of tape (operation switch).

OPERATION:

1. Type FILTAP followed by the specification selecting the files you want to back up onto tape; then type a RETURN. For example:

   .FILTAP MEMO.TXT,SCHDLE.TXT[310,2] (RET)

   Now FILTAP asks you:

   Enter tape unit number:

   Enter the device code and unit number of the magnetic tape drive containing the tape reel you want to access. The default device code is MTU.

2. Now FILTAP tells you what files it is transferring. For example:

   MEMO.TXT to MTU2:MEMO.TXT
   SCHDLE.TXT[310,2] to MTU2:SCHDLE.TXT[310,2]
   Total of 2 files transferred

3. Remember that you can use wildcard file specifications. For example, suppose you want to back up all .BAS files on the disk:

   .FILTAP *.BAS[] (RET)

4. When you use the /QUERY switch, FILTAP asks for confirmation before each transfer. Enter Y for yes, or N for no; do not type a RETURN after your answer. Remember that the placement of the /QUERY switch on the command line can affect which files it applies to. for example:

```
.FILTAP *.MAC,/QUERY *.BAS,*.TXT,*.PRG/NOQUERY,*.LST (RET)
Enter tape unit number: 1 (RET)
TERML.MAC to MTU1:TERML.MAC
C40.MAC to MTU1:C40.MAC
LSTSQR.BAS to MTU1:LSTSQR.BAS?Y
NEW.TXT to MTU1:NEW.TXT?N
STATUS.PRG to MTU1:STATUS.PRG
NEW.LST to MTU1:NEW.LST?Y
BMAC.LST to MTU1:BMAC.LST?Y
Total of 6 files transferred
```

You may enter a Control-C at any time to prevent further transfers.

5. If FILTAP cannot fit all of the files you specified on one tape, it will permit you to continue the backup on another reel. When the current tape is full, FILTAP displays:

```
%Tape is full, please mount another tape then type RETURN to
%  continue, or type Control-C to abort copy
```

If you wish to continue backing up files on another reel of tape, wait for the current tape to finish rewinding, mount a new reel of tape, then type RETURN on your teminal; the backup will continue on the new reel. If you want to abort the backup, type a Control-C.

ERRORS:

?Cannot find DSKO:SCNWLD.SYS[1,4] or MEM:SCNWLD.SYS

FILTAP needs this file to be able to process wildcard symbols in your file specifications. This message can mean that SCNWLD.SYS does not exist or that you do not have enough memory to load the file into your partition.

?Cannot READ Devn: - device does not exist
?Cannot READ Devn: - device is not mounted

You tried to copy to or from a device that is not listed in the DEVTBL command in your SYSTEM.INI, does not have a driver in account [1,6] of the System Disk, is not file-structured, or is not mounted. ("Devn:" is the device you specified.)

%No file-oriented device corresponding to Devn: is mounted

You specified a device, but left off the unit number. FILTAP tried to find a logical unit that matched the device code you specified, but failed to do so. Try mounting the device.

%Tape is full, please mount another tape then type RETURN to
% continue, or type Control-C to abort copy

There is no more room on the current reel of tape. Mount another reel and type RETURN to continue the backup process, or type a Control-C to abort the backup procedure.

CHARACTERISTICS:

A file-oriented disk backup program that allows you to copy disk files to magnetic tape. See reference sheets on TAPFIL and DIRTAP to see how to transfer files from tape to disk, and to display a directory of a magnetic tape.

Not for transferring data between Alpha Micro and non-Alpha Micro computers-- for that purpose, use the TAPE program.

Allows you to make one backup on multiple reels of tape.

Accepts wildcard file specifications. Default file specification is *.* and device and account you are logged into. Default switches are: /NOQUERY/APPEND. Default magnetic tape drive device specification is MTU.

Writes date and time of backup to tape.

fix

FUNCTION:

Symbolic debugger. Allows you to examine and execute your assembly language program and data structures in a controlled manner.

HINTS/RESTRICTIONS:

FIX is a screen-oriented debugger. Its operation is very similar to the VUE screen editor. It has two modes, Display Mode and Command Mode. In Display Mode you can examine your program and data areas, and you may single-step or proceed through portions of the program. In Command Mode you may examine and modify the AM-100 registers, examine data structures, and set up the debugging environment. You may toggle between the two FIX modes by pressing the ESCAPE key.

Both input and output values may be either in symbolic or numeric form. If FIX finds a .SYM file for the program you are debugging, it will attempt to display all values in symbolic form. Either octal or hex form may be used for numeric values; use the SET command in Command Mode to change the current radix.

To access a local symbol, specify the non-local symbol that precedes the local symbol, enter a space, then enter the local symbol. For example, in using the Search command:

>LABEL 10$ (RET)

where LABEL is the non-local symbol that precedes the local symbol 10$.

For more information, see the AlphaFIX User's Manual, (DWM-00100-69).

NOTE: To exit FIX, enter Command Mode. Now type a Q followed by a RETURN.

FORMAT:

.FIX Filespec (RET)

where Filespec selects the file that you want to debug.

DEFAULTS:

If you do not specify a file extension, FIX assumes that the file is a .PRG file. Other defaults are set by the INI.FIX initialization file.

(Changed 30 April 1981)

OPERATION:

1. Type FIX followed by the specification of the file you want to debug. Then type a RETURN. For example:

   .FIX REMOVE (RET)

   FIX loads the file into your memory partition (unless it is already there). It also attempts to locate the corresponding .SYM file and load it into memory.

2. FIX then enters Command Mode, prints the FIX status and the AM-100 registers. It then prompts you with a right angle bracket >. You may now enter FIX commands, or press ESC to enter Display Mode.

3. To exit FIX, enter Command Mode and type a Q followed by a RETURN. FIX will delete all memory modules it created (including the program you were debugging) and return you to AMOS command level.

COMMAND SUMMARY:

Below is a partial list of the FIX commands; refer to the FIX User's Manual for a full list of commands.

Display Mode:

CONTROL-J	Move cursor down one instruction
CONTROL-K	Move cursor up one instruction
CONTROL-T	Move cursor down a page (24 instructions)
CONTROL-^	Move cursor to current PC location
CONTROL-P	Toggle Breakpoint
CONTROL-X	Proceed to Breakpoint
RETURN	Single-step

Command Mode:

S	Search for symbol	Q	Leave FIX
R	Modify register	HELP	Display menu
SPEC	Set up command string	GO	Execute AMOS command

ERRORS:

If FIX does not recognize one of your commands in Command Mode, it displays the message What?

(Changed 30 April 1981)

CHARACTERISTICS:

Allows you to set breakpoints, modify and examine contents of registers, display and execute an assembly language program.

(Changed 30 April 1981)

fixdvr

FUNCTION:

Configures a floppy disk driver for a specific combination of floppy disk format, disk drive, and disk controller.

HINTS/RESTRICTIONS:

Because of the large number of possible permutations of floppy disk formats, drives and controllers, it is no longer possible to provide a separate disk driver program for each combination.

For information on the large number of floppy disk formats now available, refer to the documents Disk Drivers and Formats, (DWM-00100-32), and Configuring Floppy Disk Drivers, (DWM-00100-47), in the AM-100 documentation packet.

You MUST use FIXDVR to configure floppy disk drivers for each combination of format, drive and controller you have on your system.

NOTE: You may not use single-density AMS format if you are going to be running the drive under control of the AM-210 disk controller.

Remember to add a BITMAP command in the system initialization command file for each new format/drive/controller combination you create. You may also need to modify the DEVTBL command line.

FORMAT:

.FIXDVR (RET)

OPERATION:

1. Log into the system Device Driver Library account:

   .LOG DSK0:[1,6] (RET)

2. Type FIXDVR followed by a RETURN:

   .FIXDVR (RET)

3. FIXDVR now begins to ask you a series of questions:

   a. Controller type (A) AM-200, (B) AM-210, or (C) Icom:
      Enter the letter A, B, or C to select the type of floppy disk controller you are using for your floppy disk drive. If you select the Icom controller, FIXDVR now skips down to question #d (see below).

(Changed 30 April 1981)

b. Drive type (A) Persci, (B) Wangco, or (C) CDC:

Enter the letter that selects the type of disk drive you are using. You may not select CDC if you selected AM-200 above.

c. Double-density?

FIXDVR asks this question only if you have already selected the AM-210 as your disk controller. Enter a Y for Yes or an N for No, depending on whether or not you want to use double-density disks.

d. Format (A) STD, (B) AMS, or (C) IMG:

Enter the letter that selects the disk format you want the driver to use. You may not specify the AMS format if the driver is to use the Icom controller or if it is to use single-density format on a drive running under control of the AM-210 controller. If you have previously selected the Icom controller, FIXDVR now skips down to question #f.

e. Double-sided?

FIXDVR asks this question only if you have previously selected the AM-210 controller. Enter a Y for Yes or an N for No, depending on whether you plan to use the driver on double-sided disks.

f. Enter new driver name:

Enter the name you want to assign to the driver program. Each driver must have a unique, three-character name.

4. After you have finished answering its questions, FIXDVR configures the driver and says:

New driver is now in memory, bitmap size is nn

where nn is the number of words you must assign to the bitmap for the new device.

5. Although the new driver is in memory, it is not yet a file on the disk. Save it to the disk by using the SAVE command. For example:

.SAVE DDS.DVR RET

If you do not specify an extension, SAVE saves the file under the .DVR extension (indicating a device driver program).

6. If you are adding a new device to the system, remember to modify the system initialization command file to change the system device table and bitmap areas. (See the reference sheets for DEVTBL and BITMAP.)

ERRORS:

?Could not find xxxxxx.DVR
FIXDVR couldn't find the necessary file. If you are configuring a driver for the AM-200, FIXDVR requires that 200DVR.DVR be in DSK0:[1,6]; for the AM-210 you need 210DVR.DVR in DSK0:[1,6]; for the Icom controller you need ICMDVR.DVR in DSK0:[1,6].

Please enter Y or N
Several of the FIXDVR questions understand only an answer of Y or N for Yes or No.

?Invalid response
Several of the questions that FIXDVR asks require that you enter a letter to select an option. Check your typing.

?Invalid device
You have a bad version of 200DVR.DVR, 210DVR.DVR, or ICMDVR.DVR in DSK0:[1,6].

?Icom does not support AMS format
You tried to format an Icom floppy diskette in AMS format. (Icom floppy drives only support STD and IMG format.)

?AM-210 does not support single-density AMS format
You may not use single-density AMS format on a device that runs under control of the AM-210 floppy disk controller.

?AM-200 does not support CDC floppy disks
You may only run CDC floppy disks under the control of the AM-210 floppy disk controller.

CHARACTERISTICS:

Requires that you be logged into DSK0:[1,6].

Configures a new device driver in memory: use the SAVE command to save the driver as a disk file.

(Changed 30 April 1981)

fixmtm

FUNCTION:

Configures the program that drives the Multiterm printer so that it runs the printer with a particular I/O interface controller and connected to a specific port on that board.

HINTS/RESTRICTIONS:

Because the Multiterm driver, DSK0:MTM.DVR[1,6], does not operate through the terminal service system, you must configure it to reflect the particular I/O controller and I/O port on which you are going to run the Multiterm printer.

You must be logged into account DSK0:[1,6] to use FIXMTM.

NOTE: When you are using MTM.DVR[1,6], no other device may use the AM-300 interface controller. That means that no terminals or printers may run on the AM-300 that the Multiterm is connected to when you are printing on the Multiterm.

FORMAT:

.FIXMTM ↵

OPERATION:

1. Log into account DSK0:[1,6]:

   .LOG DSK0:1,6 ↵

2. Enter FIXMTM followed by a RETURN:

   .FIXMTM ↵

3. If the MTM driver exists in DSK0:[1,6], you now see a message that tells you how the MTM driver is currently configured (that is, what I/O interface and what port of that interface board the driver is set up to use). The interface board and port number will vary depending on your particular system, but you might see a message something like this:

   CURRENT DRIVER IS SET UP FOR AM-300 ON OCTAL PORT 375
   ENTER C TO CHANGE IT OR L TO LEAVE IT ALONE:

   Enter a C or an L.

4. If you tell FIXMTM to leave the MTM driver alone, FIXMTM returns your terminal to AMOS command level. Otherwise, you see:

   ENTER P FOR 3P+S, S FOR SIO, OR A FOR AM-300 INTERFACE:

   Enter a P, S, or A to select the specific I/O interface controller to which your Multiterm printer is going to be connected.

5. You now see a message asking you for the number of port (in octal) to which the printer is going to be connected:

   ENTER OCTAL PORT ADDRESS OF INTERFACE (0-377):

6. FIXMTM now modifies the driver and returns a message that tells you the configuration of the new driver. For example:

   MTM DRIVER MODIFIED TO RUN ON 3P+S ON OCTAL PORT 100

   FIXMTM returns you to AMOS command level.

ERRORS:

If FIXMTM cannot find the file MTM.DVR, it tells you:

CANNOT FIND MULTITERM DRIVER MTM.DVR
BE SURE YOU ARE LOGGED IN UNDER 1,6 AND TRY AGAIN

Either MTM.DVR is missing in account DSK0:[1,6] or you are not logged into DSK0:[1,6].

CHARACTERISTICS:

Requires that you be logged into DSK0:[1,6].

Returns your terminal to AMOS command level.

fmt200

FUNCTION:

Formats diskettes used in floppy disk devices that run under the control of the AM-200 Floppy Disk Controller board.

HINTS/RESTRICTIONS:

IMPORTANT NOTE: When you run FMT200, your job must be the only job running on the system.

FMT200 selects the format to use based on the device specification you provide. For example, if you have previously configured a floppy disk driver named AMS.DVR for a Persci device that uses AMS format, the device specification of AMS0: tells FMT200 to use AMS format to format the diskette in Drive Zero of that device. (For information on configuring floppy disk drivers, see the document Configuring Floppy Disk Drivers in the "System Operator's Information" section of the AMOS Software Update Documentation Packet.)

Before formatting, make sure that the format-enable switch on the AM-200 is set to "EN." As it formats, FMT200 writes over any data currently on the diskette.

FMT200 does not require that you mount a diskette before formatting it.

After formatting a diskette, use the SYSACT command to initialize the diskette (unless you are going to use DSKCPY to make a literal image of another diskette on the newly formatted diskette).

FORMAT:

.FMT200 Devn: RET

where Devn: specifies the device holding the diskette you want to format.

OPERATION:

1. Type FMT200 followed by the specification of the device holding the diskette you want to format. Then type a RETURN. For example:

   .FMT200 STD1: RET

   The command above tells FMT200 to format the diskette in Drive One of the floppy device that uses the STD driver program (and therefore, to use the STD format).

(Changed 30 April 1981)

2. Now you see the message:

   BEGIN FORMATTING

3. When FMT200 is finished, you see the message:

   EXIT

4. FMT200 now returns you to AMOS command level.

ERRORS:

You may see the following error messages when using FMT200:

?Unit number must be 0-3
: You specified a bad unit number.

?Invalid device
: The device you specified is not a floppy disk device, or is not compatible with the AM-200 Floppy Disk Controller.

[Error code x on track n]
: A media, drive, or controller error occurred. For information on the error codes, refer to the hardware documentation accompanying your disk devices.

CHARACTERISTICS:

Formats diskettes used in floppy disk devices that run under control of the AM-200 Floppy Disk Controller board. (Make sure that the format-enable switch on the AM-200 is set.)

Selects the disk format to use on the basis of the device specification you provide.

When you run FMT200, make sure that your job is the only one running on the system.

As it formats, writes over any data currently on the diskette.

(Changed 30 April 1981)

fmt210

FUNCTION:

Formats diskettes used in floppy disk devices that run under the control of the AM-210 Floppy Disk Controller board.

HINTS/RESTRICTIONS:

IMPORTANT NOTE: When you run FMT210, your job must be the only job running on the system.

The AM-210 Floppy Disk Controller handles devices that use double- or single-density diskettes and single- or double-sided diskettes.

NOTE: The AM-210 controller does not support single-density AMS format.

FMT210 selects the specific format to use based on the device specification you provide. For example, if you have previously configured a floppy disk driver named DDS.DVR for a Persci device that uses double-sided, double-density STD format, the device specification of DDS0: tells FMT210 to use STD format to format the diskette in Drive Zero of that device. (For information on configuring floppy disk drivers, see the document Configuring Floppy Disk Drivers, (DWM-00100-47), in the System Operator's Information section of the AMOS Software Update Documentation Packet.)

As it formats, FMT210 writes over any data currently on the diskette.

FMT210 does not require that you mount a diskette before formatting it.

After formatting a diskette, use the SYSACT command to initialize it (unless you are going to use DSKCPY to make a literal image of another diskette on the newly formatted one).

FORMAT:

.FMT210 Devn: RET

where Devn: specifies the device holding the diskette you want to format.

OPERATION:

1. Type FMT210 followed by the specification of the device holding the diskette you want to format. Then type a RETURN. For example:

.FMT210 DDA1: RET

(Changed 30 April 1981)

The command above tells FMT210 to format the diskette in Drive One of the floppy device that uses the DDA driver program (and therefore, to use the AMS format). (The DDA driver handles diskettes in double-density, double-sided AMS format.)

2. Now you see the message:

   BEGIN FORMATTING

3. When FMT210 is finished, you see the message:

   EXIT

4. FMT210 now returns you to AMOS command level.

ERRORS:

You may see the following error messages when using FMT210:

?Unit number must be 0-3
: You specified a bad unit number.

?Invalid device
: The device you specified is not a floppy disk device, or is not compatible with the AM-210 Floppy Disk Controller board.

[Error code x on track n]
: A media, drive, or controller error occurred at track n of the diskette. For information on the error codes, refer to the hardware documentation that accompanied your disk device.

[No interrupt received after WRITE TRACK]
: Indicates either a bad drive or a bad controller board.

CHARACTERISTICS:

Formats diskettes used in floppy disk devices that run under control of the AM-210 Floppy Disk Controller board.

When you run FMT210, make sure that your job is the only one running on the system.

Selects the disk format to use on the basis of the device specification you provide.

As it formats, writes over any data currently on the diskette.

(Changed 30 April 1981)

fmt400

FUNCTION:

Formats disks used in hard disk devices that run under the control of the AM-400 Hard Disk Interface (e.g., Century Data Trident).

HINTS/RESTRICTIONS:

Formats the disk into physical records of 512 bytes.

As it formats the disk, FMT400 writes over any data currently on that disk.

IMPORTANT NOTE: When you run FMT400, your job must be the only job running on the system.

FMT400 does not require that you mount the disk before formatting it.

After formatting a disk, use the SYSACT command to initialize the disk (unless you are going to use DSKCPY to make a literal image of another disk on the newly formatted disk.).

The AM-400 communicates with the Century Data 1150A formatter. Each Century Data Trident contains a number of logical units within one physical device. For example, the T-300 contains 19 logical units (DSK0:-DSK18:). You must format each logical unit separately as if it were a separate disk.

NOTE: If you are not using the Trident hard disk as your System Device, when you reset or turn on the machine you must run the appropriate initialization program (e.g., T80INI or TRIINI) before using the device.

FORMAT:

.FMT400 Devn: (RET)

where Devn: specifies the device holding the disk you want to format.

OPERATION:

1. Type FMT400 followed by the specification of the device holding the disk you want to format. Then type a RETURN. For example:

.FMT400 DSK1: (RET)

(Changed 30 April 1981)

The command above tells FMT400 to format the disk in Drive One of the System Device.

2. Now you see the message:

   BEGIN FORMATTING

3. When FMT400 is finished, you see the message:

   EXIT

4. FMT400 now returns you to AMOS command level.

ERRORS:

You may see the following error messages when using FMT400:

DRIVE NOT READY
Make sure that the drive has completed its power-up cycle.

CANNOT REZERO DRIVE
ERROR AFTER RETRY
COMMAND STATUS ERROR FOR COMMAND n
DRIVE STATUS ERROR FOR COMMAND n
The messages above are an indication of hardware problems; specifically, drive or interface failure.

CHARACTERISTICS:

Formats disks used in hard disk devices that run under control of the AM-400 Hard Disk Interface board.

As it formats, writes over any data currently on the disk.

When you run FMT400, make sure that your job is the only one running on the system.

(Changed 30 April 1981)

fmt500

FUNCTION:

Formats disks used in hard disk devices that run under the control of the AM-500 Hard Disk Controller (e.g., CDC Hawk hard disk).

HINTS/RESTRICTIONS:

Formats the disk into physical records of 512 bytes. Each logical unit contains 9696 records.

Before you format a disk, FMT500 requires that you use the MOUNT command to mount that disk (e.g., .MOUNT DSK2:).

As it formats the disk, FMT500 writes over any data currently on that disk.

IMPORTANT NOTE: When you run FMT500, your job must be the only job running on the system.

After formatting a disk, use the SYSACT command to initialize the disk (unless you are going to use DSKCPY to make a literal image of another disk on the newly formatted disk.).

FORMAT:

.FMT500 Devn: RET

where Devn: specifies the device holding the disk you want to format.

OPERATION:

1. Type FMT500 followed by the specification of the device holding the disk you want to format. Then type a RETURN. For example:

   .FMT500 DSK1: RET

   The command above tells FMT500 to format the disk in Drive One of the System Device.

2. Now you see the message:

   BEGIN FORMATTING

(Changed 30 April 1981)

3. When FMT500 is finished, you see the message:

EXIT

4. FMT500 now returns you to AMOS command level.

ERRORS:

You may see the following error messages when using FMT500:

?Cannot WRITE Devn: - device not mounted
The device holding the disk you want to format is not mounted. Mount the disk by using the MOUNT command (e.g., .MOUNT HWK3:).

?Cannot INIT Devn: - device does not exist]
FMT500 cannot find the device you have specified. Use the DEVTBL command to see if the device is listed as a valid system device.

?File specification error
You did not specify a device to the FMT500 command.

CHARACTERISTICS:

Formats disks used in hard disk devices that run under control of the AM-500 Hard Disk Controller board.

When you run FMT500, make sure that your job is the only one running on the system.

Writes over any data currently on the disk.

FUNCTION:

Allows you to send terminal input to another job.

HINTS/RESTRICTIONS:

FORCE has two modes; single-line and multi-line. Be careful when using FORCE to make sure that the job to whom you are forcing input has enough memory allocated to it to handle the task you are giving it.

You will often use FORCE to force input to a job that is attached to a pseudo-terminal (for example, to bring up the line printer spooler under control of a pseudo-terminal).

FORMAT:

```
.FORCE Jobname Commands-and-data ↵
```

or:

```
.FORCE Jobname ↵
Commands-and-data ↵
Commands-and-data ↵
                  ↵
```

where Commands-and-data are any valid AMOS commands or data that you can enter from the keyboard.

OPERATION:

1. Type FORCE followed by the name of the job to whom you wish to force input. If you now type a RETURN, you inform AMOS that you want to use FORCE in multi-line mode:

```
.FORCE JOB2 ↵
```

Now type one or more lines of commands and data. To end the FORCE command, enter two RETURNs (i.e., make a blank line). For example:

```
.FORCE JOB3 ↵
LOG DSK0:110,5 ↵
TXTFMT HEADER ↵
              ↵
```

2. To use FORCE in single-line mode, enter a single line of commands or data after the Jobname on the FORCE command line:

```
.FORCE JOB2 RUN SINE )
```

ERRORS:

You may see any of the system error messages that can result from the commands and data that you've forced to the other job.

You may also see the following:

[NONEXISTENT JOB]
FORCE doesn't recognize the name of the job to whom you are trying to force input. Check your spelling; then use the SYSTAT command to see what jobs are on the system.

[NO TERMINAL ATTACHED TO JOB]
You need to use the ATTACH command to assign a terminal to the job to whom you want to force input. You can assign a pseudo-terminal if you do not need to see any output from that job.

CHARACTERISTICS:

Forces input to another job on the system.

Make sure that the job to whom you are forcing input has enough memory allocated to it to handle the tasks.

Returns your terminal to AMOS command level.

global

FUNCTION:

Takes a group of assembly language object (.OBJ) files and produces an alphabetic cross reference which lists all global symbols in the files, and shows which files define them and which files accept them as externally defined symbols.

HINTS/RESTRICTIONS:

The Alpha Micro macro-assembler, MACRO, allows you to segment assembly language programs. You can assemble these segments separately, and then link them together with LINK, the linkage editor. Each segment is able to reference symbols that occur in other segments. MACRO accomplishes this by allowing you to use the INTERN and EXTERN statements.

Let's say that you define a symbol within a file. By using the INTERN statement, you can make the symbol available to other segments that will be linked together with that file. Other files that reference a symbol defined in another segment must contain an EXTERN statement for that symbol so that LINK knows that a definition for that symbol exists in another segment. MACRO also allows you to overlay a portion of a program with a program from the disk; the OVRLAY statement accomplishes this. For each OVRLAY statement, an INTERN statement must exist in another segment which identifies the start of the overlay in that segment.

GLOBAL produces a listing file that contains a cross reference of all symbols that have been referenced in an INTERN, EXTERN, or OVRLAY statement, so that you can see in which segments those references occur. (For information on INTERN, EXTERN, OVRLAY, and MACRO, see AMOS Assembly Language Programmer's Reference Manual, (DWM-00100-43).

GLOBAL produces the listing file in the account and device you are logged into. The listing file bears a .GLB extensions and the name of the first segment specified on the GLOBAL command line.

FORMAT:

.GLOBAL{/switches} Filespec1,Filespec2{,....FilespecN} RET

where /switches are optional and affect format of the information in the listing file. Filespec1...FilespecN is a list of .OBJ files for which you want a global cross reference. If there are too many files to fit on one command line, you may end the command line with a comma. Then you can resume the command line on the next line.

(1 May 1980)

You may enter as many lines of Filespecs as you wish, as long as you end each preceding line with a comma.

DEFAULTS:

If you omit portions of a file specification, GLOBAL assumes a file extension of .OBJ and the device and account you are logged into.

OPTIONS:

You may request the following options by including the appropriate switches on your command line. (Each switch must begin with a slash.)

Line width options (default is 80 characters):

/W — Wide listing (same as /w:130). Produces a listing file that may have up to 130 characters on a line.

/W:n — Characters per line, where n specifies the number of characters.

Page length (default is 60):

/L — Long listing (same as /L:80).

/L:n — Lines per page, where n specifies number of lines.

OPERATION:

1. Type GLOBAL followed by optional switches. Then supply a list of the Filespecs that specify the segments for which you want a global cross reference. Type a RETURN. For example:

```
.GLOBAL/W/L MAIN,SUB1,SUB2 (RET)
```

2. If you have more file specifications than will fit on one command line, end the line with a comma. GLOBAL now prompts you with an asterisk:

```
.GLOBAL MAIN,SUB1,SUB2,SUB3, (RET)
*
```

   Enter the next line of file specifications. You may enter as many lines as you want, as long as all preceding lines end with a comma.

3. GLOBAL reads each file you specify and builds a table of global symbols in memory. As it works, GLOBAL tells you which file it is processing. For example:

```
PROCESSING MAIN.OBJ
PROCESSING SUB1.OBJ
PROCESSING SUB2.OBJ
```

4. After processing all files, GLOBAL creates a disk file that contains the global cross reference. As it works, GLOBAL displays the name of the file it is building, and prints a dot each time it outputs a disk block. For example:

```
BUILDING MAIN.GLB .....
```

This file has the same name as the first segment you specified to the GLOBAL command and a .GLB extension.

5. When GLOBAL is finished, it prints the following and returns you to AMOS command level:

```
GLOBAL FILE FINISHED
.
```

6. Here is a portion of what a GLOBAL listing file might look like:

```
.TYPE MAIN.GLB (RET)

Global Cross-Reference (Version 1.0)

        M S S
        A U U
        I B B
        N 1 2
        - - -
ALPHA   I E .
BETA    I . E
ZETA    I O .
```

The listing file above tells us: 1) the symbol ALPHA appeared in an INTERN statement in the file MAIN and in an EXTERN statement in file SUB1; 2) the symbol BETA appeared in an INTERN statement in MAIN and in an EXTERN statment in SUB2; 3) the symbol ZETA appeared in an INTERN statement in MAIN and in an OVRLAY statement in SUB1.

ERRORS:

?Undefined switch /x - Ignored
You specified an invalid switch. The only switches GLOBAL recognizes are the /L or /W switches.

CHARACTERISTICS:

Assumes a file specification extension of .OBJ.

Produces a listing file with the same name as the first file specified in the command line and an extension of .GLB.

Returns your terminal to AMOS command level.

goto

FUNCTION:

Allows transfer of control within a command file.

HINTS/RESTRICTIONS:

Used in combination with the LOOKUP and EXIT command file commands, GOTO allows you to set up conditional transfers within a command file based on whether or not a specified file exists. (See the LOOKUP and EXIT reference sheets.)

You may only use GOTO within a command or DO file. For more information on branching within a command file, see New Features of Command Files and DO Files, (DWM-00100-63), in the "User's Information" section of the AM-100 documentation packet.

GOTOs must precede the labels they branch to-- that is, GOTO statements may only transfer forward in the command file.

FORMAT:

GOTO Argument{;comment}

where Argument is the name of the label that selects the portion of the command file you want to branch to. The optional comment must begin directly after the Argument. (That is, there must be no space between the end of the Argument and the semicolon that begins the comment.) The Argument must not end with trailing blanks-- arguments must end either with a RETURN or a semicolon (which designates the start of a comment).

The label referenced by Argument must exist in a later portion of the command file. The label uses this format:

LABEL

or:

;LABEL

The label must be the only thing on its command line. If the label does not begin with a semicolon, it is not a comment; therefore it must be an executable command file element (e.g., a program name or a command file specification). (NOTE: If the label is a comment, the Argument on the GOTO command line that referenced it must not begin with a semicolon, even though the label does.)

You may begin a label with a semicolon, spaces, or spaces followed by a semicolon; these are all ignored when GOTO compares the Argument to the label it selects. (Remember that if a label begins with a semicolon, you must not place any spaces between the semicolon and the characters that make up the rest of the label. That is, ";NOFILE" is a valid label, but "; NOFILE" is not.)

OPERATION:

1. At the place in your command file where you want to transfer control, enter GOTO followed by the argument that is the name of the label you want to branch to. For example:

```
GOTO NOFILE
```

2. At a later point in your command file, designate the portion of the command file you want to branch to by placing the appropriate label on the line before that section. For example:

```
;NOFILE
:<Couldn't find that file; enter RETURN to create it:>
```

3. You will probably want to use the GOTO command together with the LOOKUP and EXIT commands to perform conditional branching within the command file. For example:

```
LOOKUP INSTAL.MAC/?Couldn't find file
GOTO NOTFOUND
;
; If couldn't find file, GOTO NOTFOUND error routine.
; If did find it, assemble it.
TRACE ON
MACRO INSTAL
;
LINK INSTAL,SUB1,SUB2
INSTAL.PRG
EXIT *Returning you to AMOS command level*
;
;NOTFOUND
:<
If you want to create the file, type a RETURN;>
PAUSE Otherwise, type anything else.
;
VUE INSTAL.MAC
Y
```

ERRORS:

If the label you supply is not a valid comment or executable command file element, GOTO resumes executing the command file after the label.

If GOTO is not able to find the specified label, you see:

?Label not found

and you are returned to AMOS command level. This message means that you did not supply a valid argument and/or label. Make sure that your argument matches the label you want to branch to, and that the label and arguments are in proper form.

CHARACTERISTICS:

Allows transfer of control within a command or DO file. When used with LOOKUP and EXIT statements, permits conditional branching.

May be used only within a command file or DO file.

hasher

FUNCTION:

Generates a hash total for a specified disk.

HINTS/RESTRICTIONS:

A hash total is a number that is computed based on the characteristics of a group of data; the hash total thus uniquely identifies that group of data. HASHER generates a hash total for a disk; this number is based on the contents of the disk. Therefore, two disks only have the same hash total if they contain identical data.

HASHER is especially useful when you are making multiple copies of a disk. You can use HASHER to generate the hash total of the source disk. Then, every time you use DSKCPY to make copies of that disk, you can tell DSKCPY to generate a hash total for the disk copied to. If the hash total for the source disk does not match the hash totals for the disks copied to, a problem occurred during the disk copy.

NOTE FOR HAWK HARD DISK USERS:

Since you will often use HASHER to generate a hash total that you can compare against a hash total generated by DSKCPY, you should remember that DSKCPY can generate two different hash totals for the same disk, depending on whether you tell DSKCPY to use the fast copy mode or the slower (/0) copy mode.

To be compatible with DSKCPY, HASHER can also generate two different hash totals for a Hawk disk, using either the DSKCPY fast copy mode or the DSKCPY /0 mode.

Therefore, before using HASHER on a Hawk disk, you must decide whether you want the hash total computed via the fast copy technique or the /0 method. (The fast copy method is the default; to specify the /0 method, use the /0 switch on the HASHER command line.) If you generate a hash total using the fast copy method, no other users may run on the system while the hash total is being calculated.

See the DSKCPY reference sheet for more information on the two DSKCPY copy modes for the Hawk device.

FORMAT:

.HASHER{/switch} RET

where /switch selects a special mode for Hawk devices.

(Changed 30 April 1981)

DEFAULTS:

If you omit the optional /0 and you have specified a Hawk hard disk, HASHER assumes that you want the hash total computed via the technique DSKCPY uses when operating in Hawk fast copy mode. (The /0 switch has no effect when computing the hash total for other disks.)

OPERATION:

1. To generate a hash total for a floppy disk or Phoenix hard disk, first enter the command HASHER:

   .HASHER (RET)

   Now enter the specification of the device that contains that disk when you see:

   Input drive: PH01: (RET)

   As HASHER works, you see:

   [Hashing nnnn blocks]

   where nnnn is the number of blocks on the disk. When it is finished, you see:

   Hash is nnn

   where nnn is the computed hash total.

2. To generate a hash total for a Hawk disk, enter HASHER. If you want this hash total to be computed in the same way that DSKCPY generates a hash total when using the Hawk fast copy mode, do not use the /0 switch; just type a RETURN. Then enter the input drive:

   .HASHER (RET)
   Input drive: HWK1: (RET)

   Now you see:

   %All other users will be suspended while HAWK hash is running.
   Hit return to continue or control-C to abort:

   If any other user is running on the system, type a Control-C to exit HASHER. When you are sure no other users are on the system, you may use HASHER again. For example:

```
.HASHER (RET)
Input drive: HWK1:(RET)
%All other users will be suspended while HAWK hash is running.
Hit return to continue or control-C to abort:
[Hashing 9696 blocks]

Hash is 627
```

If you want HASHER to generate a hash total for a Hawk disk in the same way that DSKCPY generates a hash total when using the /0 copy mode, include the /0 switch on the HASHER command line. Remember to end the command line with a RETURN. For example:

```
.HASHER/0 (RET)
Input drive: HWK3:(RET)
[Hashing 9696 blocks]

Hash is 432
```

ERRORS:

You may see the following HASHER error messages:

?Invalid switch, please use one or more of the following:
/0 use old (slow) copy method for the AM-500

You specified a switch on the command line other than /0. Enter the command line again.

?Driver not found

HASHER couldn't find the device driver program for the specified device. That means that the driver was not in system memory, user memory, or DSK0:[1,6]. Check with the System Operator for help.

?Disk size not defined in table

HASHER doesn't know the number of disk blocks per disk for the devices you are trying to copy between. This means that you are using a device that HASHER doesn't know about. Check with the System Operator for help.

There are also several AM-500 hard-disk controller "hard error" messages that it is possible but extremely unlikely that you would ever see. (If any of these do occur, please contact Alpha Micro.)

?Sector not found during disk hash - drive N block X

where drive N is a decimal number and block X is an octal or hexadecimal number. This can only happen when you are using the fast copy mode of DSKCPY or HASHER on a Hawk hard disk system. The

contents of a disk sector have become unformatted. Verify if there is a problem by using REDALL to diagnose the disk and report any read errors. (See the REDALL reference sheet for more information about disk diagnostic tests.)

?CRC error during disk hash - drive N block X

where drive N is a decimal number and block X is an octal or hexadecimal number. This can only happen when you are using the fast copy mode of DSKCPY or HASHER on a Hawk hard disk system. The Cyclic Redundancy Check device on the AM-500 board has detected a problem in data transmission. First verify if there is a problem by using REDALL to diagnose the disk and report any read errors. (See the REDALL reference sheet for more information about disk diagnostic tests.) The problem can be fixed using DSKDDT, but the data in the block may be lost. (For more information on DSKDDT, see the DSKDDT reference sheet.)

?Sentinel field error during disk hash - drive N block X

where drive N is a decimal number and block X is an octal or hexadecimal number. This can only happen when you are using the fast copy mode of DSKCPY or HASHER on a Hawk hard disk system. The contents of a disk block have become unformatted. Verify if there is a problem by using REDALL to diagnose the disk and report any read errors. (See the REDALL reference sheet for more information about disk diagnostic tests.)

?Undefined error during disk hash - drive N block X

where drive N is a decimal number and block X is an octal or hexadecimal number. This can only happen when you are using the fast copy mode of DSKCPY or HASHER on a Hawk hard disk system. A hard error occurred which was not definable as any of the foregoing AM-500 hard-disk controller errors. Verify if there is a problem by using REDALL to diagnose the disk and report any read errors. (See the REDALL reference sheet for more information about disk diagnostic tests.)

See the documentation that accompanies the AM-500 board for more information on these errors and the conditions they report.

CHARACTERISTICS:

HASHER generates a hash total for a specified disk.

If you want a hash total for a Hawk hard disk, HASHER can compute the hash total using either the DSKCPY fast copy method or the DSKCPY /O method. If you use the Hawk fast copy mode (the default), no other users may run on the system until HASHER is finished.

(Changed 30 April 1981)

hedlod

FUNCTION:

You may optionally use the HEDLOD command in the system initialization command file to set the head load time for the AM-200 floppy disk controller. You may also use HEDLOD at AMOS command level to tell you the head load time set in the SYSTEM.INI.

HINTS/RESTRICTIONS:

When it appears in the system initialization command file, the HEDLOD command sets the number of real-time clock ticks that the AM-200 keeps the disk drive heads loaded after a data transfer. The HEDLOD command must appear after all SYSTEM commands in the SYSTEM.INI.

The HEDLOD command only affects disk drives that run under control of the AM-200 floppy disk controller and that support software control of head load timing (Persci disks only).

FORMAT:

.HEDLOD ↵

or:

HEDLOD n ↵

where n selects the number real-time clock ticks that the AM-200 keeps the disk drive heads loaded after a data transfer.

OPERATION:

1. Type HEDLOD followed by a RETURN to find out the head load time set by the SYSTEM.INI for floppy disk drives that run under the control of the AM-200 and that support software controlled head load timing.

```
.HEDLOD ↵
CURRENT HEAD LOAD TIMER IS 0 CLOCK TICKS
```

2. To define the head load time, edit the SYSTEM.INI with one of the system text editors. At any point after the last SYSTEM command, enter HEDLOD followed by the number of clock ticks you want the AM-200 to keep the heads loaded after each data transfer. For example:

```
HEDLOD 1800
```

The example above tells the system to keep the disk drive heads loaded for 1800 clock ticks (30 seconds when the real-time clock is operating at 60 Hz) after each data transfer.

ERRORS:

HEDLOD generates no error messages.

CHARACTERISTICS:

HEDLOD is both a user command and a system initialization command.

The use of HEDLOD in the system initialization command file requires that the CLKFRQ command also be in the SYSTEM.INI.

Returns the terminal to AMOS command level.

FUNCTION:

Displays text files on your terminal that contain information about the system.

HINTS/RESTRICTIONS:

The HELP command operates in two modes: 1. to tell you what files containing information on the system are available (.HLP files); and 2. to display a specific .HLP file on your terminal.

HELP looks first for the specified .HLP file in the System HELP File Library, DSK0:[7,1]; next it searches in your project library account ([your-proj#,0]), and if unsuccessful there, it looks on the device and account that you are currently logged into.

FORMAT:

.HELP ↵

or:

.HELP Topic ↵

The first format tells HELP to display a list of all of the topics for which help is available.

The second format tells HELP to display the file that contains the information on the requested topic. (The .HLP file bears the name of the topic about which it contains information.)

OPERATION:

1. Type HELP followed by a RETURN to see a list of the available topics. For example:

```
.HELP ↵
Help is available for:

APPEND  BAUD    COPY    DEL     ERASE   LOG
```

2. HELP first lists the .HLP files in account DSK0:[7,1]; then it prints a blank line and lists the .HLP files in the project library account for the account you are currently logged into. Last, it prints a blank line and lists the .HLP files in the account you are logged into.

   Type HELP followed by the name of a specific topic to see the text file associated with that topic.

   .HELP VUE ↵

   Now the screen clears and you see one or more screensful of information on the topic you have chosen.

ERRORS:

The only time that HELP generates an error message is if you ask it for information on a topic that it knows nothing about; in that case, it says:

I'm sorry, I can't help you.

Then it displays a list of the topics for which help is available.

CHARACTERISTICS:

Displays a list of topics about which it knows information, and when asked for a specific topic, displays the text file that contains the information on that topic.

Returns your terminal to AMOS command level.

hwklod

FUNCTION:

Bootstrap loader program for a system that uses the Control Data Corporation Hawk hard disk as the System Device.

HINTS/RESTRICTIONS:

The HWKLOD program is embedded in the micro-code ROM on the AM-500, which allows the system to boot off a System Disk on a Hawk hard disk when a hardware reset occurs (that is, when you hit the RESET button).

You may use HWKLOD at AMOS command level to reset the system if your System Device is a CDC Hawk hard disk. The Hawk is a 10-megabyte hard disk device (five-megabyte fixed, five-megabyte removable). The memory partition of the job that uses the HWKLOD command MUST be in Bank Zero if your system bank switches memory. (For information on bank-switched systems, refer to the document Memory Management Option, (DWM-00100-10) in the AM-100 documentation packet.)

FORMAT:

.HWKLOD ↵

OPERATION:

1. Type HWKLOD followed by a RETURN:

   .HWKLOD ↵

   The system now resets itself by reading a copy of the HWKLOD bootstrap program into system memory and executing it.

2. Once invoked, the HWKLOD program reads the operating system skeleton monitor, DSK0:SYSTEM.MON[1,4], into memory. SYSTEM.MON then brings up the system under the control of your system initialization command file, SYSTEM.INI.

3. Once the system is up and running, you see the AMOS prompt.

ERRORS:

HWKLOD generates no error messages. If it is not able to find SYSTEM.MON on the cartridge, HWKLOD searches the fixed disk. If HWKLOD doesn't find SYSTEM.MON [1,4], or if it does find SYSTEM.MON, but no SYSTEM.INI, the bootup (start-up) procedure fails.

(1 October 1979)

CHARACTERISTICS:

Boots the system from a CDC Hawk hard disk if that disk is the System Device.

Returns your terminal to AMOS command level if the system resets successfully.

ibmcpy

FUNCTION:

Transfers a copy of a file from an IBM 3740 diskette (prepared on an IBM data-entry station) to an AMOS device.

HINTS/RESTRICTIONS:

IBMCPY does not convert the file into one of the formats used by AMOS files, it simply transfers the data from the diskette to an AMOS device as is, except for translating the characters from EBCDIC to ASCII.

IBMCPY reads IBM 3740 diskettes in which the data is arranged in 80-byte records.

IBMCPY assumes that the IBM diskette is mounted on device IMG1:. You must have a valid copy of the image driver (IMG.DVR) in area [1,6] of your System Disk, and you must have IMG defined as a valid device on your system. (See the DEVTBL reference sheet for information on defining devices.)

FORMAT:

.IBMCPY Filespec ↵

where Filespec is the name (in IBM format) of the file that you want to copy.

OPERATION:

1. Type IBMCPY followed by the name of the file on the IBM diskette that you want to copy. Type a RETURN. For example:

   .IBMCPY DATA09 ↵

2. IBMCPY reads the file from the IBM diskette and copies it over to the device and PPN that you are logged into. The name of the new file is the first six characters of the IBM file's name; it has the extension .IBM.

3. Since each logical record in an IBM file is 80 characters long, the new AMOS file consists of a series of 80-character lines; each line ends with a carriage return/line feed sequence.

4. When IBMCPY is finished, it displays this message:

   Copy complete

(1 October 1979)

ERRORS:

File not found on IBM diskette

IBMCPY looked for the file that you wanted to make a copy of, but couldn't find it on the IBM diskette. Make sure that you've spelled the name of the file correctly.

IBMCPY can also display the standard system error messages if you provide an invalid device specification. For example:

?Cannot READ IMG1: - device does not exist

Make sure that IMG is defined in your system device table as a valid device. (To see a list of system devices, type DEVTBL followed by a RETURN.) Be sure to mount IMG1: before using IBMCPY.

CHARACTERISTICS:

Requires that IMG be a valid system device, and that IMG.DVR[1,6] exists on your System Disk.

For use on IBM 3740 diskettes that contain 80-byte records.

Returns your terminal to AMOS command level.

ibmdir

FUNCTION:

Prints directory of a 3740 diskette prepared on an IBM data-entry station.

HINTS/RESTRICTIONS:

IBMDIR uses the image driver (IMG.DVR) to access the data on the diskette; make sure that you have a valid copy of the image driver in area [1,6] of your System Disk. IBMDIR assumes that the diskette is mounted on device IMG1:. IMG must be a defined device on your system. (For information on defining devices, refer to the DEVTBL reference sheet.)

IBMDIR reads IBM 3740 diskettes that contain 80-byte records.

FORMAT:

.IBMDIR ↵

OPERATION:

1. Type IBMDIR followed by a RETURN:

   .IBMDIR ↵

2. IBMDIR displays the directory of the IBM diskette mounted in IMG1:. For example:

IBM 3740 Diskette Directory

Filename	start	end	used
DATA09	2 00	7 26	2 11
ORDER78	8 00	11 13	9 22

Below are the elements of the directory display:

a. Filename - The name of the IBM file. (IBM users will recognize this as the dataset name.)

b. Start - The starting address of the file on the diskette. The first file, DATA09 starts at track 2, sector 0.

c. End - The address of the last sector of the file. The ending address of DATA09 is track 7, sector 26.

(1 October 1979)

d. Used - The last disk address of the file that contains valid data. In the case of DATA09, the last sector of the file that contains valid data is sector 2, track 11.

ERRORS:

You can see several of the standard system error messages that result from invalid device specifications. For example:

?Cannot READ IMG1: - device does not exist

You must define IMG as a system device. Make sure that IMG.DVR[1,6] exists on your System Disk. (For a list of valid system devices, type DEVTBL followed by a RETURN.)

CHARACTERISTICS:

Requires that IMG be a valid system device and that IMG.DVR[1,6] exist on your System Disk.

Uses IBM 3740 diskettes that contain 80-byte records.

Returns your terminal to AMOS command level.

icmlod

FUNCTION:

Bootstrap loader program for a system that uses the Pertec Icom floppy disk as the System Device.

HINTS/RESTRICTIONS:

The ICMLOD program when contained on a 2716 PROM allows the system to boot off a System Disk on an Icom floppy disk when a hardware reset occurs (that is, when you hit the RESET button).

The program is also in account DSK0:[1,4] of the System Disk.

You may use ICMLOD at AMOS command level to reset the system if your System Device is an Icom floppy disk drive. The memory partition of the job that uses the ICMLOD command MUST be in Bank Zero if your system bank switches memory. (For information on bank-switched systems, refer to the document Memory Management Option, (DWM-00100-10) in the AM-100 documentation packet.)

FORMAT:

.ICMLOD ↵

OPERATION:

1. Type ICMLOD followed by a RETURN:

   .ICMLOD ↵

   The system now resets itself by reading a copy of the ICMLOD bootstrap program into system memory and executing it.

2. Once invoked, the ICMLOD program reads the operating system skeleton monitor, DSK0:SYSTEM.MON[1,4], into memory. SYSTEM.MON then brings up the system under the control of your system initialization command file, SYSTEM.INI.

3. Once the system is up and running, you see the AMOS prompt.

ERRORS:

ICMLOD generates no error messages. However, if it is not able to find SYSTEM.MON[1,4] and SYSTEM.INI[1,4], the start-up procedure fails.

(1 October 1979)

CHARACTERISTICS:

Boots the system from an Icom floppy disk if the Icom disk drive is the System Device.

Returns your terminal to AMOS command level if the system resets successfully.

ismbld

FUNCTION:

Builds an ISAM index file/data file combination.

HINTS/RESTRICTIONS:

ISAM (Indexed Sequential Access Method) is a method for organizing and accessing data. An ISAM file is an index file/data file combination. The index file contains pointers to records in the data file. ISAM.PRG quickly finds data records by searching the index file instead of searching the data file itself.

ISMBLD has three functions: 1. create a new ISAM file; 2. add data to the new file or to an existing file; and, 3. change the device specification of a data file.

You can use ISAM functions from within BASIC programs or assembly language programs. For information on using ISAM, refer to the ISAM System User's Guide, (DWM-00100-06, Rev A01), and the AlphaBASIC User's Manual, (DWM-00100-01). For the latest information on ISAM (e.g., the difference between Exclusive Open Mode and Counted Update Mode), refer to the document Important Notice for ISAM Users, (DWM-00100-36, Rev A01), in the AM-100 documentation packet.

NOTE: Although you call ISAM functions from within your BASIC or assembly language programs, you may not run the program ISAM.PRG directly from AMOS command level.

FORMAT:

.ISMBLD Filespec {/Switches} ↵

where Filespec selects the name you want to assign to the .IDA (data) and .IDX (index) files.

DEFAULTS:

ISMBLD assumes an extension of .SEQ when you specify a file from which to load data.

OPTIONS:

You may select the options below by including the proper switches on the ISMBLD command line. (For information on Exclusive Open Mode and Counted Update Mode, refer to the document Important Notice for ISAM Users, (DWM-00100-36, Rev A01).

(1 October 1979)

/D Change the data file device specification.

/N Prevent ISMBLD from using Exclusive Open Mode.

OPERATION:

1. If you want to use ISMBLD to create a new ISAM file, type ISMBLD followed by the name you want to assign to the file. For example:

```
.ISMBLD LABELS ↵
```

2. To create an ISAM file in Counted Update Mode, use the /N switch on the ISMBLD command line (otherwise, ISMBLD uses the Exclusive Open Mode). If the file you specify does not yet exist, ISMBLD asks you a series of questions; from the parameters you supply, ISMBLD generates a data file/primary index file combination. ISMBLD asks:

   a. Size of key: - Enter the size of the desired key in bytes. When you later access the ISAM file you are now building, you must remember to pad with blanks or other characters keys that are smaller than this size.

   b. Position of key: - Enter the location of the key within the data record. Enter the number of the first character-position in the record that the key occupies.

   c. Size of data record: - Specify in bytes the size of the records in the data file (or the maximum size in the case of variable length records).

   d. Number of records to allocate: - Number of records which the data file is to contain.

   e. Entries per index block: - Enter the number of entries per each index block.

   f. Empty index blocks to allocate: - Enter the number of additional empty index blocks you want to allocate.

   g. Primary Directory? - If you are creating a data file/primary index file combination, enter Y; otherwise, enter N.

   h. Secondary index to file: - If you are creating a secondary index file instead of a data file/primary index file combination, ISMBLD prompts you for the specification of the primary index file to which this secondary index file cross-indexes. After you supply this information,

ISMBLD returns you to AMOS command level.

i. Data File Device? - ISMBLD asks this question if you are creating a data file/primary index file combination. If the index file and data file are to be on the same device, enter just a RETURN; otherwise enter the specification of the device on which the data file is to reside.

3. Once ISMBLD has created a new data/primary index file combination, it enters file loading mode. It asks:

Load from file:

You may enter either a RETURN (to create an empty data file) or you may specify a sequential data file from which you want to load data. ISMBLD assumes a file extension of .SEQ for this data file. After loading the data file, ISMBLD returns you to AMOS command level.

4. If you are not creating a new data/index file combination (i.e., you have specified a file to ISMBLD that already exists), ISMBLD goes directly into file loading mode. Instead of asking you the list of file parameter questions above, you see:

[Processing existing file]

This notifies you that you are in file loading mode, not file creation mode. Now ISMBLD asks for the specification of the file from which you want to load:

Load from file:

ISMBLD assumes a file extension of .SEQ for this sequential data file. You may enter RETURN to tell ISMBLD not to load from a file.

5. To change the data file device after an ISAM file has been created, use the /D switch on the ISMBLD command line. For example, you want to change the device of the data file from DSK0: to DSK1:--

```
.ISMBLD MAIL/D ↵
[Processing existing file]

Current device name is DSK0:
Enter new device name: DSK1: ↵
```

ERRORS:

Below we list several of the more common ISMBLD error messages you may see:

End of input file in middle of record

You loaded an ISAM file with data from a sequential data file, but the parameters you gave to ISMBLD when you originally created the file did not exactly match the data in the data file (e.g., the record size in the data file is not the same as the size you specified to ISMBLD).

?ISBXFL Index file full

You did not specify enough additional index blocks when you built the ISAM file. When you loaded the ISAM file with data, the index was not large enough to hold the necessary entries.

?ISBAFL Data file full

You did not specify enough records when you built the ISAM file. When you loaded the ISAM file, the data file was not large enough to hold all of the data.

Duplicate secondary key left out xxxxx

You tried to use ISMBLD on a secondary index file-- the data xxxxx is already in an index file.

%Attempt to add duplicate key xxxxx

You tried to add duplicate data, xxxxx, to a data file. Make sure that you have not tried to load data twice from the same data file.

CHARACTERISTICS:

Assumes a .SEQ file extension for the data file from which you are loading data. Assumes you want to build an ISAM file in Exclusive Open Mode unless you use the /N switch.

Checks to see if the file you specify on the ISMBLD command line already exists to decide whether to build a new file or load an old one. Returns your terminal to AMOS command level.

ismcom

FUNCTION:

Compresses the upper level of an ISAM index file.

HINTS/RESTRICTIONS:

ISAM (Indexed Sequential Access Method) is a method for organizing and accessing data. An ISAM file is an index file/data file combination. The index file contains pointers to records in the data file. ISAM.PRG quickly finds data records by searching the index file instead of searching the data file itself.

NOTE: Although you can call ISAM functions from within your BASIC or assembly language programs, you may not run the ISAM.PRG program directly from AMOS command level.

For information on using ISAM, refer to the ISAM System User's Guide, (DWM-00100-06, Rev A01), and the AlphaBASIC User's Manual, (DWM-00100-01). For the latest information on ISAM, consult the document Important Notice for ISAM Users, (DWM-00100-36, Rev A01), in the AM-100 documentation packet.

Use the ISMBLD command to build the original data file/index file combination. (See the ISMBLD reference sheet.)

ISMCOM compresses the upper level of the ISAM index file. This increases the speed with which you can access data in the ISAM file and recovers space in the index file.

FORMAT:

.ISMCOM Filespec ↵

where Filespec selects the index file you want to compress.

DEFAULTS:

ISMCOM assumes a file extension of .IDX.

OPERATION:

1. Type ISMCOM followed by the specification of the file you want to compress. (This file is the index file portion of an ISAM file.) Now type RETURN. For example:

   .ISMCOM DATA.IDX ↵

(1 October 1979)

2. ISMCOM now prints a warning about file access and then tells you the compression factor it is planning to use.

   If the compression factor ISMCOM is going to use is not acceptable, you can now enter another one. ISMCOM now computes the ACTUAL compression that will be accomplished if it uses your value and gives you the chance to enter a new value. For example:

```
.ISMCOM MAIL.IDX↵

NOBODY else may use this file while I'm processing it

I am planning to compress each block to at least 90 percent full
If that is not acceptable, enter the percentage you desire 76↵
It will actually work out to be 80 percent full
If that is not acceptable, enter the percentage you desire ↵
```

3. If the actual compression value that ISMCOM computes is acceptable to you, enter a RETURN when it asks you for your own compression factor. Otherwise, enter a new compression factor and ISMCOM will compute the actual compression that it can achieve using that value. (Do not enter the percent sign, even though you are talking about percentages.) When you enter a compression factor that will achieve the result you want, type a RETURN the next time ISMCOM asks you for a compression factor.

   NOTE: A compression factor of 100% will cause a block split the next time a top level index is created. The number 95% was chosen as the optimum compression factor for most files. You may not enter a value of less than 50.

4. ISMCOM now prints some statistics that tell you how much compression was done and how much good it should do. For example:

```
No blocks unchanged, No blocks freed, No blocks compressed
.
```

   ISMCOM now returns you to AMOS command level.

ERRORS:

?File specification error
?No attempt made to close and unlock file

ISMCOM does not understand the format of your command line. For example, you see this message if you do not include a file specification on the command line.

?Cannot RENAME Filespec - file not found
?No attempt made to close and unlock file

ISMCOM did not find the index file you specified. Make sure that you supply the correct device and account specification.

CHARACTERISTICS:

Assumes a file extension of .IDX for index file.

Returns your terminal to AMOS command level.

ismdmp

FUNCTION:

For use on ISAM files; writes the contents of a data file to a sequential file or displays the structure of an index file on your terminal.

HINTS/RESTRICTIONS:

ISAM (Indexed Sequential Access Method) is a method of organizing and accessing data. An ISAM file is a data file/index file combination. The index file contains pointers to records in the data file. ISAM.PRG quickly finds data records by searching the index file instead of searching the data file itself.

NOTE: Although you call ISAM functions from within your BASIC or assembly language programs, you may not run the program ISAM.PRG itself directly from AMOS command level. For information on using ISAM and ISMDMP, refer to the document ISAM System User's Guide, (DWM-00100-06, Rev A01), and the AlphaBASIC User's Manual, (DWM-00100-01). For the latest information on ISAM, consult the Important Notice for ISAM Users, (DWM-00100-36, Rev A01), in the AM-100 documentation packet.

ISMDMP works in one of two modes: 1. it displays the structure of an ISAM index file on your terminal; or 2. it writes the data in an ISAM data file into an ordinary sequential file.

The purpose of ISMDMP in display mode is to aid in debugging programs that use ISAM. Much of the information in the display is of help only to the experienced systems programmer and some of the information is of use only to programmers working on the actual ISAM program itself.

Dumping data to a sequential file can be useful for data backup.

FORMAT:

.ISMDMP Filespec{/N} ↵

where Filespec selects the ISAM data or index file you want to dump and /N selects the Counted Update mode option.

DEFAULTS:

The default extension of the ISAM file that you want to dump is .IDX. The default extension of the file that will contain the data in the ISAM file is .SEQ.

(1 October 1979)

OPTIONS:

You can prevent ISMDMP from using Exclusive Open mode by including a /N switch on the command line. For example:

.ISMDMP MAIL/N ↵

tells ISMDMP to use Counted Update mode instead of Exclusive Open mode. (For information on Counted Update mode and Exclusive Open mode, refer to Important Notice for ISAM Users in the AM-100 documentation packet.)

OPERATION:

1. Type ISMDMP followed by the specification of the ISAM file whose contents you want to see. Type a RETURN. For example:

   .ISMDMP ADDRSS ↵

2. Now ISMDMP asks you:

   Output to:

   If you want to see a display of your index file structure, type TTY: followed by a RETURN.

   If you want to dump the contents of the data file portion of your ISAM file to a sequential file, enter a file specification. For example:

   Output to: DATA.SEQ ↵

3. If you enter a file specification, ISMDMP writes the data in the ISAM data file to the specified file. ISMDMP tells you how many data records it wrote to the file. For example:

   Output to: LABELS.TXT ↵

   15 records dumped

   You can now use TYPE or PRINT to display the file.

4. If you enter TTY: to the output prompt, ISMDMP displays the structure of your index file on your terminal screen.

The display might look something like this:

```
Size of data record:    67
Size of dir entry:      30
Size of dir block:      302
Size of key:            25
Type of key:            0
Entries per dir block:  10
Record key position:    1
Blocking factor:        7
IDA freelist pointer:   0000000000517
IDA freecount:          45
IDX freelist pointer:   000004
IDX freecount:          22
Records allocated       5
Update counter:         4
Top dir blk pointer:    000001

000001: ------------------------------  000000000002
                                        000000000000
                                        000000000000
                                        000000000000
                                        000000000000
                                        000000000000
                                        000000000000
                                        000000000000
                                        000000000000
     000000                  177777

000002: ------------------------------  000000000003
                                        000000000000
                                        000000000000
                                        000000000000
                        .
                        .
                        .
```

(Chapter 3, "Dumping an ISAM File with ISMDMP," in the ISAM System User's Guide discusses the elements of the ISMDMP display in detail.)

Several of the items of information in the upper portion of the display are parameters that you supply when you use ISMBLD to build the ISAM file (e.g., Size of data record:, Size of key:, etc.). Some of the other important elements of this display are:

a. Size of dir entry: - Amount of storage needed to store key-- key (rounded up to even) + 4.

b. Size of dir block: - (Number of entries per block * size of dir entry) + 2.

c. Type of key: - Always zero.

d. Blocking factor: - This value gives the number of logical data records that fit into each physical disk block. This number gives the blocking factor for the data file, and is thus 512/record size.

e. IDA freelist pointer: - A byte pointer into the data file.

f. IDA freecount: - Number of records free in data file.

g. IDX freelist pointer: - Logical block number of next free block in index.

h. IDX freecount: - Number of free logical index blocks.

i. Records allocated: - Number of data records used.

j. Top dir blk pointer: - Block pointer to beginning of top index block.

The rest of the display contains debugging information about the structure of the index file. This information is mostly of use to the experienced programmer who understands the internal workings of ISAM.

ERRORS:

No records dumped
You tried to dump data from an empty ISAM data file.

?Cannot RENAME Filespec - file not found
ISMDMP could not find the ISAM file you specified. Check your device and account specifications.

CHARACTERISTICS:

The default extension of the ISAM file whose data you want to dump is .IDX. The default extension of the file that will hold the contents of the ISAM data file is .SEQ. You can only dump data to a sequential file.

Returns your terminal to AMOS command level.

ismfix

FUNCTION:

Rebuilds ISAM files to recover lost indices.

HINTS/RESTRICTIONS:

ISM8LD Version 4.2 contained a bug that, under certain circumstances, resulted in indices being hidden in an ISAM index file.

ISMFIX rebuilds ISAM files processed by 4.2 ISAM and thus recovers any hidden indices.

You must run ISMFIX on all ISAM files processed by 4.2 ISAM. Using ISMFIX does no harm if you run it on a file in which indices have not been lost. Also, rebuilding a file that has already been rebuilt does not harm that file.

ISMFIX accepts a wildcarded file specification. If all of your ISAM files (both the .IDA and the .IDX portions) are on-line at the same time, you can rebuild all ISAM files on the system by logging into [1,2] and entering:

.ISMFIX ALL:↵

(It is not necessary to specify an .IDX extension, because ISMFIX only processes .IDX files and, through them, the appropriate .IDA files.) ISMFIX processes all ISAM files it finds. If ISMFIX encounters a primary index file, but that file's data file is not on-line and ready to access, an error occurs. You see a file not found error message, and ISMFIX goes on to the next ISAM file.

DEFAULTS:

When you are logged into the System Operator's account, [1,2], the default account specification is [] (that is, [*,*]). When you are logged into an account other than [1,2], ISMFIX restricts itself to the account you are logged into. (Therefore, if you are logged into [1,2], and you want to restrict ISMFIX to just that account, be sure to specify account [1,2] on the command line.)

ISMFIX forces a .IDX extension.

FORMAT:

.ISMFIX Filespec ↵

where Filespec selects the files you want to rebuild.

OPERATION:

1. To rebuild a specific file, enter ISMFIX followed by the specification of the file you want to rebuild. Type a RETURN. For example:

   .ISMFIX DSK0:MAIL↵

2. ISMFIX lists all restored indices as it re-positions them in your ISAM file. It also lists the relative record numbers of the indices.

3. You can supply wildcarded file specifications. For example:

   .ISMFIX DSK5:[110,*]↵

4. If all of your ISAM files are on-line, you can rebuild them all by logging into [1,2] and entering:

   .ISMFIX ALL:↵

ERRORS:

An error occurs if both the .IDA and .IDX portions of an ISAM file are not on-line when you run ISMFIX on that ISAM file. You see a file not found error message.

If you include a wildcarded PPN symbol, [], in your Filespec, but are not logged into [1,2], you see a Protection violation error message if ISMFIX tries to re-write a file that is not in the project you are logged into.

As ISMFIX re-positions re-discovered indices, ISAM errors can occur. In this case, you see the standard ISAM error messages. (For example, if ISMFIX relocates a key that you had already re-inserted in the index file yourself, you get a duplicate key error message. You might also see an index file full message.)

If you get an ISAM error, your best course is to re-build the ISAM file from scratch.

CHARACTERISTICS:

Understands the wildcard symbols *, ?, [], and ALL:.

You must use ISMFIX on all ISAM files that were processed with 4.2 ISAM.

jobmem

FUNCTION:

Allows you to determine what memory addresses are allocated to a job and allows you to change that allocation. For use in memory management systems (that is, systems that bank switch memory).

HINTS/RESTRICTIONS:

If you are not familiar with bank switching memory on the Alpha Micro system, see the documents Memory Management Option, (DWM-00100-10), and Setting Up Multiple Piiceon 64K Memory Boards, (DWM-00100-34), in the System Operator's Information section of the AM-100 documentation packet.

We will not discuss here the mechanics of setting up a bank-switched system, but briefly: bank switching memory allows you to access more than 64K of memory on your system (although each individual user is still limited to a maximum of 64K).

JOBMEM expects you to enter memory addresses in the number base the system is currently using for your numeric displays (usually octal). (For information on changing the display base from octal to hexadecimal, refer to the SET reference sheet.)

NOTE: The first 16K of memory cannot be allocated to the bank switchable area. It is reserved for the monitor.

Second 16K of memory:	40000-77776 (octal)	or	4000-7FFE (hex)
Third 16K of memory:	100000-137776 (octal)	or	8000-BFFE (hex)
Fourth 16K of memory (minus last 256 bytes for the I/O ports):	140000-177376 (octal)	or	C000-FEFE (hex)

IMPORTANT NOTE: You may not allocate to a job the last 256 bytes of the 64K address space. That means that the highest memory address you can allocate to a job is 177376. (The memory addresses 177400-177777 are reserved by the system for the I/O ports.)

Make sure that memory addresses for different jobs do not overlap. For example, if the last address you assign to one job is 13776, the first address you assign to the next job in the bank may not be less than 14000 (which is 13776 + 2). If the next job's memory partition were to begin at 13776 or 13777, the two jobs would be sharing a word of memory.

FORMAT:

.JOBMEM {Jobname} ↵

(1 October 1979)

or:

.JOBMEM {Jobname} Bank-#:StartAddress-EndAddress ↵

Use the first format to find out what memory is allocated to your job (omit Jobname from the command line) or the memory allocation of another job (specify the Jobname).

Use JOBMEM in the second format to allocate memory to your own job (omit the Jobname) or to another job (specify the Jobname).

Where:

Jobname	The name of the job for which memory is to be allocated (if omitted, JOBMEM uses your job as the default).
Bank-#	The number of the memory bank containing the memory being allocated to the job.
StartAddress	The beginning address of the memory block being allocated to the job.
EndAddress	The ending address of the memory block being allocated to the job.

DEFAULTS:

If you omit a jobname, JOBMEM assumes the name of your job.

OPERATION:

1. To find out what memory is allocated to your job, type JOBMEM followed by a RETURN. For example:

```
.JOBMEM ↵
CURRENT MEMORY ALLOCATION IS 0:32370-177376
```

   The message above tells you that your memory partition is from address 32370 to address 177376 in bank number zero, the first bank of the system.

2. To find out what memory is allocated to another job, type JOBMEM, the name of that job, and a RETURN. For example:

```
.JOBMEM JOB4 ↵
CURRENT MEMORY ALLOCATION IS 1:40000-177376
```

3. To allocate memory to your own job, type JOBMEM, a bank number, and starting and ending memory addresses; type a RETURN.

```
.JOBMEM 2:140000-177376 ↵
```

4. To allocate memory to another job, type JOBMEM, a jobname, a bank number, and starting and ending memory addresses; type a RETURN.

```
.JOBMEM BG23 0:140000-177376 ↵
```

ERRORS:

?Memory allocation format error
JOBMEM doesn't understand the format of your command line. For example, did you leave out the colon after the bank number?

?Non-existent job
You've specified a job that doesn't exist. Run SYSTAT to see a list of the jobs on the system.

?Non-existent bank number
You've given JOBMEM a bank number larger than the total number of MEMDEF statements in your SYSTEM.INI. (That is, you've specified a bank that has not been defined.)

?Allocation overlaps monitor or system memory
The monitor memory is used by the operating system and the system memory is sharable by all users; you must not allocate any of this memory to a user partition. You must either reduce the amount of system memory or change your user memory allocations.

?Illegal memory range (end is below base)
Ending address must be greater than starting address.

?Allocation is not within requested bank's defined memory
You've specified a valid bank number, but it is not addressed for the memory addresses you've requested. Check the addressing of your memory boards and the MEMDEF statements in your SYSTEM.INI.

?Requested allocation would overlap job JOBNAM
You must be careful not to allocate a memory address to one job that is already allocated to another. JOBNAM specifies the job whose memory partition is being overlapped.

CHARACTERISTICS:

Used in systems that bank switch memory.

Returns your terminal to AMOS command level.

jobpri

FUNCTION:

Allows you to determine the priority of your job, to change that priority, and to examine and change the priorities of other jobs.

HINTS/RESTRICTIONS:

In a timesharing environment, each user receives a certain number of real-time clock ticks before the CPU moves on to another job. If you increase this unit of time (called a quantum) for a particular user, you speed up his job and, consequently, slow down the jobs of the other users on the system.

Increasing the number of quanta the CPU spends on a job is called increasing the job's priority. You should not change your job's priority or the priority of other jobs on the system unless there is a very good reason for it; indiscriminate changes in job priorities can cause you to be unpopular with other users on the system.

The maximum priority is 65535; the minimum priority is zero. The JOBPRI command value tells the system to add that many clock ticks to the standard quantum for the specified job. If the JOBPRI command has not been used to increase a job's priority, the priority for that job is zero.

FORMAT:

.JOBPRI {Jobname}{Priority} ↵

where jobname is the name of the job whose priority you want to change and priority is the number of clock ticks you want to add to the job's quantum.

DEFAULTS:

If you omit both the Jobname and the Priority, JOBPRI assumes that you want to know the current priority of your job. If you omit just the Jobname, JOBPRI assumes that you want to change the Priority of your job. If you omit just the Priority, JOBPRI assumes that you want to know the Priority of the job you've specified.

OPERATION:

1. Type JOBPRI followed by a RETURN to find out your job priority:

```
.JOBPRI ↵
CURRENT PRIORITY IS 0.
.
```

2. To find out the priority of a job other than your own, type JOBPRI, the name of the job, and a RETURN. For example:

```
.JOBPRI ROBIN ↵
CURRENT PRIORITY IS 100
.
```

3. Type JOBPRI, a priority number, and a RETURN to change your job's priority:

```
.JOBPRI 1000 ↵
.
```

4. To change the priority of another user's job, type JOBPRI, the name of the job, a priority number, and a RETURN. For example:

```
.JOBPRI DUKE 1000 ↵
.
```

ERRORS:

JOBPRI generates no error message except: NONEXISTENT JOB if it cannot find the job you've entered on the command line.

CHARACTERISTICS:

Uses your jobname as the default.

Returns your terminal to AMOS command level.

FUNCTION:

When you use it within the system initialization command file, JOBS defines the jobs that can run on the system. At AMOS command level, JOBS tells you the name of your job.

HINTS/RESTRICTIONS:

At the time of system initialization, the JOBS command in the SYSTEM.INI file tells the system what jobs can run on the system. For information on using JOBS within the SYSTEM.INI, refer to Section 2.2, "Allocating Jobs (The JOBS Command)," in the document The System Initialization Command File, (DWM-00100-09, Rev A01), in the System Operator's Information section of the AM-100 documentation packet.

When it begins to process the SYSTEM.INI file at the time of system bootup, the system automatically attaches the first job listed in the JOBS command line and the terminals defined by the first TRMDEF command. The rest of the process of system initialization takes place under control of that job. You must explicitly attach all other jobs and terminals that you want to use on the system (via the ATTACH command). The JOBS command must be the first command in the SYSTEM.INI file after the optional :T command.

Once the system is up and running, the JOBS command takes on another function; it tells you the name of your job.

There are some functions that you might want to perform on the system (e.g., attaching jobs to terminals) that require that you know the name of your job.

To find out what jobs are defined on the system, type SYSTAT followed by a RETURN. The left-most column of the SYSTAT display lists the jobs defined by the JOBS command in the SYSTEM.INI.

FORMAT:

.JOBS ↵

or:

JOBS Job1{,Job2,...JobN}

where Job1-JobN are one- to six-character names that define the jobs that will run on the system.

OPERATION:

At AMOS command level:

1. Type JOBS followed by a RETURN. JOBS tells you the name of your job. For example:

```
.JOBS ↵
YOUR JOB NAME IS JOB5
```

Within the SYSTEM.INI:

1. Type JOBS followed by a list of the jobs you want to allocate on the system. Each job name is one to six characters in length. For example:

```
JOBS JOB1,PRINTR,JOB2,OPRTR,SPARE,SPOOLR
```

2. If you have more jobs than will fit on one line, you may use as many JOBS command lines as you want, as long as they all appear before the first TRMDEF statement. NOTE: Each job you allocate takes up about 150 words of system memory.

3. Remember to allocate memory and attach a terminal to a job before you try to use that job. (See the JOBMEM, MEMORY, and ATTACH reference sheets.)

ERRORS:

JOBS generates no error messages.

CHARACTERISTICS:

JOBS is both a user command and a system initialization command.

The JOBS command must be the first command in your SYSTEM.INI after the optional :T command.

Returns your terminal to AMOS command level.

FUNCTION:

Allows you to kill the program currently being run by another job.

HINTS/RESTRICTIONS:

The KILL command kills the program currently being executed by the specified job. If other programs are waiting in line to be executed, you must enter one KILL command for each program you want to halt.

KILL stops the execution of a program by sending a Control-C to the job; if the program running does not accept Control-Cs, a KILL command accomplishes no purpose.

NOTE TO THE SYSTEM OPERATOR:

To properly initialize the jobs on the system, your system initialization command file must contain one KILL command for each job defined on the system. The KILL command for a particular job must appear after the ATTACH command that attaches a terminal to that job, but before any FORCE commands that force input to the job. DO NOT KILL the job the system is coming up under (i.e., the first job in the JOBS command line). Use a separate KILL command for each job.

FORMAT:

.KILL Jobname ↵

where Jobname selects the job you want to kill.

OPERATION:

1. Type KILL and the name of the job whose program you want to kill; then type a RETURN:

   .KILL JOB4 ↵

ERRORS:

[NONEXISTENT JOB]
You entered an invalid job name. Check your spelling; then use the SYSTAT command to see a list of the valid jobs on the system.

CHARACTERISTICS:

You can use KILL either at AMOS command level or within the SYSTEM.INI file to send a Control-C to a job.

Returns your terminal to AMOS command level.

FUNCTION:

The LABEL program allows you to label a disk with descriptive information and to display that information. LABEL maintains as part of the label the date the disk was last mounted and the date the disk was originally labeled.

HINTS/RESTRICTIONS:

Disk labels are stored in block zero of the disk and are used both to allow the operator to easily determine what disk is mounted, and to allow programs to verify that the correct disk has been mounted.

FORMAT:

.LABEL Devn: (RET)

where Devn: specifies the device which you want to label or for which you want the current label displayed.

OPERATION:

1. Type LABEL followed by the device specification for the disk you wish to label. For example:

   .LABEL DSK4: (RET)

2. If the disk you specified is already labeled, LABEL will now display the current disk label. For example:

   .LABEL DSK1: (RET)
   Payroll Data (PAY001)
   Created on 1-Jan-80 at Computer Makers Inc. on System 1 by F. Smith
   Last access: 2-Apr-80

3. LABEL now asks for new contents for each of the label fields. If you do not wish to change the label, type a Control-C at this point, which returns you to AMOS command level. If you do wish to change the label, answer each of the following questions:

   a. Volume Name:
      Enter the textual description of the disk. This field is used by the operator to determine which disk is mounted. The MOUNT program displays this field whenever a disk is mounted. This field can be up to 40 characters long.

b. Volume ID:
Enter up to ten characters as the Volume ID. This field may be used by programs to determine if the proper disk has been mounted. The XMOUNT subroutine (which can be called from AlphaBASIC) returns this field. The MOUNT program also displays this field whenever a disk is mounted.

c. Installation:
Enter the name of your installation or company. This field, which may be up to 30 characters long, is used when exchanging disks between different installations and companies.

d. System:
Enter the name of the computer system that this disk is created on. This field, which can be up to 30 characters long, is used when a particular installation has more than one computer system.

e. Creator:
Enter the name of the person creating the disk. This field may be up to 30 characters long.

ERRORS:

You may see the following error messages when using LABEL:

?Cannot INIT Devn: - device does not exist
The device specification you gave when giving the LABEL command is not valid. Check the spelling of the device specification you gave.

?File specification error
The format of your LABEL command line was not valid. (For example, you typed LABEL followed by a RETURN.) Try again, being sure to place a colon after the device specification.

CHARACTERISTICS:

Allows you to label a disk with descriptive information.

Returns your terminal to AMOS command level.

FUNCTION:

LIB is an object file library generator. It combines separate .OBJ files into one library (.LIB) file. You may also use LIB to update an existing .LIB file.

HINTS/RESTRICTIONS:

A library file gives you a way to make available to all programmers on the system a standardized set of machine language routines. It also keeps you from having to rewrite the same utility routines again and again for multiple programs.

At the time that you link your object files into an executable program (using LINK or SYMBOL), you may include the specification of a library file on the command line. If the other files specified on the command line make use of a symbol defined in the library file, LINK or SYMBOL will process the routine within the library file that defines that symbol, linking it into your program file.

NOTE: Each object file that is placed into the library file should be a separate routine because when your program references a symbol in the library file, LINK or SYMBOL links in the entire object file required to resolve the reference. For example, if the library file UTILIT.LIB contains the object files GETNUM, SUBT, REAL, and BINWRT, and your program references a routine contained in the GETNUM file, LINK or SYMBOL links in the entire object file GETNUM from UTILIT.LIB even if that file contains more than one routine.

For more information on library files, and on LIB, MACRO, LINK, and SYMBOL, see the AMOS Assembly Language Programmer's Manual, (DWM-00100-43).

FORMAT:

.LIB{/L} output=input1{,input2,...inputN} RET

or:

.LIB{/L} inout{,input2,...inputN} RET

where output selects the name of the library file you want to build, and input specifies the object files that will make up the library. The second format is equivalent to: LIB inout=inout{,input1,...inputN}. (However, if you specify the /L switch, the second format is equivalent to: LIB/L TRM:=inout{,input1,...inputN}.)

The input specifications may take the following forms:

```
filespec
filespec\item1
filespec\(item1,item2{,...itemN})
filespec(item1,item2{,...itemN})
```

where the \ symbol designates an exception, and the parentheses designate a group of files. (For more information on the use of these symbols, see the section "OPERATION," below.)

If you want to specify more files than will fit on one command line, you may continue the command line by ending the current line with a comma. LIB will now display an asterisk and you may continue entering filespecs. You may enter as many lines of file specifications as you wish as long as the preceding line ends with a comma.

DEFAULTS:

LIB assumes an extension of .LIB for the output file and a .OBJ extension for the input files.

If you do not include an account and device specification, LIB looks for the files specified in the account and device you are logged into, then it looks in your project library account, [P,0]. Finally, LIB looks in the System MACRO Library account, DSK0:[7,7].

If you do not include an output specification, LIB creates an output file with the same name as the first input specification. If you use the /L switch to create a library listing, LIB uses the default extension .LST for that listing output file. If you use the /L switch and omit an output file specification, LIB uses the default output specification TRM: (that is, it sends the listing to your terminal display).

OPTIONS:

You may include the /L switch on the LIB command line to tell LIB to create a library listing file. This listing looks similar to a load map file, and lists all object files in the library and all INTERNed symbols.

OPERATION:

To create a library file:

1. Type LIB followed by the specification of the library file you want to create, type an equal sign, and then enter the specifications of the object files you want to combine into a library file, separating the filespecs with commas. Then type a RETURN. For example:

```
.LIB MYLIB=ADNUM,RDBIN,WRTBIN,ASCCHK,SRCH5 (RET)
```

The command line above causes LIB to create an output file named MYLIB.LIB that contains the specified object files.

2. If you want to enter more file specifications than will fit on one command line, just end the first command line with a comma, and LIB will display an asterisk on the next line, prompting you to enter more file specifications. You may enter as many lines of file specifications as you want as long as you end the preceding line with a comma.

3. LIB displays several messages as it processes your object files. For example:

```
== Object File Librarian Version 1.0 ==

Processing ADNUM.OBJ
Processing RDBIN.OBJ
Processing WRTBIN.OBJ
Processing ASCCHK.OBJ
Processing SRCH5.OBJ

Library file finished
*
```

4. To see a listing of the contents of your library, use the /L switch. For example:

```
.LIB/L NEWLIB (RET)
```

sends the listing to your terminal display. To tell LIB to place the listing in a disk file, specify an output file and the /L switch (e.g., LIB/L LIST=NEWLIB).

To update a library file:

1. To update an existing library file, enter LIB, the library file you want to modify, an equal sign, and the list of filespecs you wish to add and/or delete. For example:

```
.LIB UTILIT=UTILIT.LIB,NADDR (RET)
```

or:

```
.LIB UTILIT,NADDR (RET)
```

Both of these command lines tell LIB to take the existing library UTILIT.LIB and modify it by adding the object file NADDR to it.

2. You may specify a group of files by using the () inclusion symbols, and you may designate an exception by using the \ file restrictor symbol. For example:

```
.LIB NUMLIB\FORMAT (RET)
```

This command line tells LIB to process all of the object files in the library file NUMLIB, but to delete the object file FORMAT. The command line:

```
.LIB NEWLIB,MATH,\(BINADR,SMLLNUM,BINWRT) (RET)
```

tells LIB to process the library NEWLIB, and to add the object file MATH, but to delete the object files BINADR, SMLLNUM, and BINWRT.

3. When replacing an existing module in a library, do not just add a new version of the module without first deleting the original module of the same name. Doing so can cause problems because both versions will still be in the library. The recommended procedure is to first delete the module and then to add the new version of the module. For example, to replace the object file KEYSUB:

```
.LIB MACLIB\KEYSUB,KEYSUB (RET)
```

The command line above first deletes the existing module from the library and replaces it with the new one.

ERRORS:

You may see the following error messages when using LIB:

?Command error
LIB did not understand your command line. For example, you entered LIB followed by a RETURN.

?Undefined switch /X - ignored
where X is the switch you specified. LIB only recognizes one option switch, /L. Make sure that you did not accidently type a / when you meant to type a backslash.

?OBJ files are not libraries -- they cannot be restricted by a modifier
You may only use the \ file restrictor and the () file inclusion symbols if you are modifying a library.

?Listing aborted
LIB was not able to finish the library listing. For example, an error occurred while LIB was trying to access a file.

?The following module was not found - xxx
You tried to modify an existing library file, but the object files you specified were not present in the library file. Make sure that you did not accidentally use the \ restrictor symbol.

?Fatal error - xxx is an overlay
You may not specify an overlay as an element of an object file library.

CHARACTERISTICS:

LIB creates and modifies an object file library. SYMBOL and LINK both support the use of library files.

LIB is re-entrant, and so may be loaded into system memory. It is also serially re-usable.

LIB assumes an input file extension of .OBJ, an output file extension of .LIB, and a listing file extension of .LST. If you omit account and device specifications, LIB looks for the specified file in the account and device you are logged into; your project library account; and, the System MACRO account, DSK0:[7,7].

link

FUNCTION:

Creates executable machine language programs by linking and resolving one or more assembled object files.

HINTS/RESTRICTIONS:

When you use MACRO to assemble an assembly language program, the .OBJ file that results is in an intermediate form that is not ready for execution. Separate .OBJ files may contain symbol references to each other that cannot be fully resolved because these files cannot be assembled together.

LINK is a linkage editor. That is, it links together and resolves several .OBJ files to produce a .OVR or .PRG program file. If the program that you assembled with MACRO was made up of only one file that contains no internal or external symbol references, MACRO automatically calls LINK for you as Phase 4 of the assembly to produce a program file. Otherwise, you must use LINK yourself to link the .OBJ files that will make up the single .PRG file.

LINK links files together in the order in which you specify them on the LINK command line. LINK does not produce a program file if one or more of the files you specify is not found in its assembled object (.OBJ) form.

NOTE: You may use LINK to produce a symbol table file. And, you may use SYMBOL to produce a resolved, executable program file. See the SYMBOL reference sheet. (By using the appropriate option requests, you can make SYMBOL and LINK perform exactly the same functions.)

LINK supports the use of object file libraries. (See the LIB reference sheet for information on library files.)

For more information on MACRO, LINK and SYMBOL, see the AMOS Assembly Language Programmer's Manual, (DWM-00100-43). That manual also contains information on library, optional, and load map files.

FORMAT:

.LINK {/switches }filespec1{,filespec2,...filespecN}{/switches} RET

where filespecs select the files you want to link and /switch is an option request. You may not specify an overlay or library file as the first filespec on the command line.

(Changed 30 April 1981)

If you have too many filespecs to fit on one screen line, you may continue the LINK command line by ending it with a comma. LINK accepts as many lines of filespecs as you wish, as long as you end the preceding line with a comma.

If a /switch appears in front of a filespec (e.g., LINK MATH,/O NUM,SUB), that option request becomes the default for the rest of the command line. If a /switch follows a filespec (e.g., LINK MATH,NUM/O,SUB), it affects only that filespec. NOTE: Certain switches (identified in the discussions below as "operation switches") affect all filespecs on the command line no matter how they are placed.

DEFAULTS:

LINK uses the default extension of .OBJ, unless you are specifying a library file, in which case LINK uses the default extension of .LIB.

If your filespec does not contain an account and device specification, LINK assumes that the file is in the account and device you are logged into. Next it looks in your project library account, [P,O]. Finally, it looks in the System MACRO account, DSKO:[7,7].

The default switches are /P (generate a program file) and /R (designate a required file).

OPTIONS:

You may select one of the options below by specifying the appropriate switch:

/E	Include equated symbols in the symbols table file. (You must use /E with the /M or /S switch.) (Operation switch.)
/L	Designates a library file.
/M	Generate a load map (.MAP) file. (Operation switch.)
/N	Suppress /P switch. (Operation switch.)
/O	Designates an optional file.
/P	Generate program (.PRG or .OVR) file. The default switch. (Operation switch.)
/R	Designates a required file. The default switch. Cancels the /L and /O switches.
/S	Generate a symbol table (.SYM) file. (Operation switch.)

You may specify multiple switches by preceding each switch with a slash, /.

OPERATION:

1. Enter LINK followed by the specifications of the files you want to link together. Then type a RETURN. For example:

```
.LINK/M VISFIL,VIS1,UTILIT.LIB/L RET
```

(Changed 30 April 1981)

Notice that the command line above specifies a library file, UTILIT.LIB. By using the /M switch, we are also asking LINK for a load map file.

2. If you have more file specifications than will fit on one line, end the current command line with a comma. LINK now displays an asterisk and you may continue your list of file specifications. You may enter as many lines of file specifications as you wish as long as you end the preceding command line with a comma. Type just a RETURN at the end of the last line of filespecs.

3. Now LINK displays several messages as it process the files. (The exact messages you see depend on the options you have requested and files you have specified.) For example:

```
== Linkage Editor Version 2.0 ==

Processing VISFIL.OBJ [Base = 0, Size = 106. bytes]
Processing VIS1.OBJ [Base = 152, Size = 657. bytes]

-- Optional and Library Request --

Processing UTILIT.LIB(GETADR) [Base 1373, Size = 292. bytes]

Program and Map files finished [Program size = 1055. bytes]
*
```

4. If any errors occur during linking, LINK tells you so. For example:

```
Program file finished, 4 errors exist [Program size = 1055. bytes]
```

ERRORS:

You can see the following error messages while using LINK:

?Command error
There is something wrong with your command line. For example, you tried to use LINK without specifying a file on which to work.

?Fatal error - Insufficient memory
You must increase the size of your memory partition; there was not enough room to perform the procedure you specified.

?Undefined switch /x - ignored
LINK did not recognize the switch /x you specified. Refer to the section "OPTIONS," above for information on the valid LINK switches.

(Changed 30 April 1981)

?Fatal error - Overlays of code are not permitted
Next expected address is xxxx
Overlay code address is xxxx

Your program is trying to overlay previous code. Check your .MAC programs to make sure that your overlay references are correct.

?xxxx undefined

An external symbol is undefined. This is a very common error. You have referenced a symbol which has not previously been defined (e.g., you have made a reference to a label that does not exist). Make sure that an EXTERNed symbol in one segment is defined by an INTERN statement in another segment.

?Fatal error - First file must not be a library

To enable LINK to correctly resolve external references to a library, you must specify the program that references that library before you specify the library file itself.

?Fatal error - Attempt to specify overlay xxx as optional

You may not use the /O switch to designate a file as optional if that object file is an overlay.

?Fatal error - Overlay symbol "xxxx" in segment yyyy
was not defined in a previous input segment

You may not reference an undefined overlay. In other words, LINK is trying to process a supposed overlay file, but has seen no references to the overlay in a previous file. Without such a reference, LINK cannot construct the overlay, so it aborts and returns you to AMOS command level.

?Fatal error - First file must not be an overlay

To enable LINK to correctly resolve external references to an overlay, you must specify the program that references that overlay before you specify the overlay file itself.

?Fatal error - Expression stack error

An error occurred when LINK evaluated some expressions in your files. You should never see this error message-- it indicates an internal error.

?Fatal error - Expression stack overflow

You exceeded the number of nested expressions that LINK can handle. Try to find the exceedingly complex expression in your source file and simplify it.

CHARACTERISTICS:

LINK is re-entrant, and may be loaded into system memory; it is also serially reusable.

Creates an executable program file by linking and resolving intermediate .OBJ files.

Default extension is .OBJ for regular files and .LIB for library files. Default switches are: /P and /R.

(Changed 30 April 1981)

FUNCTION:

Invokes the LISP language processor.

HINTS/RESTRICTIONS:

LISP (LISt Processing Language) is a programming language used most often for applications that emphasize manipulation of string data (e.g., relational data base programs or natural language simulation programs) rather than programs that perform heavy numerical computation. The basic structure of a LISP program and data is the list. For information on programming in LISP, see the manual AlphaLISP User's Manual, (DWM-00100-05).

LISP.PRG is an interpreter and is reentrant. The System Operator may include LISP in system memory via the SYSTEM command in the system initialization command file.

NOTE: To leave LISP, after the LISP prompt, *, enter:

(EXIT)↵

(Remember to enter the parentheses!)

FORMAT:

.LISP ↵

OPERATION:

1. Type LISP followed by a RETURN:

   .LISP ↵

2. If the file DSK0:LISP.LSP[1,4] is present on your system, you see the question:

   LOAD EXTENDED LIBRARY?

   (If DSK0:LISP.LSP[1,4] is not present on your system, LISP does not ask this question, and does not load the extended library.) Answer a Y for Yes or an N for No; then type a RETURN. The extended library is a library of LISP functions.

(1 October 1979)

If you do not load the extended library, you have more memory space for your program, but you will be missing some of the standard features of LISP. (Refer to the AlphaLISP User's Manual for information on the contents of the extended library.)

3. Now you see the LISP herald message that tells you the version number of the LISP you are running. For example:

   "AlphaLisp 1.6 Version 1.0"

4. At this time LISP displays the LISP prompt symbol:

   *

   Now you can start entering LISP commands. For example:

   ```
   *(ADD 2 3) ↵
   5
   *
   ```

5. To leave LISP, enter:

   (EXIT) ↵

   Next you see the AMOS prompt.

ERRORS:

You may see any of the standard LISP error messages. Refer to a LISP textbook for a list of LISP error messages.

CHARACTERISTICS:

Invokes LISP language interpreter.

load

FUNCTION:

Loads disk files into your memory partition as memory modules.

HINTS/RESTRICTIONS:

You may use LOAD to load memory modules only in your own memory partition. You may use LOAD to load itself:

```
.LOAD DSK0:LOAD.PRG[1,4] RET
```

LOAD does not understand wildcard symbols.

LOAD understands ersatz devices. For example:

```
.LOAD BAS:STRIP.SBR RET
```

loads STRIP.SBR from the BASIC library account, DSK0:[7,6].

If you specify an ersatz device but omit a file extension, LOAD uses the default extension for that ersatz device. For example, the default extension for the BAS: account is .BAS. If you enter:

```
.LOAD BAS:NEW RET
```

LOAD will try to load in the file NEW.BAS.

The ersatz device default extensions are:

Ersatz Device	Extension
BAS:	.BAS
BOX:	.BOX
CMD:	.CMD
DVR:	.DVR
HLP:	.HLP
LIB:	.PRG
LSP:	.LSP
MAC:	.MAC
OPR:	.PRG
PAS:	.PAS
SYS:	.PRG
VUO:	.VUO

(Changed 30 April 1981)

FORMAT:

.LOAD Filespec (RET)

where Filespec selects the disk file you want to load.

OPERATION:

1. Type LOAD followed by the specification of the file you want to load into your memory partition; then type a RETURN. For example:

   .LOAD DSK0:ISAM.PRG[1,4] (RET)

2. LOAD does not tell you what file it has loaded into memory. You will know that it was unsuccessful if it displays this message:

   Filespec NOT FOUND

   where Filespec was the specification you supplied. To see a list of the memory modules in your partition, use the MAP command. (See the MAP reference sheet.)

ERRORS:

Filespec NOT FOUND
LOAD was not able to find the file you specified. Check your spelling and your device and account specification.

?Cannot READ Filespec - device does not exist

LOAD did not recognize the device you specified. Use the DEVTBL command to see if the device exists. Then make sure the device is mounted.

CHARACTERISTICS:

You may use LOAD to load itself into memory.

Does not understand wildcard symbols.

(Changed 30 April 1981)

log

FUNCTION:

Logs you into an account so that you can access the files in that account. Once you are on the system, you can use LOG to tell you which account you're logged into and you can use LOG to transfer between accounts and devices.

HINTS/RESTRICTIONS:

The only system commands that you may use if you are not logged into the system are: LOG, SYSTAT, MEMORY, ATTACH, SYSTEM, HELP and SET.

LOG recognizes the ersatz devices. For example:

```
.LOG BAS: (RET)
Logged into BAS:
```

logs you into the System BASIC Language Library Account, DSK0:[7,6]. For a list of the ersatz devices available on the system, see Section 1.3 of this manual. If you do not use an ersatz device specification to log into an account with which an ersatz device is associated, LOG lets you know that such an ersatz device exists:

```
.LOG DSK0:[1,4] (RET)
Logged into DSK0:[1,4]
Ersatz name is SYS:
```

Because they are such common typing mistakes, LOG treats the characters "m" and "." as commas (e.g., the command .LOG 100m1 is translated into .LOG 100,1).

You may be required to supply a password before the system will log you into an account.

If you have a command file in your account named START.CMD, LOG automatically runs that file after it logs you in. This command file can contain AMOS system commands, program invocations, the names of other command files, etc. If you want to see the processing of the command file on your terminal, remember to include the :T symbol in your command file. (See Chapter 8, "Command Files and DO Files," in the AMOS User's Guide, (DWM-00100-35), for more information on command files.)

The System-Mail function that LOG performs (see OPERATION, #4), occurs only when you log into the system after having been logged off; not when you use LOG to transfer between accounts.

(Changed 1 May 1980)

FORMAT:

.LOG {Devn:}[p,pn] (RET)

or:

.LOG {Devn:}[,pn] (RET)

or:

.LOG {Devn:}[p,] (RET)

or:

.LOG (RET)

The first format logs you into the system. [p,pn] is the project-programmer number that you want to log in under. (The square brackets enclosing the PPN are optional.) Devn: is the specification of the device that holds the disk containing the account. If you are already logged into the system under a different PPN or device, LOG transfers you over to the device and PPN you've requested.

Use the second and third formats after you've already logged into the system and want to transfer to another account. Use the fourth format after you've logged into the system to find out what PPN you're logged in under.

DEFAULTS:

If you omit a device specification when you first log into the system, LOG uses DSK0: as a default. If the account you have specified does not appear on DSK0:, LOG searches the rest of the units of that device (e.g., DSK1:, DSK2:, etc.) If you are already logged in, but omit the device specification, LOG uses as a default the device you are logged into; if the account does not appear on that device, LOG searches the rest of the units of that device, beginning with device unit #0.

If you use LOG when you are already logged into the system, LOG uses the project or programmer number of the account you are logged into as the defaults. (For example, .LOG [,3] logs you into an account whose PPN has the same project number as the account you are logged into, and a programmer number of 3.)

OPERATION:

1. If you are already logged into an account, type LOG followed by a RETURN to find out which PPN and device you are logged in under. For example:

(Changed 1 May 1980)

```
.LOG (RET)
Current login is DSK0:[12,34]
```

2. To log into the system, type LOG (optionally followed by a device specification) and a project-programmer number. Type a RETURN. For example:

```
.LOG DSK1:47,2 (RET)
```

3. In the example above, LOG looks on DSK1: for account [47,2]. If it finds it, it then looks at the account to see if a password is required. (Passwords are assigned by the System Operator in the interest of system security.)

```
.LOG AMS0:23,4 (RET)
Password:
```

The system does not display your password on the terminal as you type it; this prevents other users from seeing your password. If AMOS recognizes the password, it logs you into the system.

4. Once you are logged onto the system, LOG displays a message that tells you which device and account you're logged into. If any other users are logged into the same account, LOG warns you about it:

```
.LOG [123,4] (RET)
Logged into DSK0:[123,4]
Caution - other jobs same PPN
```

The next thing that LOG does is look in area [7,2] of the System Disk for a file named MAIL.JNK. If it finds it, LOG displays the first line of the file on your terminal:

```
.LOG 34,7 (RET)
Logged into DSK0:[34,7]
SYSTEM WILL BE DOWN FRI. 7-8 AM FOR MAINTENANCE.
```

5. The last thing LOG does is look for a command file in your account named START.CMD. If it finds the file, LOG automatically runs it for you. This is useful if you want to do something like run a business program every time you log into the system (but not when you use LOG to transfer between accounts).

6. Once you are logged into the system, you can use LOG to transfer yourself to other accounts on other disks. Type LOG followed by the device and PPN that specifies the account to which you wish to transfer; then type a RETURN.

```
.LOG 123,5 (RET)
Transferred from DSK0:[100,1] to DSK0:[123,5]
```

ERRORS:

?Disk not mounted
The disk on which LOG is searching for the account is not mounted.

?Command format error
LOG did not recognize the characters that followed LOG as being in the proper format for device or PPN specifications.

?Account number invalid
LOG doesn't recognize the PPN you've supplied. Make sure that you've specified the correct device, and that the PPN is in the proper form.

?Nonexistent device
You've tried to find an account on a device that LOG cannot access; the device is not defined in the DEVTBL command in your SYSTEM.INI, does not have a driver program in area [1,6] of your System Disk, or is not file-structured.

?Bad password
The password you gave LOG wasn't the correct one. Try again, and be careful with your spelling. If you still have no success, check with the System Operator for help.

Already logged in under Devn:[p,pn]
You've tried to log in, but you're already on the system! LOG tells you which device and account you're logged in under (Devn:[p,pn]).

Not logged in
You've typed LOG and a RETURN to find out what account you're logged in under, but you haven't logged in yet. Log in following the procedure given above.

CHARACTERISTICS:

Understands the ersatz devices. Uses as defaults: a) the project or programmer number of the account you are logged into; and b) all units of the device you are logged into (or all units of DSK:, if you are not logged in).

Automatically starts up any file named START.CMD in your account. If a file named MAIL.JNK exits in area [7,2] of the System Disk, LOG displays the first line of that file.

Returns your terminal to AMOS command level.

(Changed 1 May 1980)

logoff

FUNCTION:

LOGOFF logs you out of the account you were logged into.

HINTS/RESTRICTIONS:

Use LOGOFF whenever you leave your terminal for any length of time; this prevents other users from sitting down at your terminal and accessing the files in your accounts.

When you are logged out you are off the system. The only AMOS commands you may use when logged off are: ATTACH, SYSTEM, HELP, LOG, SYSTAT, MEMORY, and SET.

NOTE: LOGOFF deletes any memory modules in your memory partition as it logs you off the system.

FORMAT:

.LOGOFF ↵

OPERATION:

1. Type LOGOFF followed by a RETURN:

   .LOGOFF ↵

2. LOGOFF logs you off the system and lets you know which account you are leaving. (The number it displays is the PPN of that account.)

   .LOGOFF ↵
   User 110,3 logged off

ERRORS:

?Not logged in
If you try to use LOGOFF when you are not logged into the system, LOGOFF cannot log you out since it has nothing to log you out of.

CHARACTERISTICS:

Resets your default device to DSK0:. (Therefore, if you don't include a device specification the next time you use LOG, the system begins looking for your account on the System Disk, DSK0:.)

(1 October 1979)

Resets the flags in your job table to ECHO, OCTAL, NODSKERR, NOVERIFY, and NOGUARD. (See the SET reference sheet for an explanation of these flags.)

Returns your terminal to AMOS command level.

lookup

FUNCTION:

Allows you to check for the existence of a file from within a command file, and then choose whether to continue or abort execution of the command file depending on whether or not the searched-for file exists.

HINTS/RESTRICTIONS:

May be used only in a command file or DO file. Used in combination with the GOTO command file command, LOOKUP allows you to make conditional transfers within a command file. (See the GOTO and EXIT reference sheets.) For more information on LOOKUP, refer to New Features in Command Files and DO Files, (DWM-00100-63), in the "User's Information" section of the AM-100 documentation packet.

FORMAT:

LOOKUP Filespec{/}{message}

where "/" tells LOOKUP not to abort the command file if the Filespec is not found, but to skip over the next command file line. You may also supply an optional message which LOOKUP displays if the file is not found.

DEFAULTS:

If portions of the Filespec are omitted, LOOKUP assumes the account and device you are logged into and a .PRG extension. If you do not supply a message, LOOKUP allows the appropriate AMOS error message to be printed if Filespec is not found.

OPTIONS:

/	If no "/" appears after LOOKUP, if Filespec is not found, AMOS aborts the command file. If a "/" appears, LOOKUP moves directly to the next line in the command file if Filespec is not found; otherwise, AMOS skips over the next line in the command file after the LOOKUP command, and continues execution at that point.
message	An optional message that LOOKUP displays if Filespec is not found. (If a "/" appears, the message must follow the slash.) The message may be any text you wish, as long as it fits on the command line.

OPERATION:

1. Place LOOKUP anywhere in a command or DO file where you want to check for the existence of a file. To abort the command file if the specified file does not exist, do not use the "/" symbol. For example:

```
; DO file to copy a file into an archive account.
;
; (NOTE: user of file must be logged into a Project
; 200 account.)
;
; If file does not exist, abort command file so
; we don't copy everything into the archive
; account instead.  ($0 is the parameter supplied
; by the user of the DO file when the DO file is
; invoked.)
;
LOOKUP $0 ?That file does not exist-- try again.
;
COPY DSK4:[200,34]=$0
```

2. To continue execution of the command file if the specified file does not exist, include the "/":

```
; DO file to print a file.
LOOKUP $0/?That file does not exist--no file printed.
GOTO NOTFOUND;   Oops file not found.
;
; Print the file.
PRINT $0
EXIT *Your file has been printed.*
;
;NOT FOUND
:<If you would like to create the file, type a RETURN;>
PAUSE If don't want to create it, type anything but a RETURN:
VUE $0
Y
```

(For information on the EXIT, PAUSE, and GOTO statements, see the reference sheets for those commands.)

ERRORS:

If it is not able to find the specified file, LOOKUP displays the message (if any) that appears on its command line; if that message does not exist, LOOKUP displays the appropriate AMOS error message. For example:

?Cannot OPEN DSK0:LABDAT.TXT - file not found

CHARACTERISTICS:

May appear only in a command file or DO file.

If portions of a file specification are omitted, LOOKUP assumes a .PRG extension and the account and device you are logged into.

Returns your terminal to AMOS command level, if "/" symbol is not included on LOOKUP command line and if specified file is not found.

FUNCTION:

Assembles an assembly language program file into an unlinked machine language file.

HINTS/RESTRICTIONS:

MACRO is a full macro-assembler. It assembles a source text file (.MAC file) into an intermediate object form (.OBJ file) that can be linked into an executable machine language program (.PRG or .OVR) file. MACRO understands nested macro calls, nested conditionals, and program segmentation (via overlay statements, AUTOEXTERN, EXTERN, and INTERN statements).

MACRO runs in five distinct phases, some of which are called in response to optional switches or special situations. If your file contains no internal or external references, MACRO automatically calls the linkage editor to link your program as Phase 4 of the assembly; otherwise, you must explicitly use the LINK or SYMBOL commands to create a resolved, executable program from the .OBJ file created by MACRO.

NOTE: For information on using the macro-assembler, LINK, and SYMBOL, see the manual AMOS Assembly Language Programmer's Manual, (DWM-00100-43). This manual also contains information on: the object file library generator, LIB; the global cross reference generator, GLOBAL; and the dynamic symbolic debugger, DDT.

FORMAT:

.MACRO filespec{/switches} RET

where filespec selects the source file you want to assemble, and the optional /switches selects the MACRO options you want to use.

DEFAULTS:

MACRO assumes a file extension of .MAC. Unless you specify one of the listing switches, MACRO does not provide an assembly listing.

If you omit device and account specifications, MACRO looks for the specified file in the device and account you are logged into.

(Changed 30 April 1981)

OPTIONS:

You may select one of the options below by specifying the appropriate option request switch at the end of the MACRO command line. You may specify multiple switches by entering them after a single / symbol at the end of the command line.

/B text — Generate a bottom footer title on every page of the listing using the rest of the text on the command lin as the title. /B must be the last switch on the command line.

/C — Include conditionals in the assembly listing. (Conditionals are usually suppressed.)

/E — Write to the assembly listing only those lines that contain an error.

/H — List binary code in hexadecimal instead of octal form in the assembly listing.

/L — Generate a listing file by calling Phase 3 during the assembly. Creates the output file with the same name as your source file but a .LST extension.

/O — Use the current object file by omitting assembly Phases 1 and 2.

/R — Generate a symbol cross reference, which appears at the end of the regular assembly listing.

/T — Display the assembly listing on your terminal instead of writing it to a disk file.

/V{a}:X — Allows you to specify a value on the MACRO command line that can be examined during the assembly process. "a" identifies the type of value specified, and "X" is the value. (See Chapter 9 of the AMOS Assembly Language Programmer's Manual for more information on this switch.)

/X — List in your assembly listing all macro expansions. (Macro expansions are usually suppressed.)

OPERATION:

1. Type MACRO followed by the specification of the file you want to assemble. Then type a RETURN. For example:

```
.MACRO MATH.MAC (RET)
```

If you want to select one or more of the MACRO options, specify the appropriate switches at the end of your command line after a single / symbol (for example: MACRO MATH/RT).

(Changed 30 April 1981)

2. As it assembles your file, MACRO reports on the status of the assembly process. It tells you what errors occurred, how large your program is, whether it is COPYing from any copy files, and if any symbols have been automatically EXTERNed. For example:

```
.MACRO SAVTXT.MAC/L (RET)

== Macro Assembler Version 1.1 ==

Processing SAVTXT.MAC

Phase 1:   Copying from DSK0:SYS.MAC[7,7]
           Work area: 39162 bytes, 3614 used
Phase 2:   Object file finished
Phase 3:   Listing file finished
Phase 4:   Program file finished  [Program size = 60. bytes]
.
```

3. MACRO produces a .OBJ file of the same name as the file you specified on the MACRO command line. (You may modify this output file name by using the OBJNAM pseudo opcode in your source file.) If MACRO does not automatically call Phase 4 to create an executable .PRG file, you must use the linkage editor, LINK (or SYMBOL).

ERRORS:

You can see the following error codes in your assembly listing:

A	Branch address was out of the 127-word range.
B	Boundary error-- a word operand was on an odd byte address.
C	Conditional statement syntax error.
D	Duplicate user symbol. (Symbol was defined more than once.)
I	Illegal character in source line.
M	Missing term or operator in operand or expression.
N	Numeric error which indicates a digit out of the current radix range.
P	An expression which had to be resolvable on the first pass of the assembly could not be resolved.
Q	Questionable syntax. This is a general catch-all error code.
R	Register error-- a register expression was not in the range of 0-7.
T	Source line or operand terminated improperly.
U	Undefined user symbol during Pass 2.
V	Value of an absolute parameter was out of its defined range.
X	Assembler system error-- please notify Alpha Micro.

You may see the following error messages during the program assembly:

%Invalid control parameter value

You used the /V assembly switch to specify a value on the MACRO command line, but something was wrong with the format of the option request. (For example, the value after the /V switch was missing or was incorrect.)

?Cannot OPEN Devn: - invalid file name

There is something wrong with the format of your command line. For example, you may have tried to use an assembly switch but forgot to place it at the end of the file specification. All switches must appear at the end of the command line.

?File specification error

There is something wrong with the format of your command line. For example, you typed MACRO followed by a RETURN.

?MACn.OVR not found

where n is a number from 0 to 5. MACRO cannot find one of the overlays that are a part of MACRO. Make sure that the specified file is in DSK0:[1,4]. If the file is not there, contact the System Operator.

?Copy file filespec not found

where filespec is the file specification you supplied to the COPY pseudo opcode. For information on copy files, see the AMOS Assembly Language Programmer's Manual.

?Expression stack error

This is an internal MACRO error. You should never see it-- but, if you do, check your source program to see if you made any errors in specifying expressions.

NO ENDM STATEMENT

You left off the final statement, "ENDM", in a macro definition. Check your source program.

[SYNC ERROR]

MACRO generates a listing file by reading the source file and the object file and synchronizing the two to come up with the listing line data. If these two files get out of sync, there is no way that the listing may proceed, and you see this error message. MACRO then closes out the list file at the point of the sync error, but the line that caused the error will not have been included. This message probably means that you are using an object file that was generated by a different version of MACRO than the one you are using now, and you have specified the /O assembly switch. Or, you may have found an undiagnosed assembler bug. If you see no obvious errors in your source file, try to re-assemble the program again without specifying the /O switch (thus building a new object file).

CHARACTERISTICS:

A multi-pass macro assembler. Provides options that allow you to generate an assembly listing and a symbol cross reference. MACRO handles nested conditionals and nested macros. It also handles local symbols.

MACRO is not re-entrant, and may not be loaded into system memory.

The default extension is .MAC; the default device and account are the ones you are logged into.

(Changed 30 April 1981)

make

FUNCTION:

Creates the first record of a sequential file.

HINTS/RESTRICTIONS:

One of the system text editors, EDIT.PRG, will not allow you to edit a file that does not yet exist. The MAKE command "starts" a sequential file by making an entry for the file in your User File Directory and creating an empty disk record for that file.

Before editing a new file with EDIT, create the file using the MAKE command.

FORMAT:

.MAKE Filespec ↵

where Filespec selects the name, device, and account you want to assign to the new file.

DEFAULTS:

MAKE assumes an extension of .MAC and the device and account into which you are currently logged.

OPERATION:

1. Type MAKE followed by the specification of the file you want to create. Then type a RETURN. For example:

   .MAKE SRCFIL.BAS ↵

2. When MAKE is finished, you see the AMOS prompt.

ERRORS:

If you do not include a file specification on the MAKE command line (that is, you type MAKE followed by a RETURN), you see:

?File specification error

CHARACTERISTICS:

Assumes an extension of .MAC.

Returns your terminal to AMOS command level.

(1 October 1979)

FUNCTION:

Allows you to see what modules are in your memory partition and in system memory. Also provides information about those memory modules.

HINTS/RESTRICTIONS:

When you load disk files into your memory partition, those copies in memory are called memory modules. A memory module retains the same name and extension as its corresponding disk file.

You may not use MAP to see information about memory modules in other users' partitions.

MAP displays memory addresses in the number base the system is using for your numeric displays (usually octal). If you want MAP to display these memory addresses as hexadecimal numbers, make sure that the HEX option is in effect before using MAP. For information on changing the number base the system uses for your numeric displays, see the SET reference sheet.

FORMAT:

_MAP {Filespec}{/Switches} ↵

where Filespec selects the memory module about which you want information. /Switches select various options (see below).

DEFAULTS:

If you do not specify a filespec, MAP assumes that you want information on all memory modules in your partition.

If you do not specify switches, MAP assumes that you want full information on all memory modules in your partition.

The default switches are: /FSBHMU.

OPTIONS:

You may choose among several MAP options by including one or more of the following switches at the end of the command line following a slash. (MAP assumes that each character after the slash up to the RETURN at the end of the command line is a separate switch.) The MAP switches are:

/F FREE - Display number of free memory bytes available (in decimal). You must use with the /S switch.)

/S SIZE - Display number of bytes (in decimal) of each module.

/B BASE ADDRESS - Display octal base memory address for each module.

/M MODULES - Display information about modules.

/U USER MEMORY - Display information about modules that are in your memory partition.

/R RESIDENT MEMORY - Display information about modules that are in system memory.

/H HASHMARK - Display hashmarks for each memory module.

The default switches are: /FSBHMU.

OPERATION:

1. To see what modules are in your memory partition, type MAP followed by a RETURN:

.MAP ↵

You see a list of the memory modules currently in your partition. Each line of the display contains this information: 1. module name; 2. extension; 3. size in bytes (decimal); 4. octal base address (the memory address where the module begins); and, 5. hashmark (an identifying code unique to that module). The last line of the display tells you how many bytes are free in your memory partition and gives the octal memory address of the first free memory location. For example:

CREAT	PRG	566	033722	167-536-542-221
XCHG	PRG	1016	035022	513-543-124-555
QUBLK	PRG	335	037024	435-713-521-434
FREE		48416	040736	

The second line of the MAP display above tells us that the memory module XCHG.PRG is in memory, that it takes up 1016 bytes, that it begins at memory location 035022, and that its hashmark is 513-543-124-555. The last line of the display tells us that we have 48,416 bytes free and that the first free memory location is at address 040736.

2. To see information about a particular memory module, include a filespec on the MAP command line. For example:

```
.MAP LOG ↵
LOG     PRG  1016  030522   432-672-122-411
```

3. To see information about modules in system memory instead of your own partition, use these switches: /RSHMF. For example:

```
.MAP/RSHMF ↵
TRM     DVR      252      552-107-745-717
```

ERRORS:

MAP generates no error messages.

CHARACTERISTICS:

Default switches are: /FSBHMU.

Returns your terminal to AMOS command level.

memdef

FUNCTION:

As part of the system initialization command file, MEMDEF allows you to define memory banks on a system that uses memory management. At AMOS command level, MEMDEF displays the current memory configuration on your system.

HINTS/RESTRICTIONS:

If you are not familiar with bank switching memory on the Alpha Micro system, refer to the documents Memory Management Option, (DWM-00100-10), and Setting Up Multiple Piiceon 64K Memory Boards, (DWM-00100-34), in the System Operator's Information section of the AM-100 documentation packet.

We will not discuss here the mechanics of setting up a bank-switched system, but briefly: bank switching memory allows you to access more than 64K of memory on your system, although each individual user is still restricted to a 64K maximum.

When you include it in the system initialization command file, SYSTEM.INI, MEMDEF allows you to define the various banks of memory on the system. Unless the SYSTEM.INI contains MEMDEF commands, the operating system cannot recognize any memory beyond the first 64K (1K = 1024; 64K = 65536). With MEMDEF commands, the SYSTEM.INI is able to define several different sets (or "banks") of memory, each of which may be up to 64K in size. The MEMDEF commands are entered into SYSTEM.INI contiguously; the first command defines the memory configuration of the first bank (bank0), the second command defines the second bank (bank1), and so on.

At AMOS command level, MEMDEF displays the memory configuration of the system.

NOTE: Use the JOBMEM command to allocate memory to jobs on a bank-switched system. (See the JOBMEM reference sheet.)

FORMAT:

.MEMDEF↵

or:

MEMDEF Port-address,On-value,Off-value

Use the first format at AMOS command level to find out your system's memory configuration. Use the second format within the SYSTEM.INI to define the system memory banks.

(1 October 1979)

OPERATION:

At AMOS command level:

1. Type MEMDEF followed by a RETURN:

```
.MEMDEF↵
```

2. MEMDEF now displays the memory configuration of your system. (See EXAMPLES, below, for a typical display.)

In the system initialization command file:

1. One MEMDEF command must appear in the SYSTEM.INI for each memory bank on the system. All MEMDEF commands must appear before the DEVTBL and BITMAP commands.

2. The MEMDEF defines a memory bank by telling the system the following information for that bank:

   a. The I/O port address of the memory board used by the bank. This value is assigned to a specific memory board via switches on that board, and serves to uniquely identify that board to the system. Each memory board usually has its own address. The address of the first memory board is usually 100, octal.

   b. The ON and OFF constants. These values determine which sections of memory on the memory boards are assigned to the bank, and whether they are turned ON or OFF. One value specifies the number to be sent to the memory board to turn those sections of memory ON; one specifies the number to be sent to the memory board to turn them OFF. These values vary depending on the type of memory boards you use. Refer to the documentation for that memory board for this information.

3. A typical MEMDEF statement might look like:

```
MEMDEF 100,0,14
```

   where 100 selects the I/O port address used by the memory board that makes up the bank, 0 is the ON constant, 14 is the OFF constant.

4. Note that you may incorporate more than one memory board in a bank by including the slash symbol. For example:

```
MEMDEF 100,1,0/101,1,0/102,1,0
```

defines a single memory bank that uses portions of three different memory boards that are located at I/O port addresses 100, 101, and 102.

ERRORS:

MEMDEF generates the following error messages:

MEMORY MANAGEMENT NOT ACTIVE
You have used MEMDEF from AMOS command level, but your system does not bankswitch memory.

[FORMAT ERROR IN PARAMETER SEQUENCE]
MEMDEF does not understand the MEMDEF command line you supplied. Make sure that you are using the / symbols correctly, and that each MEMDEF line contains I/O port address, ON constant, and OFF constant.

?All 64K appears to be sharable memory
Because of the way the address switches on the memory board are set up, the system believes that the entire bank is supposed to be sharable (i.e., that there is no user memory). Check the memory board switches against the memory board documentation.

?No switchable memory found
MEMDEF looks for the memory boards in the system. If it can't find any memory at the I/O ports you have specified, it is unable to access the memory on the system. Check your memory board I/O address switches and make sure that the I/O port addresses in your MEMDEF statements are good.

CHARACTERISTICS:

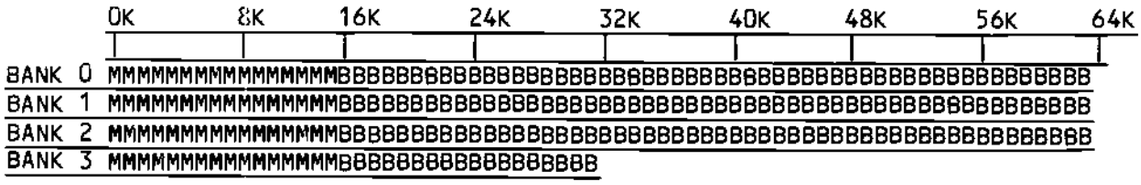
Serves both as a system initialization command and as a user command.

As part of the SYSTEM.INI, defines memory banks; however, DOES NOT allocate memory to jobs. To allocate memory to jobs on a bank switched system, use the JOBMEM command.

Returns your terminal to AMOS command level.

EXAMPLE:

When you use MEMDEF at AMOS command level, you see a representation of the memory banks on your system. This display differs depending on the particular memory configuration of your system, but a typical display might look something like this:



The display above shows a system that contains four memory banks (#0-3). The first 16K of each bank is used by the monitor; the rest is available for jobs. The actual allocation of the memory to jobs is left to the JOBMEM command.

M stands for Monitor memory; B stands for Bank-switched memory. (Another symbol that can appear in this display is S, for Sharable memory. Sharable memory is that memory which all jobs can use, but that is not allocatable to a specific job.)

memerr

FUNCTION:

Enables double-bit memory error detection.

HINTS/RESTRICTIONS:

The Piiceon 32K-word memory boards have error correction and detection capabilities. MEMERR initializes the memory board by instructing it to abort when a double-bit memory error occurs.

You may only use MEMERR with the Piiceon 32K-word memory boards (SuperMem).

Make sure that the memory board is properly jumpered for the I/O port you specify in the MEMERR command. The error interrupt-enable jumper (jumper 54) must be installed on the board.

You may assign the same I/O port to more than one memory board (you can have several 32K-word boards in a bank-switched system), because the system only issues write-status commands to the memory boards.

If you don't use MEMERR, the Piiceon 32K-memory board automatically corrects single-bit errors, but ignores double-bit errors. When you use MEMERR, single-bit errors are still corrected, but the system halts if a double-bit error occurs. (At the time of a system halt, if MEMERR is in effect, look at the Piiceon memory board. If the error light-- a red LED-- is lit, the halt was the result of a double-bit memory error.)

You will probably want to put MEMERR into your system initialization command file, although you may use it at AMOS command level. (Edit the file DSK0:SYSTEM.INI[1,4] with one of the system text editors, VUE or EDIT. Place the MEMERR command after the TRMDEF commands.)

NOTE: MEMERR was designed to be used with the AM-100/T CPU. If you want to use your Piiceon 32K-word memory boards as 64K-byte memory boards (i.e., you want to use the AM-100 CPU), you may do so by selecting a different error interrupt-enable jumper (jumper 54 is the AM-100/T interrupt line), and enabling that line on the AM-100 CPU board.

FORMAT:

MEMERR I/O-Port

where I/O-Port selects the port you have assigned to the memory board error register. (This I/O port is usually set to 250, octal or A8, hex.)

OPERATION:

1. Use one of the system text editors to edit the SYSTEM.INI. Place the MEMERR command after the TRMDEF statements. For example:

```
:T
JOBS JOB1,JOB2,SPOOL
TRMDEF TRM1,AM300=1,ADM3,100,100,200
TRMDEF TRM2,AM300=3,SOROC,100,100,100
MEMERR 250
        .
        .
        .
```

2. If the system halts, look at the error light on the memory board; if it is lit, this indicates that the system halted because of a double-bit memory error.

3. If frequent double-bit memory errors occur, you may want to replace the memory board at fault.

ERRORS:

MEMERR generates no error messages.

CHARACTERISTICS:

Should be included in your SYSTEM.INI if you are using Piiceon 32K-word memory boards.

Signals double-bit memory errors by halting the system.

Returns your terminal to AMOS command level.

memory

FUNCTION:

Allocates memory to your job.

HINTS/RESTRICTIONS:

IMPORTANT NOTE: Do NOT use the MEMORY command on systems that use memory management (that is, systems that bank switch memory) unless your job is in Bank Number Zero of that system. To allocate memory on a bank-switched system, use the JOBMEM command. (See the JOBMEM reference sheet.)

You may not increase your memory allocation beyond its current amount unless you first use the MEMORY 0 command. If you allocate yourself zero memory, the first time you try to run a command or program AMOS will reassign to you as much free memory as it has available for your memory partition.

You cannot increase your memory allocation if the other users on the system have already been allocated the rest of memory; that is, you cannot take for yourself memory that has already been allocated elsewhere.

All memory locations in your job's memory partition must be contiguous. That is, if you want to allocate 16K more memory to your job, 16K of memory must be available beginning at the end or beginning of your present memory partition. (By coordinated use of the MEMORY command, users on the system can move their memory partitions about in memory to rearrange the allocations for specific jobs.)

You must not use the MEMORY command to allocate memory in a system that bank switches memory; problems could result. (The exception to this rule is that a job in Bank Zero may use the MEMORY command.)

FORMAT:

.MEMORY ↵

or:

.MEMORY Memory-allocation{K} ↵

Use the first format to tell you how much memory is allocated to your memory partition. The second format allows you to change the amount of memory allocated to your job, where Memory-allocation is the amount of memory (in bytes) you would like to allocate to your job.

If you include a K after the amount, MEMORY multiplies the amount given in your Memory-allocation by 1000. (For example, the command .MEMORY 16K tells the system to give you 16000 bytes of memory.)

OPERATION:

1. To see how much memory has been allocated to your job, type MEMORY followed by a RETURN. For example:

```
.MEMORY ↵
CURRENT MEMORY ALLOCATION IS 42738 BYTES
```

2. To allocate memory to your job, type MEMORY followed by the amount of memory (in bytes) you want to allocate to yourself. If you type a K after the amount, MEMORY multiplies the amount by 1000.

```
.MEMORY 32K ↵
[32000 BYTES ASSIGNED]
```

ERRORS:

MEMORY generates no error messages. However, if you try to allocate a very small, nonzero amount of memory to your job, you will not have room to load the MEMORY command back into your partition to change that allocation:

```
.MEMORY 10 ↵
[10 BYTES ASSIGNED]
.MEMORY
?Insufficient memory for program load
?MEMORY?
.
```

So, be careful to check your MEMORY command line to make sure that you've made no typos. Also, be careful when attaching your terminal to another job; make sure that the job has some memory allocated to it, or you will run into the same kind of problem.

CHARACTERISTICS:

Not for use on a system that uses memory management.

Returns your terminal to AMOS command level.

mongen

FUNCTION:

Generates a new system monitor.

HINTS/RESTRICTIONS:

Generates a new system monitor by overlaying the specified disk driver program into the existing monitor.

This new monitor allows you to access any disk for which you have a disk driver program as the System Device.

The monitor that you will usually use is SYSTEM.MON in account DSK0:[1,4]. The disk driver you will use is one of the driver programs in DSK0:[1,6]. MONGEN inserts the specified driver into the monitor (overlaying the old driver) and then leaves the new monitor in memory. At that point you can save the new monitor (using the SAVE command) or test it directly from memory (using the MONTST command).

For information on MONTST and SAVE, refer to the MONTST and SAVE reference sheets. For more information on MONGEN, refer to the document Generating System Monitors, (DWM-00100-31), in the System Operator's Information section of the AM-100 documentation packet.

DEFAULTS:

The default device and account specification for the monitor file is DSK0:[1,4]. The default system monitor is DSK0:SYSTEM.MON[1,4].

The default device and account specification for the disk driver is DSK0:[1,6]. The default disk driver file extension is .DVR.

FORMAT:

.MONGEN ↵

OPERATION:

1. Type MONGEN followed by a RETURN:

   .MONGEN ↵

2. MONGEN now asks you for the specification of the system monitor you want to modify:

   INPUT MONITOR NAME:

Enter the file specification of the monitor program you are going to use. If you want to use DSK0:SYSTEM.MON[1,4], you may enter just a RETURN because that file specification is the default. The default monitor device and account is DSK0:[1,4].

MONGEN locates the specified monitor and loads it into your memory partition. Be sure that you have enough memory to accommodate the monitor and disk drivers as well as the MONGEN program itself (usually at least 16K of memory).

3. Now MONGEN asks for the specification of the disk driver you want to insert into the monitor:

NEW DISK DRIVER NAME:

Enter the file specification of the correct disk driver program. You may NOT enter just a RETURN. The default device and account specification is DSK0:[1,6]. The default file extension is .DVR.

4. MONGEN asks for a name to be given to the new monitor:

NEW MONITOR NAME:

Enter a one- to six-character name (the default extension is .MON). This name is now the name of the new monitor.

5. You can now test the new monitor by using the MONTST program or you can save the monitor as a disk file by using the SAVE command. For example:

.SAVE TRISYS.MON ↵

Remember that MONGEN does not affect the running monitor either in memory or on the System Disk. Nor does MONGEN test the new monitor; it merely builds a new monitor as a module in your memory partition.

ERRORS:

INPUT MONITOR Filespec NOT FOUND
DISK DRIVER Filespec NOT FOUND

MONGEN could not find the specified monitor or disk driver file. Make sure that you included the correct account and device specification.

You can also see several system messages if your device specifications are invalid. For example:

?Cannot READ Filespec - device does not exist
?Cannot READ Filespec - disk not mounted

Make sure that your file specifications contains valid device specifications. Check your spelling. If you see the disk not mounted message, use the MOUNT command to mount the devices you want to access.

CHARACTERISTICS:

Generates a new monitor by overlaying the disk driver program with a new disk driver.

Returns your terminal to AMOS command level.

montst

FUNCTION:

Tests a new monitor or system initialization command file by bringing up the system under the control of the new monitor or command file.

HINTS/RESTRICTIONS:

Useful for testing the new monitor generated by the MONGEN command. Also useful for testing a newly modified system initialization command file.

IMPORTANT NOTE: The system monitor and the system initialization command file you want to test, along with the the MONTST command itself must all be located on physical drive zero. (That is, for a CDC Hawk hard-disk drive system these programs must be located on the fixed-platter; for a floppy-disk system these programs must be located on the disk in the System Drive Zero.)

You can also use MONTST with MONGEN to boot up on a floppy drive even if your System Device is a hard disk. For example, say that you use MONGEN to build a monitor (named WNGMON.MON) that contains the Wangco floppy-disk device driver; you can use the MONTST command (.MONTST WNGMON.MON,SYSTEM.INI) to boot the system up on the Wangco drive.

You may only use MONTST if your job is running in the first memory partition of the system (Bank Zero for bank switched systems).

FORMAT:

.MONTST Monitorspec,Command-filespec RET

where Monitorspec specifies the monitor you want to test (which must be on DSK0:) and Command-filespec specifies the system initializaton command file you want to use (also on DSK0:).

DEFAULTS:

The default specification for the system initialization command file is DSK0:SYSTEM.INI[1,4]; the default file extension is null (that is, SYSTEM.).

The default monitor file extension is .MON.

(Changed 30 April 1981)

OPERATION:

1. Type MONTST followed by the specifications of the system monitor and the system initialization command file under whose control you want to bring up the system (separated by a comma). Then type a RETURN. For example:

```
.MONTST SYSTEM.MON,NEWSYS.INI (RET)
```

ERRORS:

You may see any of the standard system error messages when using MONTST. For example:

filespec NOT FOUND
MONTST couldn't find the monitor or the system initialization command file that you want to test.

?File specification error
MONTST did not understand your command line. For example:

```
.MONTST (RET)
?File specification error
```

CHARACTERISTICS:

MONTST brings the system up under control of the specified system monitor and system initialization command file.

Returns your terminal to AMOS command level.

mount

FUNCTION:

You can use MOUNT to mount a disk, unmount a disk, or display a list of all mounted disks on the system.

HINTS/RESTRICTIONS:

The system has no way of knowing when you change disks in your disk drives. When you change disk cartridges in your hard disk system or change floppy disks in your flopppy disk drives, you must tell the system that you have done so. (Your SYSTEM.INI can be set up to initially mount your disks for you every time the system comes up, but if you change any of your disks after the system is up and running, you still must explicitly mount them using the MOUNT command.)

If you do not mount a disk after you change it, when it comes time to transfer data it is likely that the system will write to the new disk as if it had the same free and used locations as the old disk. (That is, it will make its decision of where to write data to the disk based on the bitmap of the old disk.) Using the MOUNT command tells the system that you are changing disks and that the system must therefore look at the new disk to fetch the appropriate bitmap. (A disk bitmap is a map of the disk; it tells the system which disk blocks are free and which are used.)

IMPORTANT NOTE: Never mount or unmount a disk while someone is accessing the disk; to do so can severely damage the file structure of that disk!!

FORMAT:

.MOUNT {Devn:{/switch}} RET

where Devn: is the specification of the device you want to mount. If you do not include a device specification, MOUNT displays a list of all of the disks mounted on the system.

OPTIONS:

The /U switch tells MOUNT that you want to unmount the disk in the specified device.

The /W switch tells MOUNT that you want to wait until the specified device is ready before mounting the disk.

(Changed 1 May 1980)

OPERATION:

1. To mount a disk, enter MOUNT followed by a device specification; then type a RETURN. For example:

```
.MOUNT DSK1: RET
```

   If MOUNT successfully mounts the disk and that disk has a label, MOUNT lists the Volume Name and Volume ID of the disk mounted. For example:

```
.MOUNT DSK5: RET
Backup Disk #3 (BACKUP03) mounted
```

   MOUNT displays the disk label, so that you can verify that you have mounted the proper disk.

2. If you want to mount a disk, but you know that the device containing the disk is not yet ready, use the /W switch. MOUNT will now wait until the device is ready before mounting the disk. For example:

```
.MOUNT HWK3:/W RET
```

   Of course, you will only use the /W option if you expect the drive to be ready momentarily, since your terminal is tied up until the device is ready and the disk can be mounted.

3. If you want to unmount a disk, type MOUNT, the device specification and a /U. Then type a RETURN. For example:

```
.MOUNT AMS0:/U RET
```

4. To display a list of the disks mounted on the system, type MOUNT followed by a RETURN:

```
.MOUNT RET
```

   Now you see a list of the mounted disks. MOUNT displays the Volume Name and Volume ID of each disk that contains a label. (For information on labeling a disk, and on the Volume Name and Volume ID, see the LABEL reference sheet in this manual.) For example:

```
.MOUNT (RET)
Disks mounted:

DSK0:  System Disk (SYS001)
DSK1:  Payroll Data (PAYROLL01)
HWK2:  Transfer Disk (TRANS001)
AMS0:
AMS1:  Documentation Archives #1 (ARCHIV01)
```

ERRORS:

?Nonexistent device

You tried to mount a disk that AMOS cannot access; the device is not defined in the DEVTBL command of your SYSTEM.INI, is not file structured, or does not have a driver program in area [1,6] of your System Disk.

?No disk currently mounted

You tried to unmount a disk that is not currently mounted. Check your spelling to make sure that you specified the correct logical unit.

?Device not ready

The specified device is not ready. Try again when the device has been fully cycled up. If you wish MOUNT to wait until the device is ready, use the /W switch.

?WARNING - BADBLK.SYS has a bad hash total

When reading in the alternate track information for a device that uses alternate tracks, MOUNT discovered that the BADBLK.SYS file had been damaged and did not have the correct hash total. Check with the System Operator for instructions.

CHARACTERISTICS:

Allows you to mount and unmount disks. Also displays a list of all mounted disks.

If a disk contains a label, MOUNT displays part of the label when it mounts that disk.

Returns your terminal to AMOS command level.

newtrm

FUNCTION:

Creates a standard terminal driver.

HINTS/RESTRICTIONS:

Because of the large number of terminals available for use with the Alpha Micro Operating System, it is no longer possible to provide a separate driver program for each kind of terminal. Alpha Micro does provide terminal drivers for many popular terminals, which can be found in the system disk account [1,6].

Creating a terminal driver program, or even modifying one, has always required a great deal of programming experience and skill. Using NEWTRM, you can create a terminal driver for any terminal you choose, implementing all the features built into that terminal. No particular expertise is required; NEWTRM builds the terminal driver in a standard format by asking you a number of questions about the features of the terminal.

Refer to the user's or operator's manual for the terminal to find the information NEWTRM asks you about the terminal's features and capabilities.

Also, refer to the document Building a Terminal Driver (The NEWTRM Program) in the "System Operator's Information" section of the AMOS Software documentation packet for more information on NEWTRM, and how to find and enter the terminal's various parameters in response to its questions.

NEWTRM uses the following files. Do not erase or modify them.

DSK0:NEWTRM.PCF[7,5]	DSK0:TDV1.MAC[1,6]
DSK0:ECHO.MAC[1,6]	DSK0:TDV2.MAC[1,6]
DSK0:TABDEF.MAC[1,6]	DSK0:VARDEF.MAC[1,6]

FORMAT:

.NEWTRM RET

OPERATION:

1. Log into the system Device Driver Library account:

.LOG DSK0:[1,6] RET

(30 April 1981)

2. Type NEWTRM followed by a RETURN:

```
.NEWTRM (RET)
```

3. NEWTRM gives you a set of instructions:

```
Terminate all input lines with carriage returns.
All numeric input is in decimal.
Separate numeric answers on the same line with spaces
All NUMERIC answers default to 0.
If you have any problems, refer to the NEWTRM documentation.
```

4. NEWTRM now begins to ask you a long series of questions. These questions are structured into four groups.

   a. NEWTRM first asks questions generally applicable to all terminals. First is the unique name you want to give to the driver NEWTRM is about to build. Enter just the name, in six characters or less. Do not supply a file name extension.

```
What is the name of the driver? SOROC (RET)
```

   where SOROC is the name you have selected. The name should readily identify the terminal which the drive will support. You must include this name in the system initialization command file TRMDEF statement that defines your terminal. (For further information on modifying the system initialization command file to add a new terminal to your system, see The System Initialization Command File in the "System Operator's Information" section of the AMOS Software Update documentation packet.)

   NEWTRM asks some more questions of a general nature, dealing with time delays following linefeed and formfeed commands from the driver to the terminal.

   b. NEWTRM then moves to the second group of questions, and accepts information which identifies the terminal type and capabilities to the driver. The driver must know: if the terminal has a keyboard (that is, can be used for input as well as output); if it has a video display (via a CRT or other video display system) or is a "hard copy" (printer-type) terminal; how much to offset the CRT screen "cursor" from "home" when positioning it, and whether the terminal takes the row or the column coordinate first; and the number of rows and the number of columns on the CRT screen. If you are using NEWTRM to build a "hard copy" terminal driver, NEWTRM goes from this point to final processing.

c. In the third group of questions, NEWTRM asks what special functions are implemented by the terminal, and how the driver can access them. These questions ask about function characters, delays between function characters, characters to discard, and the function key leadin code.

In this group, NEWTRM asks you a repetitive set of questions about 28 separate commands. These commands are the TCRT commands supported by Alpha Micro. The terminal driver built by NEWTRM will support any or all of these features which your terminal implements. (NOTE: If you wish to implement other commands, you must modify the source file NEWTRM creates to contain those commands.)

d. The fourth group correlates your terminal's function commands, if any, to the Alpha Micro text editor called AlphaVUE. First you enter the decimal ASCII value transmitted by a function key on your terminal, then you enter the actual VUE control character you want to correspond to it. Do NOT type a RETURN after entering the VUE command. NEWTRM looks for a RETURN to terminate this series of questions, which you must enter after you have correlated all the function keys to VUE commands.

5. When you have entered the last return, NEWTRM creates the source file for your terminal driver. Then you see:
The driver is complete. You may assemble and test it now.

6. Now you must assemble the program by using the MACRO command. Enter:

.MACRO SOROC (RET)

where "SOROC" is the name you used in telling NEWTRM what terminal driver to build.

7. When MACRO finishes and returns you to AMOS command level, rename the .PRG file it produced to the terminal driver extension, .TDV.

.RENAME/D *.TDV=SOROC.PRG (RET)

You now have a finished terminal driver program, called SOROC.TDV, customized for your terminal.

ERRORS:

?Bad answer - try again
You typed an answer that did not start with Y, y, N or n. NEWTRM accepts a number of yes or no answers, as well as numeric ones. You

may use an upper or lower case Y or N, or spell out the word, for your answers.

CHARACTERISTICS:

Requires that you be logged into DSK0:[1,6].

Configures a standard terminal driver in the terminal driver account by asking you questions about the terminal's features and parameters. When you are ready to use the terminal, be sure to add the new terminal driver name in the system initialization command file TRMDEF statement that defines your terminal.

FUNCTION:

Allows the AlphaMAIL Operator to maintain and manage the system's use of AlphaMAIL, the Alpha Micro electronic mail system.

HINTS/RESTRICTIONS:

The AlphaMAIL Operator creates and maintains the AlphaMAIL User Directory for his or her computer system.

The OPR program allows the AlphaMAIL Operator to build and maintain the User Directory, to check the users' mail boxes, to receive messages, and to distribute a message to multiple users (and to specify how many days that message is to be on hold). For information on using OPR, refer to the AlphaMAIL User's Manual, (DSS-10000-06).

The AlphaMAIL Operator works in the AlphaMAIL account, DSK0:[7,2]. This account should have password protection to protect the account from unauthorized use.

OPR is ONLY for the use of the authorized AlphaMAIL Operator. Damage to the AlphaMAIL message system can occur if anyone but the AlphaMAIL Operator is allowed access to OPR.

See the EMAIL reference sheet for information on how the general user may access AlphaMAIL.

FORMAT:

.OPR (RET)

OPERATION:

1. Log into DSK0:[7,2] (which you can also specify as BOX:). Now, type OPR followed by a RETURN:

   .OPR (RET)

   You now see the OPR main menu:

```
OPR Version 1.0

COMMAND CODES:

A               User Directory maintenance
C               Check mail boxes
D               Distribute messages
R               Receive mail
H               Holding message maintenance
S               Display statistics (EPO Operator only)
?               Display command menu
X               Return to AMOS


OPR>
```

2. Select one of the functions listed in the menu by entering the appropriate command code after the OPR prompt symbol, OPR>; then type a RETURN. For example, to look at or change the AlphaMAIL User Directory, use the A command:

```
OPR>A (RET)
```

Now you see the Directory maintenance menu. Most of the OPR commands will cause a new menu to be displayed, along with the prompt symbol for that particular portion of OPR. For example, the Directory maintenance menu looks like this:

```
ACT Version 1.0

COMMAND CODES:

L               List User Directory
P               Print User Directory
A               Add new account to User Directory
D user name     Delete this user's account from Directory
M user name     Modify this user's account in Directory
V               Verify User Directory
?               Display command menu
Q               Return to OPR main menu
X               Return to AMOS

ACT>
```

To return to the OPR main menu from any OPR submenu, enter a Q followed by a RETURN. For example:

```
ACT>Q (RET)
```

3. To exit from OPR or any OPR submenu, enter an X followed by a RETURN. For example:

OPR>X (RET)

OPR now returns you to AMOS command level.

ERRORS:

Below are some of the error messages you can see when using OPR:

?Invalid command - Type ? for help

At any time you may ask ORP to re-display the current menu by entering a ? followed by a RETURN after the current prompt symbol. For example:

OPR>? (RET)

?Invalid entry - try again

The Directory maintenance portion of OPR did not recognize your entry. For example, you entered just a RETURN when OPR asked you for some information.

?You must enter less than 20 characters

You are trying to define a new user account or to modify an existing one, but you entered a user-ID of more than 20 characters. You will have to shorten the user-ID to 20 characters or less.

?You must enter exactly 3 digits

You are trying to define a new user account, but the box number you entered was not three numeric digits. Try again.

?This mail box already exists - try again

Each user mail box on your system must have a unique number. Pick another three-digit number and try again.

?USERNAME is already in the list

Each user in the Directory must have a unique user-ID. Pick another name and try again.

?USERNAME not found

You tried to modify an account for a nonexistent user. Check the User Directory for the list of all users.

?User Directory not found

No User Directory yet exists. You will need to build one. Check Chapter 4 of the AlphaMAIL User's Manual for instructions on building a User Directory. If you know that a User Directory does already exist, then an error has occurred in the AlphaMAIL system. Refer to the AlphaMAIL User's Manual troubleshooting section for instructions.

?FILENAME not found

An error has occurred within the AlphaMAIL system. Refer to the AlphaMAIL User's Manual for instructions.

CHARACTERISTICS:

OPR is used by the AlphaMAIL Operator to maintain and manage the Alpha Micro electronic mail system. It is ONLY for the use of the authorized AlphaMAIL Operator.

OPR displays one main menu and five submenus. To return to the main menu from any submenu, use the Q command. To exit to AMOS from any menu, use the X command. To see the current menu re-displayed, use the ? command.

OPR is re-entrant and reusable.

parity

FUNCTION:

Initializes parity error detection for the AM-710 memory board.

HINTS/RESTRICTIONS:

The AM-710 128K byte memory board has a parity error detection capability which you must enable by using the PARITY command within your system initialization command file. (For information on the AM-710 memory board, see the Alpha Micro Integrated Systems User's Guide, (DWM-00101-01). This manual tells you how to address your AM-710 memory board, and gives bank switching information for the board.)

If you don't use the PARITY command, the AM-710 memory boards will not be able to detect and report parity errors.

Make sure that the memory boards are properly jumpered for the I/O port addresses you supply to the PARITY command. For information on addressing see the Alpha Micro Integrated Systems User's Guide.

NOTE: You will probably want to put PARITY into your system initialization command file, although you may use it at AMOS command level. (For information on editing the system initialization command file, see The System Initialization Command File in the "System Operator's Information" section of the AMOS Software Update documentation packet.)

See Software Installation Instructions for the AM-710 Memory Board in the "System Operator's Information" section of the AMOS Software Update documentation packet for more information on PARITY and memory board error handling.

FORMAT:

PARITY I/O-port1{,I/O-port2,...I/O-portN}

where I/O-port1 through I/O-portN give the I/O port addresses you have assigned to the AM-710 memory boards in your system. (NOTE: This I/O port address is three digits for octal numbers, or is two digits for hexadecimal numbers. (Only enter hex numbers if you have used the SET HEX command for your job.)

OPERATION:

1. Use one of the system text editors to edit the SYSTEM.INI file. Place the PARITY command after the MEMERR command. For example:

```
MEMERR
PARITY 101,102,103
```

2. If you have more I/O port addresses than will fit on one command line, you may follow the first PARITY command with as many others as you need. For example:

```
PARITY 101,102,103
PARITY 104,105,106
```

3. Note that AM-710 memory boards require the presence of the MEMERR command as well as the use of the PARITY command. If your system contains only AM-710 memory boards, you may not include an argument on the MEMERR command line. (See #1 above.) If your system also contains other types of memory boards, you must include the proper I/O error address for those boards. For example:

```
MEMERR 250
PARITY 101,102,103
```

(See the Alpha Micro Integrated Systems User's Guide for information on the I/O error port address for Piiceon 32K word memory boards.)

ERRORS:

Your system can respond to a parity error in a number of ways depending on your system's hardware configuration. See the document Software Installation Instructions for the AM-710 Memory Board in the "System Operator's Information" section of the AMOS Software Update documentation packet for complete information on what happens when a parity error occurs, and how to handle such an error.

The PARITY command can itself generate several error messages if you use the command improperly:

?There is no AM-710 at port address xxx

Where xxx is an I/O port address you specified on your PARITY command line. This address did not match the jumpered I/O port address of any of the AM-710 boards in your system. Check the PARITY command line to make sure that you entered the port address correctly; then check the memory boards to make sure that their port addresses are jumpered correctly.

?Command format error

You did not supply any I/O port addresses, or in some other way used an improper format.

CHARACTERISTICS:

PARITY is re-entrant, reusable, and may be loaded into system memory. It may only be run with AMOS versions 4.5 and later.

PARITY is only for use with AM-710 memory boards that are used with: 1) AM-100/T CPU; or 2) an AM-100 CPU and an AM-120.

pause

FUNCTION:

PAUSE causes a temporary interruption in the processing of the command file in which it appears.

HINTS/RESTRICTIONS:

PAUSE may appear only in a command file or DO file. AMOS stops processing a command file when it reaches the PAUSE statement. At that time you may choose whether to return to AMOS command level or to continue execution of the command file. (When the PAUSE command is processed, PAUSE waits for you to enter a character: a RETURN tells PAUSE to continue execution of the file; any other character tells PAUSE to abort the command file.) When you return to AMOS command level, you may perform a series of actions such as: running a program, editing a text file, using another command file, etc. To resume executing the command file where it was interrupted, use the CONT command at AMOS command level. (See the CONT reference sheet.)

When you interrupt execution of a command file via the PAUSE command, PAUSE stores the remainder of the command file in the disk file CNT.CMD. This file appears in the account and device you are logged into. When you use the CONT command to resume execution of the file, CONT loads CNT.CMD into your memory partition and processes it. Therefore, do not erase the file CNT.CMD unless you do not want to resume execution of the command file it contains.

For more information on PAUSE and CONT, see New Features of Command Files and DO Files, (DWM-00100-63), in the "User's Information" section of the AM-100 documentation packet.

FORMAT:

PAUSE {message}

You may optionally include a message which PAUSE displays to the user of the command file.

OPERATION:

1. Place PAUSE in a command file where you want that file to interrupt execution. You may optionally include a message to be displayed to the user of the command file. For example:

   PAUSE To run the file, type RETURN; otherwise, type X.

   When a PAUSE command is reached, the user of the file may either type a RETURN (to resume execution of the command file) or any

character but a RETURN (to temporarily interrupt the processing of the command file).

2. Once a command file is interrupted via the PAUSE command, the user of the command file is returned to AMOS command level where he or she may run any group of programs or command files. The CONT command resumes execution of the file.

3. As an example of the PAUSE command, let's create a command file that allows you to compile and run BASIC programs:

```
; Command file to compile and run a BASIC program
; (MAIN.BAS).
;
TRACE OFF
LOOKUP MAIN.BAS/?MAIN.BAS does not exist
GOTO NOFILE; File doesn't exist; give user option to create
;
COMPIL MAIN.BAS
RUN MAIN.RUN
EXIT *Returning you to AMOS command level*
;
;NOFILE
:<If you want to create MAIN.BAS, type a RETURN;>
PAUSE otherwise, type any other character:
TRACE ON
VUE MAIN.BAS
Y
```

ERRORS:

PAUSE displays no error messages. However, you can see the message:

%Supersedes existing file

if the following sequence of events occur:

1. You use a command file which contains a PAUSE command that interrupts execution of the file. You then return to AMOS command level. PAUSE writes the remainder of the command file into the disk file CNT.CMD.

2. You then invoke another command file which also interrupts execution because of a PAUSE command. (However, you have not yet typed a CONT to resume execution of the first command file.) PAUSE writes the remainder of the second file into CNT.CMD, thus overwriting the contents of first command file.

If a command file is pending execution (that is, if CNT.CMD exists in the account) when another command file uses PAUSE to interrupt execution, PAUSE tells you so by displaying the above message. This message does not indicate an error, but warns you that you cannot resume execution of the first command file because CNT.CMD now contains another command file.

CHARACTERISTICS:

Used only in a command file or DO file.

Causes temporary interruption of command file execution. (You can resume execution of that command file by using the CONT command.)

May return your terminal to AMOS command level if you choose to abort command file execution.

FUNCTION:

Allows you to change your account password.

HINTS/RESTRICTIONS:

PASS cannot change the password of an account on a write-protected disk, so make sure that the correct device is write-enabled.

A password must be six characters or less in length; PASS does not check the length of the password that you enter. Remember that PASS cuts your password off at the sixth character. To use PASS, you must be logged into the system under the PPN whose password you want to change.

FORMAT:

.PASS {Devn:} ↵

where Devn: is the specification of the logical unit that contains your account.

DEFAULTS:

PASS assumes the logical unit you are currently logged into.

OPERATION:

1. Type PASS, optionally followed by a device name. Then hit RETURN. For example:

```
.PASS DSK1: ↵
```

2. PASS asks you for your old password and your new password. Then it asks you to verify your new password (in other words, to type your new password again):

```
.PASS ↵
Old password:
New password:
Verify password:
EXIT
.
```

PASS now changes your password, displays the EXIT message, and takes you back to AMOS command level.

3. So that your password remains a secret to other users, PASS does not display your old or your new password as you type them.

ERRORS:

You can encounter the following error messages:

?Cannot WRITE Devn: - write-protected

You tried to change the password of an account on a write-protected disk. Try again, but first write-enable the disk. (Devn: is the specification of the device holding the write-protected disk.)

?Bad password

Either you entered your old password incorrectly or when asked to verify your new password you made a typing mistake. In either case, PASS does not change your old password. Try again.

CHARACTERISTICS:

Protects system security by not displaying your old or new password.

Returns your terminal to AMOS command level.

FUNCTION:

Compiles a PASCAL program by invoking the AlphaPASCAL compiler, CMPILR.

HINTS/RESTRICTIONS:

PC is a command file which makes the AlphaPASCAL compiler, CMPILR, easier to use in many applications. It first invokes CMPILR, giving it a filename you specify. Then it tells CMPILR that you want the diagnostic display on the screen as CMPILR compiles the PASCAL program.

You may enter a complete file specification, including the device specification, filename, extension and project-programmer number of the PASCAL file you wish to compile.

For more information on compiling a PASCAL program, refer to the AlphaPASCAL User's Manual, (DWM-00100-07).

FORMAT:

.PC Filespec (RET)

DEFAULTS:

If you omit the device and account specifications, CMPILR assumes the device and account you are logged into. If you omit the extension, CMPILR assumes an extension of .PAS.

OPERATION:

1. Type PC followed by the specification of the file you want to compile. Then type RETURN. For example:

   .PC SALES (RET)

   The file you specify must be a compilable PASCAL program.

2. PC now invokes the AlphaPASCAL compiler, CMPILR, which proceeds to compile the program. PC causes CMPILR to display some diagnostic status messages, so you see something like this:

```
PRUN CMPILR
AlphaPascal V2.0
Source file name? SALES
Diagnostic file name (<return> for terminal?
AlphaPascal Compiler Version 2.0
          <  0>---
PROGRAM   <  3>---------
12 lines
4.82 seconds,  149.48 lines/minute
No compilation errors.
.
```

For more information on CMPILR and what each line of the display means, see the AlphaPASCAL User's Guide, (DWM-00100-08).

ERRORS:

PC generates no error messages itself. However, you may see the standard PASCAL error messages. For a list of those error messages, see Appendix C of the AlphaPASCAL User's Guide, (DWM-00100-08)

CHARACTERISTICS:

Invokes the PASCAL compiler, CMPILR.

Causes one PASCAL program to be compiled.

Causes CMPILR to send diagnostic information to your terminal display.

FUNCTION:

Compiles and links a PASCAL program by invoking the AlphaPASCAL compiler, CMPILR, and the AlphaPASCAL linker, PLINK.

HINTS/RESTRICTIONS:

PCL is a command file which makes the AlphaPASCAL compiler, CMPILR, and the AlphaPASCAL linker, PLINK, easier to use in many applications.

PCL first invokes CMPILR, giving it a filename you specify. Then it tells CMPILR that you want the diagnostic display on the screen as CMPILR compiles the PASCAL program.

Then PCL invokes PLINK, giving it the filename you specify as its code file. PLINK then proceeds to link together the files with the code file name and the extensions .P01, .P02, and P03.

You may enter only the filename and device specification of the PASCAL file you wish to link. PCL assumes the device specification (if you do not enter it) and PPN of the device and account you are logged into, and the extension .PAS.

For more information on compiling and linking program files, refer to the AlphaPASCAL User's Manual (DWM-00100-07).

DEFAULTS:

If you omit the device specification, PCL assumes the device you are logged into.

FORMAT:

.PCL Filename (RET)

OPERATION:

1. Type PCL followed by the specification of the file you want to compile. Then type RETURN. For example:

   .PCL SALES

   The file you specify must be a compilable PASCAL program.

2. PCL now invokes the AlphaPASCAL compiler, CMPILR, which proceeds to compile the program. PCL causes CMPILR to display some diagnostic status messages, so you see something like this:

```
PRUN CMPILR
AlphaPascal V2.0
Source file name? SALES
Diagnostic file name (<return> for terminal?
AlphaPascal Compiler Version 2.0
          <  0>---
PROGRAM   <  3>---------
12 lines
4.82 seconds,  149.48 lines/minute
No compilation errors.
.
```

For more information on CMPILR and what each line of the display means, see the AlphaPASCAL User's Guide, (DWM-00100-08).

3. PCL now invokes PLINK, and passes the filename SALES to PLINK. PCL erases the file SALES.PCF if it exists in the account. You see something like:

```
ERASE SALES.PCF
Total of 1 file deleted, 2 disk blocks freed
```

4. PLINK proceeds to link the files SALES.P01, SALES.P02 and SALES.P03 together into a new file SALES.PCF. As PLINK proceeds, you see a display something like this:

```
.PRUN PLINK
AlphaPascal V2.0
Code file = SALES
Creating new code file SALES.PCF
Library code file for SALES.PCF = STDLIB


Please specify files to be linked into SALES,
 one per line, ending in a blank line

  File 1 = SALES
  File 2 =
Loading program and library dictionaries
Processing SALES
  Linking in global func/proc PROGRAM
Transferring temporary file to new code file
SALES completed
.
```

For more information on PLINK and what each line of the display means, see the AlphaPASCAL User's Guide, (DWM-00100-08).

ERRORS:

PCL generates no error messages itself. However, you may see the standard PASCAL error messages. For a list of those error messages, see Appendix C of the AlphaPASCAL User's Guide, (DWM-00100-08)

CHARACTERISTICS:

Creates an executable .PCF file by compiling and linking a single source (.PAS) file.

Invokes the PASCAL compiler, CMPILR.

Causes one PASCAL program to be compiled.

Causes CMPILR to send diagnostic information to your terminal display.

Erases the .PCF version of the specified file.

Invokes the PASCAL linker, PLINK.

Links the compiled version of the specified file into a resolved, executable program file.

pcu

FUNCTION:

Compiles and updates a PASCAL program by invoking the AlphaPASCAL compiler, CMPILR, to compile a specific module, and invoking the AlphaPASCAL linker, PLINK, to link the module into the program file.

HINTS/RESTRICTIONS:

PCU is a command file which makes the AlphaPASCAL compiler, CMPILR, and the AlphaPASCAL linker, PLINK, easier to use when compiling a single module and updating a program file.

You supply PCU with the name of the module you want to compile and link, followed by the name of the .PCF file you want to link the module into.

PCU invokes CMPILR, giving it the module name you specify. PCU tells CMPILR that you want the diagnostic display on the screen as CMPILR compiles the module.

PCU invokes PLINK, giving it the filename of the .PCF file you specify as its code file. PCU tells PLINK to update the .PCF file.

PCU tells PLINK not to change the standard library code file for the .PCF file while updating it.

You may enter a complete file specification, including the device specification, filename, extension and project-programmer number, of the PASCAL module you wish to compile.

For more information on compiling and updating modules, refer to the AlphaPASCAL User's Manual (DWM-00100-07).

FORMAT:

.PCU Modulename Filename (RET)

where Modulename is the name of the module you want to update, and Filename is the name of the .PCF file you want to link the module to.

DEFAULTS:

If you omit the device and account specifications, CMPILR and PLINK assume the device and account you are logged into. If you omit the extension, CMPILR and PLINK assume an extension of .PAS.

(30 April 1981)

OPERATION:

1. Type PCU followed by the specification of the module you want to compile, followed by the name of the .PCF file you want to link the module to. Then type RETURN. For example:

```
.PCU COMPAR SALES (RET)
```

   The file you specify must be a compilable PASCAL program.

2. PCU now invokes the AlphaPASCAL compiler, CMPILR, which proceeds to compile the program. PCL causes CMPILR to display some diagnostic status messages, so you see something like this:

```
PRUN CMPILR
AlphaPascal V2.0
Source file name? COMPAR
Diagnostic file name (<return> for terminal?
AlphaPascal Compiler Version 2.0
           <  0>---
PROGRAM    <  3>---------
12 lines
4.82 seconds,  149.48 lines/minute
No compilation errors.
```

   For more information on PLINK and what each line of the display means, see the AlphaPASCAL User's Guide, (DWM-00100-08).

3. PCU now invokes PLINK, and passes the filename SALES to PLINK as its code file. As PLINK begins, you see a display something like this:

```
.PRUN PLINK
AlphaPascal V2.0
Code file = SALES
Do you wish to 1) replace or 2) update SALES.PCF? 2
```

   PCU automatically supplies the 2, telling PLINK to update SALES.PCF.

4. PLINK continues:

```
The standard library code file for SALES.PCF is STDLIB.PCF
Do you wish to change this? N
```

   PCU automatically supplies the N, telling PLINK not to change the standard library code file for SALES.PCF.

5. Now PLINK finishes:

```
Please specify files to be linked into SALES,
  one per line, ending in a blank line

   File 1 = COMPAR
   File 2 =
Loading program and library dictionaries
Keeping global func/proc SALES
Processing COMPAR
  Linking in global func/proc MAX
Transferring temporary file to new code file
SALES completed
:
```

For more information on PLINK and what each line of the display means, see the AlphaPASCAL User's Guide, (DWM-00100-08).

ERRORS:

PCU generates no error messages itself. However, you may see the standard PASCAL error messages. For a list of those error messages, see Appendix C of the AlphaPASCAL User's Guide, (DWM-00100-08)

CHARACTERISTICS:

Updates a .PCF file by causing PLINK to link an updated module to the .PCF file.

Invokes the PASCAL compiler, CMPILR.

Causes one PASCAL module to be compiled.

Causes CMPILR to send diagnostic information to your terminal display.

Invokes the PASCAL linker, PLINK.

pdlfmt

FUNCTION:

Transforms a text file into a program-design document.

HINTS/RESTRICTIONS:

The purpose of PDLFMT is to transform a program design into a formatted design document.

PDLFMT produces a final document that contains: 1. table of contents; 2. formatted design listing; 3. reference trees; and, 4. a cross-reference.

Create a text file using one of the system text editors. Insert the PDLFMT commands in that file. After exiting the text editor, you can invoke PDLFMT to format your file.

NOTE: A demonstration file, TLGRAM.PDL, is in account DSK0:[1,4]. To use it, enter:

```
.PRINT TLGRAM.PDL ↵
.PDLFMT TLGRAM ↵
.PRINT TLGRAM ↵
```

PDLFMT transforms TLGRAM.PDL into TLGRAM.LST. For information on using PDLFMT, refer to the document Program Design Language Formatting System, (DWM-00100-26), in the AM-100 documentation packet.

FORMAT:

```
.PDLFMT Filespec ↵
```

where Filespec selects the file you want to format.

DEFAULTS:

PDLFMT assumes a file extension of .PDL.

OPERATION:

1. Use one of the text editors to create a text file that contains your program design. Insert into that file the PDLFMT commands that mark the various document components. (See below for a list of the PDLFMT commands.)

2. Type PDLFMT followed by the specification of the file you want to format. Then type a RETURN. For example:

.PDLFMT DSPEC.TXT ↵

PDLFMT assumes an extension of .PDL.

3. PDLFMT now creates a .LST file with the same name as the specification you supplied on the PDLFMT command line. This .LST file contains the formatted version of your text file.

4. PRINT or TYPE the .LST file to see your formatted design document.

COMMAND SUMMARY:

Below are the PDLFMT commands you may insert in your text file. All commands must begin in the first character-position on the line and must start with a slash, /. No line of text that is not a command may begin with a slash. All PDLFMT commands except /R take a text argument.

/T text — TITLE - Specifies the name of the program design. PDLFMT writes this title at the top of every page of your .LST file. The title command must always be the first command in the file and must always be present.

/S text — SECTION NAME - Specifies the start of a new section of procedure designs. PDLFMT uses this name as a subtitle for every page in that section. After this command, enter text that describes the section.

/P text — PROCEDURE NAME - Specifies the start of a procedure design and assigns Procedure Name as the name of that design. Any time the Procedure Name occurs as a statement within a procedure design, PDLFMT considers that occurrence a reference to the procedure design.

/R — REFERENCE TREE - Specifies the start of a list of procedure names on successive lines. Each procedure name is a root of a reference tree listing. The /R command is optional, but if it appears in your file, it must be the last command in the design.

ERRORS:

X IS AN ILLEGAL COMMAND - BYPASSING LINE

PDLFMT found a command /X in the design that it does not recognize. The only legal commands are: /T, /S, /P, or /R.

REPEATED DEFINITION: xxxxxx, REFERENCES WILL BE TO LAST OCCURRENCE

xxxxxx is a section or procedure name that occurred in a section or procedure command more than once.

CHARACTERISTICS:

Assumes a file extension of .PDL.

Transforms a file that you have previously created using one of the text editors. Produces a .LST file.

Returns your terminal to AMOS command level.

perlod

FUNCTION:

Bootstrap loader program for a system that uses the Persci floppy disk as the System Device.

HINTS/RESTRICTIONS:

The PERLOD program when contained on a 2716 PROM allows the system to boot off a System Disk on a Persci floppy disk when a hardware reset occurs (that is, when you hit the RESET button).

The program is also in account DSK0:[1,4] of the System Disk.

You may use PERLOD at AMOS command level to reset the system if your System Device is a Persci floppy disk drive. The memory partition of the job that uses the PERLOD command MUST be in Bank Zero if your system bank switches memory. (For information on bank-switched systems, refer to the document Memory Management Option, (DWM-00100-10) in the AM-100 documentation packet.)

You may use PERLOD to boot either from an AMS- or STD-format diskette.

FORMAT:

```
.PERLOD ↵
```

OPERATION:

1. Type PERLOD followed by a RETURN:

   ```
   .PERLOD ↵
   ```

   The system now resets itself by reading a copy of the PERLOD bootstrap program into system memory and executing it.

2. Once invoked, the PERLOD program reads the operating system skeleton monitor, DSK0:SYSTEM.MON[1,4], into memory. SYSTEM.MON then brings up the system under the control of your system initialization command file, SYSTEM.INI.

3. Once the system is up and running, you see the AMOS prompt.

ERRORS:

PERLOD generates no error messages. However, if it cannot find SYSTEM.MON[1,4] and SYSTEM.INI[1,4], the start-up procedure fails.

(1 October 1979)

CHARACTERISTICS:

Boots the system from a Persci floppy disk if the Persci disk drive is the System Device.

Boots either from an AMS- or STD-format diskette.

Returns your terminal to AMOS command level if the system resets successfully.

FUNCTION:

Links a PASCAL program by invoking the AlphaPASCAL linker, PLINK.

HINTS/RESTRICTIONS:

PL is a command file which makes the AlphaPASCAL linker, PLINK, easier to use in many applications. If a .PCF version of the file you specify exists, PL erases it. Then PL invokes PLINK, giving it the filename you specify as its code file. PLINK then proceeds to link together the files with the code file name and the extensions .P01, .P02, and P03.

You may only enter the filename of the PASCAL file you wish to link. PLINK assumes the device and account you are logged into, and the extension .PCF.

For more information on linking program files, refer to the AlphaPASCAL User's Manual (DWM-00100-07).

DEFAULTS:

If you omit a device specification, PL assumes the device you are logged into.

FORMAT:

.PL Filename (RET)

OPERATION:

1. Type PL followed by the name of the file you want to link. Then type RETURN. For example:

   .PL SALES (RET)

   The file you specify must be a PASCAL program that has been previously compiled; that is, SALES.P01, SALES.P02 and SALES.P03 must exist.

2. PL erases the file SALES.PCF if it exists in the account. You see something like:

   ERASE SALES.PCF
   Total of 1 file deleted, 2 disk blocks freed

3. PL now invokes PLINK, and passes the filename SALES to PLINK. PLINK proceeds to link the files SALES.P01, SALES.P02 and SALES.P03

together into a new file SALES.PCF. As PLINK proceeds, you see a display something like this:

```
.PRUN PLINK
AlphaPascal V2.0
Code file = SALES
Creating new code file SALES.PCF
Library code file for SALES.PCF = STDLIB


Please specify files to be linked into SALES,
 one per line, ending in a blank line

  File 1 = SALES
  File 2 =
Loading program and library dictionaries
Processing SALES
  Linking in global func/proc PROGRAM
Transferring temporary file to new code file
SALES completed
.
```

For more information on PLINK and what each line of the display means, see the AlphaPASCAL User's Guide, (DWM-00100-08).

ERRORS:

PL generates no error messages itself. However, you may see the standard PASCAL error messages. For a list of those error messages, see Appendix C of the AlphaPASCAL User's Guide, (DWM-00100-08)

CHARACTERISTICS:

Links the specified compiled file into an executable program.

Erases the file with the name you specify and the extension .PCF.

Invokes the PASCAL linker, PLINK.

FUNCTION:

PPN displays a list of all of the project-programmer numbers associated with the user accounts on a specific logical unit.

HINTS/RESTRICTIONS:

The device that holds the Master File Directory you want to display must be mounted.

FORMAT:

.PPN Devn: ↵

where Devn: selects the device that holds the accounts whose PPNs you want to see.

OPERATION:

1. Type PPN followed by a legal device name; type a RETURN. For example:

   .PPN DDS1: ↵

2. The PPN command displays the project-programmer numbers asscciated with all of the accounts on that disk. PPN groups them by project number (i.e., all of the PPNs beginning with the same number are displayed as a group).

ERRORS:

You may encounter one of the standard system error messages if you supply an invalid device specification. For example:

?File specification error

PPN did not understand the device specification you supplied. For example, you see this error message if you enter PPN followed by a RETURN.

?Cannot READ Devn: - device does not exist

Check your spelling. The system did not recognize the device specification you supplied. For example, did you enter ASM1: instead of AMS1:?

CHARACTERISTICS:

Returns your terminal to AMOS command level.

(1 October 1979)

print

FUNCTION:

Allows you to send one or more files to a orinter.

HINTS/RESTRICTIONS:

The PRINT command sends a print request to the line printer spooler program; the spooler places your request into a queue (or waiting line). When a printer becomes available, the spooler prints your file. You may have several requests in the orinter queue waiting for available printers.

PRINT is a wildcard file command. For more information on using PRINT, refer to Chapter 9.6, "Printing a File (PRINT)," in the AMOS User's Guide, (DWM-00100-35). PRINT recognizes ersatz devices and wildcard symbols.

Use the SET command to define the type of form to be mounted on a specific printer on the system. (See the SET reference sheet for information on setting forms.)

FORMAT:

.PRINT {Printerspec=} RET

to see if any print requests are currently in the queue of a specific printer or in the queues of all printers.

.PRINT Filespec1{,Filespec2...,FilespecN}{/Switch{/Switch}} RET

to print one or more files on the printer that has the least number of blocks waiting in the queue to be printed (or on the default printer set by the System Operator). A Switch is an option request.

.PRINT Printerspec=Filespec1{,Filespec2...,FilespecN}{/Switch{/Switch}} RET

where Printerspec specifies the particular printer on which you wish to print the file(s) selected by Filespec.

DEFAULTS:

The default Printerspec is the printer with the least number of blocks waiting in the queue to be printed (unless the System Operator has used the DEFAULT command in the spooler parameter file to define another printer default).

(Changed 30 April 1981)

The default switch settings depend upon the specific printer being used. (The defaults for each printer are set by the System Operator.)

The default Filespec is a null name and .LST extension. The initial default device and account is the account and device you are currently logged into.

OPTIONS:

You may select one or more of the options below by including the appropriate switches on the PRINT command line:

/COPIES:n or /C:n	Number of copies you want to print. (A file switch.)
/DELETE or /D	Delete the file after it is printed. (A file switch.)
/NODELETE or /NOD	Turn off the /D switch. (A file switch.)
/BANNER or /B	Print a banner (identifying) page at the front of the listing. See the AMOS User's Guide for a full description of a banner page. (An operation switch.)
/NOBANNER or /NOB	Don't print a banner page. (An operation switch.)
/HEADER or /H	Print a page header at the top of every page of the listing. Page headers give the name of the file being printed, the date, and the current page number. (A file switch.)
/NOHEADER or /NOH	Don't print page headers. (A file switch.)
/FORMFEED or /FF	Print a final form feed at the end of each listing. (A file switch.)
/NOFORMFEED or /NOF	Don't print a final form feed at the end of each listing. (A file switch.)
/LPP:n	Maximum number of lines to print on each page. If /HEADER is set, PRINT prints a form feed when it outputs a full page and then it prints a page header. (A file switch.)
/WIDTH:n or /WI:n	The page width (in characters). PRINT uses this value in printing page headers. /WIDTH value must be between 80 and 132, inclusive. (A file switch.)

(Changed 30 April 1981)

/WAIT or /WA — If you try to enter more print requests into a printer queue than PRINT can handle, PRINT discards the extra print requests. The /WAIT option tells PRINT not to discard the extra print requests, but to reinsert them into the queue as PRINT finishes processing earlier requests. This option ties up your terminal while it waits for room to be made in the queue. (An operation switch.)

/FORMS:x or /FO:x — Tells PRINT the form on which you want to print your file. If that is not the form defined as mounted on the printer, then PRINT asks you to mount the proper form. "x" may be any one to six character name you choose. You can define the type of form to be mounted on a specific orinter by using the SET command. (The System Operator sets the initial form-type for each printer during system initialization.) (A file switch.)

/KILL or /K — Delete the specified file from the printer queue. /K is an operation switch; it affects all filespecs on the command line. You may not wildcard the specifications of files to be deleted from the queue. When you use /K, PRINT tells you which files were deleted from the queue.

OPERATION:

1. To find out if any print requests are waiting in the printer queues, type PRINT and a RETURN:

```
.PRINT (RET)
```

This command lists the contents of the queues for all of the printers defined on the system. If you wish to see the contents of the queue for a single printer, type PRINT followed by the name of the printer whose queue you want to see, an equal sign, and a RETURN. For example:

```
.PRINT MULTI= (RET)
```

The display you see tells you what files are waiting to be printed, what form-type they are to be printed on, how many blocks total are in the queue, how many blocks remain for each file, how many copies of each file are to be printed, and which file is currently being printed.

(Changed 30 April 1981)

If there are no print requests in the queue, you see: "The queue is empty."

2. To send files to the printer that has the least number of blocks in its queue (or to the default printer set by the System Operator), simply type PRINT followed by the specifications of the files you want to print:

```
.PRINT *.LST (RET)
HEADER.LST
MNPAK.LST
DSKCLR.LST
Total of 3 files (134 blocks) in printer request
```

3. If you wish to send files to a specific printer, type PRINT followed by the name of the printer you want to use, an equal sign, and your list of filespecs:

```
.PRINT QUME=BACKUP.*,S?CFIL (RET)
BACKUP.MAC
BACKUP.TXT
SRCFIL.LST
S2CFIL.LST
Total of 4 files (78 blocks) in printer request
```

ERRORS:

You may see any of the standard system error messages that result from invalid device and account specifications. In addition, you may encounter these messages:

;LPTSPL - Please mount form Form-name on Printer-name

The Spooler Operator job sees this message if you specify a form-type that is not defined as being mounted on the printer (e.g., FORMS has been set to NORMAL, but you type PRINT TI810=TREE.MAC/FO:CHECKS).

```
;LPTSPL - Please mount form CHECKS on TI810
```

The Spooler Operator job makes sure that the form CHECKS is indeed on the printer; then uses the SET command to inform PRINT that the forms have been changed:

```
.SET FORMS TI810 CHECKS (RET)
```

?Output printer not found

You specified a printer that PRINT does not recognize. Check your spelling; then ask the System Operator for a list of the valid printers set up for the line orinter spooler.

(Changed 30 April 1981)

If you have more than one printer defined for use with the spooler program, you can see a list of all of the printers by typing:

.PRINT RET

%No files in print request

PRINT was not able to find the files you specified. Check the command line again, and make sure that the device and account specifications are correct.

?Invalid argument for COPIES

/COPIES only understands numbers; make sure that you did not inadvertently type a letter.

?Invalid argument for LPP

You gave a non-numeric argument to the lines per page switch or the format of your command was incorrect.

?Invalid argument for WIDTH

You gave a non-numeric argument to the page width switch or the format of your command was incorrect.

CHARACTERISTICS:

PRINT is a wildcard file command. It understands ersatz devices.

Allows you to specify specific printer, form-type, number of copies, banner page, page header, form feed handling, lines per page, page width, and whether to delete the file after it is printed.

prun

FUNCTION:

Executes compiled AlphaPASCAL programs.

HINTS/RESTRICTIONS:

PRUN is the runtime portion of the AlphaPASCAL language processor. It executes your program by interpreting the .PCF file created by the AlphaPASCAL linker. Use PRUN only on programs that have previously been compiled via the CMPILR program and then linked via the linker, PLINK. (See the PC, PCL, PL, PU, and PCU reference sheets for information on command files that help you to easily compile and link PASCAL programs.)

PRUN.PRG is re-entrant; the System Operator may load it into system memory.

PRUN will not execute a program that is not compatible with the current external library. That is, you may not run a program that was linked with a certain external library with an older version of that library or with a completely different library; instead, you must re-link the program with the current library.

To interrupt program execution, type a Control-C. (For more information on interrupting program execution, see "OPERATIONS," below.)

NOTE: To compile a PASCAL program (a .PAS file), enter:

.PRUN CMPILR RET

Now the compiler, CMPILR, asks you a series of questions concerning your program file.

For full information on compiling and linking programs, see Chapter 4 of the AlphaPASCAL User's Manual (DWM-00100-08).

FORMAT:

.PRUN filespec RET

where filespec selects the .PCF file you want to run.

DEFAULTS:

PRUN uses the default extension .PCF. If you do not specify a device and account, PRUN looks for the specified file in the device and account you are logged into; next it looks in your project library account, [P,0]. Finally, it searches for the file in the System PASCAL Library account, DSK0:[7,5].

OPERATION:

1. To execute a compiled and linked PASCAL program, enter PRUN followed by the specification of the program file you want to execute; then type a RETURN. For example:

```
.PRUN SALARY (RET)
```

Now PRUN displays this message:

```
AlphaPascal V2.0
```

and then executes your program.

2. If you wish to interrupt execution of your program, type a Control-C. You now see:

```
Interrupt  (?=Help):
```

Now you may enter one of four codes: ?, Q, R, or B. Then type a RETURN. For example:

```
Interrupt  (?=Help): ? (RET)

   Q = Quit
   B = Backtrace
   R = Resume

Interrupt  (?=Help): Q (RET)
.
```

A ? tells PRUN to display a list of the codes you may enter. If you enter Q, PRUN returns you to AMOS command level; an R causes PRUN to resume program execution, and a B tells PRUN to "backtrace" your program. (To backtrace a program means to list in the order called all procedures and functions invoked during program execution up to the point of interruption, with the last-called procedure or function listed first. For more information on backtracing, see Chapter 4 of the AlphaPASCAL User's Manual.)

ERRORS:

You may see several error messages when using PRUN. The most serious of these messages has to do with library version checking:

?Wrong version of xxxx for use with yyyy

where xxxx is the external library and yyyy is the .PCF file you want to execute. A program that was linked with one external library cannot be run with an older version of that library or with a completely different library; instead, you must re-link the program with the current external library you want to use.

Error messages may also take the form:

Message in Function-or-procedure at IPC = xxx within Filespec

where "Message" indicates the error that occurred, "Function-or-procedure" indicates the function or procedure in which the error occurred, "IPC = xxx" indicates the Interpreter Program Counter number (i.e., the place in the program) at which the error occurred. Lastly, PRUN gives the "Filespec," the file specification of the program you were executing. For example:

?SQRT(x) where x < 0 in PROGRAM at IPC = 64 within TEST.PCF

For a list of the possible error messages, see the section titled "Error Handling Procedures and Variables," in Chapter 14 of the AlphaPASCAL User's Manual.

You may also see standard AMOS monitor messages. For example:

?Cannot OPEN filespec - file not found

PRUN could not find the program file you specified. Make sure that you are logged into the proper account and device. Check the section titled "DEFAULTS" above for information on the default device and account specifications used by PRUN.

?Cannot OPEN filespec - device does not exist

You included a device specification in your file specification, but that device does not exist. Make sure that you have correctly specified the device (e.g., make sure that you did not type "DKS1:" instead of "DSK1:"). If you are sure you are specifying the device correctly, check with the System Operator to see why the device is not available.

CHARACTERISTICS:

PRUN executes compiled and linked PASCAL programs.

Assumes the default extension .PCF. If you omit an account and device specification from your file specification, PRUN searches in: the account and device you are logged into; your project library account; and, the PASCAL Library account, DSK0:[7,5].

PRUN is re-entrant and serially re-usable.

pu

FUNCTION:

Updates a PASCAL program by linking a specific module into the program file.

HINTS/RESTRICTIONS:

Invokes the AlphaPASCAL linker, PLINK, to link a module into a program.

PU is a command file which makes the AlphaPASCAL linker, PLINK, easier to use in many applications when linking a single module into a program.

You supply PU with the name of the module you want to update, followed by the name of the .PCF file you want to link the module into.

PU invokes PLINK, giving it the name of the .PCF file you specify as its code file. PU then tells PLINK to update the .PCF file.

PU tells PLINK not to change the standard library code file for the .PCF file while updating it.

For more information on updating a PASCAL program, refer to the AlphaPASCAL User's Manual (DWM-000100-07).

FORMAT:

.PU Moculespec Filespec (RET)

where Modulespec specifies the module you want to update, and Filespec specifies the .PCF file you want to link the module into.

DEFAULTS:

If you omit the file extension, PU assumes the extension .PCF. If you omit the device and account specifications, PU assumes the device and account you are logged into.

OPERATION:

1. Type PU followed by the name of the module you want to update, followed by the name of the .PCF file you want to link the module to. Then type RETURN. For example:

```
.PU COMPAR SALES (RET)
```

The module name you specify must be a PASCAL module that has been previously compiled; that is, COMPAR.P01, COMPAR.P02, and COMPAR.P03 must already exist.

2. PU now invokes PLINK, and passes the filename SALES to PLINK as its code file. As PLINK begins, you see a display something like this:

```
.PRUN PLINK
AlphaPascal V2.0
Code file = SALES
Do you wish to 1) replace or 2) update SALES.PCF? 2
```

PU automatically supplies the 2, telling PLINK to update SALES.PCF.

3. PLINK continues:

```
The standard library code file for SALES.PCF is STDLIB.PCF
Do you wish to change this? N
```

PU automatically supplies the N, telling PLINK not to change the standard library code file for SALES.PCF.

4. Now PLINK finishes:

```
Please specify files to be linked into SALES,
  one per line, ending in a blank line

  File 1 = COMPAR
  File 2 =
Loading program and library dictionaries
Keeping global func/proc SALES
Processing COMPAR
  Linking in global func/proc MAX
Transferring temporary file to new code file
SALES completed
.
```

For more information on PLINK and what each line of the display means, see the AlphaPASCAL User's Guide, (DWM-00100-08).

ERRORS:

PU generates no error messages itself. However, you may see the standard PASCAL error messages. For a list of those error messages, see Appendix C of the AlphaPASCAL User's Guide, (DWM-00100-08)

CHARACTERISTICS:

Updates a PASCAL program by linking a specific module into the program file.

Invokes the PASCAL linker, PLINK.

Updates a .PCF file by causing PLINK to link a module to the PCF. file.

FUNCTION:

Allows you to examine and modify locations in memory.

HINTS/RESTRICTIONS:

If your system bank switches memory, you may only examine and change locations in your own memory bank and in system memory.

QDT understands only upper case letters. All numeric input must ALWAYS be in octal, even if the system is using hexadecimal for your numeric displays.

NOTE: This is a very dangerous program to use, because you can easily destroy the monitor in system memory. (Of course, you can reload a valid copy by resetting the system.)

The primary use for QDT is not to debug programs, but to examine the contents of memory locations in the monitor or at the locations used by the I/O ports. (I/O ports start at 177400 in memory and proceed up to 177777.)

You can use QDT in much the same way that you would use DDT's absolute mode, but QDT is much smaller than DDT and allows you to examine and change memory locations anywhere in memory.

FORMAT:

```
.QDT ↵
```

OPERATION:

1. Type QDT followed by a RETURN:

```
.QDT ↵
```

2. Now you can begin to enter QDT commands. Unless you specify a memory address, QDT assumes that the first memory location you want to display is at address zero.

3. To exit QDT, use the E command:

```
E
.
```

QDT now returns you to AMOS command level.

(1 October 1979)

COMMAND SUMMARY:

B	Change to byte mode so that you can modify single bytes. The current location counter changes by one.
W	Change to word mode so that you can change full words. The current location counter changes by two.
E	Exit from QDT.
#	Display the current word as two octal bytes.
TAB	Takes the contents of the current location and uses those contents as the address of the next location to be displayed.
RUBOUT	Cancels current command or number. (Displays an XXX symbol to tell you that the command line is being ignored.)
n/	Changes current location to the one specified by the octal number that precedes the slash. No change if you do not include a number.
nRETURN	Stores the specified octal number in the current location.
nLineFeed	Works like a RETURN, but increments current location counter.
^	Works like Line-feed, but decrements current location counter.

ERRORS:

If QDT does not understand a command, it echoes it back to you with a question mark. For example:

```
Y
Y?
```

CHARACTERISTICS:

Useful for testing devices on the S-100 bus, by allowing you to examine and modify the memory locations used by the I/O ports.

Returns your terminal to AMOS command level.

FUNCTION:

Use within the system initialization command file to increase the number of blocks allocated to the monitor queue. Use at AMOS command level to find out how many monitor queue blocks are available for use.

HINTS/RESTRICTIONS:

Various portions of the system (e.g., the line printer spooler) use the monitor queue system. The monitor queue is also available for use by user programs. The initial size of the queue is 20 eight-word blocks. The number of queue blocks you need depends on the tasks you need to perform.

Use the QUEUE command in the system initialization command file to increase the size of the monitor queue. (Remember, however, that increasing the number of queue blocks increases the amount of memory used by the monitor.)

To find out how many queue blocks are available, use the QUEUE command at AMOS command level.

FORMAT:

.QUEUE ↵

or:

QUEUE n

OPERATION:

1. To find out how many queue blocks are available: at AMOS command level type QUEUE followed by a RETURN:

```
.QUEUE ↵
97 Queue blocks available
```

2. To increase the number of blocks in the queue, use one of the system text editors to edit DSKO:SYSTEM.INI[1,4]. Before any SYSTEM commands, add one QUEUE command followed by the number of blocks that you want to add. For example:

```
QUEUE 15
```

The example above adds 15 eight-word blocks to the 20 blocks initially assigned to the queue; this generates a queue of 35 blocks.

ERRORS:

[No queue blocks allocated]
No queue blocks are currently available for use.

CHARACTERISTICS:

QUEUE is both a user command and a system initialization command.

QUEUE returns your terminal to AMOS command level when you use it as a user command.

FUNCTION:

Performs a random write/read test on disk media.

HINTS/RESTRICTIONS:

NOTE: RAZA destroys the data on the disk you are testing.

RAZA tests your disk by randomly writing and reading data on it; therefore do not test a disk that contains data you want to keep! If you tell RAZA to test DSK0:, it destroys your System Disk and you cannot boot the system from that disk again until you copy new system software over to it from another disk.

FORMAT:

.RAZA ↵

OPERATION:

1. Type RAZA followed by a RETURN:

```
.RAZA ↵
```

2. RAZA asks you for the specification of the logical unit you want to test:

```
.RAZA ↵
DEVICE NAME (DSK,HWK,ETC): DSK↵
DRIVE NUMBERS TO RAZA (0,2,3,ETC): 2↵
```

In the example above, we've asked RAZA to test the disk in DSK2:. To end the test, type a Control-C.

ERRORS:

VERIFICATION ERROR FOR DRIVE n RECORD x

If RAZA finds that any of the data it has written does not verify, it tells you which drive (e.g., DRIVE 2) holds the bad disk, and tells you which record of the disk contains the error (e.g., RECORD 1000).

CHARACTERISTICS:

Destroys any data already on the device. Continues until you type a Control-C.

(1 October 1979)

redall

FUNCTION:

Performs a disk diagnostic test by reading all (or a specified number) of the records on a disk and reporting any read errors.

HINTS/RESTRICTIONS:

REDALL does not harm the data on your disk.

FORMAT:

```
.REDALL Devn:N ↵
```

where Devn: specifies the disk you want to read and N gives the number of disk records you want to read. If you omit N, REDALL reads all records on the disk.

OPERATION:

1. Type REDALL followed by the specification of the device you want to test and the number of records you want to read. DO NOT separate the device specification and N with a space. Type a RETURN. For example:

```
.REDALL DSK1:100 ↵
```

2. REDALL now tells you the number of records it is reading. If you omit a number after the device specification, REDALL reads all records on the disk. After it has finished, it displays an EXIT message and returns to the monitor level.

```
.REDALL DSK1: ↵
READING 9696 RECORDS
EXIT
.
```

3. REDALL tells you if any read errors occurred by displaying the appropriate error messages on the screen.

ERRORS:

You may see the standard system error messages that result from invalid device specifications. For example:

?Cannot READ Devn: - device does not exist

You specified a device that REDALL does not recognize. Check your spelling. Then use the SYSTAT command to see a list of valid, mounted devices on the system.

(1 October 1979)

?Cannot READ Devn: - disk not mounted

The device you specified is a valid device, but it is not presently mounted. Use the MOUNT command.

CHARACTERISTICS:

Performs a disk-read diagnostic test that does not harm the data on your disk.

Returns your terminal to AMOS command level.

rename

FUNCTION:

Renames one or more files that exist within the same account.

HINTS/RESTRICTIONS:

The new file specification you assign to a file cannot contain a PPN or a device specification. You can only rename files within an account.

RENAME recognizes special devices. For example:

```
.RENAME Newfilespec=MEM:Oldfilespec ↵
```

renames memory modules. (MEM: must be defined in your system device table.) RENAME also recognizes wild card symbols and ersatz devices. For example:

```
.RENAME *.OLD=BAS:*.SBR ↵
```

renames each file in the BASIC Library Account, DSK0:[7,6], that has a .SBR file extension to a .OLD extension.

You may use the standard wildcard symbols in your file specifications. If you use wildcard symbols in your Newfilespec, the wildcard symbols are replaced by the appropriate elements specified by the Oldfilespec(s). For example:

```
.RENAME *.OLD=*.BAS ↵
```

renames the files TEST.BAS to TEST.OLD, DIGIT.BAS to DIGIT.OLD, and UNI.BAS to UNI.OLD. RENAME is a wildcard file command. For more information on the use of RENAME, refer to Section 9.3, "Renaming Files (RENAME)," in the AMOS User's Guide, (DWM-00100-35).

FORMAT:

```
.RENAME Newfilespec=Oldfilespec1{,...OldfilespecN}{/Switch1{/Switch2}} ↵
```

where Oldfilespec1-OldfilespecN selects the one or more files that you want to rename and Newfilespec determines their new names. A /Switch is an option request (see OPTIONS).

DEFAULTS:

RENAME uses the device and PPN you are currently logged into as file specification defaults. Default switches are /NODELETE and /NOQUERY. Defaults for unspecified filenames and extensions are *.*.

(1 October 1979)

OPTIONS:

You may select the following options by including the appropriate option code in your command line:

/QUERY	or /Q	Request confirmation to rename.
/NOQUERY	or /NOQ	Rename without asking for confirmation.
/DELETE	or /D	If file with new name already exists, delete it before performing renaming.
/NODELETE	or /NOD	If new name is in use, don't perform renaming.

OPERATION:

1. Type RENAME followed by a Newfilespec, an equal sign, and one or more Oldfilespecs. Type a RETURN. For example:

```
.RENAME TTY??.*=DVR??.* ↵
DVR12.PRG to TTY12.PRG
DVRAA.TXT to TTYAA.TXT
DVR67.MAC to TTY67.MAC
Total of 3 files renamed
```

As RENAME renames the files, it tells you both the old and the new names of the files. At the end of the process, it tells you how many files were renamed.

2. If you use the /QUERY option, RENAME asks you to confirm the renamings. (To request confirmation of each renaming, place the /Q option directly after the RENAME command; to request confirmation of the renaming of a particular file, place the /Q option after the specification of that file.)

```
.RENAME/Q BASTXT.*=WRKFIL.*,WRKTXT.* ↵
WRKFIL.BAS to BASTXT.BAS?Y
WRKFIL.RUN to BASTXT.RUN?Y
WRKTXT.TXT to BASTXT.TXT?N
WRKTXT.LST to BASTXT.LST?Y
Total of 3 files renamed

.RENAME *.FTX=*.TXT,*.LST/Q ↵
REPORT.TXT to REPORT.FTX
TABLE.TXT to REPORT.FTX
REPLST.LST to REPLST.FTX?Y
TABLST.LST to TABLST.FTX?Y
Total of 4 files renamed
```

Answer the question mark with a Y for YES, or an N for NO; do not type a RETURN after your answer. You may type a Control-C to stop further renamings and return to AMOS command level.

ERRORS:

?Cannot find DSK0:SCNWLD.SYS[1,4] or MEM:SCNWLD.SYS

RENAME needs this file to process wildcard symbols in file specifications. The system can't find the file or you do not have enough room in your memory partition to load the file.

?Specification error ^

RENAME doesn't understand the format of your command line; the ^ symbol points to the location of the item that confused it.

?Cannot READ Devn: - device does not exist
?Cannot READ Devn: - disk not mounted

The device specified by Devn: is not present in the DEVTBL command of the SYSTEM.INI, does not have a driver in area [1,6] of the System Disk, is not file-structured, or is not mounted.

%No file-oriented device corresponding to Devn: is mounted

You specified a device, but did not include a unit number. RENAME is not able to find a logical unit that matches your specification. Check your spelling.

%Account does not exist - [p,pn]

The indicated PPN does not exist; to create it you must use SYSACT.

?More than one output specification

You can't rename a file to two names; use only one Newfilespec.

?Device or [p,pn] specifications on output are illegal

Your Oldfilespecs and Newfilespec must contain the same PPN and device specifications.

?Missing output specification

You must give an output specification so that RENAME knows what to use as the new names for your files.

%No files renamed

RENAME couldn't find any files that matched your Oldfilespecs or wasn't able to carry out the renaming procedure because of an error.

?Cannot RENAME Oldfilespec - file already exists

You tried to rename a file, but a file of that name already exists. If you want RENAME to delete existing files when it renames, use the /D option.

CHARACTERISTICS:

You can only rename files within the same account.

RENAME is a wildcard file command. It recognizes special devices, ersatz devices, and wildcards.

The default switches are /NOQUERY/NODELETE. The default file specifications are *.* and the account and device you are logged into.

Returns your terminal to AMOS command level.

FUNCTION:

Loads 18 frequently used system programs into your memory partition from the System Disk.

HINTS/RESTRICTIONS:

RES.CMD is a command file that loads into your memory partition these programs from account DSK0:[1,4]:

LOG, QDT, SYSTAT, SYSACT, DSKDMP, DSKFIL, DSKANA, DSKCPY, MAP, DIR, SCNWLD.SYS, DEL, REDALL, COPY, RENAME, ERASE, TYPE, MOUNT.

Users with floppy-disk based systems will use RES.CMD to load important programs into memory from their System Disk so that they can remove the System Disk and still have the use of their system software.

NOTE: You can use one of the system text editors to create your own .CMD file to load into memory the specific programs you need.

FORMAT:

.RES ↵

OPERATION:

1. Type RES followed by a RETURN:

   .RES ↵

2. The RES command file uses the LOAD command to load frequently used programs into your memory partition.

ERRORS:

For each program that RES tries to load into memory which does not exist, you see this error message:

Filespec NOT FOUND

For example:

DSK0:QDT.PRG[1,4] NOT FOUND

RES then goes on to load the next program.

CHARACTERISTICS:

RES is a command file in the system Command File Library account, DSK0:[2,2].

Returns your terminal to AMOS command level.

revive

FUNCTION:

Allows you to wake up a job that has been put to sleep by the SUSPND command.

FORMAT:

.REVIVE Jobname ↵

where Jobname specifies the job you wish to return to active status.

OPERATION:

1. Type REVIVE followed by the name of the job you want to awaken; then hit RETURN. For example:

.REVIVE JOB7 ↵

ERRORS:

[NONEXISTENT JOB]
You entered an invalid job specification. (For example you typed REVIVE followed by a RETURN or you misspelled the job name.) Check your spelling; if it is correct, use the SYSTAT command to see a list of valid jobs on the system.

CHARACTERISTICS:

Returns your terminal to AMOS command level.

rewind

FUNCTION:

Rewinds a magnetic tape unit operating under the control of the AM-600.

HINTS/RESTRICTIONS:

IMPORTANT NOTE: DO NOT use REWIND when another job is using the tape unit. Such an action causes unpredictable results.

You may specify one tape drive with a unit number of 0 to 7. (e.g., MTU0: or MTU5:).

The magnetic tape unit you access must be defined in the system device table. (See the DEVTBL reference sheet.) The program MTSTAT.SYS must be in system memory. (Use the SYSTEM command within the system initialization command file to include MTSTAT.SYS in system memory.)

NOTE: A magnetic tape is said to be at load point when the metallic film at the start of the tape is positioned at the read head.

FORMAT:

.REWIND Devn: ↵

where Devn: is the magnetic tape unit you want to rewind.

OPERATION:

1. Make sure the tape you want to rewind is mounted.

2. Type REWIND followed by the specification of the magnetic tape unit you want to rewind. Then type RETURN. For example:

   .REWIND MTU6: ↵

3. You now see one of the following REWIND messages:

   a. Tape is rewinding now

      This message is the most common and indicates that the tape is in the process of rewinding. The rewind will be finished in a moment.

   b. Tape is already rewinding

      You are trying to rewind a tape that is already in the process of rewinding. The rewind will be finished in a moment.

c. Tape cannot be rewound - it is at load point.
The tape has already been rewound or has never been moved from its initial position at load point.

REWIND returns you to AMOS command level while the tape is still rewinding.

ERRORS:

You can see the following system error messages:

?File specification error
REWIND did not understand your command line (e.g., you typed REWIND followed by a RETURN). Re-type the line.

?Cannot OPEN Devn: - device does not exist
REWIND did not recognize the magnetic tape unit you specified. Make sure that the device you specified is defined in the system device table. (See the DEVTBL reference sheet.)

You can also see the following REWIND error message:

Tape cannot be rewound - it is at load point
The tape has already been rewound.

CHARACTERISTICS:

Requires the magnetic tape unit you access to be defined in your system device table.

MTSTAT.SYS must be in system memory.

rndred

FUNCTION:

Performs a random-read disk diagnostic test.

HINTS/RESTRICTIONS:

RNDRED randomly selects a disk track and performs a seek and read operation on a random record on that track. It lists any read errors that it finds.

Runs until you type a Control-C or reset the system.

Does not harm the data on the disk.

FORMAT:

.RNDRED Devn: ↵

where Devn: specifies the device you want to test.

OPERATION:

1. Type RNDRED followed by the specification of the logical unit you want to test. Type a RETURN. For example:

   .RNDRED AMS1: ↵

2. RNDRED displays no data unless a read error occurs, in which case it displays the appropriate error message.

ERRORS:

You can see the standard system error messages that result from an invalid device specification. For example:

?Cannot READ Devn: - device not mounted
The device you want to test is not mounted, and RNDRED cannot read data on the disk until you mount it using the MOUNT command.

CHARACTERISTICS:

Does not harm the data on the disk you are testing.

Returns your terminal to AMOS command level when you type a Control-C.

(1 October 1979)

FUNCTION:

Runs compiled BASIC programs.

HINTS/RESTRICTIONS:

Invokes the runtime package portion of the BASIC language processor.

RUN.PRG is reentrant; the System Operator may include it in system memory via the SYSTEM command in the system initialization command file. If RUN is in system memory, each user who wants to use it may then use the copy in system memory instead of having to load it into his own memory partition.

Use RUN only on programs that you have previously compiled. For information on writing BASIC programs, using BASIC in interactive mode (BASIC.PRG), and using the compiler portion of the BASIC language processor (COMPIL.PRG), refer to the AlphaBASIC User's Manual, (DWM-00100-01).

FORMAT:

.RUN Filespec ↵

where Filespec specifies the compiled BASIC program you want to execute.

DEFAULTS:

Assumes a file extension of .RUN.

OPERATION:

1. Type RUN followed by the specification of the file that contains the compiled BASIC program you want to execute. Then type a RETURN. For example:

   .RUN INVOIC.RUN ↵

   (NOTE: Compiled programs usually have the .RUN extension.) The BASIC runtime package now executes your program.

2. To interrupt program execution, type a Control-C.

   .RUN ACCNT.RUN ↵
   Enter new account number: [You type a Control-C here]
   Operator interrupt in ACCNT.RUN
   .

(1 October 1979)

ERRORS:

You may see any of the standard BASIC error messages. Refer to the back of the AlphaBASIC User's Manual for a list of the BASIC messages.

CHARACTERISTICS:

Assumes a file extension of .RUN.

FUNCTION:

Allows you to save memory modules in your memory partition as disk files.

HINTS/RESTRICTIONS:

Memory modules bear the same name and extension as their corresponding files on the disk. (Use the MAP command or DIR MEM: to find out what modules are in your memory partition.) SAVE will not save modules as disk files in accounts or on devices other than the ones into which you are currently logged.

SAVE understands the wildcard symbols ? and *.

FORMAT:

.SAVE Filespec1{,Filespec2,...FilespecN}{/Rename-extension} ↵

where Filespecs select the modules you want to save as disk files.

DEFAULTS:

SAVE assumes a file extension of *. If you specify just a filename, SAVE saves all memory modules of that name regardless of their extensions.

OPERATION:

1. Type SAVE followed by a list of Filespecs identifying the memory modules you want to save. Then type a RETURN:

```
.SAVE *.SYS,*.PRG ↵
SCNWLD.SYS
LOG.PRG
QDT.PRG
.
```

SAVE displays a list of the memory modules being saved as disk files. These files appear in the account into which you are currently logged.

2. If you ask SAVE to transfer a copy of a module to your account and a disk file of that name already exists, SAVE erases the original disk file and replaces it with a copy of the memory module:

```
.SAVE TLC ↵
ERASE TLC.PRG, SAVE TLC.PRG
.
```

In the example above, a file named TLC.PRG already existed in the account; SAVE erased it before saving the memory module TLC.PRG.

3. If you do not want SAVE to erase an existing file, use the rename option. Follow the list of Filespecs with a slash and any extension you choose. Before saving any memory modules, SAVE will rename to the new extension any existing files with the same names as those memory modules. For example, assume that you have a file in your account named EXIT.SBR:

```
.SAVE EXIT.SBR/OLD ↵
RENAME EXIT.SBR,        SAVE EXIT.SBR
```

The example above renames your existing file EXIT.SBR to EXIT.OLD; then it saves the memory module in your partition as file EXIT.SBR. (If a file named EXIT.OLD already exists in the account, the example above erases it before renaming EXIT.SBR to EXIT.OLD.)

ERRORS:

SAVE generates no error messages. SAVE lists all modules that it saves; if you do not see such a list, it means that SAVE wasn't able to save the modules you specified.

CHARACTERISTICS:

Allows you to rename an existing disk file if the module you save has the same specification as that disk file.

Returns your terminal to AMOS command level.

send

FUNCTION:

Allows you to exchange messages between terminals on the same system.

HINTS/RESTRICTIONS:

Terminals which are not connected to the same computer may not exchange messages and a terminal will only receive a message if it is not protected by a "guard" program and if it is at AMOS command level or in terminal input mode. Messages cannot be longer than one line.

Your terminal must be at AMOS command level if you want to use SEND.

FORMAT:

.SEND Jobname Message ↵

where Jobname is the job to whom you want to send a Message.

OPERATION:

1. Type SEND, the name of the job that you want to communicate with, and the message. Type a RETURN. For example, assume that you are JOB1:

   .SEND JOB3 HOW LONG IS THAT LISTING YOU'RE PRINTING? ↵

2. If the terminal attached to JOB3 is not guarded and is at AMOS command level or in terminal input mode, the message appears on JOB3's terminal:

   JOB1 - HOW LONG IS THAT LISTING YOU'RE PRINTING?

   (NOTE: the name of the job that sent the message appears at the front of the message.) Now JOB3 can return an answer by also using the SEND command (e.g., .SEND JOB1 ABOUT 17 PAGES).

3. When SEND is finished, it returns your terminal to AMOS command level.

ERRORS:

You may see the following error messages:

?Job not found
: You tried to send a message to a nonexistent job. Make sure that you spelled the name of the job correctly and that the job does indeed exist on this system. (You can use the SYSTAT command to see what other jobs are on the system.)

?Job specification error
: SEND is confused by the job name that you supplied. For example, you see this error message if you type SEND followed by a RETURN.

?Busy
: SEND cannot send the message to the job you specified because that job is not at AMOS command level or is not in terminal input mode.

?Job has no terminal attached
: The job with which you want to communicate has not been attached to a terminal; there is nothing on which to display your message.

?Guarded
: You are trying to send a message to a job that is protected by a program that guards it from messages.

CHARACTERISTICS:

Requires that the job with whom you want to communicate be on the same system as your terminal, not be guarded, and be in terminal input mode or at AMOS command level.

Returns your terminal to AMOS command level.

FUNCTION:

You can use SET to choose various system terminal-handling options for your terminal or to view the options previously set. You can use SET to specify the type of form that ought to be mounted on a specific system printer. Using SET, you can also set the bits per inch rate for reading or recording a specific magnetic tape unit.

HINTS/RESTRICTIONS:

SET sets various flags in your job table so that each user of the system can choose different terminal-handling options. For example, the default number base the system uses for numeric display is octal. You can use the SET command to change the base that the system uses for your job's numeric displays by using the .SET HEX or .SET OCTAL commands.

You can also use SET to specify the kind of form you want mounted on a printer. The PRINT command then checks to make sure that the form-type specified by the user is the same as the type of form you have defined with SET for that printer. (See the PRINT reference sheet for information on the PRINT/FORMS option.)

You can set the density at which data is read or recorded by a magnetic tape unit to either 800 or 1600 bits per inch (BPI) using the .SET BPI command. This command sets a flag in MTSTAT.SYS indicating the desired density. The density is actually set at the time of the next I/O operation by the magnetic tape unit driver.

NOTE: An error occurs if you try to read a magnetic tape when a SET BPI value is in effect which is different than the BPI at which the tape was recorded.

FORMAT:

.SET RET

to display the current options; or:

.SET Option RET

where Option tells SET which flag to modify in your job table; or:

.SET FORMS Printer-name Form-name RET

where Form-name specifies the kind of form you want to assign to a specific printer (specified by Printer-name); or:

.SET BPI Devn:NNNN (RET)

where Devn: is a device specification that selects a magnetic tape unit (drives 0 through 7). NNNN is the desired tape density of either 800 or 1600 bits per inch.

DEFAULTS:

The system comes up with the following options as the default: OCTAL, ECHO, NOVERIFY, NOGUARD, NODSKERR, CTRLC. The default form-type for a specific printer is set by the printer initialization file for that printer. The initial tape density at the time of system start-up is 1600 BPI.

OPTIONS:

The current options that you may choose are:

CTRLC	Enable Control-C (the user-interrupt command).
NOCTRLC	Disable Control-C (the user-interrupt command).
OCTAL	Display all non-decimal numeric displays in octal.
HEX	Display all non-decimal numeric displays in hexadecimal.
ECHO	Display terminal input.
NOECHO	Silence terminal input.
DSKERR	Report soft disk errors and retries that occur.
NODSKERR	Silence reports of soft disk errors and retries.
VERIFY	Verify every write operation by re-reading the data and making sure it has a correct checksum. This does not compare the written data with data in memory. (Only supported by the AM-500 and AM-410.)
NOVERIFY	Don't verify write operations.
GUARD	Don't allow other terminals to send messages to your terminal via the SEND command.
NOGUARD	Allow other terminals to send messages to your terminal.
FORMS	Specify the kind of form to assign to a specific printer.
BPI	Sets the magnetic tape unit bits per inch rate for data reading or recording on tape.

Remember: These options are only set for the job that used the SET command.

OPERATION:

1. Type SET followed by a RETURN:

```
.SET (RET)
Current settings are:
    OCTAL  ECHO    DSKERR  NOVERIFY  NOGUARD  CTRLC
```

SET displays the options currently in effect.

(Changed 30 April 1981)

2. To change the SET options, type SET followed by an option. Then type a RETURN. For example:

```
.SET HEX (RET)
```

See OPTIONS, above, for a list of the options you may select.

3. To assign a form-type to a printer, type SET FORMS followed by the name of the printer, the form-name, and a RETURN. For example:

```
.SET FORMS QUME CHECKS (RET)
```

The command above assigns the form-type CHECKS to the printer named QUME. (For a list of names of printers on the system, check with the System Operator.) The form-type can be any one- to six-character name you choose.

4. To set the recording density of a magnetic tape, type SET BPI followed by the device specification that selects the magnetic tape unit you want to access (drives 0-7) and the density required (800 or 1600 BPI). For example:

```
.SET BPI MTU1:800 (RET)
```

This command prepares MTU1 to change the read or write data at a density of 800 bits per inch. To find a list of magnetic tape units on the system, type DEVTBL followed by a RETURN. The system will handle a maximum of 8 magnetic tape drives.

ERRORS:

[INVALID FUNCTION]
SET doesn't recognize the option you've chosen. Check your spelling and retype the command line. If you are using the SET BPI option, this message indicates that you have supplied an invalid device specification or that you have entered an invalid BPI value. Make sure that you have designated the magnetic tape unit correctly and that you have specified a BPI of 800 or 1600. (For a list of valid MTUs, type DEVTBL followed by a RETURN.)

?Printer not found
You tried to use SET to associate a form-type with a printer, but SET doesn't recognize the printer name you specified. Check your spelling; then check with the System Operator for a list of printers on the system.

CHARACTERISTICS:

Allows you to: choose various system output options for your job; see the options currently in effect; define a form-type to be mounted on a specific printer; or SET the read or record bits per inch rate on a magnetic tape unit.

FUNCTION:

Tells you the size (in bytes) of your disk file.

HINTS/RESTRICTIONS:

Tells you the number of bytes in the specified file (in decimal).

FORMAT:

.SIZE Filespec ↵

where Filespec specifies the file whose size you want to know.

DEFAULTS:

SIZE assumes the account and device you are logged into and a file extension of .PRG.

OPERATION:

1. Type SIZE followed by the specification of the file whose size you want to see. Then type a RETURN. For example:

```
.SIZE FORM.TXT ↵
SIZE IS 1483 BYTES
.
```

ERRORS:

You may see several of the more common system error messages when you use SIZE. For example:

?Cannot OPEN Filespec - file not found
SIZE was not able to find the file you specified. Check your spelling and make sure that the account and device specifications were correct.

?Cannot READ Filespec - device does not exist
SIZE doesn't recognize the device you specified. Check your spelling. To see a list of valid system devices, type DEVTBL followed by a RETURN.

CHARACTERISTICS:

Returns your terminal to AMOS command level.

(1 October 1979)

skip

FUNCTION:

Skips to the next file mark on a tape mounted on a magnetic tape unit.

HINTS/RESTRICTIONS:

Use SKIP to skip over files or a header block on a magnetic tape.

IMPORTANT NOTE: DO NOT use SKIP while another job is accessing the specified tape drive. Such an action causes unpredictable results.

You may specify 0 to 7 tape drives (e.g., MTU0:-MTU7:).

The magnetic tape drive you specify must be defined in the system device table as an unsharable device. (See the DEVTBL reference sheet.) The program MTSTAT.SYS must be in system memory. (See the SYSTEM reference sheet for information on modifying the system initialization command file to include a program in system memory.)

FORMAT:

.SKIP Devn: ↵

where Devn: specifies a magnetic tape drive.

OPERATION:

1. Make sure that the tape you want to access is mounted.

2. Type SKIP followed by the specification of the tape you want to access. Then type RETURN. For example:

   .SKIP MTU5: ↵

3. SKIP issues a command to the magnetic tape driver to read forward on the currently mounted tape until it detects a file mark. If the drive is positioned at the front of a file, this command causes the tape driver to skip to the next file.

4. When SKIP finishes, it returns you to AMOS command level.

ERRORS:

?File specification error

SKIP did not understand your device specification. Make sure that the device you specified is defined in your system device table. (Type DEVTBL followed by a RETURN for a list of the valid devices on the system.)

(1 October 1979)

CHARACTERISTICS:

Requires that MTSTAT.SYS be in system memory and that the device you want to access be defined in your system device table.

Do not use while other jobs are accessing the specified device.

Returns your terminal to AMOS command level.

sleep

FUNCTION:

Allows you to put your own job to sleep for a specified amount of time.

HINTS/RESTRICTIONS:

The number you give SLEEP tells the operating system how many seconds to keep your job inactive.

The SLEEP command is useful in command files for keeping your job inactive until another procedure completes.

FORMAT:

.SLEEP N ↵

where N is a decimal number that specifies the number of seconds you want your job to stay inactive.

DEFAULTS:

If you do not specify an amount of time to the SLEEP command, SLEEP assumes a value of 0 seconds.

OPERATION:

1. Type SLEEP followed by the number of seconds you want your job to remain inactive. Then type a RETURN. For example:

   .SLEEP 60 ↵

2. The example above tells AMOS to put your job to sleep for one minute.

ERRORS:

SLEEP displays no error messages.

CHARACTERISTICS:

Puts your job into inactive status for a specified number of seconds.

Returns your terminal to AMOS command level.

(1 October 1979)

FUNCTION:

Bootstrap loader program for a system that uses the Control Data Corporation Phoenix hard disk as the System Device.

HINTS/RESTRICTIONS:

The SMDLOD program when contained on a 2716 PROM allows the system to boot off a System Disk on a Phoenix hard disk when a hardware reset occurs (that is, when you hit the RESET button).

The program is also in account DSK0:[1,4] of the System Disk.

You may use SMDLOD at AMOS command level to reset the system if your System Device is a CDC Phoenix hard disk. The Phoenix is a 90-megabyte hard disk device (75-megabyte fixed, 15-megabyte removable). The memory partition of the job that uses the SMDLOD command MUST be in Bank Zero if your system bank switches memory. (For information on bank-switched systems, refer to the document Memory Management Option, (DWM-00100-10) in the AM-100 documentation packet.)

FORMAT:

.SMDLOD ↵

OPERATION:

1. Type SMDLOD followed by a RETURN:

   .SMDLOD ↵

   The system now resets itself by reading a copy of the SMDLOD bootstrap program into system memory and executing it.

2. Once invoked, the SMDLOD program reads the operating system skeleton monitor, DSK0:SYSTEM.MON[1,4], into memory. SYSTEM.MON then brings up the system under the control of your system initialization command file, SYSTEM.INI.

3. Once the system is up and running, you see the AMOS prompt.

ERRORS:

SMDLOD generates no error messages. However, if it cannot find SYSTEM.MON[1,4] and SYSTEM.INI[1,4], the start-up procedure fails.

(1 October 1979)

CHARACTERISTICS:

Boots the system from a CDC Phoenix hard disk if that disk is the System Device.

Returns your terminal to AMOS command level if the system resets successfully.

FUNCTION:

Alphabetically and numerically sorts data records in a sequential text file.

HINTS/RESTRICTIONS:

Sorts only sequential files.

Sorts according to the ASCII values of the characters in the data records. (Therefore all data records that begin with upper case letters will come before all data records that begin with lower case letters-- or vice versa, depending on whether you are sorting in ascending or descending order.)

You may sort a text file that is too large to fit into memory all at one time.

SORT does not understand wildcard symbols.

SORT replaces the file you specify with another file of the same name in which the data are arranged in the proper order.

For more information on SORT and for an example of its use, refer to Section 10.4, "Sorting a File (SORT)," in the AMOS User's Guide, (DWM-00100-35).

FORMAT:

.SORT Filespec ↵

where Filespec selects the file you want to sort.

DEFAULTS:

File specification defaults are a file extension of .DAT (indicating a data file) and the account and device you are currently logged into.

OPERATION:

1. To sort a file, type SORT, the specification of the file you want to sort, and a RETURN. For example:

   .SORT LABELS.DAT ↵

2. SORT now asks you a series of questions:

   a. Record size: - SORT recognizes a carriage return symbol

as the end of each data record, but it also needs to know the size of the largest data record it is going to be dealing with. Enter the maximum size (in bytes) of the logical data records in your file. (Every character in the record is one byte of data, including spaces and punctuation. Exclude carriage return and line-feed bytes.)

b. Key size: - The key is the field in the logical record on which you wish to sort (e.g., customer name). SORT asks this question once for each key that you define. (You can define up to three keys.) If you want SORT to use less than three keys, answer this question with a RETURN after you have defined all of the keys you want to use. Enter the size (in bytes) of the key.

c. Key position: - SORT asks this question for each of the keys you define. Enter the column number in the data record where the first byte of the sort key occurs. (The first byte of a record is position #1.)

d. Key order: - SORT asks this question for each of the keys you define. Enter an A if you want that key sorted in ascending ASCII order or enter a D if you wish a descending order sort.

3. SORT now sorts the file. After the sort has finished, SORT reports back with some statistics:

a. Sorted n records - This statistic tells you how many logical records SORT processed.

b. n Runs - SORT tells you how many passes it made through the data to sort the file. If the entire file fits into memory, SORT performs the sort in memory and you see 1 RUN; if your file is too large to fit into memory all at one time, SORT performs the sort on the disk (a modified poly-phase merge sort), and you see that SORT performed its sort in more than one Run.

c. n Key comparisons, m per record - SORT tells you how many comparisons it made while doing the sort-- an indication of how out of order the file was.

d. hh:mm:ss Elapsed time, n ms per record - Elapsed time (rather than compute time) taken by the file sort. (Elapsed time will be affected by the number of other users on the system, and on the type of processing they are doing.)

ERRORS:

You can see the following error messages when using SORT:

?Enter A or D
SORT wants to know whether you want to sort the key in ascending (A) or descending (D) order based on the ASCII values of the characters in the data record. You must enter an A or a D.

?Insufficient memory for sort
SORT must be able to fit at least five of your data records into memory to perform a sort; if it cannot, you see this message.

?Record size must be >0
?Key size must be >0
?Key size must be less than record size
SORT checks to see that you are giving it reasonable data. If you see any of these messages, it probably means that you made a typing error. Key and record size must be at least one byte in length and (since the key is an element of the record) the key must be smaller than the record.

?Entire key must be within record
If the [(start-position in the record of the key + the length of the key)-1] is beyond the end of the record, you see this message. (The minus one comes from the fact that the first position in the record is a one and not a zero.) SORT thinks that the record size you gave was too small or that the key size was too big.

You can also see several system messages. For example:

?Cannot DELETE Filespec - write protected
You have tried to sort a file on a write-protected disk. SORT cannot replace your unsorted file with the new, sorted file because it can't write on the disk. Write-enable your disk and try again.

?Cannot OPEN Filespec - file type mismatch
You see this message if you try to sort a random file (that is, a file that has been allocated contiguously on the disk). You may only sort sequential files.

CHARACTERISTICS:

Sorts logical records in a sequential file in ascending or descending order.

Allows you to specify up to three keys on which to sort.

Replaces your original file with a new, sorted file.

Returns your terminal to AMOS command level.

(1 October 1979)

FUNCTION:

Compares two sequential files and lists the differences between them.

HINTS/RESTRICTIONS:

Useful for distinguishing between two versions of the same program so that you can determine what changes have been made.

FORMAT:

.SRCCOM {Listfilespec}=Oldfilespec,Newfilespec{/Switch} RET

where Oldfilespec and Newfilespec select the files that you want to compare and Listfilespec is the file that SRCCOM will create to hold the comparison. You may choose one of two switches (see OPTIONS).

DEFAULTS:

SRCCOM assumes a file extension of .MAC for Oldfilespec and Newfilespec and .LST for the Listfilespec. "Listfilespec=" may be omitted, in which case the listing will be displayed on the terminal (i.e., the default Listfilespec is TRM:).

OPTIONS:

You may choose one of the two options below by ending the SRCCOM command line with a slash (/) and the appropriate switch:

/Q Quick listing. The comparison listing is in the same format as the standard listing, but SRCCOM lists only the differences between the two files.

/B Brief listing. The comparison listing is in the same format as the Quick listing, but all line numbers are omitted.

OPERATION:

1. Type SRCCOM followed by the specification of the file you want to create to hold the file comparison. Now type an equal sign followed by the specifications (separated by a comma) of the two files you want to compare. (If you want to ask for a /B or /Q option, enter it here.) Type a RETURN. For example:

.SRCCOM CMPARE=CTLOG1.TXT,CTLOG2.TXT RET

(Changed 1 May 1980)

2. After SRCCOM has made the comparison, it returns you to AMOS command level. Now you can either print the comparison file (your Listfile) or display it on the terminal.

3. When you display the Listfile, you see a display something like this:

```
SRCCOM Version 2.0 Comparison - .SRCCOM CMPARE=CTLOG1.TXT,CTLOG2.TXT

00001  00001  First line of text
00002  00002  So far, all the same.
00003  -----  !! 2 lines that occur
00004  -----  only in CTLOG1.TXT !!
00005  00003  Now the text is again
00006  00004  the same, but the line
00007  00005  is different between
00008  00006  the two files.
+++++  00007  !! A line only in CTLOG2.TXT !!
00009  00008
```

The numbers on the left of the display are the line numbers of the Oldfile; the next column of numbers are the line numbers of the Newfile. The ---- symbols indicate a line that does not appear in the Newfile, but does appear in the Oldfile. The ++++ symbols indicate a line of characters that does not appear in the Oldfile, but does appear in the Newfile. In the example above, the two files are exactly the same until the third line; then the Oldfile has an extra two lines that the Newfile doesn't have. The files are again the same (although their line numbers are now different) until the second from the last line, where the Newfile has a line that the Oldfile doesn't have.

If you ask for the /Q option, the SRCCOM display looks much like the one above, except that the Listfile contains only lines where differences occur. The /B option tells SRCCOM to create a Listfile just like the /Q comparison, but without line numbers.

ERRORS:

If you enter a line that SRCCOM doesn't understand, it replies:

```
?Enter:     .SRCCOM listfile=oldfile,newfile
  or        .SRCCOM listfile=oldfile,newfile/Q
  or        .SRCCOM listfile=oldfile,newfile/B
(/Q produces a 'Quick' listing of differences only)
(/B produces a 'Brief' listing - quick and unnumbered)
```

(Changed 1 May 1980)

You may also see the standard system error messages that result from invalid file or device specifications. For example:

?Cannot OPEN Filespec - file not found

CHARACTERISTICS:

Assumes a file extension of .MAC for Oldfilespec and Newfilespec and an extension of .LST for Listfilespec. If you omit Listfilespec, assumes TRM:.

Returns your terminal to AMOS command level.

(Changed 1 May 1980)

FUNCTION:

Allows you to suspend the operation of a job for an indefinite period of time.

HINTS/RESTRICTIONS:

Use in combination with the REVIVE command to put a job in suspension and awaken it again. Once you use SUSPND, the specified job stays inactive until someone wakes it again.

FORMAT:

.SUSPND Jobname ↵

where Jobname specifies the job you want to suspend.

OPERATION:

1. Type SUSPND followed by the name of the job you want to suspend.

   .SUSPND JOB5 ↵

   The job remains suspended until you issue a REVIVE command for that job.

ERRORS:

[NONEXISTENT JOB]
You specified an invalid job name. Check your spelling. If the job name looks all right, use the SYSTAT command to see a list of valid jobs on the system.

CHARACTERISTICS:

Returns your terminal to AMOS command level.

To wake up the job you suspended, use the REVIVE command.

symbol

FUNCTION:

Creates a symbol table file for one or more machine language object files to allow you to reference user defined symbols when you use the symbolic debugging programs DDT or AlphaFIX. Also provides several options that let you generate a program file or a load map file.

HINTS/RESTRICTIONS:

The object files that MACRO (the assembler) creates contain complete information about the symbols used in your program, as well as the actual generated code. To make this list of symbols available to the debugging programs DDT and AlphaFIX, you must use the SYMBOL program to generate a symbol table file.

SYMBOL processes files in the order in which their specifications appear on the command line. You may not specify an overlay or library file as the first file on the command line.

NOTE: You may also use SYMBOL to generate a program file. (And, you may use LINK to produce a symbol table file. LINK and SYMBOL can be made to perform exactly the same functions if you use the appropriate option requests.)

SYMBOL supports library and optional files, and can generate a load map file. For information on library, optional, and load map files, see the AMOS Assembly Language Programmer's Manual, (DWM-00100-43). This manual also contains information on SYMBOL, MACRO, LINK, GLOBAL, LIB, and DDT.

FORMAT:

.SYMBOL {/switches }filespec1{,filespec2,...filespecN}{/switches} RET

where filespecs select the files you want to process and /switch is an option request. You may not specify an overlay or library file as the first filespec on the command line.

If you have too many filespecs to fit on one screen line, you may continue the SYMBOL command line by ending it with a comma. SYMBOL accepts as many lines of filespecs as you wish, as long as you end the preceding line with a comma.

If a /switch appears in front of a filespec (e.g., SYMBOL MATH,/O NUM,SUB), that option request becomes the default for the rest of the command line. If a /switch follows a filespec (e.g., SYMBOL MATH,NUM/O,SUB), it affects only that filespec. NOTE: Certain switches

(Changed 30 April 1981)

(identified in the discussions below as "operation switches") affect all filespecs on the command line no matter how they are placed.

DEFAULTS:

SYMBOL uses the default extension of .OBJ, unless you are specifying a library file, in which case it uses the default extension of .LIB.

If your filespec does not contain an account and device specification, SYMBOL assumes that the file is in the account and device you are logged into. Next it looks in your project library account, [P,0]. Finally, it looks in the System MACRO account, DSK0:[7,7].

The default switches are /S (generate a symbol table file) and /R (designate a required file).

OPTIONS:

You may select one of the options below by specifying the appropriate switch:

/E	Include equated symbols in the symbol table file. (You may also use /E with the /M switch to tell SYMBOL to put equated symbols in the load map file.) (Operation switch.)
/L	Designates a library file.
/M	Generate a load map (.MAP) file. (Operation switch.)
/N	Suppress /S switch. (Operation switch.)
/O	Designates an optional file.
/P	Generate program (.PRG or .OVR) file. (Operation switch.)
/R	Designates a required file. The default switch. Cancels the /L and /O switches.
/S	Generate a symbol table (.SYM) file. The default switch. (Operation switch.)

You may specify multiple switches by preceding each switch with a slash, /.

OPERATION:

1. Enter SYMBOL followed by the specifications of the files you want to process together; then type a RETURN. For example:

   .SYMBOL/M VISFIL,VIS1,UTILIT.LIB/L RET

   Notice that the command line above specifies a library file, UTILIT.LIB/L. By using the /M switch, we are also asking SYMBOL for a load map file.

2. If you have more file specifications than will fit on one line, end the current command line with a comma. SYMBOL now displays an asterisk and you may continue your list of file specifications.

(Changed 30 April 1981)

You may enter as many lines of file specifications as you wish as long as you end the preceding command line with a comma.

3. Now SYMBOL displays several messages as it processes the files. (The exact messages you see depend on the options you have requested and the files you have specified.) For example:

```
== Linkage Editor Version 2.0 ==

Processing VISFIL.OBJ
Processing VIS1.OBJ

-- Optional and Library Request --

Processing UTILIT.LIB(GETADR)

Symbol and Map files finished
.
```

4. If any errors occur during linking, SYMBOL tells you so. For example:

```
Symbol file finished, 4 errors exist
```

ERRORS:

You can see the following error messages while using SYMBOL:

?Command error
There is something wrong with your command line. For example, you tried to use SYMBOL without specifying a file on which to work.

?Fatal error - Insufficient memory
You must increase the size of your memory partition; there was not enough room to perform the procedure you specified.

?Undefined switch /x - ignored
SYMBOL didn't recognize the switch, /x, you specified. Refer to the section "OPTIONS," above for information on the valid SYMBOL switches.

?Fatal error - Overlays of code are not permitted
Next expected address is xxxx
Overlay code address is xxxx
Your program is trying to overlay previous code. Check your .MAC programs to make sure that your overlay references are correct.

?xxxx undefined
An external symbol is undefined. This is a very common error. You have referenced a symbol which has not previously been defined

(e.g., you have made a reference to a label that does not exist). Make sure that an EXTERNed symbol in one segment is defined by an INTERN statement in another segment.

?Fatal error - First file must not be a library

To enable SYMBOL to correctly resolve external references to a library, you must specify the program that references that library before you specify the library file itself.

?Fatal error - Attempt to specify overlay xxx as optional

You may not use the /O switch to designate a file as optional if that object file is an overlay.

?Fatal error - Overlay symbol "xxxx" in segment yyyy
was not defined in a previous input segment

You may not reference an undefined overlay. In other words, SYMBOL is trying to process a supposed overlay file, but has seen no references to the overlay in a previous file. Without such a reference, SYMBOL cannot construct the overlay, so it aborts and returns you to AMOS command level.

?Fatal error - First file must not be an overlay

To enable SYMBOL to correctly resolve external references to an overlay, you must specify the program that references that overlay before you specify the overlay file itself.

?Fatal error - Expression stack error

An error occurred when SYMBOL evaluated some expressions in your files. You should never see this error message-- it indicates an internal error.

?Fatal error - Expression stack overflow

You exceeded the number of nested expressions that SYMBOL can handle. Try to find the exceedingly complex expression in your source file and simplify it.

CHARACTERISTICS:

SYMBOL is re-entrant, and may be loaded into system memory; it is also serially reusable.

Creates a symbol table file by linking and resolving intermediate .OBJ files.

Default extension is .OBJ for regular files and .LIB for library files. Default switches are: /S and /R.

(Changed 30 April 1981)

sysact

FUNCTION:

Use SYSACT to add or delete user accounts on a specific disk, add or change account passwords, or initialize a disk.

HINTS/RESTRICTIONS:

You must be logged into account [1,2] to run SYSACT. SYSACT works by modifying the Master File Directory of a specific disk. The MFD contains a list of all accounts on that disk (listed by their project-programmer numbers) and the password (if assigned) associated with each one. You may add or delete user accounts (by adding or deleting Project-programmer numbers) and add or change passwords.

If a disk has never been used before, you must first format the disk (or certify it, in the case of hard disks that run under control of the AM-410 controller). Then use SYSACT to initialize the disk. Next you must establish the account structure on the disk by using SYSACT to add PPNs to the disk's MFD.

The project-programmer number that you assign to the user is in the form of two numbers, each of which may range from 0 to 377, octal. The first number of the PPN is the project number; numbers 1 through 77 are reserved by Alpha Micro for system use. The second number is the programmer number. Accounts whose PPNs share the same project number are said to be in the same project; users may transfer files into another account from his own if both accounts share the same project number.

If while using SYSACT you decide that you've made a mistake, leave SYSACT by typing a Control-C instead of using the SYSACT exit command; the changes you've made will not get written out to the Master File Directory.

FORMAT:

.SYSACT Devn: RET

where Devn: is the specification of the logical unit whose MFD you want to modify.

OPERATION:

1. Type SYSACT followed by the specification of logical unit whose MFD you wish to modify. Type a RETURN. For example:

.SYSACT DSK1: RET

(Changed 1 May 1980)

2. The SYSACT prompt symbol is an asterisk: *. If you forget the SYSACT commands, type an H; SYSACT will list them for you. (See below, COMMAND SUMMARY.)

COMMAND SUMMARY:

The SYSACT commands:

A p,pn — Add account p,pn to the disk. If the account does not already exist, SYSACT adds it and then asks you for a password. If you do not want the account to have a password, enter just a RETURN. Any password you enter must be six characters or less.

C p,pn — Change the password of account p,pn. If the account exists, SYSACT asks for a new password (0-6 alphanumeric characters).

D p,pn — Delete account p,pn. If the account exists and it has no files, SYSACT deletes it.

E — Exit to monitor. SYSACT rewrites the MFD, incorporating your changes, before it takes your terminal back to AMOS command level.

H — Display a list of the SYSACT commands.

I — Initialize disk. Erases the data on the disk (including the contents of files and directories. Initializes bitmap. SYSACT asks you to confirm this command before it will initialize the disk: Initializing the disk clears all files - enter Y to confirm: Unless you enter a Y and a RETURN, SYSACT will not initialize the disk; instead it says: %No initialization performed and prompts you for another command.

L — Display a list of all of the accounts in the MFD and their passwords. For example:

```
*L (RET)
1,2
110,6     ROB
```

ERRORS:

?Account already exists
You've tried to create an account that already exists. Use L to see what accounts already exist in the MFD.

?Account does not exist
You've tried to delete an account that doesn't exist or you've tried to change the password of a nonexistent account.

(Changed 1 May 1980)

?Illegal account PPN - format is P,PN (P = octal 0 to 377)
You've entered a bad PPN as an Add, Change, or Delete parameter. Check the number and enter it again.

?Account has files on it
You tried to delete an account that still had files in it. Exit SYSACT, log into the account you tried to delete, and erase all files (.ERASE *.*). Now, you can invoke SYSACT again and delete that account.

?SYSACT is a privileged program (may only be run from PPN 1,2)
You must log into account [1,2] to run SYSACT.

?Invalid command - type H for help
SYSACT did not understand the command you entered. To see a summary of all SYSACT commands, type H followed by a RETURN.

?MFD is full - no new accounts may be added
The Master File Directory for each disk can contain no more than 63 entries. You tried to allocate the 64th account on the disk.

?Account does not exist
You tried to change the password of an account or tried to delete an account, but that account does not exist. Use the L command to see a list of all PPNs on the disk.

CHARACTERISTICS:

If a disk has never been used before (and it does not run under control of the AM-410), after you format the disk you must use SYSACT to initialize the disk. Then use SYSACT to add accounts to the disk's MFD.

Returns your terminal to AMOS command level if you use the Exit command (E) or a Control-C.

(Changed 1 May 1980)

syscpy

FUNCTION:

Copies all files from DSK0: to DSK1:.

HINTS/RESTRICTIONS:

SYSCPY is a command file that uses the DSKCPY command to make a literal image of DSK0: onto DSK1:. DSKCPY verfies that copy by checking the backup disk against the original.

IMPORTANT NOTE: Do NOT use on devices that run under control of the AM-410. The Phoenix disk contains information about bad blocks on that disk (the BADBLK.SYS[1,2] file); making a literal copy of that disk transfers over that information to a device to which it does not apply.)

SYSCPY writes over all data on DSK1:, so make sure that DSK1: is a blank or scratch disk.

FORMAT:

.SYSCPY ↵

OPERATION:

1. Make sure that DSK1: is a blank or scratch disk. If you are using a Hawk-based system, make sure that you are not running off the cartridge or you will erase all data on the fixed platter. To see if you are running off the cartridge instead of the fixed disk, type SYSTEM followed by a RETURN. If you are running off the cartridge you see:

   *** SYSTEM IS RUNNING FROM CARTRIDGE DISK ***

   If you do not see this message, enter SYSCPY followed by a RETURN:

   .SYSCPY ↵

2. You see the message:

   Insure scratch disk on drive 1 and hit RETURN:

   Type RETURN after you have made sure that the scratch disk is in drive number one.

3. Now you see:

   [Begin copy and verify of system disk from drive 0 to drive 1]

4. SYSCPY now uses the DSKCPY command to make a literal image of DSK0: onto DSK1:. When the copy is complete you see:

   [Duplication completed]
   [Verification completed]

   Now SYSCPY tells you that the disk copy is done by ringing your terminal bell five times.

ERRORS:

If a hard disk-error occurs while SYSCPY is using DSKCPY to make a literal image of DSK0:, you see a device error message. Check with the System Operator for help.

CHARACTERISTICS:

Command file uses DSKCPY to make literal image of DSK0: onto DSK1:.

Verifies the copy.

Assumes that you are not running off the cartridge.

DO NOT use on devices that run under control of the AM-410 Hard Disk Controller or that have a file BADBLK.SYS[1,2].

Returns terminal to AMOS command level.

sysmem

FUNCTION:

As part of the system initialization command file, defines which memory locations will be allocated to the bank switchable area of system memory. (For use only in systems that bank switch memory.) At AMOS command level, tells you what memory locations are assigned to the system area of switchable memory.

HINTS/RESTRICTIONS:

Switchable system memory is used to allocate BITMAPs and other system areas in bank switchable memory, reducing total resident monitor size.

When SYSMEM appears within the system initialization command file, it must appear after MEMDEF commands and before any BITMAP commands. If you are not familiar with bank switching memory on the Alpha Micro system, see the Alpha Micro Integrated Systems User's Guide, (DWM-00101-00).

We will not discuss here the mechanics of setting up a bank-switched system, but briefly: bank switching memory allows you to access more than 64K of memory on your system (although each individual user is still limited to a maximum of 64K).

NOTE: The first 16k of memory cannot be allocated to the bank switchable system area; it is reserved for the monitor.

Second 16K of memory:	40000-77776 (octal)	or	4000-7FFE (hex)
Third 16K of memory:	100000-137776 (octal)	or	8000-BFFE (hex)
Fourth 16K of memory (minus last 256 bytes for the I/O ports):	140000-177376 (octal)	or	C000-FEFE (hex)

IMPORTANT NOTE: You may not allocate to switchable system memory the last 256 bytes of the 64K address space. That means that the highest memory address you can allocate is 177376. (The memory addresses 177400-177777 are reserved by the system for the I/O ports.)

You may enter more than one SYSMEM command in the SYSTEM.INI if you want to allocate more than one area of system switchable memory.

FORMAT:

```
.SYSMEM (RET)
```

or:

```
SYSMEM Bank-#:StartAddress-EndAddress
```

(Changed 30 April 1981)

Use the first format to find out what memory is allocated to the system area. Use SYSMEM in the second format in SYSTEM.INI to allocate memory to the system area, where:

Bank-#	The number of the memory bank containing the memory being allocated to the system area.
StartAddress	The beginning address of the memory block being allocated to the system area.
EndAddress	The ending address of the memory block being allocated to the system area.

OPERATION:

1. At the AMOS command level, to find out what memory is allocated to the system area, type SYSMEM followed by a RETURN. For example:

```
.SYSMEM (RET)
Current system memory allocations are:
     1:32370-177376
```

   The message above tells you that switchable system memory is from address 32370 to address 177376 in Bank One, the second bank of the system.

2. To allocate memory to the switchable system area from within the system initialization command file, use SYSMEM, a bank number, and starting and ending memory addresses. For example:

```
SYSMEM 4:100000-160000
```

   SYSMEM expects you to enter the starting and ending memory addresses in the number base the system is currently using for your numeric displays. The default number base (if you do not use the SET command) is octal. (For information on changing the display base from octal to hexadecimal, refer to the SET reference sheet.)

3. If you want to allocate more than one area to switchable system memory, use more than one SYSMEM command. When you use the BITMAP command in the SYSTEM.INI to allocate bitmaps, if you use the /S switch, the monitor will try to place that bitmap in the switchable area allocated by the first SYSMEM; if there is not enough room left in that area, the monitor will try the area allocated by the second SYSMEM, and so on.

ERRORS:

?Memory allocation format error

SYSMEM doesn't understand the format of your command line. For example, did you leave out the colon after the bank number?

(Changed 30 April 1981)

?Non-existent bank number
You've given SYSMEM a bank number larger than the total number of MEMDEF statements in your SYSTEM.INI. (That is, you've specified a bank that has not been defined.)

?Allocation overlaps monitor memory
The monitor memory is used by the operating system. You must not allocate any of this memory to switchable system memory. You must either reduce the size of the monitor or change your user memory allocations.

?Illegal memory range (end is below base)
Ending address must be greater than starting address.

?Allocation is not within requested bank's defined memory
You've specified a valid bank number, but it is not addressed for the memory addresses you've requested. Check the addressing of your memory boards and the MEMDEF statements in your SYSTEM.INI.

?Allocation overlaps memory previously allocated to a job.
You've tried to allocate to switchable system memory an area that has already been allocated (via the JOBMEM command) to a user job. Check your bank number and memory addresses. If they're OK, check the user memory partition allocation.

CHARACTERISTICS:

Allows you to reduce the size of the resident monitor by defining an area of switchable memory in which bitmaps can be placed. Only for use on systems that bank switch memory.

Acts both as a user command and as a system initialization command.

(Changed 30 April 1981)

systat

FUNCTION:

Provides information about the jobs running on the system.

HINTS/RESTRICTIONS:

May be run by any user on the system, whether logged in or not. The job status codes which appear in the SYSTAT display are as follows:

TI	Terminal Input wait state
TO	Terminal Output wait state
LD	Program Load state
SL	Sleep state
DS	Disk access in progress
IO	I/O access other than terminal or disk
EW	External Wait state
RN	Running
SP	Suspended state
^C	Control-C

For more information on job status codes, see Chapter 12, "System Information Commands," in the AMOS User's Guide, (DWM-00100-35).

SYSTAT displays memory addresses in the number base the system is using for your numeric displays (usually octal). If you wish to see these addresses as hexadecimal numbers, make sure that the HEX option is set before using SYSTAT. (See the SET reference sheet for information on changing the numeric display base.)

FORMAT:

.SYSTAT{/switch} RET

where the optional /switch requests a brief status display.

OPTIONS:

You may select a brief status display by using the /N switch. This switch tells SYSTAT to omit the list of devices and blocks free on those devices.

OPERATION:

1. Type SYSTAT followed by a RETURN:

.SYSTAT RET

(Changed 1 May 1980)

The display you see gives this information for each job running on the system: 1. the name of the job; 2. the terminal to which the job is attached; 3. the device and account into which the job is logged; 4. the octal memory address where the Job Control Block for the job is located; 5. the terminal status for that job; 6. the last program run by the job before you used the SYSTAT command; 7. the number of bytes of memory (in decimal) allocated for that job; and, 8. the octal memory address at which the job's memory partition begins (including the bank number in which that partition resides, if the system bank switches memory). SYSTAT also tells you which devices are mounted on the system and how many blocks on those devices are free for use.

2. If you wish a brief status display, use the /N switch. This tells SYSTAT to omit the list of devices on the system and blocks free on those devices.

ERRORS:

SYSTAT generates no error messages.

CHARACTERISTICS:

Returns your terminal to AMOS command level.

EXAMPLES:

A typical SYSTAT display might look like this:

```
Status of AMOS Version 4.4

JOB1    TERM1   DSK4:310,2      023056  RN  SYSTAT   50176 bytes at 0:  35400
JOB2    TERM3   not logged in   023522  ^C           48896 bytes at 1:  40000
JOB3    TERM2   DSK1:20,1       024166  TI  VUE      48896 bytes at 2:  40000
SPOOL   NULL    DSK0:1,2        024632  EW  LPTSPL    4352 bytes at 3: 167000
4 jobs on system

DSK0   26254 Blocks free
DSK1   29032 Blocks free
DSK2   17121 Blocks free
DSK3   16542 Blocks free
DSK4   20795 Blocks free
DSK5    3642 Blocks free
AMS0   not mounted
AMS1   not mounted
8 devices on system
```

(Changed 1 May 1980)

The first line tells us about the first job on the system: 1. it's named JOB1; 2. it is attached to terminal TERM1; 3. it is logged into account DSK4:310,2; 4. its Job Control Block appears at memory address 23056; 5. it is in a RUN state; 6. it is running the SYSTAT program; and 7. it has 50176 bytes of memory in its partition which starts at memory address 35400 in Bank Zero.

Note that JOB2 and JOB3 both have memory partitions that start at the same address; this is because the system represented above is a "bank-switched" system. Both jobs' memory partitions start at the same address but in different memory banks.

We also see that devices DSK0: through DSK5: are mounted. Devices AMS0: and AMS1: are not mounted.

system

FUNCTION:

As part of the system initialization command file, SYSTEM tells the operating system what programs to include in system memory. At AMOS command level, SYSTEM tells you what programs currently reside in system memory.

HINTS/RESTRICTIONS:

The SYSTEM command performs an important part of the system initialization process by allowing you to add programs to the monitor at the time of system start-up. In addition to that function, a SYSTEM command without a file specification MUST appear within the SYSTEM.INI to tell the system to close out the monitor expansion process that takes place at system start-up. (See the document The System Initialization Command File, in the AMOS Software Update documentation packet, for information on the system boot up process.)

After the system is up and running, the SYSTEM command takes on a new purpose; it allows you to see what programs are resident in system memory. If your System Device is a hard disk, SYSTEM also tells you if the system is running off the cartridge.

FORMAT:

.SYSTEM RET

or:

SYSTEM Filespec

where Filespec selects the program to be included in system memory. (You may only use the second format within the system initialization command file, SYSTEM.INI.)

DEFAULTS:

When you use SYSTEM within the system initialization command file, you may optionally include a file specification; the default file extension is .PRG and the default account is DSK0:[1,4].

(Changed 30 April 1981)

OPERATION:

At AMOS command level:

1. Type SYSTEM followed by a RETURN. SYSTEM now lists all of the programs that are currently in system memory and tells you how many bytes of memory are used by the monitor:

```
The following programs are allocated in system memory:
        VUE     PRG
        TRM     DVR
Total resident monitor size is 21607 bytes.
Monitor version is 4.5
```

2. If there are no programs in system memory, SYSTEM says:

```
No programs allocated in system memory
```

3. If you have a hard-disk system, SYSTEM lets you know if the system is using your disk cartridge as the System Disk.

```
* * * System is running from cartridge disk * * *
```

Within the SYSTEM.INI:

1. You can use one or more SYSTEM commands to tell the system to include programs within system memory. (Use one SYSTEM command for each program.) Follow the SYSTEM command with the file specification of the program you want to include. For example:

```
SYSTEM RUN.PRG
```

   Any programs that you include in the system monitor in this way must be reentrant.

2. Place any SYSTEM commands after the TRMDEF, BITMAP, and DEVTBL commands.

3. Whether or not you use SYSTEM to include programs in the monitor area of memory, you MUST use a single SYSTEM command without a file specification after all commands that expand the monitor size (including any other SYSTEM commands that include a file specification). For example:

```
QUEUE 10
SYSTEM RUN.PRG
SYSTEM BASIC.PRG
SYSTEM
```

(Changed 30 April 1981)

ERRORS:

No error messages.

CHARACTERISTICS:

Serves both as a system initialization command and as a user command.

MUST appear within the SYSTEM.INI to tell the system to close out the monitor expansion that occurs at system start-up.

(Changed 30 April 1981)

tapdir

FUNCTION:

Allows you to display a list of the files on a magnetic tape reel. Also allows you to create a disk file containing the tape directory.

HINTS/RESTRICTIONS:

Used in combination with FILTAP (to write disk files to magnetic tape) and TAPFIL (to copy files from tape to disk). For more information on these programs, see The Magnetic Tape File Backup Programs in the "System Operator's Information" section of the AMOS Software Update Documentation Packet.

The tape you read via TAPDIR must previously have been written by FILTAP. (See the FILTAP and TAPFIL reference sheets.)

Accepts full wildcard specifications, which select the files to be listed in the directory. (The account and device portions of the specifications refer to the disk account and device from which the files were originally backed up.)

TAPDIR is a wildcard command. See Chapter 9 of the AMOS User's Guide, (DWM-00100-35), for information on wildcard file specifications. Similar in use to DIR. (See the DIR reference sheet.)

FORMAT:

.TAPDIR {/switch}{Listfilespec=}{inspec1}{,inspec2...,inspecN} RET

where /switch is an option request. The optional Listfilespec specifies a disk file to contain the tape directory display. Inspec specifies the files on the tape whose directory listing you want to see.

DEFAULTS:

The default Listfilespec is DIRECT.LST in the account and device you are logged into. The default inspec is *.* and the account and device you are logged into. The default magnetic tape drive device specification is MTU.

OPTIONS:

The single TAPDIR switch is /KILL: it tells TAPDIR to delete any existing Listfile that matches the specified Listfilespec before creating the new Listfile. /KILL is an operation switch; it may appear anywhere on the command line.

(30 April 1981)

OPERATION:

1. Type TAPDIR followed by an optional Listfilespec and equal sign. Then type any input specifications. Type a RETURN. The Listfilespec specifies a disk file that will contain the directory display. If you omit the Listfilespec and the equal sign, TAPDIR sends the display to your terminal. If you omit the Listfilespec and include the equal sign, TAPDIR uses the default Listfilesoec DIRECT.LST and the account and device you are logged into.

   For example, to send a directory display of all the files on a tape to your terminal, enter:

   ```
   .TAPDIR ALL:[] RET
   ```

2. Now TAPDIR asks you for the specification of the magnetic tape drive that contains the tape you want to access:

   ```
   Enter tape unit number:
   ```

   The default device code is MTU, so to access a tape in drive MTU3:, you can either enter "3" or "MTU3:".

3. The display you see looks something like this:

```
.TAPDIR ALL:[]*.DAT,*.MAC RET
Enter tape unit number: 0 RET
   1        DSK0: SYS    MAC   140,1   16 L  14-May-80  14:52:23
   2        DSK0: NBSORT MAC   140,1    4 L  14-May-80  14:52:25
   6        DSK0: FILTAP MAC   140,1   23 L  14-May-80  14:52:25
   7        DSK0: JANE   DAT   140,1   99 C  14-May-80  14:52:27
Total of 4 files in 142 blocks
```

   The first number on the line tells you the file's relative position on the tape. Next you see the device specification, the filename and extension, and the account specification of the file as it appeared on the disk it was being backed up from. The next number tells you the number of disk blocks the file takes up. TAPDIR now tells you whether the file is a sequential, or linked file ("L"), or a random, or contiguous file ("C"). Finally, TAPDIR gives the date and time of the backup.

   For example, in the display above, TAPDIR tells us that the file NBSORT.MAC is the second file on the tape, that it was originally backed up from DSK0:[140,1], that it contains 4 disk blocks, that it is a linked file, and that it was backed up May 14th at 14:52:25.

   At the end of the directory display, TAPDIR tells you how many files were listed in the display.

4. To create a disk file containing the directory display, specify a Listfile. For example:

```
.TAPDIR = ALL:[]*.DAT (RET)
Enter tape unit number: MTU4: (RET)
```

creates the file DIRECT.LST in the account and device you are logged into that contains a directory display for all .DAT files on the tape in drive MTU4:. (NOTE: If your printer has been defined on your system as a terminal, you may send the display directly to a printer by using an output specification of "TRM:Printer-name", where printer-name is the name assigned to the printer by the TRMDEF command in the system initialization command file.)

ERRORS:

?Cannot find DSK0:SCNWLD.SYS[1,4] or MEM:SCNWLD.SYS

The TAPDIR program needs this file to be able to process wildcarrd symbols in you file specification. This message can indicate that SCNWLD.SYS does not exist, ot that you do not have enouugh memory to load the file into your partition.

?Cannot READ Devn - device does not exist
?Cannot READ Devn - device is not mounted

You tried to copy to or from a device that is not listed in the DEVTBL command in your SYSTEM.INI, does not have a driver in area [1,6] of the System Disk, is not file-structured, or is not mounted. ("Devn:" is the device you specified.)

%No file-oriented device corresponding to Devn: is mounted

You specified a device, but left off the unit number. TAPDIR cannot find a logical unit that matches your specification. Try mounting the device.

?Tape is not file structured

The tape you are trying to read was not written by the FILTAP program. The TAPDIR program can only read tapes written by FILTAP. Check to make sure you have mounted the correct reel of tape.

?More than one output specification

You may not supply more than one output specification.

?Device full

There is no more room on the disk.

%No such files

TAPDIR was unable to find any files matching your input specification.

CHARACTERISTICS:

A file-oriented tape directory program that displays a directory of a tape that was written by FILTAP. Allows you to specify wildcard file specifications.

Sends display to your terminal, a disk file, or a printer.

Default Listfilespec is DIRECT.LST in the account and device you are logged into. Default input file specification is *.* and the account and device you are logged into.

The device and account portions of the input file specification specify the disk account and device from which the files were originally backed up by FILTAP.

tapfil

FUNCTION:

Writes files from magnetic tape to disk. Allows you to restore file-oriented disk backup from tape to disk.

HINTS/RESTRICTIONS:

Used in combination with FILTAP (to write disk files to the tape) and TAPDIR (to see a list of files on magnetic tape). (See the reference sheets for TAPDIR and FILTAP.) Only reads tapes that have been written via FILTAP. Not for transferring data between Alpha Micro and non-Alpha Micro computers. (Use the TAPE program for that purpose-- see the TAPE reference sheet.)

Use a separate TAPFIL command for each magnetic tape reel.

Similar to the COPY command. (See the COPY reference sheet.) You may not copy to a disk account if it is not in the project you are copying from unless you are logged into the System Operator's account, [1,2], or are logged into the account you are writing to. If you are logged into [1,2], the default output account specification is [], and if the account you are copying to does not exist, TAPFIL creates it. You may copy files into the account you are logged into from any other account, regardless of project number. TAPFIL is a wildcard file command. (See Chapter 9 of the AMOS User's Guide, (DWM-00100-35), for information on using wildcard commands.)

The input specification must give the exact specification of the files you want to transfer from the tape, including device and account specifications of the files as they were written to the tape. The output specification allows you to specify the device and account the files are to be written to on the disk, and to rename the files as they are written out to the disk.

FORMAT:

.TAPFIL{/switches}{outspec}={inspec{/switches}{,inspec2{/switches}...}} RET

where /switches specifies a TAPFIL option, outspec specifies the files to be created on the disk, and inspec specifies the files to be transferred from the tape (including the disk device and account specifications originally associated with the files when they were backed up onto tape by FILTAP.)

(30 April 1981)

DEFAULTS:

The output specification defaults to the input specification. If you are logged into [1,2], the default output account specification is []. The input specification defaults to *.* and the account and device you are logged into. The default switches are /NOQUERY/DELETE. The default magnetic tape drive device specification is MTU.

OPTIONS:

Use the switches below to select TAPFIL options. Each switch must begin with a slash. Remember that the placement of the switch on the command line modifies its effect.

/QUERY	or /Q	Ask user for confirmation before copying files (file switch).
/NOQUERY	or /NOQ	Don't ask for confirmation before copying files (default, file switch).
/DELETE	or /D	Copy over to an existing file, thus deleting it (default, file switch).
/NODELETE	or /NOD	Don't copy over to an existing file (file switch).

OPERATION:

1. Type TAPFIL followed by the optional output specification. Then type an equal sign followed by any input specifications. Type a RETURN. The output specification assigns the specifications of the new files you are creating; the input specification selects the files to be copied. For example, to copy all .PRG files from the tape that were originally backed up from account [110,2] on DSK2: over to your current account DSK3:[110,5], enter:

   .TAPFIL DSK3:[110,5]=DSK2:*.PRG[110,2] RET

   Now TAPFIL asks you for the specification of the magnetic tape drive that holds the tape reel you want to read:

   Enter tape unit number:

   TAPFIL assumes a default device code of MTU. So, to read the tape in the magnetic tape drive MTU1:, you may either enter "1" or "MTU1:".

2. Remember that you may specify wildcard file specifications to TAPFIL (as in the input specification *.PRG, above).

3. Use the /QUERY switch to ask for confirmation before each transfer. Enter a Y for Yes or an N for No after each TAPFIL prompt. Do not type a RETURN after your answer. For example, suppose you are logged into DSK2:, and you are copying files from the magnetic tape drive MTU0:

```
.TAPFIL *.OLD[]=DSK3:*.MAC[10,*]/QUERY (RET)
Enter tape unit number: 0 (RET)
MTU0:DSK3:NEW.MAC[10,3] to DSK2:NEW.OLD[10,3]?Y
MTU0:DSK3:B32.MAC[10,4] to DSK2:B32.OLD[10,4]?Y
MTU0:DSK3:SCRTCH.MAC[10,6] to DSK2:SCRTCH.OLD[10,6]?N
Total of 2 files transferred
```

You may enter a Control-C at any time to stop the file transfer. Remember, the placement of the TAPFIL switches modifies their effect.

4. As with the COPY command, you may not copy files into an account that is outside of the project you are copying from unless you are logged into [1,2] or are logged into the account you are writing to. If you are logged into [1,2], TAPFIL will copy to any account, and will create the account you are writing to if it does not already exist. When you are logged into [1,2], the default output account specification is [].

ERRORS:

?Cannot find DSK0:SCNWLD.SYS[1,4] or MEM:SCNWLD.SYS

TAPFIL needs this file to be able to process wildcard symbols in your file specifications. This message can mean that SCNWLD.SYS does not exist or that you do not have enough memory to load the file into your partition.

?Cannot READ Devn: - device does not exist
?Cannot READ Devn: - device is not mounted

You tried to copy to or from a device that is not listed in the DEVTBL command in your SYSTEM.INI, does not have a driver in account [1,6] of the System Disk, is not file-structured, or is not mounted. ("Devn:" is the device you specified.)

%No file-oriented device corresponding to Devn: is mounted

You specified a device, but left off the unit number. TAPFIL tried to find a logical unit that matched the device code you specified, but failed to do so. Try mounting the device.

?Tape is not file-structured

The tape you are trying to read was not written by FILTAP. TAPFIL can only read tapes written by FILTAP. Check to make sure you have mounted the correct reel of tape.

?Missing output specification
You omitted the equal sign in the TAPFIL command line; TAPFIL couldn't tell which information was your input specification and which was your output specification.

?More than one output specification
You may not supply more than one output specification.

?Files may not be transferred to RES:
You may only add programs to system memory by using the SYSTEM command within your SYSTEM.INI.

%Not copied - Destination file already exists
You tried to copy to an existing file while the /NODELETE option was in effect.

?You are not logged in under [1,2], can't create [p,pn]
You cannot copy from an account to a nonexistent account unless you are logged into the System Operator's account, [1,2]. If you copy to a nonexistent account while logged into [1,2], TAPFIL will create the account for you.

?Output MFD is full
The Master File Directory only has room for 64 entries. The transfer in progress would have created a new account, but there is not enough room in the MFD.

?Device full
No more room on the disk.

%Bypassing BADBLK.SYS[1,2]
% BADBLK.SYS exists to prevent bad blocks
% on a device from being allocated, and
% should never be directly accessed.
You cannot copy the BADBLK.SYS[1,2] file, since this would lead to the corruption of the file system.

?Cannot OPEN Devn: - protection violation
You tried to copy into an account you are not logged into that is not in the same project as the account you are copying from, and you are not logged into [1,2]. Either log into [1,2] or the account you want to write into, and try again.

CHARACTERISTICS:

A file-oriented program that transfers files from a magnetic tape to the disk. You must specify the account and device that the files were originally backed up from. Allows you to rename files being copied.

Not for transferring data between Alpha Micro and non-Alpha Micro computers-- use the TAPE program for that purpose.

Accepts wildcard file specifications. Output file specificaton defaults to input file specification; input specification defaults to *.* and account and device you are logged into. Default switches are: /NOQUERY/DELETE. Default magnetic tape drive device code is MTU.

FUNCTION:

Magnetic tape utility program that copies data from disk files to tape or vice versa.

HINTS/RESTRICTIONS:

TAPE copies data from disk files to magnetic tape or from tape to disk files. It can perform ASCII to EBCDIC data conversion and vice versa.

If you are copying a magnetic tape created on another system, chances are that the data is incompatible with the AMOS system. AlphaBASIC requires that all data records end with a carriage return/line-feed pair. If these are not present, the BASIC may be unable to read the mag tape data you have written into a file. The screen-oriented text editor, VUE, requires that each line end with a carriage return/line-feed pair, and that the line be less than 510 characters. Beyond converting data from EBCDIC to ASCII or vice versa, the task of modifying data so that it is compatible with the AMOS system is left up to you.

MTSTAT.SYS must be in system memory. (Use the SYSTEM command in the system initialization command file to add this module to system memory.) If you are not sure whether or not MTSTAT.SYS is already in system memory, type SYSTEM followed by a RETURN. SYSTEM tells you what modules are in system memory.

Make sure that the magnetic tape drives you want to access are defined in your SYSTEM.INI device table. These devices are NOT sharable, so place them after a slash in the DEVTBL command line. For example:

```
DEVTBL DSK1,STD0,STD1,MEM,RES,/MTU0,MTU1,MTU2,MTU3
```

NOTE: TAPE only works with sequential disk files that contain fixed-length data records. It does not support any random-access file format on the tape. Before you use TAPE, use the SET BPI command (see the SET reference sheet) to set the recording density of the tape you are using.

Remember to make note of the size of the data records you write to tape and the number of records you write per tape block; when you read the data back to a disk file, you will need that information.

You can read one file after another on the magnetic tape by using TAPE several times in a row without rewinding the tape. You can also use the SKIP command to skip over files. See the SKIP reference sheet. (NOTE: Some tapes contain a one-block header file at the front of each file. You can skip over this header block by using TAPE once or via

the SKIP command.) To rewind tapes, use the REWIND command. (See the REWIND reference sheet.)

If at any time you make a mistake in answering TAPE questions, you may type a Control-C to exit TAPE and return to AMOS command level.

FORMAT:

.TAPE ↵

OPERATION:

1. Type TAPE followed by a RETURN:

   .TAPE ↵

2. Now the screen clears and TAPE begins to ask a series of questions, clearing the screen after each display:

   ```
   This is the magnetic tape program
   It can copy files from disk to tape or it can copy files
   from tape to disk.
   *
   *
   *
   Which do you want to do?
   1- copy a file from disk to tape
   2- copy a file from tape to disk

   Type the number of the option you wish
   Answer?
   ```

   Enter the appropriate number followed by a RETURN to select the function you want to perform. You may only enter a 1 or a 2; any other answer will cause TAPE to display the message above again. You may type a Control-C to interrupt the TAPE command and return to AMOS command level.

3. Now you see:

   ```
   Tape can do character code conversion
   What type of conversion do you want to do?
   1- None.
   2- Convert the ASCII file to an EBCDIC file.
   3- Convert the EBCDIC file to an ASCII file.

   Type the number of the option you wish.
   Answer?
   ```

   Enter 1, 2, or 3 followed by a RETURN. Remember that the conversion (if any) will be done on the data going to the

output device. Whether this device is the disk or tape unit depends on your answer to question #1, above.

4. If you asked to copy a disk file to tape, TAPE begins to ask a new series of questions:

   a. TAPE requests a file specification (give the AMOS specification of the file you want to copy) and a device specification (give the specification of the tape drive to which you want to copy).

   b. Now TAPE asks if you want to do reblocking. (Reblocking consists of specifying the number of file data records to place in each tape block.) If you say no to this question, TAPE assumes that each data record is 512 bytes long, and that the blocking factor is 1 (that is, that you want to write one disk block of 512 bytes into one tape block). If you tell TAPE that you want to do reblocking, it asks you the size of the data records in the file (in bytes, including record delimiters) and the blocking factor (the number of data records in each tape block).

   c. TAPE now displays a message that tells you how many characters will be written in each tape block. For example:

      You are writing 20000 characters in a tape block

      If this number is not satisfactory (e.g., it is zero), you have made a mistake in answering the TAPE questions. Type a Control-C to return to AMOS command level and try using TAPE again.

   d. Now TAPE is ready to write to the tape. You see:

      *
      *
      *
      Is the tape loaded?
      Type return if it is

      Make sure that the tape has been physically loaded on the tape drive, and that the tape is at load point (i.e., the metallic film at the start of the tape is positioned at the read heads). Type a RETURN when you are ready.

      TAPE now displays this message:

      Is the drive on-line?
      Type return if it is

Make sure that the drive is on-line. If the drive is ready to write, type a RETURN.

e. TAPE now transfers your file to the tape. When it is finished, you see a message that tells you how many blocks were written and how many errors were encountered. For example:

```
There were 100 blocks written or read
There were 0 errors
```

5. If you asked to copy data from tape to a disk file, TAPE asks you a series of questions:

   a. TAPE asks for a file specification (give the AMOS specification of the file you want to create) and a device specification (give the specification of the tape drive you want to use).

   b. You must now supply blocking information to TAPE. Note that when writing data from a disk file to tape this information is optional; in that case, TAPE has a default record size and blocking factor that it can use. However, when writing from tape to a disk file, you MUST supply blocking information. TAPE asks you the size of the data records (in bytes, including record delimiter characters) and the number of data records in each tape block. (These values were established when you first wrote the data onto the tape.)

   c. TAPE now tells you how many characters it is going to read per tape block. For example:

```
You are reading 1000 characters in a tape block
```

   If this number is not satisfactory (e.g., it is zero), you made a mistake in answering the earlier TAPE questions. Type a Control-C to exit TAPE and return to AMOS command level. Then try using TAPE again.

   d. TAPE now asks you how many tape blocks you want to read. When you enter this value, remember that TAPE will read fewer blocks than that if: 1. it finds an end-of-file marker; or 2. it reaches the end of the tape.

e. TAPE now asks you to type RETURNs when the tape is loaded and when the drive is on-line. After TAPE reads the data from the tape, it displays a message that tells you how many tape blocks were read and how many errors were encountered. For example:

```
There were 200 blocks written or read
There were 0 errors
```

ERRORS:

You can see the standard system error messages if you provide invalid file or device specifications. For example:

?Cannot READ Devn: - device does not exist

Make sure you have defined the magnetic tape drive in your system device table.

You can also see the following TAPE message:

You are writing 0 characters in a tape block

This message indicates that you made an error in entering values to the TAPE questions that asked for blocking information. Type a Control-C to exit to AMOS command level and try using TAPE again.

CHARACTERISTICS:

MTSTAT.SYS must be in system memory.

Make sure that you convert the data you transfer so that it is compatible with the AMOS system.

TAPE works only with sequential disk files that contain fixed-length data records.

The default blocking factor and data record size for writing data from file to tape is: blocking factor = 1; data record size = 512 bytes.

Returns your terminal to AMOS command level.

FUNCTION:

Displays or sets the system time of day and resets the time of day on the AM-120 Auxiliary I/O Controller board.

HINTS/RESTRICTIONS:

Displays time in AM/PM format; allows you to set the time either in that format or in military (i.e., 24-hour) format.

Requires the file TODCNV.PRG in account [1,4] of your System Disk.

Certain features are enabled only if your system contains an AM-120 Auxiliary I/O Controller board. (The AM-120 board performs several functions, including providing a battery-supported calendar and clock which maintain and update the date and time even when your system is not on.) For information on the AM-120, see the document Software Installation Instructions for the AM-120

You may only set the system time and reset the AM-120 time if you are logged into the System Operator's account, [1,2].

From within the system initialization command file, TIME allows you to set the system time automatically from the AM-120 clock/calendar.

TIME is not re-entrant, but is re-usable.

FORMAT:

.TIME RET

or:

.TIME HH:MM:SS{AM{PM}} RET

or:

TIME

where HH is hours, MM is minutes and SS is seconds. You may optionally enter AM or PM if you do not use military (i.e., 24-hour) format.

You may use the first format when logged into any account. This format displays the time (if it is set).

You may only use the second format if you are logged into the System Operator's account, [1,2], or if you are using TIME in response to a :K command that appears before the final SYSTEM commmand in your system initialization command file. This format sets the system time and optionally resets the time on the AM-120 board.

(Changed 30 April 1981)

Use the third format from within the system initialization command file to set the system time from the AM-120 Auxiliary I/O Controller board.

DEFAULTS:

If you have entered the time in military form, TIME sets the display to AM or PM as dictated by your entry; if you have not used military format, the default is AM.

OPERATION:

At AMOS command level from any account:

1. To see the time of day, type TIME followed by a RETURN. For example:

```
.TIME (RET)
12:30:12 PM
```

At AMOS command level from the System Operator's account, [1,2]:

1. To set the system time of day, type TIME followed by the time (in the format HH:MM:SS) optionally followed by an AM or PM. (Remember: if you are using military format, do not include an AM or PM on the TIME command line.) For example:

```
.TIME 03:12:56 PM (RET)
```

or:

```
.TIME 16:00:00 (RET)
```

2. The system time is now set. If your system contains an AM-120 Auxiliary I/O Controller board, TIME now asks you if you want to reset the time on the AM-120:

```
Do you wish to reset the AM-120 board also?
```

Answer by entering a Y or an N followed by a RETURN. If you answer N, the system time is still set to the time you specified, but the time is not changed on the AM-120 board.

In the SYSTEM.INI file:

1. To automatically set the system time of day from the AM-120 board when your system is rebooted, include the TIME command in your system initialization command file before the final SYSTEM command but after the CLKFRQ command. For example:

```
CLKFRQ 60
TIME
SYSTEM
```

(Changed 30 April 1981)

NOTE: This feature requires that your system contain an AM-120 Auxiliary I/O Controller board.

ERRORS:

?Couldn't load TODCNV

TIME couldn't find TODCNV.PRG in system memory, in your memory partition, or in the disk account DSK0:[1,4]. Check your System Disk to make sure that it contains TODCNV.PRG[1,4]. You may also see this message if there is not enough room in your memory partition to load TODCNV. You can use the MAP command to see what modules are in your memory partition; if there are modules in your memory that you don't need, make more room by using the DEL command to delete them. Or, check with the System Operator to see if you can expand the size of your memory partition.

?Improper time format

You entered the time of day in an illegal format. (For example, you may have used slashes instead of colons.) Look at the FORMAT section above for information on the proper TIME command line format.

?Please enter Y or N

You have entered an invalid response to the question "Do you wish to reset the AM-120 board also?". Enter only a Y or an N.

?You must be logged in to [1,2] to reset the date

You may only use TIME to set the time if you are logged into [1,2] or if you are using TIME in response to a :K command that appears before the final SYSTEM commmand in your system initialization command file.

CHARACTERISTICS:

Displays the time of day in AM/PM format.

If you are logged into [1,2], you can use TIME to change the system time of day using military format or AM/PM format. Also allows you to reset the time stored by the AM-120 Auxiliary I/O Controller board.

From within the system initialization command file, TIME allows you to set the system time from the AM-120 board.

NOTE: Some features require that your system contain an AM-120 Auxiliary I/O Controller board.

Uses the TODCNV.PRG program.

TIME is not re-entrant, but is re-usable.

(Changed 30 April 1981)

FUNCTION:

TRACE allows you to turn the command file trace flag on and off within a command file, thus affecting what command file elements the user of the command file sees.

HINTS/RESTRICTIONS:

May appear only in a command file or DO file. When the command file trace flag is "on," the user of command file can see all of the contents of the command file as the file is processed; the user does not see all of the contents if the trace flag is "off." (Even when the trace flag is off, certain special items in the command file can be seen by the user of the file-- e.g., messages to the user, program output, etc.-- if you use the :R, :S, and :<> symbols.) For information on setting up command files, see Chapter 8, "Command Files and DO Files," in the AMOS User's Guide, (DWM-00100-35). For a more detailed discussion of the command file trace flag, see New Features of Command Files and DO Files, (DWM-00100-63), in the "User's Information" section of the AM-100 documentation packet.

TRACE allows you to turn the trace flag on or off at any time within the command file, as many times as you want. You may also use TRACE to switch the current state of the trace flag, regardless of what that state may be. Note that the standard command file symbols :S, :R, and :<> work with TRACE in the same way that they worked with :T (an earlier predecessor of the TRACE command).

FORMAT:

TRACE Argument

where Argument tells TRACE to turn the trace flag on or off, or to switch its current state.

DEFAULTS:

If you do not include an argument on the TRACE command line, the command is ignored. If no :T or TRACE command appears in the command file, the trace flag is off.

OPTIONS:

TRACE takes one of three arguments:

ON	Turns the trace flag on.
OFF	Turns the trace flag off.
SWITCH	Switches (that is, reverses) the current state of the trace flag.

OPERATION:

1. Place the TRACE command anywhere in your command file where you want to affect the status of the trace flag. For example:

```
; Command file to compile BASIC programs.
; (Want user to see that we are using compiler.)
TRACE ON
COMPIL ARTBL
COMPIL IDXGL
COMPIL OEMENU
TRACE OFF
; Don't want user to have to see cleanup.
ERASE *.BAK
COPY [110,0]*.OLD=*.BAS
```

ERRORS:

TRACE displays no error messages. If you do not supply the proper argument, the TRACE command is ignored.

CHARACTERISTICS:

Uses three arguments (ON, OFF, and SWITCH) to turn on, turn off, or switch state of command file trace flag.

May only be used in a command file or DO file.

FUNCTION:

Troubleshoots a Trident system by allowing you to communicate with the Century Data 1150A Formatter.

HINTS/RESTRICTIONS:

Before trying to use TRIDDT, read the Installation and Operating Instructions for the AM-400 System that accompanies the AM-400 Hard Disk Interface and read the Century Data 1150A Formatter manual.

For information on the error and status codes returned by the Formatter and a list of TRIDDT commands, refer to Section 3, "Functional Description," of the document Trident 1150 Formatter Performance Specification Bulletin No. 346. This document is available from Alpha Micro, and should be included with each Trident system.

All TRIDDT commands and status codes are in hexadecimal, so be sure to use the SET HEX command before using TRIDDT. These commands and statuses are those values the system usually uses to transfer data and information between the computer system and the Century Data Formatter.

FORMAT:

.TRIDDT ↵

OPERATION:

1. Type TRIDDT followed by a RETURN:

   .TRIDDT ↵

2. Now TRIDDT asks you for a command:

   COMMAND:

   Enter the appropriate TRIDDT command. For example, the 80 command is a sixteen-bit command whose lower byte contains zeros in bits 7-4. Bit 3 of that command contains a zero if we are checking formatter status or a one if we are checking drive status. To request information on drive status, then, enter an 8008 after the command prompt. For example:

```
.TRIDDT ↵
COMMAND: 8008 ↵
STATUS: 00C0
```

The status code that TRIDDT returns in the example above tells us that the addressed drive (drive zero in this case) is READY and ON-LINE.

3. To exit TRIDDT, enter a Control-C.

ERRORS:

UNKNOWN STATUS BYTE
You see this error message if the Trident encounters a status byte from the AM-400 that it does not recognize.

CHARACTERISTICS:

Allows you to communicate with the Century Data 1150A Formatter to troubleshoot a Trident hard disk system.

Returns your terminal to AMOS command level.

FUNCTION:

Performs initialization for Century Data T25, T50, and T200 hard disks.

HINTS/RESTRICTIONS:

If you are using models T25, T50, or T200 of the Century Data Trident series of hard disks as peripheral disk drives (that is, you are not using these drives as your System Device), you must use the TRIINI command every time you turn on or reset the system to initialize the Century Data 1150A Formatter.

If you do not use the TRIINI command, the disk drive will not be able to achieve READY status (i.e., a SYSTAT command shows the drive to be NOT READY).

FORMAT:

.TRIINI ↵

OPERATION:

1. Type TRIINI followed by a RETURN:

   .TRIINI ↵

   The TRIINI command initializes the 1150A formatter.

2. TRIINI returns a numeric code that identifies the status of the formatter. The message takes this form:

   STATUS: n

   where n is a 16-bit status code. For a discussion of the formatter status codes, refer to the documentation accompanying the 1150A formatter.

ERRORS:

TRIINI generates no error messages.

CHARACTERISTICS:

Communicates with a Century Data 1150A formatter working with the AM-400 Trident Hard Disk Interface.

Returns your terminal to AMOS command level.

(1 October 1979)

FUNCTION:

Bootstrap loader program for a system that uses the Century Data Trident T25, T50, or T200 hard disk as the System Device.

HINTS/RESTRICTIONS:

The TRILOD program when contained on a 2716 PROM allows the system to boot off a System Disk on a T25, T50, or T200 hard disk when a hardware reset occurs (that is, when you hit the RESET button).

The program is also in account DSK0:[1,4] of the System Disk.

You may use TRILOD at AMOS command level to reset the system if your System Device is a T25, T50, or T200 hard disk. The memory partition of the job that uses the TRILOD command MUST be in Bank Zero if your system bank switches memory. (For information on bank-switched systems, refer to the document Memory Management Option, (DWM-00100-10) in the AM-100 documentation packet.)

TRILOD only tries to boot off surface Zero. It will not operate if the drive is write-protected.

FORMAT:

.TRILOD ↵

OPERATION:

1. Type TRILOD followed by a RETURN:

   .TRILOD ↵

   The system now resets itself by reading a copy of the TRILOD bootstrap program into system memory and executing it.

2. Once invoked, the TRILOD program reads the operating system skeleton monitor, DSK0:SYSTEM.MON[1,4], into memory. SYSTEM.MON then brings up the system under the control of your system initialization command file, SYSTEM.INI.

3. Once the system is up and running, you see the AMOS prompt.

ERRORS:

TRILOD generates no error message, but if it does not find SYSTEM.MON[1,4] and SYSTEM.INI[1,4], the start-up procedure fails. TRILOD only tries to boot off surface Zero.

(1 October 1979)

CHARACTERISTICS:

Boots the system from surface Zero of a Century Data Trident T25, T50, or T200 hard disk if that disk is the System Device.

Does not operate if the drive is write-protected.

Returns your terminal to AMOS command level if the system resets successfully.

triset

FUNCTION:

Initializes the Century Data 1150A Formatter to handle the specific mix of Century Data hard disks on your system.

HINTS/RESTRICTIONS:

The AM-400 Hard Disk Subsystem allows you to run different Century Data hard disks under control of a single Century Data 1150A Formatter. For example, you can run T-25, T-50, T-80, T-200, and T-300 drives on the same system in any mix, so long as you do not exceed the maximum of four drives per formatter. (With the optional 1150A expansion, you may use eight drives with one formatter.) NOTE: If you do mix drives, you may reference only one type of drive as DSK.

The purpose of the TRISET command is to tell the Century Data Formatter the specific mix of drives that it is handling. You must run TRISET each time you reset or turn on the system. (You can do this from AMOS command level or you may include TRISET in your system initialization command file after the SYSTEM commands.)

Reference each type of drive in the DEVTBL and BITMAP commands with a different three-letter code. For example, the commands below set up a system that contains a T-300 (DSK0:-DSK18:, the System Device), a T-25 (TRA5:-TRA9:), a T-50 (TRB10:-TRB14:), and a T-80 (TRC15:-TRC19:):

```
DEVTBL DSK0,DSK1,DSK2,DSK3,DSK4,DSK5,DSK6,DSK7
DEVTBL DSK8,DSK9,DSK10,DSK11,DSK12,DSK13,DSK14
DEVTBL DSK15,DSK16,DSK17,DSK18
BITMAP DSK,1630,0,1,2,3,....18
BITMAP TRA,562,5,6,7,8,9
BITMAP TRB,1022,10,11,12,13,14
BITMAP TRC,1534,15,16,17,18,19
```

Remember to rename the drivers in account DSK0:[1,6] for the Century Data drives to the three-letter names by which you referenced those drives in the DEVTBL and BITMAP commands. For example, in the sample commands above, we reference the T-50 drive as device TRB. Rename the T-50 driver, DSK0:TRIT50.DVR[1,6], to DSK0:TRB.DVR[1,6].

Before using TRISET, you may have to modify the source program TRISET.MAC to reflect the mix of drives on your system if that program is not already correctly set up. TRISET.MAC is located in account DSK0:[10,2].

NOTE: For more information on modifying TRISET.MAC and setting up your system initialization command file to handle a mix of Century Data drives, refer to the document Installation and Operating Instructions for the AM-400 System, that accompanies your AM-400 Hard Disk Interface.

The portion of code that you must modify is titled "Configuration Table." Each line in the table below defines a type of Century Data Trident drive being used on the system. (The first four entries represent the four drive-types in the example above.) The high byte of the last word in each entry represents the physical drive number of the device. For example, the last word of the fourth entry is 1400. If you translate this number into binary, the top byte (the top eight bits) represents the number 3. The fourth entry, then, defines the drive physically addressed as drive #3 on the system.

A sample configuration table in TRISET.MAC:

```
; CONFIGURATION TABLE FOR EIGHT DEVICES
; (DRIVES 0-7)
; THE SIXTH WORD IN EACH ENTRY IS THE
; SECTOR-PER-TRACK CONSTANT
;       -- USE 26 FOR T25, T50, T200 DRIVES
;       -- USE 40 FOR T80 AND T300 DRIVES
; THE LAST WORD IN EACH ENTRY IS THE DRIVE
; NUMBER (HIGH BYTE).
;
CHGTBL: WORD    0,0,0,0,400,40,400,0        ; Drive 0
        WORD    0,0,0,0,400,26,400,400      ; Drive 1
        WORD    0,0,0,0,400,26,400,1000     ; Drive 2
        WORD    0,0,0,0,400,40,400,1400     ; Drive 3
        WORD    0,0,0,0,400,26,400,2000     ; Drive 4
        WORD    0,0,0,0,400,26,400,2400     ; Drive 5
        WORD    0,0,0,0,400,26,400,3000     ; Drive 6
        WORD    0,0,0,0,400,26,400,3400     ; Drive 7
        END
```

Use MACRO to re-assemble the TRISET program after changing the table. Then replace the version of TRISET.PRG already in DSK0:[1,4].

FORMAT:

.TRISET ↵

OPERATION:

1. Type TRISET followed by a RETURN:

.TRISET ↵

(1 October 1979)

The TRISET command now tells the Century Data 1150A Formatter what kinds of Century Data Trident disks are hooked up to your system. TRISET and the Trident initialization programs (T80INI and TRIINI) display standard completion codes.

2. Now that the formatter is properly initialized, use the MOUNT command to mount the devices you want to access.

ERRORS:

TRISET generates no error messages.

CHARACTERISTICS:

Initializes the Century Data 1150A Formatter to handle the specific mix of Century Data hard disks you have on your system.

You may include TRISET in your system initialization command file after any SYSTEM commands. You must run TRISET each time you reset or turn on your system.

Returns your terminal to AMOS command level.

FUNCTION:

As part of the system initialization command file, TRMDEF defines the terminals to be used on the system. At AMOS command level, it gives you information about the system terminals.

HINTS/RESTRICTIONS:

Once the system is up and running, TRMDEF is used to find out information about the terminals defined on the system. As part of the system initialization command file, the TRMDEF command actually defines those terminals. (For information on defining terminals, see the document The System Initialization Command File, (DWM-00100-09), in the "System Operator's Information" section of the AM-100 documentation packet.) Each terminal on the system must have its own TRMDEF command in the SYSTEM.INI.

FORMAT:

.TRMDEF (RET)

or:

TRMDEF Terminal-name, Interface=port#, In-width, In-buffer, Out-buffer

where the various parameters define the terminal to the system. (See OPERATION, below, for information on these parameters.)

OPERATION:

At AMOS command level:

1. Type TRMDEF followed by a RETURN. Each line in the display gives the following information about a terminal on the system: 1. terminal name; 2. job to which it is attached; 3. memory address (in octal) at which the Terminal Definition Block for that terminal exists; 4. interface driver used by the terminal; 5. number of the port on the I/O board the terminal is using; 6. terminal driver used by that terminal; and 7. terminal parameters. For example:

```
.TRMDEF (RET)
TRM1 JOB1     025574  AM300   1   SOROC   100,100,100
TRM2 JOB2     027764  AM300   2   HAZEL   100,100,100
TRM3 JOB3     030712  AM300   3   SOROC   100,100,100
TRM6          032306  AM300   6   SOROC   50,50,50
QUME          033230  AM300   5   TELTYP  50,50,100
NULL SPOOL    033646  PSEUDO  0   NULL    17,17,2
```

(Changed 1 May 1980)

The example above shows that we have six terminals defined on the system. TRM1 is attached to JOB1. The Terminal Definition Block for TRM1 appears at memory address 025574 (this address can be of use to system's programmers; the general user may ignore it.) The type of interface driver being used by TRM1 is the program that drives the AM-300 board. TRM1 is connected to port #1. TRM1 uses the terminal driver program for a SOROC terminal. The three terminal parameters specify the size (in bytes) of the terminal's input buffer, type-ahead buffer, and output buffer.

Note that QUME in the example above is not attached to a job (in fact, it is a printer). Also, the terminal named NULL is defined as a pseudo terminal. A pseudo terminal (i.e., a terminal that uses the PSEUDO interface driver) is not an actual hardware device, but is a software simulation of a terminal. A pseudo terminal allows you to use jobs that don't require terminal I/O without tying up a real terminal. The job that runs the line printer spooler program, for example, generally uses a pseudo terminal.

In the SYSTEM.INI file:

1. Before AMOS can transfer data between the system and a terminal, AMOS must know that the terminal exists, what driver program to use to access the device, what kind of terminal it is, etc. The TRMDEF statement performs this terminal definition process.

   The TRMDEF statements must appear directly after the JOBS statement in the SYSTEM.INI. Insert one TRMDEF statement for each terminal you want to define.

2. To enter a TRMDEF command line, type: 1. TRMDEF; 2. a terminal name; 3. the name of the interface driver program the terminal uses; 4. the name of the terminal driver program the terminal uses; 5. the maximum terminal line width; 6. the size of the terminal input buffer; and, 7. the size of the terminal output buffer. Separate the items on the TRMDEF command line with commas. For example:

   ```
   TRMDEF TERM5,AM300=2:6,SOROC,100,100,100
   ```

   The next few paragraphs discuss the elements of the TRMDEF command line:

   Terminal-name - A one- to six-character name you want to assign to the terminal. (This is the name used by the ATTACH and PRINT commands and any commands that specify the TRM: device.)

(Changed 1 May 1980)

Interface - Type of I/O board to which the terminal is connected. (For example, this might be an AM-300 or a PS3.) This tells the system what kind of interface driver program to load in for the terminal. (All terminal interface drivers must be in account DSK0:[1,6] and have a file extension of .IDV.) After the Interface, enter an equal sign and the number of the port on the interface to which the terminal is connected. For example, "AM300=2" tells the system that the terminal is connected to port #2 on the AM-300 interface board. The AM-300 board also allows you to include an optional code that selects the terminal baud rate. If you include the optional buad rate code, place it after the I/O port number. Separate it from the I/O port number by a colon (e.g., AM300=2:6). (The default AM-300 baud rate code is 19200 baud.) For a list of the baud rate codes used by the AM-300 and AM-310 interface drivers, see The System Initialization Command File, (DWM-00100-09), in the "System Operator's Information" section of the AM-100 documentation packet.

Terminal - Type of terminal. This tells the system what kind of terminal driver program to load in to access the terminal. The terminal driver programs must be in account DSK0:[1,6] and have file extensions of .TDV.

In-width - Specifies the maximum number of characters per line before the system starts to discard characters.

In-buffer - Specifies number of characters system will store in input buffer before discarding characters. (Also known as "type-ahead" buffer.) Sometimes the system cannot process characters as quickly as you type them. The in-buffer holds characters till the system can get around to them.

Out-buffer - Specifies size of the terminal output buffer (the buffer that holds the characters that the system sends to the terminal).

ERRORS:

TRMDEF generates no error messages.

CHARACTERISTICS:

Serves both as a system initialization command and a user command. One TRMDEF statment MUST appear in the SYSTEM.INI for each terminal you use on the system.

Returns your terminal to AMOS command level.

(Changed 1 May 1980)

txtfmt

FUNCTION:

Formats text files as directed by TXTFMT commands embedded in the files.

HINTS/RESTRICTIONS:

Use one of the system text editors to create a text file. Enter TXTFMT commands directly into the text file. (TXTFMT commands must be the only text on a line and must begin in the first character position at the left of the line. You must precede each command with a slash, /.)

After exiting the editor (i.e., you are at AMOS command level), you may now invoke TXTFMT to format the text file(s) you have created.

TXTFMT formats one or more text files into a single, formatted file. This formatted file bears the same name as the first file specified on the TXTFMT command line, but it has a .LST extension.

For information on using TXTFMT and on the TXTFMT commands, refer to TXTFMT User's Manual, (DWM-00100-07, Rev. B00).

FORMAT:

.TXTFMT Filespec1{,Filespec2,....FilespecN} RET

where Filespec selects the text file(s) you want to format.

DEFAULTS:

TXTFMT assumes file extensions of .TXT. Many of the TXTFMT commands have default values. Refer to the TXTFMT User's Manual, for information on the TXTFMT commands.

OPERATION:

1. Type TXTFMT followed by the one or more file specifications that select the text files you want to format. Then type a RETURN. For example:

   .TXTFMT HEADER,COPYRT,TITLE,PRFACE,MAIN RET

   The command above formats the files HEADER.TXT, COPYRT.TXT, TITLE.TXT, PRFACE.TXT, and MAIN.TXT into the single file HEADER.LST.

(Changed 30 April 1981)

2. TXTFMT tells you as it begins to process each file:

```
.TXTFMT HEADER,MEMO (RET)
Processing HEADER.TXT
Processing MEMO.TXT
.
```

3. When TXTFMT has finished formatting all the files you specified, you see the AMOS prompt.

4. To see your formatted file, you can either print it (using the PRINT command) or display it on the monitor screen (via the TYPE command). Both PRINT and TYPE use the default file extension of .LST.

COMMAND SUMMARY:

Below is a list of some of the TXTFMT commands; for a full list, refer to the TXTFMT User's manual.

/LINESIZE n	Sets width of text line	/BREAK	Start new line
/PAGESIZE n	Set # of lines per page	/INDENT n	Indent line
/MARGIN n	Set margin	/JUSTIFY	Right-justify
/CENTER text	Center text line	/PAGE	Begin new page
/TITLE text	Set page header	/PAGE n	If not n lines on page, begin new page
/NUMBER n	Set page number		
/FORMAT	Format text	/LIST	Begin list
/UNFORMAT	Do not format text	/LE	A list element
/PARAGRAPH	Start new paragraph	/END LIST	End the list
/CHAPTER	Start new chapter	/INDEX	Mark as index entry

Other commands allow you to set the page number (Roman numeral lower, Roman numeral upper, chapter-oriented, absolute, etc.) for the top or bottom of the page. You may also create a table of contents, appendices, an index with entries and sub-entries, double indents, nested lists whose elements are numbered with letters, Roman numerals, etc. You can also reset page numbers, header levels, and chapter and appendix numbers.

ERRORS:

Below we list the TXTFMT error messages you are most likely to encounter. For a full list of TXTFMT error messages, refer to the TXTFMT User's Manual.

(Changed 30 April 1981)

?Illegal command X

TXTFMT did not recognize a command, X, that you had in the text file. For example:

```
.TXTFMT SRCFIL.FNX
Processing SRCFIL.FNX
?Illegal command /CHEPTER
.
```

TXTFMT inserts the illegal command into your .LST file so that you can locate the problem. Check your spelling.

%Line too long - remainder of line ignored

TXTFMT found a line in your text file of more than 300 characters. It ignored everything past the 300th character.

You can also see the usual system error messages. For example:

?Cannot OPEN Filespec - file not found

You tried to format a file that does not exist. Check your spelling. Make sure that you specified the proper account and device.

CHARACTERISTICS:

Produces a formatted .LST file; does not change the input files.

type

FUNCTION:

Displays a text file on your terminal screen.

HINTS/RESTRICTIONS:

If the file display covers more than one screen page, type a Control-S to freeze the display; type a Control-Q to release the display. Type a Control-C to interrupt the display.

TYPE works only on sequential files.

Use TYPE only on files in which the data is in ASCII form (i.e., files with the extensions: .TXT, .LST, .BAS, .MAC, .LSP, .HLP, .CMD, .DO, etc.).

FORMAT:

.TYPE Filespec ↵

where Filespec selects the file you want to see.

DEFAULTS:

TYPE assumes a file extension of .LST.

OPERATION:

1. Type TYPE followed by the specification of the file you want to display. Then type a RETURN. For example:

   .TYPE PSTINV.BAS ↵

   You now see the file displayed on your terminal screen.

ERRORS:

?Cannot OPEN Filespec - file not found
TYPE wasn't able to find the file you specified. Check your spelling.

?Cannot INIT Filespec - device does not exist
TYPE cannot find the device you specified in your filespec. Use the SYSTAT command to see if the device is a valid, mounted system device.

(1 October 1979)

?Cannot OPEN Filespec - disk not mounted

The system can't access the device specified by your filespec; use the MOUNT command to mount the disk.

?Cannot OPEN Filespec - file type mismatch

You tried to display the contents of a random file. You will have to use another method to display the contents of that file.

?File specification error

TYPE did not understand the format of your command line. Check your spelling and retype.

?Cannot OPEN Filespec - illegal user code

You've specified an account that does not exist. Check your typing. If that's OK, check to see that you are trying to access the proper device.

CHARACTERISTICS:

Assumes that the specified file contains data in ASCII form.

You may not use TYPE to display a random file.

Returns your terminal to AMOS command level.

FUNCTION:

Performs initialization for Century Data T80 and T300 hard disks.

HINTS/RESTRICTIONS:

If you are using models T80 or T300 of the Century Data Trident series of hard disks as peripheral disk drives (that is, you are not using these drives as your System Device), you must use the T80INI command every time you turn on or reset the system to initialize the Century Data 1150A Formatter.

If you do not use the T80INI command, the disk drive will not be able to achieve READY status (i.e., a SYSTAT command shows the drive to be NOT READY).

FORMAT:

.T80INI ↵

OPERATION:

1. Type T80INI followed by a RETURN:

   .T80INI ↵

   The T80INI command initializes the 1150A formatter.

2. T80INI returns a numeric code that identifies the status of the formatter. The message takes this form:

   STATUS: n

   where n is a 16-bit status code. For a discussion of the formatter status codes, refer to the documentation accompanying the 1150A formatter.

ERRORS:

T80INI generates no error messages.

CHARACTERISTICS:

Communicates with a Century Data 1150A formatter working with the AM-400 Trident Hard Disk Interface.

Returns your terminal to AMOS command level.

(1 October 1979)

FUNCTION:

Bootstrap loader program for a system that uses the Century Data Trident T80 or T300 hard disk as the System Device.

HINTS/RESTRICTIONS:

The T80LOD program when contained on a 2716 PROM allows the system to boot off a System Disk on a T80 or T300 hard disk when a hardware reset occurs (that is, when you hit the RESET button).

The program is also in account DSK0:[1,4] of the System Disk.

You may use T80LOD at AMOS command level to reset the system if your System Device is a T80 or T300 hard disk. The memory partition of the job that uses the T80LOD command MUST be in Bank Zero if your system bank switches memory. (For information on bank-switched systems, refer to the document Memory Management Option, (DWM-00100-10) in the AM-100 documentation packet.)

FORMAT:

.T80LOD ↵

OPERATION:

1. Type T80LOD followed by a RETURN:

   .T80LOD ↵

   The system now resets itself by reading a copy of the T80LOD bootstrap program into system memory and executing it.

2. Once invoked, the T80LOD program reads the operating system skeleton monitor, DSK0:SYSTEM.MON[1,4], into memory. SYSTEM.MON then brings up the system under the control of your system initialization command file, SYSTEM.INI.

3. Once the system is up and running, you see the AMOS prompt.

ERRORS:

T80LOD generates no error messsages. However, if it cannot find SYSTEM.MON[1,4] and SYSTEM.INI[1,4], the start-up procedure fails.

(1 October 1979)

CHARACTERISTICS:

Boots the system from a Century Data Trident T80 or T300 hard disk if that disk is the System Device.

Returns your terminal to AMOS command level if the system resets successfully.

FUNCTION:

Allows you to temporarily save AMOS command lines and to later invoke them by typing a single character.

HINTS/RESTRICTIONS:

You can use the U command to temporarily save any AMOS command line. To invoke that command line, type a U. For example, rather than typing the command line:

.ERASE *.BAK,*.OBJ,*.LST ↵

every time you want to erase those types of files, you can use the U command to save that command line and (later) to invoke it. U saves the command line in your memory partition (in a memory module named U.SCS), so when you clear your memory partition or turn off the system, the command line you saved is gone.

FORMAT:

.U Command-line ↵

where Command-line is the AMOS command line that you want to save; or:

.U ↵

when you want to invoke the command line that you previously saved.

OPERATION:

1. To save an AMOS command line - Type U followed by the command line. Then type a RETURN. The command line may be any legal AMOS command line (that is, it may contain an AMOS command with file specifications, a command file specification, etc.). For example:

   .U PRINT/COPIES:2 *.LST,*.RST/WAIT/NOBANNER ↵

   U remembers the command line, but does not send it to AMOS. To change the saved command line, use U again and enter a new command line.

2. To invoke a command line that you've saved - Type U followed by a RETURN. U sends to AMOS the command line you've previously saved.

3. To save more than one command line at a time - Make copies of the U.PRG program under different names; then you can use those programs to save command lines too.

   For example, if you make a copy of U.PRG under the name A.PRG, you can save an AMOS command line by typing:

   .A Command-line↵

   The command line is now temporarily saved in the memory module A.SCS. To invoke it, type:

   .A ↵

   You may copy U.PRG under any name you choose; the new program builds a memory module to hold the command line you save that has the same name as the program with an .SCS extension.

ERRORS:

[NO PREVIOUS COMMAND]
You've typed U and a RETURN, but you have not previously saved a command line; U has no command line to repeat back to you.

CHARACTERISTICS:

You can save as many command lines as you wish by making copies of U.PRG under different names, then using those copies of the program.

When you turn off or reset the system or when you erase all modules from your memory partition, the command lines you've saved disappear.

The U command returns your terminal to AMOS command level.

FUNCTION:

Creates and edits text files.

HINTS/RESTRICTIONS:

VUE is a screen-oriented text editor. You see the text you are editing displayed on the screen, and move the screen cursor to the location in your file that you want to change.

In Command mode, VUE allows you to search for specific strings of text, perform local and global replacements, move and delete blocks of text, and change various editing parameters. VUE is re-entrant and may be loaded into system memory by the System Operator.

VUE copies into memory the file you want to edit and makes a backup file by renaming the disk file to a .BAK extension. When you exit VUE, the new, edited version of the file goes out to disk under the original name and extension of the file. You may edit a file too big to fit into your memory partition.

A VUE initialization file, INI.VUE, allows you to set your own default editing parameters. Refer to the manual AlphaVUE User's Manual, (DWM-00100-15, Revision B00), for information on using VUE and setting up INI.VUE.

NOTE: To exit VUE, enter Command mode. (If you see a screenful of asterisks or if you see your text, you are in Screen mode. Type an Escape to enter Command mode.) Now type an F followed by a RETURN to exit and update your file or type a Q followed by a RETURN if you want to exit without updating.

FORMAT:

.VUE Filespec RET

where Filespec selects the file you want to edit.

DEFAULTS:

The defaults VUE uses (for example, the default file extension), are set in the VUE initialization file, INI.VUE.

OPERATION:

1. Type VUE followed by the specification of the file you want to edit. Then type a RETURN. For example:

(Changed 1 May 1980)

.VUE XREF.BAS (RET)

2. If the file you specify (in this case, XREF.BAS) does not yet exist, you see:

XREF.BAS DOES NOT EXIST - CREATE IT?

Enter a Y for Yes or an N for No. If you enter N, VUE returns your terminal to AMOS command level. If you enter Y, you see a screenful of asterisks; this tells you that VUE has created an empty file, and that you can start to enter text. Just start typing, hitting a RETURN when you want to start a new line. You must not enter more than 510 characters between RETURNS.

3. If you are editing an existing file, VUE loads a copy of that file into memory and takes you directly into Screen mode. Use the cursor control keys (labeled with arrows) to move the cursor to the point in your text where you want to make changes.

4. To leave Screen mode and enter Command mode, type an Escape. Now you see several lines of data. The cursor is waiting at the VUE prompt symbol, >. This display might look something like this:

AlphaVue 2.4 Status: space insert fold Sblk

Editing XREF.BAS 34875 bytes free Margin 0 Page 3

>

Below this display you might also see a summary of VUE editing commands. (If the file MENU.VUE exists in account DSK0:[7,0] or the account you are currently logged into.) You may now use various Command mode commands. To re-enter Screen mode, type an Escape.

5. To exit VUE, enter Command mode and type an F followed by a RETURN. To exit without updating your file, enter Command mode and type a Q followed by a RETURN.

COMMAND SUMMARY:

Below is a partial list of the VUE commands; refer to the VUE manual for a full list of commands. Some terminals allow you to use the cursor control keys (marked with arrows) to move the cursor; others require that you use Control-J, Control-K, Control-H, and Control-L to move the cursor. You may also use the RUBOUT key to delete characters.

Control-A	Move cursor to front of last word
Control-W	Move cursor to front of next word
Control-U	Move cursor to beginning of line
Control-N	Move cursor to end of line
Control-R	Move cursor to last page
Control-T	Move cursor to next page
Control-E	Move cursor to end of file
HOME	Move cursor to beginning of file
Control-B	Insert blank line
Control-F	Insert space
Control-O	Append lines
Control-D	Delete character and space
Control-Y	Delete text on line to end of line
Control-Z	Delete line (including RETURN at end of line)
Control-RUBOUT	Same as Control-U followed by a Control-Y.
Escape	Enter Command or Screen mode (toggle mode)

In Command mode:

F	Finish and update	S text	Search for string
Q	Finish without updating	R text	Search and Replace
COPY	Copy block of text	DELETE	Delete block of text

ERRORS:

If you specify an invalid command, you can see the usual system messages (e.g., file not found). VUE itself can generate several error messages:

!!!Insufficient space to complete transfer!!!
Your file cannot fit into memory. Use the Unyank command to write out the front part of the file to disk. Then you can use the Yank command to bring in more text.

Line over 510 characters long!!
You cannot enter a text line of more than 510 characters. Use the Control-B command to break up the line.

Disc write-protected, please unprotect it
VUE cannot transfer your edited file to the disk. Write-enable the disk and try again.

(Changed 1 May 1980)

If for some reason VUE cannot transfer your file to the disk, it names the module in memory IMAGE.VUE. (You can use SAVE to save the module as a file in the account and device you are logged into; then you can edit it.) You see the message:

MEMORY IMAGE STORED IN MEM:IMAGE.VUE

NOTE: IMAGE.VUE contains no line-feeds. When you re-VUE the file after you have saved it via SAVE, VUE re-inserts line-feeds in the proper places in the file.

CHARACTERISTICS:

Screen-oriented text editor.

Creates backup file with .BAK extension.

Returns your terminal to AMOS command level.

wait

FUNCTION:

Allows you to delay the execution of any programs by your job until another job completes its current task.

HINTS/RESTRICTIONS:

A job's task is said to be completed when that job enters one of the following states: terminal input, sleep or external wait status.

FORMAT:

.WAIT Jobname ↵

OPERATION:

1. Type WAIT, the name of the job for whom you want to wait, and a RETURN. For example:

   .WAIT JOB3 ↵

2. When the specified job finishes its current task, you see the AMOS prompt. Your job now proceeds.

ERRORS:

WAIT displays no error messages.

CHARACTERISTICS:

Delays the execution of any program by your job until the specified job has finished its current task.

Returns your terminal to AMOS command level.

(1 October 1979)

wnglod

FUNCTION:

Bootstrap loader program for a system that uses the Wangco floppy disk as the System Device.

HINTS/RESTRICTIONS:

The WNGLOD program when contained on a 2716 PROM allows the system to boot off a System Disk on a Wangco floppy disk when a hardware reset occurs (that is, when you hit the RESET button).

The program is also in account DSK0:[1,4] of the System Disk.

You may use WNGLOD at AMOS command level to reset the system if your System Device is a Wangco floppy disk drive. The memory partition of the job that uses the WNGLOD command MUST be in Bank Zero if your system bank switches memory. (For information on bank-switched systems, refer to the document Memory Management Option, (DWM-00100-10) in the AM-100 documentation packet.)

You may use WNGLOD either to boot from an AMS- or STD-format diskette.

FORMAT:

.WNGLOD ↵

OPERATION:

1. Type WNGLOD followed by a RETURN:

   .WNGLOD ↵

   The system now resets itself by reading a copy of the WNGLOD bootstrap program into system memory and executing it.

2. Once invoked, the WNGLOD program reads the operating system skeleton monitor, DSK0:SYSTEM.MON[1,4], into memory. SYSTEM.MON then brings up the system under the control of your system initialization command file, SYSTEM.INI.

3. Once the system is up and running, you see the AMOS prompt.

ERRORS:

WNGLOD generates no error messages. However, if it does not find SYSTEM.MON[1,4] and SYSTEM.INI[1,4], the start-up procedure fails.

(1 October 1979)

CHARACTERISTICS:

Boots the system from an AMS- or STD-format Wangco floppy disk if the Wangco disk drive is the System Device.

Returns your terminal to AMOS command level if the system resets successfully.

wng210

FUNCTION:

Bootstrap loader program for a system that uses the Wangco floppy disk as the System Device running under the AM-210 floppy disk controller.

HINTS/RESTRICTIONS:

The WNG210 program when contained on a 2716 PROM allows the system to boot off a System Disk on a Wangco floppy disk when a hardware reset occurs (that is, when you hit the RESET button).

The program is also in account DSK0:[1,4] of the System Disk.

You may use WNG210 at AMOS command level to reset the system if your System Device is a Wangco floppy disk drive. The memory partition of the job that uses the WNG210 command MUST be in Bank Zero if your system bank switches memory. (For information on bank-switched systems, refer to the document Memory Management Option, (DWM-00100-10) in the AM-100 documentation packet.)

You may use WNG210 to boot from double- or single-sided diskettes that are in single- or double-density AMS format.

FORMAT:

.WNG210 (RET)

OPERATION:

1. Type WNG210 followed by a RETURN:

   .WNG210 (RET)

   The system now resets itself by reading a copy of the WNG210 bootstrap program into system memory and executing it.

2. Once invoked, the WNG210 program reads the operating system skeleton monitor, DSK0:SYSTEM.MON[1,4], into memory. SYSTEM.MON then brings up the system under the control of your system initialization command file, SYSTEM.INI.

3. Once the system is up and running, you see the AMOS prompt.

ERRORS:

WNG210 generates no error messages. However, if it does not find SYSTEM.MON[1,4] and SYSTEM.INI[1,4], the start-up procedure fails.

(1 May 1980)

CHARACTERISTICS:

Boots the system from a Wangco floppy disk if the Wangco disk drive is the System Device and runs under control of the AM-210 floppy disk controller.

Returns your terminal to AMOS command level if the system resets successfully.

FUNCTION:

Allows you to position the screen cursor on your terminal display.

HINTS/RESTRICTIONS:

To be able to use the XY command, your terminal driver (the program that takes care of the screen-positioning functions of your terminal) must allow use of TCRT screen calls. Use the XY command at the AMOS command level either directly or within a command file. You will probably find this command most useful within a command file; you can use XY to help you position the display of your command file messages.

FORMAT:

.XY Row-number Column-number ↵

where Row-number is the horizontal position and Column-number is the vertical position of the cursor on the screen. The Row-number may be between 1 and 24. The Column number may be between 1 and 80.

.XY = Screen-function ↵

where Screen-function is the number of the screen function that you want to perform. See page 59 of the AlphaBASIC User's Manual, (DWM-00100-01), for a list of the screen functions you can use. (Also see page 2 of the Addendum at the back of that manual.)

OPERATION:

1. Type XY followed by the numbers of the screen row and column where you want to position the cursor; next type a RETURN. Example:

   .XY 12 40 ↵

   This positions the cursor to the 12th line on the screen and the 40th character position (about the middle of the screen on most terminals).

2. Type XY, an equal sign, and the number of the screen function you want to perform. Then type a RETURN. Example:

   .XY = 0 ↵

   This uses the screen function #0 to clear the screen.

ERRORS:

XY generates no error messages. If you specify an invalid function number or row and column numbers that are out of range, XY generally ignores your command or performs some random function. So, be careful to check for validity the values that you give XY.

CHARACTERISTICS:

Requires that the driver program for your terminal be able to use TCRT screen calls.

Returns your terminal to the AMOS command level.

APPENDIX A - THE ASCII CHARACTER SET

APPENDIX A

THE ASCII CHARACTER SET

Many of the commands in this manual display data in numeric form. In the case of text files that contain data encoded in ASCII, these commands display the ASCII values of the data. (ASCII, American Standard Code for Information Interchange, provides a standard set of values for representing text characters in numeric form.)

The next few pages contain charts that list the complete ASCII character set. We provide both the octal and hexadecimal representations of the ASCII values. The number base in which you see data displayed is usually octal, but you can tell the system to use hexadecimal as your display base. (See the SET reference sheet.)

Note that the first 32 characters are non-printing Control-characters.

THE CONTROL CHARACTERS

CHARACTER	OCTAL	HEX	MEANING
NULL	000	00	Null (fill character)
SOH	001	01	Start of Heading
STX	002	02	Start of Text
ETX	003	03	End of Text
ECT	004	04	End of Transmission
ENQ	005	05	Enquiry
ACK	006	06	Acknowledge
BEL	007	07	Bell code
BS	010	08	Back Space
HT	011	09	Horizontal Tab
LF	012	0A	Line Feed
VT	013	0B	Vertical Tab
FF	014	0C	Form Feed
CR	015	0D	Carriage Return
SO	016	0E	Shift Out
SI	017	0F	Shift In
DLE	020	10	Data Link Escape
DC1	021	11	Device Control 1
DC2	022	12	Device Control 2
DC3	023	13	Device Control 3
DC4	024	14	Device Control 4
NAK	025	15	Negative Acknowledge
SYN	026	16	Synchronous Idle
ETB	027	17	End of Transmission Blocks
CAN	030	18	Cancel
EM	031	19	End of Medium
SS	032	1A	Special Sequence
ESC	033	1B	Escape
FS	034	1C	File Separator
GS	035	1D	Group Separator
RS	036	1E	Record Separator
US	037	1F	Unit Separator

PRINTING CHARACTERS

CHARACTER	OCTAL	HEX	MEANING
SP	040	20	Space
!	041	21	Exclamation Mark
"	042	22	Quotation Mark
#	043	23	Number Sign
$	044	24	Dollar Sign
%	045	25	Percent Sign
&	046	26	Ampersand
'	047	27	Apostrophe
(	050	28	Opening Parenthesis
)	051	29	Closing Parenthesis
*	052	2A	Asterisk
+	053	2B	Plus
,	054	2C	Comma
-	055	2D	Hyphen or Minus
.	056	2E	Period
/	057	2F	Slash
0	060	30	Zero
1	061	31	One
2	062	32	Two
3	063	33	Three
4	064	34	Four
5	065	35	Five
6	066	36	Six
7	067	37	Seven
8	070	38	Eight
9	071	39	Nine
:	072	3A	Colon
;	073	3B	Semicolon
<	074	3C	Less Than
=	075	3D	Equal Sign
>	076	3E	Greater Than
?	077	3F	Question Mark
@	100	40	Commercial At

CHARACTER	OCTAL	HEX	MEANING
A	101	41	Upper Case Letter
B	102	42	Upper Case Letter
C	103	43	Upper Case Letter
D	104	44	Upper Case Letter
E	105	45	Upper Case Letter
F	106	46	Upper Case Letter
G	107	47	Upper Case Letter
H	110	48	Upper Case Letter
I	111	49	Upper Case Letter
J	112	4A	Upper Case Letter
K	113	4B	Upper Case Letter
L	114	4C	Upper Case Letter
M	115	4D	Upper Case Letter
N	116	4E	Upper Case Letter
O	117	4F	Upper Case Letter
P	120	50	Upper Case Letter
Q	121	51	Upper Case Letter
R	122	52	Upper Case Letter
S	123	53	Upper Case Letter
T	124	54	Upper Case Letter
U	125	55	Upper Case Letter
V	126	56	Upper Case Letter
W	127	57	Upper Case Letter
X	130	58	Upper Case Letter
Y	131	59	Upper Case Letter
Z	132	5A	Upper Case Letter
[	133	5B	Opening Bracket
\	134	5C	Back Slash
]	135	5D	Closing Bracket
^	136	5E	Circumflex
_	137	5F	Underline
`	140	60	Grave Accent
a	141	61	Lower Case Letter
b	142	62	Lower Case Letter
c	143	63	Lower Case Letter
d	144	64	Lower Case Letter
e	145	65	Lower Case Letter
f	146	66	Lower Case Letter
g	147	67	Lower Case Letter
h	150	68	Lower Case Letter
i	151	69	Lower Case Letter
j	152	6A	Lower Case Letter
k	153	6B	Lower Case Letter
l	154	6C	Lower Case Letter
m	155	6D	Lower Case Letter
n	156	6E	Lower Case Letter
o	157	6F	Lower Case Letter

CHARACTER	OCTAL	HEX	MEANING
p	160	70	Lower Case Letter
q	161	71	Lower Case Letter
r	162	72	Lower Case Letter
s	163	73	Lower Case Letter
t	164	74	Lower Case Letter
u	165	75	Lower Case Letter
v	166	76	Lower Case Letter
w	167	77	Lower Case Letter
x	170	78	Lower Case Letter
y	171	79	Lower Case Letter
z	172	7A	Lower Case Letter
{	173	7B	Opening Brace
|	174	7C	Vertical Line
}	175	7D	Closing Brace
~	176	7E	Tilde
DEL	177	7F	Delete