ALPHA MICROSYSTEMS
RIGHT. FROM THE START

AMOS User's Guide

DSM-00042-02

REVISIONS INCORPORATED	
REVISION	DATE
00	March 1988
01	August 1988
02	April 1991

AMOS User's Guide
To re-order this document, request part number DSO-00042-00.

This document applies to AMOS versions 2.2 and later.

AMIGOS	AMOS	Alpha Micro	AlphaACCOUNTING
AlphaBASIC	AlphaCALC	AlphaCOBOL	AlphaFORTRAN 77
AlphaLAN	AlphaLEDGER	AlphaMAIL	AlphaMATE
AlphaNET	AlphaPASCAL	AlphaRJE	AlphaWRITE
CASELODE	OmniBASIC	VER-A-TEL	VIDEOTRAX

AlphaBASIC PLUS	AlphaVUE	AM-PC	AMTEC
DART	ESP	MULTI	inSight/am
inFront/am			

ALPHA MICROSYSTEMS
P.O. Box 25059
Santa Ana, CA 92799

PREFACE

The *AMOS User's Guide* contains instructions for the operation and use of your Alpha Micro computer system. Not only does it contain guidelines for operating the various pieces of equipment attached to your system, but also rules for entering the many available AMOS commands, creating your own commands, and manipulating files and data sets.

When you're through with this book, you can move on to more specialized documentation (on programming languages, word processing, and much more) save the Alpha Micro Documentation Library.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION TO THIS MANUAL

The first book in this series, the *Introduction to AMOS*, gave you the theoretical background you need to understand why the computer acts the way it does, and why you have to treat it the way you do. This manual gives you the opportunity to put the theory into practice.

The topics you'll read about in this chapter are:

- The Contents of This Manual
- The AMOS Operating System
- A List of Common Abbreviations and Symbols

1.1 WHAT'S IN THIS BOOK

This book is a step by step guide to AMOS, the Alpha Micro Operating System. It is necessarily general since there is such a wide variety of possible ways to configure your Alpha Micro computer system. But whatever size computer you have, and whatever types of equipment you have attached to your system in the way of terminals, disk units, tape units, and so on, the same basic principles apply.

First, we'll tell you how to start up the computer in the morning and turn it off at night. Then, how to use your terminal, how to mount and unmount disks, how to request system information from the computer, and how to use the various commands that come with the system. Next, we'll describe how you can store, manipulate, and retrieve data in the form of files.

In Part 2, you'll find out how to make things even easier for yourself by using the wildcard features of certain commands, how to rearrange the data in your files, and how to create duplicate or backup copies of your data. We also describe how you can use different disk accounts to your advantage, how to protect them from unauthorized eyes, and how to construct special files of system commands.

You will also get a brief introduction to several programming languages you can use to write your own programs, and finally, we'll point you in the direction of additional documentation you can read to get more in-depth information on many of the topics we've covered only briefly in this manual.

But before we go any further, let us introduce you to AMOS.

1.1.1∞AMOS

A computer can do nothing without a set of instructions to tell it what to do. The main set of instructions, or program, that controls the entire operation of the computer system is the System Monitor. As you learned in *Introduction to AMOS*, all the other programs that run on your computer are governed by this monitor program.

AMOS is the Alpha Micro Operating System, and it's designed to work with the entire range of Alpha Micro Computers, from the smallest to the largest. As a result of differences in hardware design, you may find that your computer's displays are slightly different from the ones we use as examples. Other differences, such as the file extensions used for certain files, are pointed out as we go along.

1.2∞SYMBOLS AND ABBREVIATIONS

Like all the other manuals in the Alpha Micro Documentation Library, this manual contains a number of standardized symbols and abbreviations that make the examples easier to read and understand.

SYMBOL	MEANING
devn:	Device-Name. The "dev" is the three letter physical device code, and the "n" is the logical unit number. Examples of device names are DSK0:, DSK5:, WIN1:, and MTU0:. Usually, device names indicate disk drives, but they can also refer to magnetic tape drives and video cassette recorders.
filespec	File Specification. A file specification identifies a specific file within an account. A complete filespec is made up of the devn:, the filename, the file extension, and the project-programmer number. For example: devn:filename.ext[p,pn] -or- DSK0:SYSTEM.INI[1,4]
[p,pn]	This abbreviation represents an account on a disk that you can store files and data in. An actual disk account number looks like this: [100,2] or [1,4]. Disk account specifications are sometimes referred to as "Project-programmer numbers."

SYMBOL	MEANING
{°}	Braces are used in some examples to indicate optional elements of a command line. In the example: **`DIR{/switch}`** the braces tell you that "/switch" is not a required portion of the DIR command line.
/	The slash symbol precedes a command line switch or "option request." For example: **`DIR/WIDE:3`**RETURN This command requests a directory display of the disk account you are currently logged into. The switch (/WIDE:3) indicates that you want the display to be three columns wide.
TEXT	Bold text in an example of user/computer communication represents the characters you type.
`text`	Text like this in an example of user/computer communication represents the characters the computer displays on your terminal screen.
.	The AMOS prompt symbol. Most of the time you'll see the AMOS prompt shown as a dot (.), although you can change the prompt symbol to something else, if you wish. The prompt is the signal AMOS gives you that it's ready to perform your commands.
KEY	In our examples, the keycap symbol appears whenever you need to press a certain key on your terminal keyboard. The name of the key you need to press appears inside the keycap symbol, like this: RETURN. If you need to press the TAB key, you would see TAB, or the ESCAPE key, ESCAPE. (Sometimes the ESCAPE key is labeled ESC or ALT MODE.)
^	This symbol in front of a capital letter means the letter is a "control character." That is, you held down the CTRL key on your terminal keyboard while you typed the letter. For example, when you press CTRL/C, it appears on your screen as `^C`. (^C is the control character used like the CANCEL key to cancel most programs and return you to AMOS command level.) You'll find out more about control characters in Chapter 2.
✋	This symbol means "halt!" It indicates an important note you should read carefully before going further in the documentation. Usually, text next to this symbol contains instructions for something you MUST or MUST NOT do, so read it carefully.

SYMBOL	MEANING
✋	This symbol means "hint." It indicates a helpful bit of information.
☝	This symbol means "remember." It indicates something you should keep in mind while you are following a set of instructions.

But before you can start writing your own programs, entering your own commands, and creating your own files, you have to know how the equipment operates. The next chapter tells you everything you need to know.

PART 1

Getting Your Hands On the System

CHAPTER 2

THE EQUIPMENT

In this chapter, you will find out how to get started using your Alpha Micro computer. Before you can begin using the computer for programs and text processing, you have to know a bit about the equipment you are going to use.

Computers are generally discussed in terms of **hardware** and **software**. The term "hardware" refers to the machines and devices—the computer, your terminal, the printer, disk drives, etc.

"Software" refers to the sets of coded instructions that tell the hardware what to do. A set of instructions that performs a specific task or a series of related tasks is a program. Software is further divided into system software which supervises the general operation of the equipment and controls the running of other programs, and application software which performs the routine processing of data under the control of the system software.

This chapter will show you how to:

- Find out your user name and disk accounts.
- Turn your system (your hardware) on and off.
- Use your terminal.
- Log on and off the computer.
- Use your keyboard.
- Change disk cartridges and floppy diskettes.

2.1 FINDING OUT YOUR USER NAME

Before you begin to use your computer system, consult with the person designated as your System Operator. Your System Operator will give you a user name to use when you log onto the system. You may be assigned a user password if required. Your System Operator will add your user name to the list of valid users of the system.

You will be assigned a root account which will be your central or home disk account. Other accounts on the system will also be allocated for you. These project accounts will be yours to use when you log into the system and when you are creating files. Your System Operator will assign you as many project accounts as you need.

2.2∞TURNING THE SYSTEM ON

Below are explanations for turning on most of the equipment you will need. The owner's manual for your computer contains additional information about the operation of your particular model.

Check with your System Operator or refer to the instructions that came with each device for information on how to turn the devices on and off. The procedure is different for different brands of devices. In this manual, we cannot detail all of the procedures for all of the equipment that we support, but each device should come with complete instructions for use.

The procedure for turning on your system will depend on what types of devices are on your particular system. Since Alpha Micro supports many different types of storage devices (such as Winchester disks, floppy disks, and video cassettes), you will have to adjust your procedure according to the needs of the specific devices you have.

When you come to the point of turning on your disk drive, use the instructions for the type of system that you have. Generally, you should turn the devices on in the following order:

YOUR TERMINAL

There are several things you can do to extend the life of your terminal. First, you can make your terminal last longer and be more reliable if you don't turn it off and on frequently. The surges of electricity that occur when it's turned on and off might eventually cause certain components to wear out. Also, you can turn the screen contrast down so that the characters don't become etched on the screen. It is a good idea, too, to clear the screen of characters when you are going to be away from the terminal for any length of time. To do this, make sure that you are at AMOS command level, and press the "CLEAR" or "CLEAR ALL" key, or press RETURN until the screen clears.

If you do turn your terminal off, it is best to do so after the computer itself has been shut off, and to turn it on again before turning on the computer. Turning a terminal on or off while the computer is powered on can sometimes cause electrical "noise" on the connecting cable that might interfere with your computer.

If your terminal has a REMOTE/LOCAL switch, the switch must be turned to REMOTE in order to communicate with the computer. If it has a HALF DUPLEX/FULL DUPLEX switch (sometimes labeled just HALF/FULL), turn that to FULL DUPLEX.

THE PRINTER

If you are going to be using a printer, turn it on. Like terminals, you might have problems if you turn your printer on or off while the computer is on. You may want to turn off your printers if you also turn off your computer.

If it has a REMOTE/LOCAL switch, turn the switch to REMOTE. If it has an ONLINE button or switch, turn that to ONLINE.

OTHER DEVICES

If you have any other devices on your system, such as magnetic tape drives, video cassette recorders, modems, etc., turn them on now. Check the instructions that came with the device to find out the procedure for turning them on.

POWERING UP DISK DRIVES

There are many different configurations of Alpha Micro computers available. In some of these configurations, the disk drives are powered up when you turn on the power to the computer main enclosure; other configurations require that you power up the disk drives separately.

If your disk drives power up separately from the computer, turn them on before you turn on the computer. In fact, with many disk drives, you may wish to keep them powered up all the time. For example, we recommend that you leave power on to your Winchester disks.

If you are going to be using floppy-disk drives in addition to your hard-disk drives, turn them on now.

As a general rule, floppy disk drives are the last components of the system to be powered up, and the first items to be shut off.

THE COMPUTER

Turn on the power to your computer. With most Alpha Micro systems, you have to hold down the RESET button while you turn on the power. In all cases, the RESET button is on the computer's front panel.

In many Alpha Micro systems, your Winchester or floppy disk drives will be powered up automatically when you turn on the power to your computer.

You may find that you do not want to turn your computer off very often. Since the Video Cassette Recorder software and the Task Manager are time controlled, you may find it convenient to schedule tasks and file-backups to run in the evening.

It generally does not harm the computer to leave it on for long periods of time. And, in fact, turning power on and off frequently to electronic circuits creates more wear and tear than if you leave them on continuously.

FLOPPY DISK DRIVES

After the computer and floppy disk drives are powered up:

If your system boots from a floppy disk drive, insert your System Disk into drive zero (the System Drive). (The System Disk is the disk that contains the programs that comprise AMOS.) Then close the drive door and push the RESET button.

If you have more than one drive in your disk unit, you can go ahead and insert other floppy disks that you are going to use. (Refer to Section 2.6, "Changing Disk Cartridges and Floppy Disks.")

Never turn the power on or off while there are floppy disks in the drive.

2.3∞TURNING THE SYSTEM OFF

If you have a floppy disk unit, you should remove the diskettes when the system is not in use. (You should remove and insert floppy diskettes only when power is on.) These measures help to protect your stored data from being damaged, in case of power surges or failures.

Follow the procedures below (which also tell you how to spin down the disk drives).

One of the reasons for turning off the power is if you are going to move the computer equipment. You MUST turn off the power any time you move the equipment, or you may damage your disk storage and/or your Central Processing Unit. Users of AM-1000 and AM-1200 desktop systems should especially take note of this warning, since the these computers are lightweight and easy to move.

Before you turn off your computer or disk drives, make sure that everyone using the system is at AMOS command level, and make sure that AMOS is not in the process of transferring data between the computer and the disk.

The procedures for shutting off both Hard disk and Floppy disk systems follow. Use the instructions for whichever type of system you have.

HARD DISK SYSTEMS

1. On some disks, such as the Winchesters, you have to spin down the disk. Use the MOUNT command on your terminal (with the proper option for your system) to unmount the disk. Refer to the instructions shipped with your drive for the exact power-down procedure for your disk drive.

   In some cases, powering down the drives and shutting off the power to your computer are done in the same step. If this is true for your system, just go on to the next step.

2. Turn off the power to your computer.

FLOPPY DISK SYSTEMS

1. Remove and store the floppy disks that are in the drives.
2. Turn off the power to your floppy disk drives.
3. Turn off the power to your computer. In many cases, turning off the power to the disk and the computer is done in one step.

When these major components of your system have been powered down, you can turn off your terminals and printer.

2.4 WHAT IS A TERMINAL?

A terminal is the device you use for communicating with AMOS. It has a keyboard upon which you type commands and enter data to the computer. The terminal also has some method of displaying to you what you type in and what is printed out by the computer.

Since the terminal displays its output on a video screen, it’s called a Cathode Ray Tube Terminal (CRT), or a Video Display Terminal (VDT).

Section 2.4.1 will explain how to use your terminal.

2.4.1 Prompt and Cursor Symbols

When your system is turned on, you should see a prompt symbol on the left side of your terminal screen. The usual prompt symbol is a dot (.), and indicates that AMOS is waiting for a command. AMOS allows you to define your own prompt symbol, up to 19 characters long, using the SET PROMPT command—so, you may see a different prompt.

If you do not see a prompt symbol, type a CTRL/C (hold down the CTRL key on the keyboard and type a C).

You should now see an AMOS prompt on the screen.

If you *still* do not see the AMOS prompt, check to see that your terminal is on, and that the cables running between the computer and the terminal are firmly in place. If these are okay and you still do not see the prompt symbol, see your System Operator for advice.

When the AMOS prompt is displayed, we say that you are at "AMOS Monitor command level"; that is, you are communicating directly with AMOS.

At other times you may see other prompt symbols that indicate that you are communicating with a program that AMOS is executing.

For example, when you use the text editing program, AlphaVUE, you see the AlphaVUE prompt symbol: >. At that point you must enter commands that the AlphaVUE program understands.

When you exit AlphaVUE, you return to AMOS command level. (You see the AMOS prompt again. The prompt can be defined by you, but starts out as a dot.) The various prompt symbols remind you which program you are communicating with.

Remember: To enter AMOS commands, you must be at AMOS command level.

If you are using a CRT or VDT terminal, you also see another symbol, the cursor, next to the prompt symbol. The cursor may be a small white rectangle, a triangle, a blinking line, etc., depending on the type of terminal you have.

The cursor always marks your current screen position. Any characters that you type appear at that position on the screen. As you type, the cursor moves forward as the new characters appear on the screen.

2.5∞LOGGING ON AND OFF THE SYSTEM

Now you are ready to sit down at your terminal and begin working. The first procedure you need to know is how to log onto and log off the computer. The commands are LOG and LOGOFF. On a user names system, you can use LOGON in addition to LOG. Let’s go over the two ways and you’ll soon see how easy it is to sign on using your user name.

Logging On With The LOG Command

Once you see the AMOS prompt, you are ready to log onto the system. Type LOG and your user name, and press the RETURN key. If required, LOG asks for your user password. For example:

```
LOG BOB SMITH RETURN
User Password:
```

After checking your identification, the system then logs you into your root account:

```
Logged into DSK2:[200,5]
```

Before going on to the LOGON command below, log off the system. Type LOGOFF. (We'll discuss LOGOFF in more detail in the next section.)

If you are on a system that does not support user names, you may use the LOG command by specifying the disk account you want.

Type LOG, the device and account number you are going to use, and press the RETURN key. For example:

```
LOG DSK2:[200,5] RETURN
```

The device this account is on is DSK2:, and the [200,5] is the number of the disk account you want to work in. Your System Operator should have already assigned you one or more of these account numbers for your use. If this account is protected with a password, LOG prompts you for it:

```
LOG DSK2:[200,5] RETURN
Password:
```

LOG verifies your identification and logs you in to the requested account.

Logging On With The LOGON Command

LOGON automatically logs you in to your root account. If you want to retype or ask for HELP or use the function keys while in LOGON's menu display, you are allowed to do so.

To sign on, type LOGON and press the RETURN key.

```
LOGON RETURN
```

The program then displays a menu request for your user name and user password:

```
Please enter your user name and press RETURN:
Please enter password and press RETURN:
```

Enter your response starting at the column marked by the blinking cursor.

After checking your identification, LOGON verifies that you are now in your root account. For example, if your root account is DSK2:[200,5], LOGON displays:

```
Logged into DSK2:[200,5]
```

Now you are ready to begin processing data on your computer system. But before we get into using the computer, you should know more about the LOGOFF command.

Logging Off With The LOGOFF Command

You should log off your computer whenever you are going to be away from your terminal for an extended period of time. If you leave your terminal logged into an account while you are gone, someone else could gain access to your files.

The LOGOFF process looks like this:

```
LOGOFF RETURN
User Bob Smith logged off on Monday, May 9, 1988
1199 disk reads, 120 disk writes
CPU time: 00:02:07.9, Connect time: 04:31:33
```

You are now off the system. If you try to perform most commands or access any programs when you are not logged in, you will see:

```
?Login please
```

To get back on the system, use the LOGON or LOG command again.

Once you are logged in, try typing a few characters on the keyboard. As you type, the system displays the characters on your terminal display.

If you should see each character displayed twice on the screen as you type, check your terminal for a switch labeled FULL/HALF or FULL DUPLEX/HALF; turn the switch to FULL or FULL DUPLEX.

Now you are ready to begin entering commands and instructions on your keyboard.

2.5.1∞Using Your Keyboard

The first step in communicating with AMOS is typing your instructions on the terminal keyboard. The keyboard is very similar to that of a standard typewriter, but you will find a few extra keys that have special functions.

Take a moment to look at the keyboard of your terminal so that you can easily locate these keys later:

RETURN (or RET)	The carriage return key. Just as you press the carriage return on a typewriter to begin a new line on the page, a RETURN tells AMOS that you are ending a line of input and that you want to begin a new line. AMOS won't process an instruction from you until you press RETURN to let it know that you are finished with that line.
RUBOUT°(or RUB, DELETE,°or°DEL)	The deletion key; it backspaces AND deletes. If you make a mistake while typing an instruction to AMOS, you can erase it by using the RUB key. Press the RUB key to remove the last character you typed. If you are using a CRT terminal, you will see the cursor move to the left, erasing the character in its new position. Keep pressing the RUB key until the incorrect characters are gone, then type the correct characters.
SHIFT	The shift key on most keyboards acts much like the shift key on a typewriter. By holding down the shift key, you can type upper case letters and the symbols that occur on the upper half of the keys that bear two symbols.
ALPHA (or CAPS)	While the ALPHA key is locked down, all of the letters you type appear as upper case letters. The keys that have two symbols on them, however, are not affected by the ALPHA key on most keyboards. That is, to type a %, you must hold down the shift key while you press the 5 key, even if the ALPHA key is down. (NOTE: Not all terminals have ALPHA keys.)

ESCAPE°(or ESC) or°ALT)

The ESC key is used with several applications on the system to signal the end of input, or to switch between command modes; however, you do not use ESC at AMOS command level.

CONTRL (or CTRL)

The CTRL key is used in conjunction with other keys to give you the ability to enter a different kind of character—a Control-character.

To enter a Control-character, hold down the CTRL key while you press another key. For example, to type a CTRL/C, hold down the CTRL key while you type a C.

2.5.2∞Control Characters

Control characters allow you to perform special functions. The following list contains the most important control characters. See Appendix D for a list of all the Control characters and their function.

CTRL/C

The CTRL/C is the system interrupt command. Use it to interrupt whatever program is in progress and return to AMOS command level. After typing CTRL/C to interrupt a program, you cannot resume execution of that program; you must start it over from the beginning.

You may usually interrupt programs even when they are displaying data.

Some programs, such as AlphaVUE, do not recognize a CTRL/C as an exit command; instead you must use the exit command for that program if you want to return to AMOS command level.

Other programs do recognize a CTRL/C however, if an exit command exists for a program, it's usually better to use that command than to enter a CTRL/C. Many programs perform various closing functions when you use their normal exit commands and would not have a chance to perform those procedures for an orderly exit if you bypass them by using CTRL/C.

CTRL/U

At AMOS command level, you can return to the beginning of the command line you are typing by entering a CTRL/U.

The cursor moves to the left edge of the display and waits for you to retype the command.

CTRL/I

The tab character. (Many terminals have a TAB key that you can use instead of typing CTRL/I.) A tab moves the cursor to the next tab stop on your terminal display.

CTRL/S

If you have a CRT or VDT terminal, it often happens that a program or command displays more data on your terminal than will fit on one screen-page. To stop the screen display, type a CTRL/S.

You may now read the data on the screen at your leisure. Not only does the display freeze, but AMOS actually stops sending data to your terminal until you type a CTRL/Q (see below); at that point, AMOS resumes sending information where it left off.

While a CTRL/S is in effect, AMOS stores (but does not act upon) anything that you type except for a CTRL/Q. There is, however, a limit to how much can be stored. The exact number of characters depends upon your initial system set-up.

CTRL/Q

When you type a CTRL/S (described above) to freeze the screen display, you must type a CTRL/Q to resume the screen display. If you have typed anything while the CTRL/S was in effect, a CTRL/Q tells AMOS that it can now go ahead and act upon that input.

Try this sample. Enter a CTRL/S, then type DIR and press RETURN, and then PRNT and press RETURN. The commands aren't displayed on the screen and it looks like nothing happened. Now use CTRL/Q to release the display, and you will see first a list of the files in your account printed on the screen, and then a display of the files waiting to be printed. (You'll learn more about the DIR and PRNT commands later on.)

CTRL/R

If the Command Line Editor is active on your terminal, you have the ability to reprocess previously typed commands by pressing CTRL/R. Usually, you can only go back two or three commands. If you make a typing mistake while entering a command and get an error message as a result, press CTRL/R to re-display the command. Now you can correct the error, press RETURN, and the command will execute correctly.

Otherwise, this control function shows you what commands are in your command buffer (the command buffer is an area of memory where the computer stores commands that have been entered, but not yet executed). If you type commands faster than the system can process them (for instance, you type a command to "PRINT" while AMOS is still processing your last command to "TXTFMT"), your command is stored in the command buffer.

Since AMOS is still working on the one command, the characters you typed when you entered your latest command will be intermixed with the output of the command that is processing.

2.5.3∞Correcting Typing Mistakes

If you make a mistake while typing a command line, you may always correct any mistakes in that line if you have not yet typed a RETURN. Use the RUB key to backup and erase single characters, use the left arrow key to backup without erasing, or type CTRL/U to take you back to the beginning of the command line so you can start over. (See the paragraphs above for an explanation of the RUB key and CTRL/U.) You can also use a CTRL/C to tell AMOS to ignore the current line.

If you press the RETURN key before correcting your mistake, and the command you entered was not a valid AMOS command, AMOS tells you that it did not recognize the command:

```
DOR RETURN
?DOR?
```

(You meant to type "DIR".) After letting you know that it does not understand DOR, AMOS displays its prompt symbol. You are now free to try again.

Your terminal may be set up for you to edit and recall command lines. You can easily test to see if it is by typing Control-R. If the most recent command you typed (such as "DOR") reappears next to the cursor, your terminal is set up to recall and edit command lines.

By recalling the command line which contained the error, you can back space (using the left arrow key) and correct it. Now when you press RETURN, AMOS can recognize the DIR command and display a list of the files in your account. It doesn't matter where the cursor is on the command line; if the command is valid, AMOS will execute it.

This feature is going to be even more useful when you type longer commands. We'll talk more about recalling and editing command lines in Chapter 4.

2.6∞CHANGING DISK CARTRIDGES AND FLOPPY DISKS

To change disk cartridges or floppy disks, follow the instructions that accompanied your disk drives. You do not need to reset the computer in order to change a disk; but before you cycle it down, you should consult other users who might be accessing the same device.

DO NOT turn off the disk drive units or the computer when changing a disk.

After changing a disk cartridge or floppy disk, you must always "mount" that new disk by using the MOUNT command. AMOS always mounts the System Disk (DSK0:) for you when you first turn on or reset the system.

2.6.1∞The MOUNT Command

Type MOUNT and the name of the device which contains the new disk. (Remember to include the colon.) Then press the RETURN key on your terminal keyboard. For example:

```
MOUNT AMS1:RETURN
```

This command mounts the floppy disk that is in drive AMS1:.

Using a MOUNT command is the only way you have of letting AMOS know that you have changed a disk. If you do not use MOUNT after changing a disk, AMOS assumes that the disk that was previously in the drive is still there.

AMOS stores information in an area called a bitmap that tells it where data and files are located on a disk. Each disk, having different numbers and lengths of files saved on it, has a different bitmap.

Therefore, if AMOS is assuming that a certain disk is in the drive, it writes to and reads from that disk according to the bitmap of the disk. If the disk in the drive was not mounted when it was loaded, AMOS will be using the wrong bitmap when it reads from and writes to the disk.

This can cause your data to be overwritten or destroyed. When you use the MOUNT command, AMOS reads the bitmap from the new disk, and uses that when reading from and writing to the disk.

If you accidently attempt to mount a disk that someone else is already using, you could cause damage to the bitmap of the disk, which will probably cause loss of data. It does not harm a disk if it is already mounted and you use the MOUNT command, *unless* someone else is using that disk.

You will get a warning that other users are accessing the disk, and asking you if you want to go ahead with the MOUNT. Make sure the other users are aware of what you are doing before you proceed.

If you should incorrectly enter the name of the device you want to mount, you see a message something like this:

```
MOUNT ASM1:RETURN
?Cannot mount ASM1: -device does not exist:
```

If you get this message, it means that you misspelled the device name or that the device is not defined on your system. To find out what devices are defined, use the DEVTBL command:

```
DEVTBL RETURN
```

and you may see something like this:

```
DSK0 (sharable) (physical unit 0, logical unit 0)
DSK1 (sharable) (physical unit 0, logical unit 1)
DSK2 (sharable) (physical unit 0, logical unit 2)
DSK3 (sharable) (physical unit 0, logical unit 3)
TRM0 (sharable)
RES0 (sharable)
MEM0 (sharable)
VCR0 assigned to NETSER
```

These are the devices that are currently defined for your system. If you do not have a Winchester disk drive, the display you see may be substantially different from the first four lines of the example above.

Now that you are familiar with the system hardware, we can show you some of the wide variety of commands and programs you can use to retrieve information and process data. In the next chapter we'll show you a group of commands that display valuable system information.

CHAPTER 3

REQUESTING INFORMATION FROM THE COMPUTER

Some of the commands that you can use on the system are designed so that if you enter only the command itself (with no specifications), AMOS will display information pertaining to that command. With others, AMOS will display instructions for the command if you leave out vital information. The rest will give you some kind of error message if you do not enter the command properly, or leave out vital information.

The *System Commands Reference Manual* contains information about all of the commands available.

This chapter describes some of the commands that will help you find out information about the system. You can experiment with these commands to familiarize yourself with using your terminal and entering commands. The following topics are discussed in this chapter:

- The HELP command
- The LOG command
- The STAT and SYSTAT commands
- The JOBS and JOBALC commands

3.1 THE HELP COMMAND

The purpose of the HELP command is to give information about the system and the system commands to a person who is not familiar with using the system.

You don't have to be logged in to use HELP. To find out what topics the HELP command knows about, type HELP followed by a RETURN:

```
HELP RETURN
```

Depending on the selection of topics on your particular system, you may see a display that looks something like this:

```
Help is available for the following:

APPEND  BAUD  COPY  DEL  ERASE  FORCE
```

To see information about one of the topics listed, type HELP followed by the name of the topic:

```
HELP COPY RETURN
```

The screen clears and the terminal displays information on that topic. If you ask for information on a topic that HELP doesn't know about, it will display the subjects it does have information about.

Of course, there are many more HELP files than in the display above, and you may create your own HELP topics by using one of the system text editors to create files with .HLP extensions in the HELP library file on DSK0:[7,1]. (See Chapter 4 for a discussion of file extensions.)

The HELP command automatically includes the .HLP files that you create in its list of topics, and displays your HELP files on command.

3.2∞FINDING OUT WHAT ACCOUNT YOU ARE LOGGED INTO (LOG)

If you don't know or cannot remember which account your terminal is logged into, type LOG and a RETURN, and AMOS will tell you:

```
LOG RETURN
Current login is DSK0:[321,10]
```

If you are not logged in, you will see the message:

```
Not logged in
```

3.3∞THE STAT AND SYSTAT COMMANDS

The System Status commands give you a quick summary of tasks the computer is currently performing. The displays list all the jobs on your computer system and tell you what programs they are running. Jobs, you recall from *Introduction to AMOS*, are subdivisions of your computer's memory and each is usually attached to a terminal.

SYSTAT displays on your terminal screen one line for each job on the system. Each line gives the name of the job, the name of the terminal that the job is using, the account that the job is logged into, the octal base address in system memory where the system maintains information about the job, the status of the job, the name of the last program run by the job, the number of bytes (decimal) in the memory partition being used by the job, and its base address in memory (octal).

Below the job information lines is a list of all storage devices currently mounted on the system and the number of blocks free on those devices.

To see the SYSTAT display, enter:

```
SYSTAT RETURN
```

The STAT command is much like the SYSTAT command, except that the display is updated continuously until you press ESC/Q. This can be very useful if you wish to watch the progress of a task, or to see what other users or devices are doing. To see the STAT display, enter:

```
STAT RETURN
```

Consult the Reference Sheets for SYSTAT and STAT in your *System Commands Reference Manual* for examples and further explanations.

3.4∞FINDING OUT THE STATUS OF YOUR JOB (JOBS AND JOBALC)

To interpret the SYSTAT and STAT displays mentioned above, you will need to know your job name. Also, if you wish to use the SEND command (explained in Section 10.5, SEND), all users on the system will need to know the names of their jobs.

To find out your job name, type JOBALC and a RETURN:

```
JOBALC RETURN
YOUR JOB NAME IS JOB1
```

The JOBS command shows you how many jobs are allocated on your system, and also how many are currently assigned. The command and display look something like this:

```
JOBS RETURN
13 jobs allocated, 10 currently assigned
```

Now that you know the basics of how to operate your terminal and keyboard, and how to use a few commands to extract information from the system, we can move on to the structure of more complex commands. The next chapter tells you how to arrange various pieces of data when you enter commands and how the arrangement affects the commands' behavior.

CHAPTER 4

COMMAND FORMATS, SWITCHES, AND DEFAULTS

Every AMOS command is actually an individual assembly language program or command file stored on the disk that processes data according to the arguments and switches you specify when you enter the command name. Most commands require that you enter these arguments and switches in a relatively fixed sequence.

One measure of your computer's versatility is that, in addition to the many commands supplied with the Alpha Micro Operating System, you can write your own commands, and we'll tell you how to do that in Chapter 11, "Command Files, Do Files, and Control Files."

In the previous chapter we discussed several commands that did not need additional information to perform their job. You merely entered the command name and pressed RETURN. However, most commands require certain data, entered in a certain sequence, to produce the desired results.

In this chapter, we'll talk about how commands expect to receive this additional information. Here are the topics this chapter contains:

- Command Syntax
  (How to fit the pieces of a command together.)
- Command Switches
  (Special options that change the way a command works.)
- Command Defaults
  (What AMOS assumes if you leave something out.)
- Command Line Editing
  (How to recall previous commands and modify them.)

First, let's look at how the elements of a command are arranged.

4.1∞FORMAT

Although some commands consist of the command name alone, many commands need additional information to do what you want them to. How you arrange this additional information is determined by the format (syntax) of the individual command.

Each command requires different types of information depending on what the command does. Some pieces of data are required, while others are optional. Also, the amount of data you give on a command line and the order you enter it in determine how the command performs.

Here is a command format that shows what order the options must be in.

```
COMMAND {outfilespec=}{infilespec1{...,infilespecN}}{/switch}
```

The command name (six characters or less) is required, but everything else (as indicated by the {º} symbols) on this sample command line is optional. In this case, the "outfilespec=" is followed by the "infilespecs" and the "/switches." In this example, the phrase {...,infilespecN} means that you can specify any number of infilespecs, each separated by commas.

An infilespec is an input file specification—a file the command takes and manipulates in some way. An outfilespec is an output file specification—a file the command creates or a file where it places its results. We'll explain swtiches in just a moment.

Notice that the outfilespec comes before the equal sign, and the infilespecs come after it. If you're used to scientific equations, this arrangement may seem strange to you. But computers usually treat an equal sign as an "is replaced by" sign, rather than "is equal to."

That's why you'll often hear the phrase "replacement statement" used instead of "equation" when talking about programming instructions or system commands. All you have to remember is that the output of a command or instruction comes before the equal sign and the inputs or factors the command uses to get that result come afterward.

You've already been introduced in Chapter 1 to the abbreviations and symbols we use in our examples, but here's a quick review.

{º}	Braces. Anything enclosed in braces in a command syntax example is optional.
/	Slash. A command switch is always preceded by a slash. Some commands allow you to string several switches after a single slash, while others require a separate slash ahead of each switch you use.
RETURN	Return. This symbol appears at the end of a command line and tells you when to press the RETURN key.

4.2 SWITCHES

Switches are indicators that turn on and off various command options. They are always preceded by a slash. The command reference sheets in the *System Commands Reference Manual* tell you how to specify switches for each command.

You can specify switches in commands by their full name (such as /QUERY or /KILL), or you can use their abbreviation (such as /Q or /K). Many switches have counterparts that reverse their effect. For example, the /NOQUERY switch reverses the effect of the /QUERY switch, and it can be abbreviated /NOQ.

4.3 DEFAULTS

You don't always have to specify complete file specifications because every command assumes certain values unless you tell it otherwise. These assumed values are called defaults, and they enable you to use the command in many situations without having to type in a long string of additional information.

Let's look at the DIR command for a moment. This command displays a list of the files in certain accounts, and we'll discuss it more full in the next chapter. For now, just enter the command name on your terminal, like this:

```
DIR RETURN
```

The DIR program assumes that you want to see a directory of all the files in the account you're currently logged into. This is the most general usage of DIR.

If you want to see a directory listing of just the files in your account that have a file name of MEMO, you can enter the specific filename to limit the display.

```
DIR MEMO RETURN
```

The DIR command still assumes you want to see the files in your own account. But if you want to see a list of the MEMO files in another account (we'll talk more about disk accounts in Chapter 9), all you have to do is add an account number. Like this:

```
DIR MEMO[220,15] RETURN
```

DIR assumes the account is on the same logical device as your own account. So if you want to see a list of the MEMO files in a certain account on another device, add a device specification. For example:

```
DIR DSK1:MEMO[200,15] RETURN
```

As you can see, the more information you give a command, the more easily you can control the results. As we explain the use of each command, we'll tell you the default values each command uses and what other information each command assumes.

4.4∞INPUT/OUTPUT REDIRECTION

Any command in AMOS that allows switches before the file specification can use Input/Output Redirection. This lets you redirect the input and/or output of an AMOS command.

In order to use redirection, redirection must be set on for your system (this is the default condition). If you try using redirection and it does not work, use the SET REDIRECTION command (see the SET reference sheet).

4.4.1∞Redirection Symbols

>	Send output to the following file.
>>	Append output to the end of the following file.
<	Input the contents of the following file into the AMOS-command.
|	Place the output of the following file into the input of the AMOS-command.
#	Removes terminal dependent escape sequences from the redirected data. Must be the first symbol following REDIR on the line.
\	Quotes the character following it so it is not interpreted as a command.

The default extensions are .INP for input files, .LST for output files.

4.4.2∞Using Redirection

To use redirection, enter the AMOS command followed by the redirection symbol and the redirection command. For example:

```
SYSTAT/N >OUT.LSTRETURN
```

The above command places the output of SYSTAT into the file OUT.LST. Another example:

```
VUE INPUT.TST <VUECMD.INPRETURN
```

The above command takes the contents of VUECMD.INP and uses it as input to the VUE program. You can also combine the type kinds of redirection. For example:

```
VUE INPUT.TST <VUECMD.INP >OUT.LSTRETURN
```

There is a REDIR command which forces redirection if NOREDIRECTION has been set for your job. REDIR is useful in command files when it is not known if redirection is on or off. If you are redirecting input from a file, and the end of the file is reached, the input source reverts to the keyboard.

I/O redirection can both execute DO files and be used within CMD or DO files. Because input redirection works regardless of terminal input mode, redirection works where DO file won't—for example, connecting across a network, running VUE, etc.

I/O direction uses process spawning. This means you must have extra jobs available on your system (allocated in a JOBS statement in your system initialization file).

Redirection reduces the size of your memory partition by about 8K while it is being used.

Some programs, such as AlphaWRITE, use terminal features that require responses back from a real terminal. Since I/O redirection does not provide such responses, these programs will hang. You can free it (at least until the next time it asks for a response) by using CTRL/F (ACK).

If a program is run with the # option, and that program requests the number of rows and columns on the terminal, it receives a value of zero for both. This is technically correct, but it causes some programs to lock up and die.

With redirection you can use "piping"—redirecting the output of one program into the input of another. For example:

```
REDIR ERSATZ | TYPE/P
```

This command take the output of the ERSATZ program (typically a very long list) and feeds it to the TYPE command which, because of /P, lets you see a paginated display on your terminal.

In this example:

```
#PROCES >OUTPUT.LST RETURN
```

The command file ***PROCES*** is run, and the output (minus terminal-dependent escape sequences) is sent to ***OUTPUT.LST***.

In another example:

```
REMOVE \<THE MEMO\> >OUTPUT.LST RETURN
```

the phrase ***<THE MEMO>*** is an argument for the ***REMOVE*** .DO file. If you left out the "\" before the < and > brackets (needed to indicate to the .DO file that THE MEMO is one argument, not two), the command would try to find a file called THE to get input for REMOVE.DO. The above command calls REMOVE.DO with THE MEMO as an argument, and puts the results in OUTPUT.LST.

4.5∞EDITING A COMMAND LINE

As you type more and more information on a command line to customize its results, the easier it is for a mistake to creep in. And nothing is more frustrating than to type a long command with lots of switches and file specifications, and get a "?Specification error" because you put a single comma in the wrong place.

Your terminal might already be set up for you to edit command lines; if not, ask your System Operator to modify the System Initialization Command File to activate this feature on your terminal. Actually, the command line editor feature has two parts:

- ∞∞You can edit the command line you're currently working on.
- ∞∞You can recall previously entered command lines and edit them.

With the command line editor active, you can insert characters, delete sharacters, space forward and backward, and perform other operations on AMOS command lines in much the same way that the AlphaVUE text editor allows you to modify lines in a text file. (You'll get a brief introduction to AlphaVUE in the next chapter.)

You can use the designated keys on your terminal keyboard to perform these operations, or if your terminal does not have these keys, you can use Control key and Escape key sequences. Here's a list of key sequences that correspond to these special keys:

CTRL/H	Left Arrow
CTRL/L	Right Arrow
CTRL/W	Next Word
CTRL/A	Previous Word
CTRL/F	Insert Character
CTRL/D	Delete Character
CTRL/V	Delete Word
CTRL/Y	Delete to End of Line
CTRL/Z	Delete Entire Line
CTRL/Q	Enter and Exit Character Insert Mode
ESC/>	Insert Word (Press space bar to exit)
ESC/?	Help

4.6∞RECALLING A COMMAND LINE

In addition, you can call back one or more previously entered command lines to your screen and edit them or re-enter them. (Your System Operator can tell you how many command lines are stored for your terminal.) You can press PREV SCREEN (CTRL/R) to recall previously entered lines, and NEXT SCREEN (CTRL/T) to get back to the present. Note that these two keys operate differently when you use AlphaVUE to edit a text file.

Don't worry if you press your RETURN key several times; you haven't cleared out any stored command lines. ***Only non-blank command lines are saved.*** Also, if you use AlphaWRITE, AlphaBASIC, or some other application program, your previously entered AMOS commands will still be waiting for you when you return to AMOS command level.

After you've retrieved a previous command line, press RETURN to perform that command again. Here's an example you can try:

DIR/WRETURN

This command displays the names of the files in the account you're currently logged into. At the end of the display, the cursor is positioned next to the prompt symbol.

Now, press the PREV SCREEN key (or CTRL/R) to recall the last command you entered. DIR/W will appear on your screen again.

Use the right arrow key to move to the end of the command line and add *.CMD, like this:

DIR/W *.CMDRETURN

Press RETURN and a list of all the files in your account that have .CMD extensions appears on your screen. (If there aren't any, you see "No such files" instead.) The cursor is positioned at the end of the display next to the prompt.

Once again, press PREV SCREEN and the command DIR/W *.CMD reappears on your screen. Move the cursor to the asterisk (*) and press CTRL/Q to enter character-insert mode. Type DSK0:[2,2] and press CTRL/Q again to exit character-insert mode, so the command now looks like this:

DIR/W DSK0:[2,2]*.CMDRETURN

When you press RETURN this time, you should see a list of several files with .CMD extensions since this account is the system command file library. Although these examples are relatively simple, you can see how useful the command line editor will become when you need to enter commands with long strings of switches and file specs.

4.7∞GETTING HELP FROM THE EDITOR

If your terminal uses the command line editor feature, you can also take advantage of a special Help facility. By pressing HELP (or ESC/?) the editor takes as much of a command as you've typed so far and tries to tailor the help message to what you're trying to do. This type of help is only available at AMOS command level. Other help facilities are available while you're using AlphaVUE, AlphaWRITE, AlphaCALC, etc. You'll learn about them in conjunction with those topics.

The next step is for you to create some files to use various AMOS commands on. The next chapter introduces you to AlphaVUE, an extremely useful and powerful text editor program that enables you to create and modify files. You'll also learn about a number of system commands that make it easy for you to handle these files once you've created them.

CHAPTER 5

CREATING AND USING FILES

Files are the basic units for organizing and storing related data. All the data you work with on the computer is eventually stored on your disk in the form of files, so before we do anything else, let's find out how to create files and retrieve data from them.

This chapter covers the following topics:

- Creating a new file in your disk account
- Finding out what files you already have
- Which file extensions have reserved meanings
- Displaying a file on your terminal
- Making another copy of a file
- Changing the name of a file
- Erasing a file
- Printing a copy of a file on your printer

Let's start off by creating a new file.

5.1 CREATING FILES WITH ALPHAVUE

To create a new file, you must first select a file name six or fewer characters long. Every file name also has an extension of three or fewer characters. The extension is useful for identifying the type of data a file contains and is separated from the file name by a dot, like this:

```
DATA.LST
```

Duplicate extensions are acceptable within an account as long as the file names preceding them are different. And you can have files with the same name and extension in different accounts, but not in the same account.

Although you can theoretically use any file extension you wish, there are a number of file extensions that have special meanings.

5.1.1 Reserved File Extensions

A file extension can be any combination of letters you choose. For many files that you need to process routinely, there are certain file extensions that AMOS automatically generates or expects to receive.

Appendix B contains a brief description of each reserved file extension and where it's used. For example, AlphaVUE uses the extension .BAK. AlphaBASIC uses .BAS, .RUN, .SBR, and .DAT. TXTFMT uses .TXT and .LST. Other reserved extension include .HLP, .CMD, .PAS, and .LSP. Glance over this appendix to familiarize yourself with the various extensions that have special significance.

After you've selected a filename and extension, you might wish to find out if the name you've chosen is already being used.

5.1.2 Finding Out What Files are in Your Account (DIR)

To see if the file name is in use, you can request a display of your account directory with the DIR command. Entering DIR by itself causes a single column list of the files in your account to appear on your screen. In this is longer than one screenful, you can use CTRL/S to stop the display and CTRL/Q to start it up again.

Perhaps an easier way to view the account directory is by using the /W switch with the DIR command. This switch widens the display to four columns. Just enter:

```
DIR/W RETURN
```

The display that appears on your screen now is much easier to read, especially if your account contains a large number of files.

5.1.3 Creating a Sample File

Let's say you need to write a letter to a customer, and the file you want to create is called LETTER.TXT. Enter:

```
VUE LETTER.TXT RETURN
```

AMOS loads the AlphaVUE program into memory, and AlphaVUE searches for the file name you entered. If there were already a file by that name in your account, you'd see it displayed on your screen. (To get back to AMOS command level from here, you could press the ESCAPE key, then press the letter Q, and finally press RETURN. Section 5.1.4 goes into this procedure in more detail.) But since it's a file that doesn't exist yet, AlphaVUE responds:

```
LETTER.TXT does not exist, do you wish it created?
```

You do, so enter a Y for yes and press RETURN. (AlphaVUE asks this question just in case you make a typing mistake when asking for an existing file.) Your screen clears and fills with asterisks, and the cursor moves to the upper left corner, ready for you to start typing your note. If a different display appears—one where the cursor appears next to a > sign—press the ESCAPE key to see the screenful of asterisks.

Now you can enter the sample letter, just to get the feel of creating a document. Just type it in, and don't worry about mistakes. It might seem a bit different, especially if you are used to a typewriter. Later on, you'll learn more about AlphaVUE from the *AlphaVUE/TXTFMT Training Guide*and the *AlphaVUE User's Manual*. These manuals will show you how useful and easy AlphaVUE can be for entering, correcting, and editing documents.

Here is the sample letter for you to type:

```
Mr. John R. Baxter
Autonomic Neurosystems
P.O. Box 12
Commerce, CA   92714

Dear Mr. Baxter:

Your recent order helped us surpass our sales goal
for last month, and we wish to express our sincere
appreciation.  Thank you for your confidence in our
products.  We are looking forward to your continued
patronage.

                      Sincerely,




                      George Morton
```

5.1.4∞Getting Back to AMOS Command Level

Now that you have typed your note into memory, you need to store it on your disk and return to AMOS. Pressing the ESCAPE key brings you to AlphaVUE command level. The AlphaVUE prompt is a right angle-bracket (>) and if you enter an F (for Finish) and a RETURN, your file is stored on the disk. As AlphaVUE stores your file, it prints one dot on your screen for each block of the file it transfers.

Your cursor appears next to the AMOS prompt indicating that you have left AlphaVUE and are once again talking to AMOS. The whole operation looks like this:

```
>F RETURN . . .
```

If you decide that you don't want to save what you just typed, you can still return to AMOS by entering a Q (for Quit) at AlphaVUE command level instead. Like so:

```
>Q RETURN
```

You can do many more things with AlphaVUE than we've described here, and you can learn more about both AlphaVUE and TXTFMT by reading the *AlphaVUE User's Manual* the *TXTFMT User's Manual* and the *AlphaVUE/TXTFMT Training Guide*.

5.2∞DISPLAYING FILES (VUE, TYPE)

Once your files are stored in your account on the disk, you can look at them and modify them whenever you wish by using the VUE command or the TYPE command.

The VUE command is the same one you used to create the file in the first place. Here's how to look at that note you wrote earlier:

```
VUE LETTER.TXT RETURN
```

Since it now exists in your account, AlphaVUE has no trouble locating it and displays it on your screen. If your file is bigger than one screenful, you can page back and forth using PREV SCREEN (CTRL/R) and NEXT SCREEN (CTRL/T). To go directly to the end of the file, use CTRL/E; and to return to the front of the file, use CTRL/^ or the HOME key.

When you're through looking at your file or changing it, you use the same method of storing it on the disk as you did when you created it. That is, you use the Escape key to get to the command level, and enter "F" to write the file to the disk and bring you back to AMOS.

AlphaVUE won't let you look at files with certain extensions, either because they contain data your terminal can't display (such as .RUN files), or because they should not be modified (such as .BAK files). If you try to look at a file you shouldn't, AlphaVUE sends you an error message. For example:

```
?Cannot VUE or UNYANK file with .BAK extension
```

Another way to look at a file is with the TYPE command:

```
TYPE LETTER.TXT RETURN
```

You can't modify a file with the TYPE command the way you can with the VUE command, and instead of showing you only one screenful at a time, TYPE scrolls through the entire file from beginning to end without stopping.

The only way you can halt the display is with a CTRL/S. To start it up again, use CTRL/Q. (You can use the /P switch with the TYPE command to display one page of data at a time.)

Also, TYPE does not allow you to go backward in the file or jump around the way the VUE command does. Although you'll probably use the VUE command most often, TYPE is useful when you just want to verify the contents of a particular file.

5.3∞COPYING FILES (COPY)

If you need to send the same letter you wrote at the beginning of this chapter to a different customer, you can easily make another copy of it and just change a few of the words. The COPY command lets you do this with a minimum of bother. Decide on another file name that does not already exist in your account and enter the following command.

```
COPY NOTE.TXT=LETTER.TXT/NOD RETURN
```

where NOTE.TXT is the name of the new file, and LETTER.TXT is the name of your original letter. Now you have two copies of the same thing and you can use AlphaVUE to change one or the other.

The /NOD on the end stands for /NODELETE. It means that if, by chance, the file name you chose (NOTE.TXT) was already in use in your account, the file you copy (LETTER.TXT) will not wipe out whatever was in that file before. Rather, AMOS will inform you that the file NOTE.TXT exists, and not do any copying. You can then choose an unused name for your file.

5.4∞RENAMING AND USING FILES

There are times when you may wish to change the name of a file, and the RENAME command was developed for this purpose. It's a good idea to verify that the new file name you picked is not already being used; RENAME sends you an error message if it is.

If you suspect the file name is in use and you don't wish to save the contents of the file if it is, you can use the /D switch which deletes the file before renaming your current file to the new name. Here's how it works:

```
RENAME/D MISSIV.TXT=NOTE.TXT RETURN
```

where MISSIV.TXT is the new name given to the current file, and NOTE.TXT is the current file name. When the command is finished, the file NOTE.TXT no longer exists and you can use that file name again for another file.

5.5∞PRINTING FILES (PRNT)

Now that you've finished your letter, you need a printed copy of it to put in the mail. To use the PRNT command, enter the name of the command and the name of the file:

```
PRNT LETTER.TXT RETURN
```

The command above causes your letter to be sent directly to the printer. If you have more than one printer attached to your system, your printing will automatically go to whichever one the System Operator has defined as the default printer. If you want to use a printer other than the default, you must specify the printer name in your PRNT command. If it's a Diablo printer, for example, your PRNT command might look like this:

```
PRNT DIABLO=LETTER.TXT RETURN
```

The file automatically begins printing.

To find out the names of the printers defined on your system, or to check on the progress of a file being printed, enter just the command:

```
PRNT RETURN
```

The resulting display tells you the name of each printer and how much of each file is left to print.

5.6∞ERASING FILES (ERASE)

When you want to delete a file from your account, you can use the ERASE command to get rid of it. Suppose you want to get rid of the LETTER.TXT file. Just enter:

```
ERASE LETTER.TXT RETURN
```

The file is gone and does not appear on the directory listing of your account. If you want to create another LETTER.TXT file, you can do so by following the same steps you used to create the first one.

Depending on how your system is set up, you may see an error message if you try to ERASE, RENAME, or access a file. It's possible that the file has been protected during system initialization from being accessed. If you need to access such a file, contact your System Operator for help.

These first five chapters have given you a broad, basic knowledge of how AMOS computer systems work, and how you can use them to your advantage. The next seven chapters tell you how to add leverage to the commands you've already used, and introduce you to some new commands that make your job even easier.

CHAPTER 6

FILE PROTECTION

AMOS supports two types of files: traditional files which have a maximum size of 65,535 disk blocks (32 megabytes), and extended files which have a maximum size of 4,294,967,295 disk blocks (2,097,152 megabytes) in any one disk file or on a single logical unit. Even if all your files do not approach these maximum sizes, extended files possess other features which make them more desirable than traditional files. Eventually, you will want all your files to be in the extended file structure.

- Interpreting the file protection codes.
- Changing a file protection code.
- Other extended directory information.
- Mixed traditional and extended file systems.

6.1 FILE PROTECTION

Each file and directory in the extended format has a ten-digit code number associated with it. This code number is made up of five groups of two digits each. Each two digit group specifies the protection level for a certain groups of users trying to access the file.

AMOS automatically determines the class of the user, and accesses the appropriate group of bits to decide what type of access is allowed. A typical protection code of 0505051717 is composed of five groups of two digits.

```
| 05 | 05 | 05 | 17 | 17 |
   5    4    3    2    1
    Protection Groups
```

The five groups correspond to:

Group 1 -	Users within the same directory (account)
Group 2 -	Users at the same directory level (same project number)
Group 3 -	Users at a different level (different project number)
Group 4 -	Users within the same network level
Group 5 -	All other users not included in groups 1-4

The code for each group is a number which represents the type of access members of the group have to that file. The four basic number codes are:

01 -	File may be read
02 -	File may be written
04 -	File may be executed
10 -	File may be deleted, renamed, and have its protections changed

Note that you can add codes together to combine their properties. For example, 03 allows the file to be both read *and* written. A code of 06 allows the file to be both written and executed. And a code of 17 combines all the access types into one.

Each of the five groups defined above can have different combinations of access to the file. The typical protection code we mentioned earlier, 0505051717, enables the first two groups complete access to the file, while groups 3, 4, and 5 can only read and execute the file.

An interesting sidelight to this scheme of protection codes is that if a particular group does not have "read" access to the file (a code of 00, for example), the file will not appear on any directory displays requested by members of that group—it is essentially invisible to them.

The only access type that can be overridden is 10 (delete, rename). The System Operator, by logging into OPR: (the System Operator's account, DSK0:[1,2]), can delete and rename files that would otherwise be inaccessible.

If the ten digit protection code is preceded by a "1", it indicates the file should be "zeroed" when it is erased. (The sample code above would become 10505051717.) Normally, erased files are simply removed from the directory, but the data that was in the file is left intact on the disk. If this "Zero Flag" is present, the file will be removed from the directory, and the data blocks which once comprised the file will be filled with zeros, thus obliterating the data.

6.2 CHANGING PROTECTION

The RENAME command enables you to change the protection of the files for which you have "write" access. Since RENAME is a wildcard command, you can change the protection of an entire group of files or change one file at a time. The /PROTECTION: (or /PROT:) switch, followed by the new protection code, updates the file protection code in addition to whatever else you ask RENAME to do. For example:

```
RENAME CUSTMR.SAV/D=CUSTMR.LST/PROT:10001010317 RETURN
```

The command above renames CUSTMR.LST to CUSTMR.SAV (and deletes the existing CUSTMR.SAV file), and it gives CUSTMR.SAV the new protection code 10001010317. This code can be interpreted like this: 1 00 01 01 03 17. The first digit is the "zero flag" indicating the data will be overwritten with zeros when it's erased. The next group, "00," indicates that group 5 has absolutely no access to the file. Groups 3 and 4 are only allowed to read the file, group 2 can read and write the file, and group 1 (anyone who can login to the account where the file resides) can do anything they want to it.

RENAME can also be used just to change protection like this:

```
RENAME DATA.S87/PROT:0303030717 RETURN
```

Keep in mind that protection is only available under extended directories. The DIR program (and others) shows the default protection 0505051717 for all files under the traditional directory format since that is the equivalent protection code.

6.3 ADDITIONAL DIRECTORY INFORMATION

Along with its file protection code, each file in the extended format also has additional information stored with it which is automatically maintained by AMOS.

★ Date and Time of File Creation

★ Date and Time of Last File Update

★ Date and Time of Last Backup

★ Record Size

All dates and times are stored to the nearest minute. Creation date and time is the moment at which the file was originally created. The COPY command preserves this date and time if you make additional copies of the file. The update date and time correspond to the last time the file was written to. And the backup date and time record the moment at which the file was most recently backed up using the standard AMOS BACKUP utilities.

By comparing the backup and update dates and times, you can select files for your next backup that have been updated since the previous backup, thereby providing a convenient way of reducing the amount of data backed up on a regular basis.

The record size is used to store the size of the data records defined in a record I/O file.

6.4∞MIXED FILE SYSTEMS

Any logical disk unit can use either the traditional file structure or the extended file structure.

Whenever a disk is mounted with the MOUNT command, AMOS checks to see which file system is present on the disk. Based on which file system it finds, AMOS then adapts its routines to support that file system.

File system information that is present only in the extended file system (such as date and time stamps) is returned as zeros when accessing a traditional file. Protection is returned as 0505051717 which corresponds to the protection used in the past.

PART 2

Making the Computer Do the Work

CHAPTER 7

USING WILDCARD SYMBOLS

So far, you've learned the basic operation of several system commands, but useful as these commands are, they can become tedious if you have to repeat them for a large number of files. If you have to make duplicate copies of twenty AlphaBASIC programs, or rename all your .TXT files, or erase the .BAK files in your account one by one, you could become extremely frustrated if it weren't for wildcard symbols.

A select group of the most often used commands recognize these special symbols and allow you to process groups of files without having to specify each and every file name. Many commands do not recognize these wildcard symbols, so if you're ever in doubt, check the appropriate command reference sheet in the *System Commands Reference Manual*.

This chapter contains the following information:

- Definitions of the various wildcard symbols.
- Some rules for using wildcard symbols in account specs
- Examples of how wildcarding works

Before we do anything else, we should define wildcard symbols.

7.1 WHAT ARE WILDCARD SYMBOLS?

In some card games, you can use a joker (or some other card) as a wild card which can be used in place of any other card in the deck. Likewise, you can substitute wildcard symbols for various parts of file and account specifications in certain AMOS commands and enable the command to process several files with a single file specification.

The list below explains what each of the wildcard symbols means.

*	Asterisk. This symbol stands for any group of characters in file names, file extensions, and account numbers. For example, *.BAS means every file that has the extension .BAS. Also, PR*.BAS means all files that start with PR and have .BAS extensions like PRIVAT.BAS, PROOF.BAS, PRT23.BAS, and PR.BAS.

?	Question mark. This symbol stands for a single character in file names, file extensions, and account numbers. For example, EX?PT.LST includes all files whose names begin with EX and end with PT (with only one character in between), and have .LST extensions, such as EXMPT.LST, EXRPT.LST, and EX9PT.LST, but not EXERPT.LST. A series of question marks at the end of a file name indicates that many or *fewer* characters.
ALL:	All Logical Devices. This symbol stands for any file-structured, mounted device. For example, ALL:START.DO refers to all START.DO files on all logical devices.
dev:	All logical devices whose names begin with the code "dev" where "dev" is the device abbreviation. If you leave off the unit number from a device specification, wildcard commands search for the files you specify on all units of the specified device type. For example, WIN:BAKUP.CMD stands for any file named BAKUP.CMD on any logical device whose name begins with "WIN" (such as WIN0:, WIN1:, WIN2:, etc.).
[°]	Brackets. This symbol stands for any account number. For example, SORT.HLP[°] means all SORT.HLP files in any account. [°] is the same as [*,*].

The system commands that recognize these wildcard symbols as valid arguments allow you to manipulate numerous files with a single command. But before we give specific examples of how each of these wildcard commands operates, there are a couple of general rules that you need to keep in mind when using wildcard symbols in account numbers.

7.2 SOME RULES FOR USING WILDCARD SYMBOLS

First, when you use an asterisk (*) in an account number, you can either use it to represent the entire project or programmer number (as in [*,5] and [45,*]), or you can use it as a partial project or programmer number with one or more numbers following it (as in [*41,12] and [22,*2]). The catch is that you cannot put numbers *ahead* of the asterisk in a partial project or programmer number. (This wildcard account number is illegal: [25*,61.].)

Here are some examples of valid wildcard account numbers:

FOLIO.LST[*,12]	**covers**	**FOLIO.LST[5,12] and FOLIO.LST[120,12]**
PROG.BAS[15,*]	**covers**	**PROG.BAS[15,1] and PROG.BAS[15,34]**
WWTIRE.TXT[*3,*]	**covers**	**WWTIRE.TXT[3,0], WWTIRE.TXT[23,5], and WWTIRE.TXT[103,14]**

When you use an asterisk as part of a file specification, just the opposite is true. You can still use the * to represent the entire file name or extension (as in *.TXT or PRGRAM.* or *.*), but you cannot put any characters *behind* the asterisk in a partial file name or extension. (These are valid wildcard file names: PAY*.LS* and X*.B*, but these are invalid: F*M.*AS and *RX.F*L.)

The question mark symbol (?) at the beginning of a project or programmer number stands for that many or *fewer* numbers. (For example, EASY.PRG[??1,??], covers EASY.PRG[1,2] and EASY.PRG[21,14].) If you use the ? symbol in the middle or at the end of a project or programmer number, the *exact* number of characters must match. (Thus, FINDIT.LST[1?1,2?] covers FINDIT.LST[101,20], FINDIT.LST[141,24], and FINDIT.LST[111,26], but not FINDIT.LST[11,2] or FINDIT.LST[1,245].)

7.3∞WILDCARD FILE SPECIFICATIONS

The way you specify files in wildcard commands determines the way the commands perform. The paragraphs below describe how the placement of input file specifications (infilespecs) and output file specifications (outfilespecs) alters the results of these commands.

Keep in mind as you read through this chapter that our are hypothetical. Because of the file protections discussed in the previous chapter, you might not get the same results if you try the same commands with your own files. Our examples presume that you have complete access to all files.

7.3.1∞Input File Specifications

Most commands, wildcard commands included, use infilespecs. These are the file specifications that the command works upon to perform its designated function. In a wildcard command, you are able to use one infilespec to represent a group of files. For example, if you want to refer to DSK0:MCSAM.RUN and DSK1:MCELI.RUN, you could condense them into a single filespec: DSK:MC*.RUN.

There are certain situations, however, when wildcard commands should be used with caution since you might unintentionally include more files in the wildcard filespec than you bargained for. Such is the case when you are ERASEing or RENAMEing files.

7.3.2∞Output File Specifications

A few of the wildcard commands require you to supply an outfilespec under certain circumstances. The outfilespec, if used, always comes immediately after the command name itself, ahead of any infilespecs, and indicates the destination or result of the command's designated outfilespec and an infilespec:

```
RENAME *.SAV=*.BAK RETURN
```

In this command, *.SAV is the outfilespec and *.BAK is the infilespec. All the files having .BAK extensions in the account you're currently logged into are automatically renamed with .SAV extensions.

If we changed the command to say:

```
RENAME *.SAV=BRUT.BAK,SEC.BAK RETURN
```

...only the two specified files in the account you're currently logged into would be renamed with .SAV extensions.

Another way to use wildcard symbols in outfilespecs is this:

```
RENAME AR*=*.FIN RETURN
```

This RENAME command takes all the files with .FIN extensions in the account you're currently logged into and changes the first two letters of the filename to AR.

7.3.3 Ersatz Device Specifications

The more data you have to enter on your screen, the easier it is for you to make a typing mistake. The less you have to type, the better. So for your convenience, several abbreviated substitutes have been invented for the device and account specifications that you use most frequently. These ersatz specifications are only four characters long and reduce the possibility of error a great deal.

The following table lists some of the standard ersatz specs and the devices and accounts they correspond to:

ERSATZ	DEVICE SPEC	LIBRARY NAME
RES:	System Memory	
MEM:	User Memory	
OPR:	DSK0:[1,2]	System Operator's Account
SYS:	DSK0:[1,4]	System Program Library
DVR:	DSK0:[1,6]	Device Driver Library
CMD:	DSK0:[2,2]	Command File Library
LIB:	DSK0:[7,0]	Miscellaneous File Library
HLP:	DSK0:[7,1]	Help File Library
BOX:	DSK0:[7,2]	Mailbox Data File Library
LSP:	DSK0:[7,4]	AlphaLISP Language Library
PAS:	DSK0:[7,5]	AlphaPASCAL Language Library
BAS:	DSK0:[7,6]	AlphaBASIC Language Library
MAC:	DSK0:[7,7]	Macro Language Library

You can use these ersatz specifications anytime you want to refer to one of the library accounts listed above. For example, to see a directory listing of all the files in the Help File Library, you could enter either:

`DIR DSK0:[7,1]`RETURN -or- **`DIR HLP:`**RETURN

7.4∞WILDCARD SWITCHES

Most ordinary commands permit you to put switches only at the end of a command line. Wildcard commands give you added flexibility by permitting you to place switches anywhere in the command line. Where you place a particular switch within a command will then affect how the switch operates.

Wildcard commands recognize two types of switches: file switches that pertain to individual file specifications in the command line, and operation switches that affect the entire command line regardless of their placement.

7.4.1∞File and Operation Switches

A file switch usually applies to a single file specification, and you can place it anywhere in a command line depending on the results you want. We'll talk more about file switches and where to put them when we begin setting switch defaults.

You can place an operation switch anywhere in the command line and its effect is always the same. For example, the /K switch in the PRNT commands below is an operation switch:

```
PRNT/K MEMO.LST,REPORT.LST,LETTER.LSTRETURN
```

is the same as...

```
PRNT MEMO.LST/K,REPORT.LST,LETTER.LSTRETURN
```

and is the same as...

```
PRNT MEMO.LST,REPORT.LST,LETTER.LST/KRETURN
```

The description of each command will tell you whether its switches are File Switches or Operation Switches.

7.5°WILDCARD DEFAULTS

We discussed command defaults and rules of syntax back in Chapter 4, and those same basic rules apply to wildcard commands as well. But there are some significant additional features of wildcard command syntax that increase their power tremendously.

Setting the Device Defaults:

If you do not specify the device or account in an infilespec, most commands default to the device and account you are logged into. Wildcard commands allow you to reset these default values within a command line. Let's take an example:

```
ERASE *.BAK,WIN1:*.LST,*.BAK,*.CMD RETURN
```

The first thing this ERASE command does is get rid of all the files with .BAK extensions in the account and device you are currently logged into, say DSK1:. Then the device specification changes to WIN1:, so ERASE deletes all the files with .LST extensions on that device, but still with the same account number as the account you are logged into.

For the next filespec, ERASE stays on WIN1: (it's now the default device, and deletes all the files with .BAK extensions, and *.CMD extensions.

Setting the Account Defaults:

But that's not all. Here's another example:

```
ERASE° DSK0:SAE.*,STOR.*,[200,1]PT4.LST,MGT.LST,DSK1:*.RST RETURN
```

This ERASE command sets the default device right away to DSK0: and deletes all files named SAE and STOR from the account on that device with the same number as the account you're currently logged into. So far, so good.

Now, notice that there is an account number ahead of the next filespec. Placing the account number in this position resets the default account number, so that the files PT4.LST and MGT.LST are both deleted from account [200,1] on DSK0:. The last filespec contains a different device name, but since the account number default has been changed to [200,1], ERASE deletes all the files with .RST extensions from that account on DSK1:.

That example might seem complicated at first, but if you analyze it one part at a time, you'll see how much you can accomplish with a single command.

Setting the Project-Programmer Number Defaults:

Here's another default feature you can use.

```
ERASE° [24,0]TAX.BAS,JR.LST[,5],[211,3]GRT.PGM,S101.SLR[212,]RETURN
```

If you look closely at the account numbers in this command, you'll notice that one is missing a project number and one is missing a programmer number. If you leave one or the other of these numbers out of an account number, wildcard commands default to the currently set default project or programmer number. In this example, the project number for JR.LST and the programmer number for S101.SLR default to the previously set project and programmer numbers. This command could also have been written:

```
ERASE° TAX.BAS[24,0],JR.LST[24,5],GRT.PGM[211,3],S101.SLR[212,3]RETURN
```

As your use of files becomes more extensive and your need to manipulate a lot of files quickly increases, you'll find that the ability to condense commands into the shortest possible space (especially in Command and Do files which we'll discuss in Chapter 11), while having them process the maximum number of files, is extremely important.

Setting File Switch Defaults:

If you put a wildcard file switch right after the command name like this:

```
ERASE/Q° TEXT1.TXT,RPT23.TXT,SUMX.TXT,FILE.TXT,GAME.TXTRETURN
```

the switch applies to everything on the command line. (In this case, the Query switch will cause the ERASE command to pause after each file for your confirmation before actually erasing it.)

If the switch comes after one of the file name,...

```
        ...RPT23.TXT/Q,SUMX.TXT,FILE.TXT,...
```

the command only stops for confirmation for that one file, since it is a file switch.

When you place the switch ahead of the file specification, it remains in effect for the rest of the command line. Such as:

```
...RPT23.TXT,/Q SUMX.TXT,FILE.TXT,...
```

From these examples, you can see that the placement of the switches in the command line can greatly alter the outcome of the command.

The preceding examples have given you just a small taste of how wildcard symbols operate. Now we can look at some of the wildcard commands themselves to see how they operate and how they can save you hours of work.

7.6∞WILDCARD SYMBOLS IN COMMANDS

All the commands described in the remainder of this chapter recognize wildcard symbols. There are countless ways to use these commands and we'll try to show you as many uses for each one as we can. It would be impossible for us to give examples of all the possible combinations of options and wildcard symbols, so we encourage you to experiment and use your imagination.

The ultimate source of information on the features and use of any command is, of course, the Command Reference Sheet for that command. Each reference sheet contains a complete listing of command characteristics, switches, defaults, etc., so refer to your *System Commands Reference Manual* for all the details on each command.

7.6.1∞DIR - File Directory Display

You've already seen a few of the things the DIR command can do for you, but now let's explore some of this command's other capabilities. Before we go any further, though, here is a list of the command's syntax and defaults.

COMMAND SYNTAX:

```
DIR° {{listfilespec}=}{filespec1{,...filespecN}}{/switches}
```

COMMAND DEFAULTS:

listfilespec	DIRECT.LST, and the device and account you're currently logged into.
filespec	*.*, and the device and account you're currently logged into.
/switch	/WIDE:1/NOBASE

How to Create a Directory File

The first thing you probably noticed about the syntax of this command is that it contains an optional listfilespec. This corresponds to the outfilespec you read about earlier in this chapter. If you specify a filename followed by an equal sign (or just an equal sign if you want to use the default DIRECT.LST), DIR creates for you a file containing the directory display you requested. Here are some examples:

```
DIR/W =RETURN
```

(Creates a file named DIRECT.LST containing a four column wide list of the files in the account you're currently logged into.)

```
DIR/W:3 FILES.LST=*.TXT,*.LSTRETURN
```

(Creates a file named FILES.LST containing a three column wide list of the files in your current account that have .TXT and .LST extensions.)

```
DIR TOC.LST[20,2]=MY??.*[24,1]RETURN
```

(Creates a file named TOC.LST containing a single column directory list in account [20,2] on your current device. This file contains a list of all the files in account [24,1] on your current device whose names contain four characters or fewer, start with the letters MY, and have any extension.)

How to Print a Directory List on the Printer

If you ever need to retain a permanent, printed record of a directory display, you can do it two ways.

a. Use the DIR command with a listfilespec to create a disk file that contains the display you want to print, and then print it with the PRNT command (more on PRNT later), or...

b. Substitute the name of the printer (or any other terminal) for the listfilespec in the DIR command. Ask the System Operator what the terminal identification of your printer is, or type TRMDEF followed by a RETURN to see a list of all the terminals attached to your system. Then to send a list, say, of all the .CMD files in all the accounts on the device you're logged into, to the printer TI810, enter:

```
DIR TRM0:TI810=*.CMD[]RETURN
```

The device spec TRM0: tells DIR that the following name applies to a terminal rather than a disk file.

How to Find a Lost File

If you're looking for a specific file, but you can't remember where it is, you can use the DIR command to locate it for you. For example, if the file you're searching for is called LOST.FIL, enter:

```
DIR ALL:LOST.FIL[] RETURN
```

This DIR command searches through every account ([]) on every disk device (ALL:) and displays the location of every LOST.FIL it finds.

How to Use Switches with DIR

The DIR command allows you to use a variety of switches. You can see a complete list of these switches and their capabilities in the DIR command reference sheet, but here are some highlights.

1. /DATA - (an operation switch). This switch causes DIR to list the complete file specification for each file it selects. Instead of displaying just the file name, for example: TARIFF.AMT[143,10]. If you create a list file of a directory display using the /DATA switch, your AlphaBASIC programs can read the file to access these file specifications and determine which files to open for data transfer.

2. /KILL - (an operation switch). Normally, when you use DIR to create a file containing a directory list, DIR won't create the file if one alredy exists with the same name in that particular device and account. The /KILL switch tells DIR to go ahead and destroy the previously created file, if there is one.

3. /WIDE - (an operation switch). This switch tells DIR to display the list of file names in four columns instead of one. You can specify a different number of columns by placing a colon and desired number after the switch. Like so:

```
DIR/WIDE:3 RETURN
```

7.6.2∞COPY - Transferring Files

Many times, you'll find it's more convenient to make a duplicate copy of a file and change it to suit a new purpose than it is to create a new file from scratch. It's also desirable to make backup copies of important files frequently to prevent their loss in case of accident or emergency. The COPY command, by virtue of its wildcard features, is suited perfectly to this task.

COMMAND SYNTAX:

```
COPY {outfilespec}={infilespec1{,...infilespecN}{/switches}
```

COMMAND DEFAULTS:

outfilespec	*.*, and the device and account you're currently logged into. If the System Operator uses COPY from account [1,2] on any disk, the same defaults apply except the default account number is the wild-card symbol [º]. This means that unless the device and account are explicitly stated, the files the System Operator copies are placed on the device the System Operator is logged into, in the account number corresponding to the one they came from.
infilespec	*.*, and the device and account you're currently logged into.
/switch	/DELETE/NOQUERY

How to Copy a File Within Your Account

To make a duplicate copy of a file, you can use the COPY command. Choose a filename that's not already being used in your account (use the DIR command to see the file names), and enter the command like this:

```
COPY NEWFIL.DAT=OLDFIL.DAT RETURN
```

where OLDFIL.DAT is the existing file and NEWFIL.DAT is the file you want to create. Now you have two copies of the same file under two different names, and you can modify one or the other to suit your needs.

How to Copy a File from Another Account

Many times the file you want a copy of is in a different account. You can copy it into your own account under the same name (if that name is not used already in your account), or you can give it a new name. Note that if a file already exists in your account with the same name, the copy is written over it and the file's former contents are lost. Here's an example of copying a file from another account:

```
COPY MINE.TXT=DSK3:YOURS.TXT[120,2]RETURN
```

In this example, the file YOURS.TXT was copied from DSK3:, account [120,2] into a file called MINE.TXT on the device and account you're currently logged into.

If you want to copy several files into your account keeping their names the same, the command might look like this:

```
COPY =DFCLT.*[120,2],WIN0:EASY.*[50,6]RETURN
```

Here we copied all files named DFCLT from account [120,2] on the device you're currently logged into, and all the files named EASY from account [50,6] on WIN0:, into the account you're currently logged into keeping the same names they had before.

How to Copy a File Between Accounts

Most of the time, you'll be copying files to and from the account you're logged into. You can also copy files between accounts without being logged into either one as long as they share the same project number as the account you are logged into.

The System Operator has the added ability to copy files to and from accounts that he or she is logged into, regardless of project number. The System Operator must be logged into account number [1,2] on any device to do this, however. If someone needs a copy of all the program files in a certain account written in AlphaBASIC, the System Operator could enter a command like this:

```
COPY DSK2:[44,10]=DSK0:*.BAS[12,3]RETURN
```

The System Operator's account should have special protection so no one can log into it without knowing the special password. If you need something like this done, submit your request to the System Operator.

Also, if the System Operator is logged into [1,2] and tries to copy files into an account that hasn't been established on the disk, COPY automatically creates the new account, and gives it the same password as the original, if any.

How to Use Switches with COPY

Like the other wildcard commands, COPY allows you to use a number of switches which enable you to alter the effect of the command. Check the COPY command reference sheet for a complete list of switches.

1. /QUERY - (a file switch). This switch tells COPY to pause before copying a file and wait for you to enter a Y (Yes) or an N (No). You don't need to press RETURN.

   If you enter Y, COPY goes ahead and copies the file, but if you enter N, it doesn't. You can deactivate the /QUERY switch with the /NOQUERY switch. Here's an example:

   ```
   COPY/Q *.MAJ=*.MIN,*.INT/NOQ,*.KEY,*.FLT,*.SHP RETURN
   ```

   This command pauses before copying all the files with .MIN, .KEY, .FLT, and .SHP extensions, but copies the .INT files without stopping.

2. /NODELETE - (a file switch). The /NODELETE switch tells COPY not to delete a file if it has the same file name and extension as the outfilespec in the command line. This prevents you from losing a file that might have a duplicate filename. The /NODELETE switch counteracts the /DELETE switch, the default, which causes the COPY command to erase an existing file before copying one with the same filename and extension to that same account and device.

7.6.3 ERASE - Deleting Files

After you're finished using a particular file, there is no need to keep it on your disk any longer. You might as well get rid of it and make room for new files. This is especially true if you already have a backup copy of the file on a removable disk or video cassette, so here is the ERASE command. We used ERASE in several examples near the beginning of this chapter, so you already know a lot about this command.

COMMAND SYNTAX:

```
ERASE {outfilespec=}filespec1{,...filespecN}{/switch}
```

COMMAND DEFAULTS:

outfilespec	*.*, and the device and account you're logged into.
filespec	the account and device you're logged into. The extension defaults to a null extension (one that's zero characters long).
/switch	/NOQUERY

How to Delete Files from Another Account

You can erase files from your own account or from another account as long as the project number is the same as your own, even if the accounts are on different devices. For example, if you are logged into account [200,2] on DSK5:, you can enter the following command:

```
ERASE *.BAK,DSK4:CRT?.*[200,5],VCR?.*[200,6]RETURN
```

In this command, ERASE deletes all the files having .BAK extensions from the account you're logged into, [200,2]. DSK4: sets the default device for the rest of the command line, and ERASE deletes all files with four character names that start with CRT from account [200,5] and all files with four character names that start with VCR from account [200,6], both on DSK4:.

If you try to delete a file that doesn't share the project number of your own account, you see an error message like this:

```
?cannot delete FILESPEC - protection violation
```

The System Operator, when logged into DSK0:[1,2], can use the ERASE command to delete any file in any account.

How to Delete Duplicate Files

When you specify an outfilespec using wildcard symbols, the ERASE command compares the infilespecs to the outfilespecs and if there is a match, it deletes the matching outfile. Here's how it works:

```
ERASE *.GRT=*.NOM[105,0]RETURN
```

ERASE selects all the files with .NOM extensions in account [105,0] and compares them to all the files with .GRT extensions in the account you're currently logged into. If there are two files with the same filename, ERASE deletes the one with the .GRT extension from your account.

You can use this feature within your own account to get rid of duplicate files you may no longer need.

How to Use Switches with ERASE

The only switches available with the ERASE command are the /QUERY and /NOQUERY switches. Both are file switches. The query switch forces ERASE to pause before erasing the files you specify. ERASE waits for you to enter either Y (Yes) or N (No) before erasing the file and proceeding to the next. It's not necessary to press RETURN after the Y or N. Suppose you're logged into account [20,3]...

```
ERASE *.BAK,/Q A*.BAS,DEBIT.*,CREDIT.*,SUM??.*/NOQ RETURN
```

In the command above, ERASE deletes all the files with .BAK extensions in the account you're logged into. The /Q switch sets the default to QUERY for the next group of files. ERASE pauses before erasing all the .BAS files that begin with the letter A, all the DEBIT, and all the CREDIT files. Finally, the /NOQ switch counteracts the /Q switch for the last group of files, all those with five character filenames that begin with the letters SUM.

7.6.4∞RENAME - Changing File Names

The RENAME command operates much the same as the COPY command. The difference is that RENAME gets rid of the file under the old name. It's really a combination of the COPY command and the ERASE command, but it saves you from having to perform those two separate steps to change the name of a file or group of files.

COMMAND SYNTAX:

```
RENAME outfilespec=infilespec1{,...infilespecN}{/switches}
```

COMMAND DEFAULTS:

outfilespec	The default device and account is ALL:[°], and the default filename is *.*.
infilespec	*.*, and the device and account you are currently logged into. NOTE: You must specify at least a partial outfilespec and partial infilespec for this command to work correctly.

/switches	/NODELETE/NOQUERY

How to Change Filenames in Your Own Account

Occasionally, you may wish to rename a group of files in your account. If you should want to change the extension of all your .BAK files to something else, like .SAV, you could enter:

```
RENAME *.SAV=*.BAK RETURN
```

If you want to organize a variety of files with different extensions into a group with the same extension, you could say:

```
RENAME *.GRP=HEADER.TXT,AR*.SAV,MKT??.LST RETURN
```

Here, we renamed the HEADER.TXT file, all the files whose names start with AR and have .SAV extensions, and all the files whose names begin with MKT, are five characters or fewer, and have .LST extensions. Now they all have .GRP extensions.

How to Change Filenames in Other Accounts

It's possible for you to rename files in accounts that have the same project number as the one you're logged into. The other account can be on the same device as your account, or on a different device, as in this example:

```
RENAME MFG*.*=DSK1:[55,2]BOM?.BAS,WRK??.LST,COG.L? RETURN
```

Assuming your project number is also 55, this rename command changes the names of all the files in account [55,2] on DSK1: whose four character names start with BOM and have .BAS extensions; all the files in that account whose five character names start with WRK and have any extension, and all the files in that account named COG whose two character extensions start with L.

These files all stay in the same account you found them in, and they keep their original extensions, but now the first three characters of all their names are MFG.

You can't move a file to a different account using the RENAME command, so don't specify an account in the outfilespec. If you do, an error message like this appears on your screen:

```
?cannot rename FILESPEC - protection violation
```

The System Operator can rename files in any account, regardless of project number, when logged into DSK0:[1,2].

How to Use Switches with RENAME

The RENAME command recognizes two pairs of switches.

1. /QUERY - (a file switch). Depending on where you place this switch in the command line, this switch forces RENAME to pause before renaming certain files and wait for you to respond with either Y (Yes) or N (No).

   This switch enables you to selectively rename files within a large group without having to enter each filename in a RENAME command line. The effect of this switch can be reversed by using the /NOQUERY switch.

2. /DELETE - (a file switch). If you specify this switch, RENAME deletes an existing file if you are renaming another file to the same name. You can turn off this switch by using the /NODELETE switch.

7.6.5 PRNT - Getting a Printed Listing

Nine times out of ten, you'll want to get a printed listing of your file. Often it will be the final copy of a letter or memo, a listing of your AlphaBASIC or AlphaPASCAL program, or perhaps a copy of the report that your program generates.

Since a printer can only print one file at a time, a special program known as the Spooler maintains a list of the surplus files until the printer finishes printing the file it's currently working on. The list of pending files waiting to be printed is called the "print queue." And as soon as one file is finished printing, the spooler checks to see if there is another file waiting in the queue to print. If there is, it automatically sends this next file to the printer.

COMMAND SYNTAX:

```
PRNT {printerspec=}{infilespec1{,...infilespecN}{/switches}}
```

COMMAND DEFAULTS:

printerspec	The default printerspec is the printer having the fewest blocks of data currently in its print queue or, more likely, the printer set as the default by the System Operator.
infilespec	The default infilespec is the account and device you're currently logged into, and extension of .LST, and a null filename.
switches	The default switches are set up by the System Operator when the Printer Spooler is installed on your system. Check with your System Operator for the current switch defaults.

How to Check the Print Queue

To find out how many files are currently waiting to be printed, or to find out how many blocks of the currently printing file remain, you can use the PRNT command without any arguments:

```
PRNT RETURN
```

If the print queue is empty, a message on your screen will tell you so.

How to Print Multiple Files

When you have a series of files to print, you can use a single PRNT command to do so. If the files in that series share common elements in their names, you can make the command even simpler by using wildcard symbols. For example, if you have written a series of chapters for a certain document, and their filenames all begin with CHAP, to print them you might enter:

```
PRNT CHAP* RETURN
```

This command causes all the files in the account you're currently logged into, whose filenames begin with CHAP and have .LST extensions, to be placed in the print queue.

If you have more than one printer attached to your system, you can specify which one you want the file printed on by entering a printerspec:

```
PRNT DIABLO=*.BAS,*.TXT RETURN
```

The command above places all the files with .BAS and .TXT extensions into the print queue for the Diablo printer.

Check with the System Operator to find out the names of the printers on your system.

How to Use Switches with PRNT

The PRNT command has a large number of switches that you can use to control the files you send to the printer. They are all described in detail on the PRNT reference sheet in the *System Commands Reference Manual*. The most important ones are described below.

1. /COPIES:n - (a file switch). Prints the number of copies you specify.

2. /KILL - (an operation switch). Deletes the files specified from the print queue. If the file is currently printing, it will stop printing when it reaches the end of the current block.

3. /FORMS:xxxxxx - (a file switch). Forces the printer to pause before printing a file if the forms name is not the same as for the previously printed file. This allows you time to change the paper. The typical is /FORMS:NORMAL if you do not use this switch. The System Operator must specify the default form name when the printer spooler is first set up.

4. /WAIT - (an operation switch). If you want to print more files than the spooler can keep track of, the extra file requests are discarded. You can use the /WAIT switch to tell PRNT not to discard these file requests if there is not room for them in the print queue. The catch is that PRNT ties up your terminal until it's able to print enough of the earlier files to make room in the queue for the extras.

In this chapter, you've read about several useful commands that use wildcard symbols. But there are many more commands in Alpha Micro's repertoire, and in the next chapter we'll discuss two of them that can lighten your workload a great deal.

CHAPTER 8

COMBINING AND SORTING FILES

This chapter introduces you to two commands that you can use to combine several files together and organize according to conditions you specify. These two commands do not recognize wildcard symbols.

Many times, a large file can be organized into several smaller sub-files which are easier to keep track of and to work with. Then, when you're ready to process all the files together, you can combine them into one. If the records in these files need to be arranged in a certain sequence, you can sort them into the proper order. Here are the two topics we'll talk about in this chapter:

- Appending several files together
- Sorting data within files

Let's start with the APPEND command.

8.1 THE APPEND COMMAND

This command allows you to combine two or more files into one. You can use APPEND to create an entirely new file that contains the same data as its components, or you can add additional data to an existing file.

COMMAND SYNTAX:

```
APPEND outfilespec=infilespec1{,...infilespecN}
```

COMMAND DEFAULTS:

outfilespec	At least a partial filespec is required. The extension defaults to a null extension (that is, an extension zero characters long), the account and device default to the account and device you're currently logged into.

infilespec	At least one partial filespec is required. The account and device default to the account and device you're currently logged into. The extension defaults to the extension of the outfilespec, and each subsequent infilespec defaults to the extension of the infilespec preceding it.

8.1.1∞How to Combine Files

For certain purposes, such as formatting a group of text files, you may wish to combine the files into a completely new file rather than processing them separately. You can do it like this:

```
APPEND BOOK.TXT=INTRO,CHAP1,CHAP2,CHA3[RETURN]
```

In this command, we combined four individual text files into a new file called BOOK.TXT. All these files defaulted to the device and account we were currently logged into, and their extensions all defaulted to .TXT. BOOK.TXT now contains a copy of each file in the order you specified, and the original files reamin intact.

8.1.2∞How to Add Files to an Existing File

You don't necessarily have to create a new file each time you append files together. You can simply append additional files to an existing one. Here's how:

```
APPEND PRGM.BAS=PRGM.BAS,RTN1.BAS,RTN2.BAS,RTN3.BAS[RETURN]
```

Note that we used the same file, PRGM.BAS, as both the outfilespec and the first infilespec. All the subsequent files are thus added on at the end of the existing PRGM.BAS file. (We could have placed PRGM.BAS anywhere in the string of infilespecs depending on what order we wanted the files arranged in.) PRGM.BAS now contains its former self plus the contents of the other three files. Those other three files remain as they were.

8.2∞THE SORT COMMAND

The SORT command arranges the records in a single sequential file in the order you specify. SORT asks you several questions to determine your criteria and arranges the records in the file accordingly. SORT only works on sequential files, and like APPEND, SORT does not recognize wildcard symbols.

COMMAND SYNTAX:

```
SORT filespec
```

COMMAND DEFAULTS:

filespec	SORT assumes the device and account you're currently logged into. The file extension defaults to .DAT (indicating a data file).

8.2.1∞How to Use the SORT Command

Every file is made up of individual records, and each record is a collection of related information about a certain subject. SORT enables you to reorganize the records within a single file in either ascending or descending alpha-numeric sequence. Each record in your file must be terminated with a RETURN. Here's a sample file that contains customer address records.

```
         1    1    2    2    3    3    4    4    5    5    6    6    7
....5....0....5....0....5....0....5....0....5....0....5....0....5....0

001   Carlson, Ander      43 Beekman Place    New York   NY   12/24/84
002   Lorenson, Will      P.O. Box 1221       San Diego  CA   04/30/82
003   Mathews, Michael    6701 23rd Street    Denver     CO   04/27/82
004   Lincoln, Thomas     12992 Dover Place   Concord    MA   08/01/83
005   Young, Helen        3225 Cutty Sark     Las Vegas  NV   06/30/83
006   Carter, Marilyn     5443 Via Verde      Tucson     AZ   02/19/84
007   Kleburg, Lou        4216 Larimore       Omaha      NE   06/30/86
008   O'Grady, Rosie      R.R. 1, Box 143     Goleta     CA   06/02/86
009   Walker, Joan        4 Pleasant Circle   Newport    RI   09/14/85
010   Smith, Steven       1675 Seaside Drive  Monterey   CA   01/25/87

....5....1....1....2....2....3....3....4....4....5....5....6....6....7
         0    5    0    5    0    5    0    5    0    5    0    5    0
```

(NOTE: The scales at the top and bottom are not part of the file; they're there to make it easier for you to figure out the relative positions of the data within each record.)

Each of the ten records in this file contains the same pieces of data: a customer ID number, the name of the customer, the customer's address, and the date of the customer's last purchase.

Take a minute or two to use AlphaVUE to create a file called CUST.DAT containing the ten records shown above.

When you're ready to sort them, enter:

```
SORT CUST.DAT RETURN
```

Now SORT asks you several questions to determine how you want the file sorted.

1. `Record size:`

   Although the records in our file are all the same length (70 characters), that might not always be the case. Some files may contain records of differing lengths, so you must enter the maximum record size in your file. If you underestimate, any records larger than the size you specify will be truncated to that length in the sorted file. The records in our file are 70 characters long, so enter that number.

2. `Key size:`

   A key is a piece of data within each record that you want the file sorted by. Let's put the file in alphabetical order by customer name. We've allowed for twenty characters in the name field of our file, so enter 20 here.

3. `Key position:`

   Now you have to tell SORT where that field begins in the records to be sorted. Looking back at the listing of the individual records, you can see that the customer name starts in position 7 of the record. Enter that number.

4. `Key order:`

   Do you want the records sorted in ascending or descending order? Answer A for ascending or D for descending.

5. `Key size:`

   If you want to specify an additional sort key, you can enter the size of a second key field here. SORT then requests the Key position and Key order for the second key. You can specify up to three key fields each time you sort a file. Or just enter a RETURN, and SORT begins rearranging your file right away.

It takes SORT only a short time to reorganize your file, and when it's done, SORT displays a group of statistics similar to the ones shown below:

```
Records sorted:  10
Runs generated from sort phase:  0
Passes over data in merge phase:  0
Record comparisons made:  20
Sector reads:  0
Sector writes:  0
```

8.2.2∞Some SORTing Hints

You can sort files that are larger than available memory (as little as 6K is enough in some cases), but as a general rule the more memory you have available, the more efficiently (faster) SORT works.

If your file won't fit into memory, SORT performs the sort on disk. This takes longer, but it still gets the job done.

When you specify the dimensions of the key fields, make sure they fall completely within the record to be sorted. For example, in our little customer file, you could not specify a key size of 6 and a key position of 69 since the record is only 70 characters long.

You can specify as many as three key fields for SORT to use. The first is the major sort key and determines how the entire file is sorted. Those records that have duplicate information in the major sort key field are arranged in order by the second key field you specify. And finally, records that have duplicate information in both the major and intermediate sort keys are arranged in order by the third, or minor, sort key field.

If you use a date as a sort key, keep the format of the date field in mind. In our customer file the date field is in the MM/DD/YY format, where MM represents the month, DD represents the day, and YY represents the year. When we sort by this field, the year is more important than the month and day. We therefore have two choices: we could either convert all the dates on our file to YY/MM/DD format, or we could split the date field into two separate keys.

If we decided to specify two separate keys, two digits for the year and five digits for the month and day, the process would look something like this:

```
SORT CUST.DAT RETURN

Record size:  70 RETURN

Key size:  2 RETURN
Key position:  69 RETURN
Key order:  A RETURN

Key size:  5 RETURN
Key position:  63 RETURN
Key order:  A RETURN

Key size: RETURN
```

And the file would be arranged as shown below:

```
         1    1    2    2    3    3    4    4    5    5    6    6    7
....5....0....5....0....5....0....5....0....5....0....5....0....5....0

003   Mathews, Michael    6701 23rd Street    Denver      CO   04/27/82
002   Lorenson, Will      P.O. Box 1221       San Diego   CA   04/30/82
005   Young, Helen        3225 Cutty Sark     Las Vegas   NV   06/30/83
004   Lincoln, Thomas     12992 Dover Place   Concord     MA   08/01/83
006   Carter, Marilyn     5443 Via Verde      Tucson      AZ   02/19/84
001   Carlson, Ander      43 Beekman Place    New York    NY   12/24/84
009   Walker, Joan        4 Pleasant Circle   Newport     RI   09/14/85
008   O'Grady, Rosie      R.R. 1, Box 143     Goleta      CA   06/02/86
007   Kleburg, Lou        4216 Larimore       Omaha       NE   06/30/86
010   Smith, Steven       1675 Seaside Drive  Monterey    CA   01/25/87

....5....1....1....2....2....3....3....4....4....5....5....6....6....7
         0    5    0    5    0    5    0    5    0    5    0    5    0
```

The two commands we've discussed in this chapter are extremely useful, but they can also cause some confusion if you use them incorrectly. Before you start manipulating your files drastically, it's a good idea to make a backup copy that you can always go back to, in case something unforeseen happens. The next chapter explains several different ways to back up your data.

CHAPTER 9

BACKING UP YOUR FILES

One of the most valuable lessons that everyone learns early in their computer career is this:

BACK UP YOUR DATA !!!

Backing up your programs and data by regularly copying them onto a spare disk, magnetic tape, or video cassette is perhaps the most important habit you can develop.

Computers are generally very reliable machines, but nobody can completely guard against such an accident as someone tripping over the power cord or spilling a cup of coffee into the disk drive. There are also natural disasters such as power outages, floods, fires, etc., that cannot be prevented, but must be planned for.

Your programs and data are irreplaceable, so keeping a current copy of them in a safe place will save you the frustration of losing several days worth of programming or data entry.

In this chapter, we will discuss the following topics:

- Backup Procedures
- Backing up a single account (COPY)
- Backing up an entire disk (DSKCPY)
- Using the BACKUP Command
- Other types of backup

9.1 BACKUP PROCEDURES

We will discuss specific backup techniques in a later section of this chapter. For now, let's talk about some other things that you can do when using the system to keep data loss to a minimum.

Since Alpha Micro supports a number of types of data recording devices, we will refer to those devices as "backup media" or "media." These include such methods of data storage as hard disks, floppy disks, 1/2" magnetic tape drives, 1/4" streaming tape drives, and video cassette recorders.

1. Whenever you change a disk cartridge or floppy disk, you **MUST** use the MOUNT command; if you don't, AMOS may well write over data that is already on the disk. (See Section 2.5.1 for an explanation.)

2. If your text files begin to show strange typographical errors (that you know you didn't make), ask the System Operator to run a memory diagnostic test program—you may have a memory problem. (The System Operator should be running memory and disk diagnostic tests on a regular schedule, anyway.)

3. Keep your text editing sessions short; never use one of the text editors for more than half an hour or so without exiting and saving your file.

   If that unexpected catastrophe should occur, it is better to have lost a half an hour's worth of work instead of a day's. (The changes and additions that you make while you are editing are only changed in your temporary memory. You have to "save" the file to permanent storage in order to make those changes permanent.)

4. As a general rule, never leave your terminal without logging off the system (using the LOGOFF command). Change your password occasionally (if you have one). These measures protect your system and accounts from unauthorized use.

5. Treat your backup media with respect. Whether your backup media are disk cartridges, floppy disks, magnetic tape, or video cassettes, handle them carefully and gently.

   DO NOT stack your media on top of the disk drives, and DO NOT leave them lying about where they can be knocked off counters or desks.

If at all possible, you should have several spare disks, tapes, or video cassettes that you can use for backup. You should never have just one backup media. (What happens if something should really go wrong while you are in the process of backing up, and you lose the data on both your backup and original media?)

The usual procedure is to follow the "grandfather-father-son" philosophy of backup. That is, you always have several backups of varying age. The next time you back up, use the media with the oldest version of your data (the "grandfather").

By rotating your backup media, you can ensure that you always have several fairly recent backups of your data.

9.1.1∞Backing Up the Files in Your Account

The command that you will use to transfer copies to your backup media will vary depending on what the media is. COPY is one of the commands you use to copy between disks.

You can use the COPY command to make backup copies of all the files in your account, all of your accounts, and even of all accounts on the disk. The COPY command is a wildcard command, so you may want to review Chapter 6, "Using Wildcard Symbols," before proceeding.

Use the SET command to tell AMOS to notify you of any soft disk errors encountered during the backup procedure:

```
SET DSKERR RETURN
```

Now, decide what files you want to back up. For example, let’s say that you are logged into DSK0:[300,4], and want to copy all files in that account to the same account on DSK1:

```
COPY DSK1:= RETURN
CRMINV.BAS to DKS1:CRMINV.BAS
RESET.BAS to DSK1:RESET.BAS
CRITEM.BAS to DSK1:CRITEM.BAS
PRINT.TXT to DSK1:PRINT.TXT
Total of 4 files transferred
```

The fact that nothing appears on the right side of the equal sign in the command line above tells COPY that we want to copy ALL the files from the device and account we are logged into.

The DSK1: on the left hand side of the equal sign tells COPY to transfer copies of those files under their own names to the same account on DSK1:.

The next command:

```
COPY DSK1:[]=[300,*] RETURN
FUNC.BAS[300,2] to DSK1:FUNC.BAS[300,1]
PRTYP.LIT[300,2] to DSK1:PRTYP.LIT[300,2]
IBUF.LIT[30,2] to DSK1:IBUF.LIT[300,2]
PARSER.LIT[300,10] to DSK1:PARSER.LIT[300,10]
Total of 5 files transferred
```

tells AMOS to copy of all the files from all of the accounts in Project 300 on the device you are logged into (DSK0:) over to the same accounts on DSK1:.

You must remember to include the wildcard account symbol [°] on the left side of the equal sign—if you do not, AMOS will copy every file on the device you are logged into (DSK0:) into the account that you are logged into on DSK1:. (In the example above, you must be logged into an account in Project 300 to avoid a protection violation error.)

You should perform file backup frequently if you create and change files often. Perhaps at the end of every working day you should copy those files onto a backup disk.

Creating a command or DO file to perform this function for you can make your task simpler; see "BACKUP.DO" in Section 10.3.3 for an example of a DO file that backs up files from one disk cartridge to another.

9.1.2∞The System Operator and the COPY Command

The System Operator (if logged into [1,2]) can use COPY to back up all of the accounts on a disk regardless of their project numbers.

For example:

```
COPY WIN1:[]=DSK1:[] RETURN
```

copies all of the files in all of the accounts on DSK1: over to their corresponding accounts on the device WIN1:.

The System Operator may omit the wildcard account symbol in the outfilespec—when it is used from account [1,2], COPY assumes a default outfilespec of *.*, the device you are logged into, and an account specification of [°].)

9.2∞BACKING UP ENTIRE DISKS (DSKCPY)

Rather than copying over individual files from one device to another, DSKCPY makes a literal image copy of an entire disk.

You can *only* copy between devices of the same type; that is, where the first three characters of the device name are the same, for example, AMS1: to AMS2:, DSK2: to DSK3:, DDS1: to DDS2:, etc. And both disks must be in the same file format, either traditional or extended.

If you are using a floppy disk-based system, place the backup disk in a drive other than the System Drive.

The backup disk must have been formatted at some time in the past. (Check with the System Operator for information on what formatting program to run to format a brand new disk.) Note that DSKCPY completely obliterates any data previously on the backup disk.

For example, let’s say you want to back up the System Disk, DSK0:, onto DSK1:.

First, MOUNT the backup disk:

```
MOUNT DSK1:RETURN
```

Then enter the DSKCPY command. The DSKCPY program first asks you for the INPUT DRIVE. Enter the specification of the device you are going to back up. Then the DSKCPY program asks you for the OUTPUT DRIVE. Enter the specification of the disk you are going to copy onto the backup device.

The entries would look like this on your terminal screen:

```
DSKCPYRETURN
Input drive:  DSK0:RETURN
Output drive:  DSK1:RETURN
```

DSKCPY begins to make a literal image of DSK0: onto DSK1:, and displays how many blocks it is copying. After it has finished the backup, DSKCPY tells you that it has finished copying the disk; then it proceeds to verify the duplication by checking the data on the back-up disk against the data on the source disk.

After this verification is complete, DSKCPY returns you to AMOS command level. The entire process looks something like this:

```
DKSCPYRETURN
Input drive:  DSK0:RETURN
Output drive:  DSK1:RETURN
[Copying 9696 records]

[Duplication completed]
[Verification completed]
```

9.3∞USING THE BACKUP COMMAND

The BACKUP command enables you to back up selected files from several disks onto a single removable disk or tape. BACKUP is a very flexible command that takes advantage of the time and date stamps associated with extended format files. BACKUP works on files in both traditional and extended format.

Enter BACKUP followed by the file specifications of the files you want to back up.

```
BACKUP DSK0:MEMO*.TXT[150,0],DSK1:TEST*.*[200,3]RETURN
```

BACKUP responds with a menu of the available backup devices. After you make your selection, BACKUP asks you to confirm that the device specification is correct. If it is, press RETURN. If it isn't, enter the correct specification.

BACKUP then gives you step-by-step instructions for completing the backup. These instructions vary according to the type of backup device you're using.

9.3.1∞BACKUP Options

BACKUP lets you customize the command line further by providing several switches:

/AFTER:date&time	Backup only files modified after the specified date and time.
/BEFORE:date&time	Backup only files modified before the specified date and time.
/BOOT	Generate warm boot tape. (Not for floppy disks.) See Appendix F.
/MODIFIED	Backup only files modified since last back-up.
/QUERY	Confirm files before backup.
/WAIT:+@HH:MM	Wait specified time before backup begins. (VCR only.)

Note: The dates and times in the /AFTER and /BEFORE switches only apply to extended format files that contain date and time stamps, and must be specified like this:

```
/switch:{month-day-year}{@hour:minute{AM/PM}}
```

Either a date or time or both must be specified. A time without a date defaults to the current date, while a date without a time defaults to zero time. A typical BACKUP command might look like this:

```
BACKUP /AFTER:01-01-1988@12:00AM DSK0:[]RETURN
```

This command backs up all the files on DSK0: that were modified after 12 midnight on the morning of January 1, 1988.

(The reference sheet for BACKUP in the *System Commands Reference Manual* for your system contains a complete list of the switches and their functions.)

9.3.2∞Restoring Files from Backups

To restore files onto your disk that were backed up using the BACKUP command, you need to use the RESTOR command. The operation of the RESTOR command is very similar to the operation of the BACKUP command.

Type the RESTOR command followed by switches, and output file specification, and the file specifiations for the files you want to restore:

```
RESTOR /DELETE DSK1:[200,0]=DSK0:*.BAS[110,2]RETURN
```

In the example above, the /DELETE switch allows RESTOR to copy over existing files on the disk with files of the same name from the backup. The specification DSK1:[200,0] is the account into which you want the files restored. And DSK0:*.BAS[110,2] is the specification from which the files were originally backed up. In this case, we are restoring all the files with .BAS extensions that were backed up from DSK0:[100,2], and we are placing them in account DSK1:[200,0].

After you enter the RESTOR command, RESTOR responds with a menu of the available backup devices, just as BACKUP did. Select the backup device you want. RESTOR asks you to confirm the specification of the device you selected. If it is correct, press RETURN; if not, enter the correct specification.

RESTOR will prompt you to prepare the backup device—read the instructions RESTOR gives you, and do what they say. RESTOR displays a list of the files on the backup medium that fit the file specifications you gave. RESTOR also displays a dot on your terminal screen for each file transferred from the backup to the disk.

RESTOR recognizes many of the same switches as BACKUP. See the RESTOR command reference sheet in the *System Commands Reference Manual* for a list of the switches you can use with the RESTOR command, and a list of the messages RESTOR might display to you.

Note that when you restore a file that was backed up with the BACKUP command, you **MUST** use the RESTOR command.

9.4∞OTHER BACKUP COMMANDS - TRADITIONAL FILES ONLY

AMOS also supports other VCR backup commands (VCRSAV, VCRDIR, and VCRRES), magnetic tape backup commands (MTUSAV, MTUDIR, and MTURES), and streamer back-up commands (STRSAV, STRDIR, and STRRES). However, these commands only work on traditional format files. If you try to backup an extended format file, these commands will ignore it.

The VCRxxx, MTUxxx, and STRxxx commands are supported for those systems that still have old backups that were created with them, and for those systems which still use command files that contain these commands.

IMPORTANT NOTE: You can only restore a file from a backup if you use the related command that was used to save it. That means, to restore a file that was backed up with the VCRSAV command, you **MUST** use VCRRES. To restore a file that was backed up with the MTUSAV command, you **MUST** use MTURES. And to restore a file that was backed up with the STRSAV command, you **MUST** use STRRES.

9.5∞FOR FURTHER READING

If you want to learn more about any of the commands we've discussed in this chapter, you can look up their reference sheets in the *System Commands Reference Manual* for your system.

Also, information on the 1/2 inch magnetic tape drive can be found in the manual *Magnetic Tape Backup Software* for your system.

Information on the backup of traditional format files onto video cassettes can be found in the manual *Video Cassette Recorder Backup Software*.

Information on the backup of traditional format files onto 1/4 inch streaming tape drive can be found the manual in *1/4" Streamer Backup Software*.

Finally, there are programs for backing up onto floppy disks from Winchester disks. See the *System Operator's Guide* for your system.

CHAPTER 10

ACCOUNTS AND PASSWORDS

Every user of the system has one or more accounts in which to create, store, and edit files. When you create files on a disk, the system marks those files as belonging to the account you are logged into.

Each account has a directory that lists all of the files in that account. It may seem easier to have just one account, but it can get very hard to find your files if you have a lot of them. The best way to organize your accounts is by subject. If you set up each account so that it contains the same type of files (for instance, one account for storage, one for memos, one for letters, one for inventory documents, one for purchase order records, etc.), it will be easier to find the files you are looking for.

To provide security for your programs and files, you can password-protect your accounts. In this chapter, we will discuss accounts and passwords and their use. The subjects covered will be:

- Project-Programmer numbers
- Passwords
- Logging into another account
- Start command files
- Sending messages to other system users

10.1 PROJECT-PROGRAMMER NUMBERS

Accounts are assigned to each user on the system by the System Operator. When you are assigned an account, you are given the account number, which consists of a project number and a programmer number.

Here are some examples of project-programmer numbers:

```
[110,5]   [334,7]   [250,12]   [200,1]   [100,100]
```

The account number is a unique, two-part number that distinguishes your accounts from all other accounts on the same device. The first number is called the project number.

If several users' accounts have the same project number, those users are said to be in the same project. Users in the same project have certain privileges when it comes to transferring files between each others' accounts.

For instance, it is easier to copy files between accounts that are in the same project. See Chapter 6, "Using Wildcard Symbols", the COPY command, for example.

Here is an example of several accounts that are in the same project (100):

```
[100,1]   [100,7]   [100,44]
```

The second number is called the programmer number, and is separated from the project number by a comma (in the example above, the programmer numbers are 1, 7, and 44). You will usually see account numbers enclosed within brackets.

The project number may range from 0 to 377, octal; and the programmer number may range from 0 to 376, octal. The largest account number possible, then, is [377,376]. (For a discussion of the octal numbering system, see the manual *Introduction to AMOS*; for now, the fact that the numbers are octal just means that no single digit may be greater than 7.)

A programmer number of zero usually indicates a library account for that project (for example, the account [311,0] contains files of interest to all of the users in project 311).

Project numbers 1-77 are reserved by Alpha Micro for system software and for the AlphaACCOUNTING business package. See Section 7.3.3 for a list of the system software accounts and their ersatz device specifications.

10.1.1∞The Root Account

On a system which supports the user names feature, everyone has a special root account in addition to one or more project accounts. The root account is much the same as all the rest of the accounts except that it has a special role. Associated with your user name, the root account identifies you personally to system and application software. This central or home account is where you can touch base regularly to read reminders and messages sent to you.

10.1.2∞The Mail Account

If AlphaMAIL is supported on your system, your System Operator will give you a mail account where you can receive incoming mail. See your System Operator for more information.

10.2∞PASSWORDS

The System Operator may assign you an account password. The password is for your protection; if you maintain its secrecy, other users cannot log into your account.

You do not have to enter a password if you are transferring to an account that is in the same project. That is, if you are in account [100,2] and you want to log into account [100,1], you would not have to enter the password if account [100,1] were password protected. You would have to know the password if you were transferring from an account such as [105,5].

If an account is password protected, you will be asked to enter the password when you log in:

```
LOG 100,3RETURN
Password:
```

The system does not display your password on the terminal screen as yo type it. This prevents anyone from seeing what you entered. Remember that the purpose of the password is to keep unauthorized users from gaining access to the computer through your account. Keep you password a secret!

Once you type your account number and password correctly, AMOS will tell you which account you are logged into. On a system that supports the user names feature, AMOS will check for your user name and optional user password before verifying that you are logged in. Below is a summary of logging in with the log command using numeric accounts:

```
WITHOUT USER NAMES                 WITH USER NAMES

LOG 100,1RETURN                    LOG 100,1RETURN
Password:                          Password:
Logged into DSK0:[100,1]                User name:
                                             User Password:
                                        Logged into DSK0:[100,1]
```

If you make a mistake in typing your account number or password, you will see one of the following error messages:

```
?Account number invalid
?Bad password
?Command format error
```

and you will have to re-enter the LOG command with the correct information.

Logging in with the LOGON command requires a user name. The user password is optional. When you have made yourself known to the system, you are automatically logged into your root account.

10.3∞LOGGING INTO ANOTHER ACCOUNT

Once you are logged into an account on the system, you can transfer to another account by using the LOG command with the account number of the account you want to transfer to:

```
LOG 100,3RETURN
Password:
Transferred from [321,10] to [100,3]
```

You can take shortcuts if you are transferring to an account with either the same project number or the same programmer number. For example, if you are in account [100,1] and wish to transfer to account [114,1], you can type:

```
LOG 114,RETURN
```

and AMOS will log you into [114,1] since you were already using programmer number 1. Likewise, if you are in [100,1] and you want to transfer to [100,4], type:

```
LOG ,4RETURN
```

and AMOS will log you into [100,4].

If the account you wish to log into is on another disk, you might have to specify that device in the LOG command. You have to specify the device if there is another account with the same account number somehwere else, or if the device is on a different type of disk (for instance, transferring from an account on DSK0: to one on WIN1:).

For example, if you are currently logged into DSK3:[K107,3], and you wish to transfer to DSK2:[100,5], enter:

```
LOG DSK2:[100,5] RETURN
```

You should enter the DSK2: here to make sure that you get to the right place. If you did not, and DSK3: also contained an account [100,5], that is where you would end up.

If you're absolutely sure that there is only one account [100,5] on the system, you don't have to enter the DSK2:—AMOS will search the other logical units of the same physical device beginning with unit #0; however, if that account is on a different physical device, such as WIN1:, you **must** include the device specification in your LOG command.

10.4 THE START COMMAND FILE

Whenever you log into an account, AMOS looks for a file in that account named START.CMD. If such a file exists, the system assumes that it is a command file, and begins executing it as such.

You can use this special command file to perform certain procedures automatically every time you log into an account, such as erasing all backup files, displaying a directory listing for that account, etc. It is also useful for placing a message on the screen that gives the purpose of the account, and perhaps what files are contained in it.

Command files of any other name will not execute automatically when you log into the account, only the file named START.CMD. Every account can have its own START.CMD file, if you wish. We discuss command files further in the next chapter.

10.5 SENDING MESSAGES TO OTHER USERS (SEND)

The SEND command allows you to send messages to other jobs while you are logged into the system.

You simply type SEND, a space, the job name of the person you wish to send the message to, and the message you would like to send.

The message can be one line long. The length of the line on your screen is defined in the system initialization command file. This is usually a little longer than the width of the screen. You can, of course, send more than one message, if it will not fit on one line. A typical SEND command might look like this:

```
SEND BILL WILL YOU BRING OVER THE SALES REPORT FOR LAST WEEK? RETURN
```

This message will print out on the terminal of the user whose job name is BILL. It is usually effective, and an easy way to communicate short messages to other people on your system.

The person whose jobname is BILL will see:

```
JOB -- WILL YOU BRING OVER THE SALES REPORT FOR LAST WEEK?
```

where JOB is the jobname of the person who sent the message—in this case, you.

This is not a foolproof method, because sometimes the other user will be doing something that prevents the message from appearing, such as running a program. In this case, you will see a message on your terminal:

```
        ?Busy
or
        ?Guarded
```

indicating that your message did not get through. You can try again later, or use another method of contacting that person.

CHAPTER 11

COMMAND FILES, DO FILES, AND CONTROL FILES

One of the constants of everyday life is repetition. Especially in business, you find that certain tasks have to be done on a regular basis. The mail has to be opened every day, somebody has to make the coffee, and if you don’t do the payroll each week, people tend to get upset.

Some of these tasks are done on your computer. One of the great strengths of computers is their ability to free people from repetitious tasks, and to speed up the completion of those tasks. However, sometimes the operation of the computer itself can be repititious.

Many times a certain task on the computer is done by using the same string of commands and operations every time. Alpha Micro provides special files called command, DO, and control files that can contain these strings of commands and operations, making the ordinary operation of your computer easier.

This chapter will discuss the following points:

- What is a command file?
- How to create command files
- How to create DO files
- The Task Manager

11.1 WHAT IS A COMMAND FILE?

One of the important features of the Alpha Micro system is that rather than having a set of "built in" commands that AMOS recognizes, all of our commands are actually files stored on the disk.

In examples in earlier chapters, it looked like you were just typing the name of the command, but what you were really doing was telling AMOS to find and execute the file with that name.

Because of this feature, you can extend and customize the set of AMOS commands that already exist on your system by creating your own executable files.

And, you can design your own commands, combining the existing AMOS commands with special instructions. You can also erase or rename AMOS commands that do exist, if there are some commands you do **not** want people on your system to use, or if you want to control **who** knows and can use certain commands.

And the easiest way to do these things is to build command files.

A command file is a file that contains the same kinds of input that you might enter from the keyboard. You can tell AMOS to read its instructions from a command file instead of having to enter those commands and data yourself.

For example, let's say that you often erase all of your backup files, and then view the directory of your account. The process that you must go through to accomplish this task is:

```
ERASE *.BAK RETURN
MEMDOC.BAK
COMF.BAK
Total of 2 files deleted, 3 disk blocks freed

DIR RETURN
MEMDOC   TXT    12             DSK0:[100,4]
COMF     TXT     5
VAR      BAS     10
VAR      RUN     7
Total of 4 files in 34 blocks
```

The example above is a simple one—sometimes a frequently used sequence of commands may be quite long and tedious to type. An extremely powerful tool for dealing with this problem is the command file.

Let's say that you create the command file CLEAN.CMD to perform the functions in the example above. The file contains the following lines of text:

```
;This file erases .BAK files and
;   displays account directory.
:T
ERASE *.BAK     ; Get rid of the backup files.
DIR             ; Display Files
```

The :T at the front of your command file is a special symbol that allows you to see the lines of your command file on your terminal display as AMOS processes them. The semi-colon (;) indicates that the text following it is a comment. All of these special symbols will be discussed later in this chapter.

With this command file, instead of entering what you entered in the first example each time, you merely have to enter:

```
CLEAN RETURN
```

and AMOS reads the file CLEAN.CMD and performs the process for you. With this small example, you do not save much typing, but imagine the typing and time you would save if the sequence of commands you regularly enter is ten or twenty lines (or more) long!

If the extension of a command file is .CMD or .DO, you do not have to include the extension when entering the name of the command file (you saw that you only had to type CLEAN rather than CLEAN.CMD).

Otherwise, you must specify the file extension; that is, if your command file is called DOIT.TXT, you must include the .TXT when specifying the file. This is why it’s easiest to use .CMD (or .DO) extensions.

11.2 HOW TO CREATE A COMMAND FILE

To create a command file, use AlphaVUE to make a text file (usually with a .CMD extension). Fill the file with the commands you would ordinarily enter from the keyboard, plus any special symbols.

A command file can contain most commands or data that you might enter from the keyboard; the file can even contain the name of another command file.

AMOS continues to read lines of text from the command file until the file ends. You may run, enter, and exit programs; supply data to programs, or perform system functions—all under the control of a single command file.

WARNING: Do not use the MEMORY command inside a command file. This command changes the memory allocation of your job; that is, it changes the section of computer memory that you have been assigned. Since your command file and any other files it is using are stored in memory, changing that memory may cause serious problems.

11.3 HOW IS A .DO FILE DIFFERENT FROM A COMMAND FILE?

A DO file is a special type of command file that allows you to pass arguments to the file. This type of file has a .DO extension and contains exactly the same type of elements as a regular command file (including the special symbols mentioned above), but also includes some additional symbols that allow you to specify items of text to be substituted into the command file at the time that you run the file.

Because you can pass text items to a DO file, you can use DO files in many different situations in which a regular command file would be too specific.

11.4 HOW DO I FIND OUT MORE ABOUT COMMAND AND DO FILES?

Because of the great variety of commands and symbols that can be used in command and DO files, Alpha Micro has a manual specifically on this subject, the *Command Files User's Manual*. Please see this document for further information.

11.5 TASK MANAGER CONTROL FILES

Alpha Micro computers also support a system called the Task Manager, which allows you to perform tasks without tying up your terminal.

The Task Manager is a batch processing program that allows you to use real or pseudo terminals to run tasks without human control.

If some tasks that you do take a long time to run (an hour, for instance), you can have the Task Manager perform them and still be able to use your terminal for other things.

You can also schedule tasks to run at specific date and times, and you can schedule tasks to remain permanently in the queue, running at specified intervals (such as weekly, monthly, etc.).

See the *Task Manager User's Manual* for more complete information about the use of the Task Manager.

Check with your System Operator to see if the Task Manager is installed on your system.

11.5.1 How are Control Files Used?

The Task Manager uses control files to give it directions. These control files are similar to command files, but have a different set of special symbols, and are more versatile. Almost anything you do on your terminal can be done in a control file.

Commands are entered into the control file just as they would be on the terminal.

11.5.2∞Submitting Tasks

The SUBMIT command is used to sumbit control files to the Task Manager. The format is:

```
SUBMIT {Control filespec}{/switch1.../switchN}
```

For example:

```
SUBMIT INVEN.CTL/RESTART RETURN
```

If you do not specify a control filespec, the program will display the contents of the queue file. See the SUBMIT command reference sheet in the *System Commands Reference Manual* for the switches and their use.

Once the control file has been submitted, the Task Manager executes the commands in the control file to complete the task. The Task Manager allows you to control exactly when the task is performed, and gives you the capability of having tasks repeat automatically at regular intervals.

CHAPTER 12

USING ALPHA MICRO PROGRAMMING LANGUAGES

In this chapter, we will discuss the use of the various Alpha Micro programming languages. Of course, we will leave detailed instructions to the specific manuals for each language, but we will give you an idea of what each language can do, what applications each language would be best for, and how to access each language.

These are just the standard languages that are generally supported. Your Alpha Micro dealer may have other languages available. See him or her for information about optional languages and their capabilities.

The topics discussed will be:

- AlphaBASIC
- Alpha Micro Assembly Language

In addition, there are several other programming languages that are available individually, such as AlphaCOBOL, AlphaFORTRAN, and AlphaC. Consult your Alpha Micro dealer for further information.

12.1 WHAT IS ALPHABASIC?

The BASIC language is probably the most commonly used computer language, especially for microcomputers.

The acronym BASIC stands for *Beginners' All-purpose Symbolic Instruction Code*. BASIC is a higher level programming language that was designed to be versatile tool for learning computer programming, and also to provide a relatively simple language for a wide variety of applications.

BASIC is an easy to use programming language primarily because of its similarity to the English language. Don't let the word "Beginners" in its name fool you. BASIC is a good language for newcomers, but it is also a powerful language used by all types of programmers.

Over the years since its inception, BASIC has been added to and modified as new concepts of programming have emerged. Some versions of BASIC are more extensive than others; the use of these extended versions allows the programmer a wider range of applications, greater ease in programming, and/or greater efficiency and speed.

AlphaBASIC is just such an extension of the BASIC language, with several features not found in other implementations. For example, AlphaBASIC provides a method for defining sophisticated data record structures that adds flexibility and efficiency to business-oriented programs.

These features not only enchance the performance of traditional uses of the language but also make business applications easier to program.

12.1.1∞How to use AlphaBASIC

You can use AlphaBASIC in either of two modes: interactive or compiler mode.

INTERACTIVE MODE

In Interactive mode you create, edit and test your program while it resides in temporary memory. This mode is convenient for the creation and debugging of new programs or for learning AlphaBASIC.

Type BASIC and a RETURN, and you see:

```
BASIC RETURN
READY
```

There is no prompt symbol, just type in either single lines, or your program. To exit, type:

```
BYE RETURN
```

Here is a small example of how an AlphaBASIC program might look in interactive mode:

```
BASIC RETURN
AlphaBASIC Version xx.(xxx)

READY
10 ! Program to print name in reverse RETURN
20      STRSIZ 20 RETURN
30 START:  INPUT LINE "Enter your name: ",NAME$ RETURN
40      IF LEN(NAME$) = 0 THEN GOTO START RETURN
40 LOOP:   COUNTER = LEN(NAME$) RETURN
60      FOR I = 1 TO COUNTER RETURN
70        PRINT NAME$[COUNTER;1]; RETURN
80        COUNTER = COUNTER - 1 RETURN
90      NEXT RETURN
100     PRINT RETURN
110     INPUT "Another one? (Y or N):  ",QUERY$ RETURN
120     IF UCS(QUERY$) = "Y" GOTO START ELSE PRINT "ALL done." RETURN
130     END RETURN
RUN RETURN
COMPILING
Compile time was 0.008 seconds, elapsed time was 0 seconds
Enter your name:  ALPHA MICRO RETURN
ORCIM AHPLA
Another one? (Y or N):  N RETURN
All done.

CPU time was 0.043 seconds, elapsed time was 1 seconds

READY
BYE RETURN
```

COMPILER MODE

Compiler mode is more useful for programs which are to be put into production use, or for testing programs which are too large to fit in memory in the interactive mode.

In compiler mode, you compile the program at AMOS command level and store the compiled program on the disk.

During the actual running of the compiled program, only the compiled program and a minimal run-time execution package need to be in memory, thereby conserving memory space.

To use compiler mode, use one of the text editing programs (AlphaVUE is the best for programming) to create your program. When you are back at AMOS Command level, enter:

```
COMPIL MYPROG RETURN
```

Once your program is successfully compiled, you can execute it by typing:

```
RUN MYPROG RETURN
```

COMPIL assumes an extension of .BAS, and RUN assumes an extension of .RUN.

12.2∞ALPHA MICRO ASSEMBLY LANGUAGE

All of the programming languages discussed above are called "higher-level" languages. That means that they are one step removed from the basic machine language the computer understands. This makes them easier to use and more efficient for normal programming needs than the more hardware-oriented machine language.

However, it means that they are not appropriate for programming certain applications that require high speed execution and the ability to communicate with hardware devices.

The programming language that you use to create machine language programs is called "assembly language."

For information on the assembly language for AMOS/L, AMOS/32, and VMEbus computers, refer to the *AM-100/L Instructions Set* and the *Assembly Language Programmer's Manual*.

CHAPTER 13

WHERE TO GO FROM HERE

In the previous chapters you've learned many of the most useful commands and features of the Alpha Micro Operating System, but of course there is still much more. By now, you've mastered the basic skills you need to operate your computer system and to create and edit your files and programs.

You know everything that all Alpha Micro users need to know, and now you can move on to the more specialized topics that apply to your particular equipment configuration and your specific business requirements.

Whether it is a programming language, a text-editing program, or a hardware technical manual, we have documentation available. This chapter will give you some guidelines for finding the right document for your needs. The points we will discuss are:

- Other useful commands
- References to other manuals
- Guidelines for the System Operator

13.1 OTHER USEFUL COMMANDS

There is such a wide variety of commands already available on your Alpha Micro computer system that you will find yourself using some commands all the time, and others not at all. To familiarize yourself with all the available commands, skim through the *System Commands Reference Manual* for your system.

Here are a few commands we feel might be particularly useful.

SRCCOM	Compares the text of two files and shows you where they are similar and different. This is useful if you are not sure which of two files is the latest version.

CREATE	This command creates an empty random file, which you can then access later and fill.
MAKE	This command creates an empty sequential file, which you can then access later and fill.
SET	This command has many options. It allows you to set certain characteristics of your terminal and other devices.
DSKANA	This command scans the specified disk for errors. It is a privileged program that can only be used from account DSK0:[1,2], and is usually run by the System Operator. See the Cautions in the DSKANA reference sheet before attempting to use this command.
DATE	Displays the time on your terminal. For example:

```
DATE RETURN
Friday, April 22, 1988
```

TIME	Displays the time on your terminal. For example:

```
TIME RETURN
3:00:14 PM
```

TRMDEF	Displays a list of the terminals that are defined on your system.
DEVTBL	Displays a list of all devices defined on your system.
PPN Devn:	Displays the current accounts for a specified disk or device.

13.2∞OTHER REFERENCES

The Alpha Micro Software Documentation Library is made up of eight volumes of manuals. The first two volumes, "System Tools," Volumes 1 and 2, are included with each new computer system. In them, you will find a document entitled "Current Revisions of AMOS Software Documentation" which lists the titles, part numbers, and revision levels of all the currently available documentation in alphabetical order. This document also lists the contents of each volume of the Documentation Library. Please refer to this list when ordering manuals from your Alpha Micro Dealer.

13.3 IF YOU ARE THE SYSTEM OPERATOR

On Alpha Micro multi-user systems, one user is usually designated as the System Operator. If you are the System Operator, you have a few additional responsibilities, and you must have knowledge of some things that other users don't have to worry about.

The System Operator:

★ Maintains the system initialization command file.

★ Assigns accounts and passwords to the other users.

★ Sets up the hardware (like printers, tapes, etc.).

★ Manages system memory.

★ Controls the special system libraries.

★ Periodically tests memory and disk for errors.

You can find special instructions for these and other tasks in the *System Operator's Guide* for your computer system.

APPENDIX A

AMOS ERROR MESSAGES

Below is a list of error messages that you might see when operating at AMOS command level. For information on the specific error messages you might see when using a particular command, see the documentation for that command.

The format of the error message may differ slightly among different versions of AMOS, but the errors and their solutions are the same.

A.1∞COMMON ERROR MESSAGES

The most common kind of error message that you will see on the system consists of two parts. The message begins with:

```
?Cannot {init, open, close, read, write, input, output, wait,
delete, rename, assign, deassign, allocate record on,
deallocate record on, read bitmap on, write bitmap on, lock
directory on, unlock directory on, allocate random file, mount,
unmount, load, or access} Filespec or Devn: -
```

and ends with one of the messages below (for example: `cannot read AMS3: - disk not mounted`):

`- BADBLK.SYS not found`

The program BADBLK.SYS was not found on the disk, meaning that it was erased or somehow lost. See your System Operator for help.

`- BADBLK.SYS has a bad hash total`

The BADBLK.SYS program is damaged. See your System Operator for help.

`- BADBLK.SYS is in wrong (unsupported) format`

You will only see this error if you have an AMOS/L system and your version of BADBLK.SYS was certified on an AMOS system that was version 4.2 or earlier. The AMOS/L system will not support it. To correct this problem, your System Operator will have to transfer your files to an AMOS system that has version 4.3 or later, and then re-certify.

`- bitmap kaput`

Every time the system writes data to a file-structured device (such as DSK0:, AMS1:, etc), it checks that device's bitmap to determine where on the disk to write the data. (A bitmap is a map of the device—it tells the system what records on the device already contain data, and which are free for use.)

Every time the system checks the bitmap, it totals the data in that bitmap and checks it against the value it has previously computed; if there is a discrepancy, it means that something in the bitmap has changed, and you see the bitmap kaput error message.

To solve this problem, MOUNT the disk. If you still get this message, check with the System Operator for help—he or she can run the DSKANA program that will reset the bitmap.

`- buffer not INITed`

Before your assembly language program can perform an open, read or write operation, the program must assign a buffer to the file DDB. (Because you can use the monitor call INIT to do this, we say that the buffer has been INITed.) If you fail to do this procedure, you will see this message.

If you should see this message within AlphaBASIC, the message may indicate memory problems; check with the System Operator.

`- deadly embrace possible`

You are trying to OPEN a file out of sequence from the sequence defined in the LOKINI initialization file. This message originates from the AMOS file-locking system. Check the LOKINI initialization file for the proper file-opening sequence.

`- device does not exist`

You've tried to access a device that the system does not recognize. For example:

```
TYPE ASM1:FILE.TXT RETURN
?Cannot init ASM1:FILE.TXT - device does not exist
```

You probably mis-typed the device name. You can use the DEVTBL command to see a list of all the devices on the system.

`- device error`

A hard disk error has occurred; that is, the system was not able to read data from a disk. Try to perform the operation again. If you still have no success, check with the System Operator for help; this message may indicate a hardware problem.

`- device full`

There is not enough room on the disk to complete the data transfer. Start over again with another device, or make room on the first device by erasing unnecessary files.

`- device in use`

Another user is using the non-sharable device that you wish to access (such as the VCR). Wait and try again later.

`- device not mounted`

You have tried to access a valid system device, but that device is not mounted. Use the MOUNT command to mount the device, and try again.

`- device not ready`

You are trying to access a device that is not ready. For example, you will see this message if you try to mount a disk before that device is powered up and ready.

`- disk not mounted`

The disk drive that the command tried to access is not mounted. Use the MOUNT command to mount that disk, then try again.

`- file already exists`

You've tried to create a file that already exists. For example, if the file NEWCPY.MAC already exists in your account, and you try to rename another file to that name:

```
RENAME NEWCPY=WRKFIL RETURN
WRKFIL.MAC to NEWCPY.MAC
?Cannot rename WRKFIL.MAC - file already exists
```

`- file already open`

Your assembly language program tried to open a file that is already open. Check your program to see if you are opening the file twice.

`- file cannot be DELETEd`

You tried to delete a file that was coded as undeletable. Check the protection code of the file.

```
- file cannot be RENAMEd
```

You tried to rename a file that was coded as unrenamable. Check the protection code of the file.

```
- file in use
```

You tried to OPEN a file which is being used by another user. Either you or the other user requested an exclusive lock on the file. This messages originates from the AMOS file-locking system.

```
- file not found
```

AMOS cannot find the file you've specified. For example:

```
TYPE LABDAT RETURN
?Cannot open DSK0:LABDAT.LST - file not found
```

Check your spelling, and make sure that you've specified the correct device and account.

```
- file not open
```

Your assembly language program has tried to access a file that is not open for input. Check your program to see if you are accessing the correct file.

```
- file specification error
```

You did not properly specify the command. Check your format and spelling and try again.

```
- file type mismatch
```

You've tried to use a program designed for sequential files on a random file (or vice versa). For example, the TYPE program works only on sequential files; if you try to use it on a random file, you see:

```
TYPE PRTIDX.DAT RETURN
?Cannot open DSK0:PRTIDX.DAT - file type mismatch
```

```
- first logical unit is not mounted
```

The first logical unit accessed by the command is not mounted. Use the MOUNT command to mount the unit once you have determined which device or unit is not mounted.

`- illegal block number`

Your program has tried to access a disk block that doesn't exist, or that is beyond the range of the file you are using.

`- illegal user code`

You've tried to access an account that does not exist. For example, if there is no account [100,2] on DSK1:

```
TYPE DSK1:MYFILE.TXT[100,2]
?Cannot open DSK1:MYFILE.TXT[100,2] - illegal user code
```

`- insufficient free memory`

There is not enough room in memory to complete the operation you are attempting. Make sure that no unnecessary modules are in your memory partition.

`- insufficient queue blocks`

There were not enough queue blocks to perform the requested operation. Try the command again, and if that does not work, have the System Operator increase the amount of queue block space in the system initialization command file.

`- invalid filename`

You've specified a filename that AMOS does not understand. For example:

```
MAKE  .TXT
?Cannot open - invalid filename
```

In this case, the filename is a space which AMOS does not recognize as a filename.

`- MFD is damaged`

Your Master File Directory is damaged. Copy as many programs as you can onto a backup media, and then run the SYSACT program with the I (initialize) option.

`- PPN does not exist`

You've tried to access a user account that does not exist. For example, if there is no account [100,2] on DSK1:

```
TYPE DSK1:MYFILE.TXT[100,2]
?Cannot open DSK1:MYFILE.TXT[100,2] - PPN does not exist
```

```
- protection violation
```

You've tried to create a file in an account other than your own that is outside of your project. For example:

```
COPY [210,3]=ELIPSE.BAS[100,5]RETURN
ELIPSE.BAS[100,5] to ELIPSE.BAS[210,3]
?Cannot open ELIPSE.BAS[100,5] - protection violation
```

```
- record in use
```

You tried to access a record inside a file that is already locked by another user. Either one or both users have requested an exclusive lock on that record.

```
- record not locked for output
```

You are attempting to output a record that was not previously read and locked via an INPUTL statement.

```
- write protected
```

You are trying to write data to a device that is write-protected. Make sure the write-protect buttonis off and try again. Make sure also that you were writing to the correct disk.

A.2∞OTHER ERROR MESSAGES

In addition to the error messages above, you can also see:

```
?Address error at nnnnnnnn
```

An address error occurred while your job was running. This error appears whenever a program tries to access a word or longword on an odd address boundary.

```
AMXXX ERROR n FOR DRIVE n BLOCK n (CYLINDER n HEAD n SECTOR n)
```

You will see a message something like this if you have SET DSKERR, and a hard error occurs on a system that uses an AM-500, AM-420, or AM-410 Disk Controller Hard Disk Subsystem (the XXX stands for whatever controller you have). This message indicates a hardware problem—the Disk Controller wasn't able to successfully read data from the disk.

The message tells you what kind of error occurred (refer to the information supplied with the disk drive to find out what error conditions corresponds to that error code), and where on the device the data transfer operation failed (the drive, record, cylinder, head and sector). Check with the System Operator for help.

```
Bus error - PC #
```

A bus error indicates that an illegal condition was recognized on the data bus. The number following the letters "PC" tell you the memory address the Program Counter was set to when the bus error occurred.

```
?Bus error at nnnnnnnn
```

A bus error occurred at location nnnnnnnn. This is a special type of error generated by the central processor hardware.

```
?CHK instruction at nnnnnnnn
```

Your job executed a CHK instruction which caused the trap to be taken. (This instruction is not currently supported on AMOS/L systems.)

```
?Command terminated - insufficient memory
```

You tried to execute a program from inside a command file, but there was not enough room to load the program into your memory partition.

Use the MAP command to see what modules are in your memory partition; delete those you do not need. If you still cannot use the command file, talk to your System Operator about getting more memory area allocated to your job.

```
?Divide by zero at nnnnnnnn
```

Your job attempted to perform a divide (DIV or DIVS) instruction with a divisor of zero.

```
?EM1111 at nnnnnnnn
```

Your job executed one of the EM1111 instructions which are not currently used by AMOS/L; they are reserved for future use. This error is commonly caused by a program which begins executing in non-existent memory.

```
ERROR n
```

You see this message if you SET DSKERR and a soft disk-error occurs on a system using an AM-200 or or AM-210 floppy disk Subsystem. Frequent soft errors can indicate hardware problems with the drives; check with the System Operator.

The number following ERROR indicates the type of soft error that occurred; refer to the instructions supplied with the disk drive to see what error condition corresponds to that error code.

Because this message reports a soft error (the system had to retry a data read) and not a hard error (the system could not read the data at all), you do not necessarily have to worry unless you see a DEVICE ERROR message (see above), which indicates a hard error.

```
ERROR n CMD n, STS n, RECORD n (TRACK n, SECTOR n)
```

You see this message if you have SET DSKERR and a soft disk-error occurs on a system that is using an AM-400 Hard Disk Subsystem. The number following ERROR indicates the type of soft error that occurred; refer to the instructions supplied with the disk drive to find out what error condition corresponds to that error code. The message tells you where on the disk the error occurred. Check with the System Operator.

```
?Illegal instruction at nnnnnnnn
```

Your job attempted to execute an illegal instruction. Any bit pattern not currently recognized by the CPU is considered an illegal instruction.

```
?Illegal user interrupt on level n
```

An interrupt occurred on level "n" when none was expected. This can be caused by a program that does not initialize the interrupt vector locations properly, or by faulty hardware that generates spurious interrupts.

```
?Insufficient memory for program load
```

You do not have enough memory in your partition to load the program you want to execute. Use the MAP command to make sure that no unnecessary modules are in your memory partition. If you still receive this message, check with the System Operator to see if he or she can allocate more memory to your job.

```
?Login please
```

You've tried to enter an instruction to AMOS, but you are not logged into the system. Use the LOG command (see Section 2.5, "Logging on and off the System"). If you need help in figuring out what to do, you can use the HELP command even if you are not logged in (see Section 3.1, "The HELP Command").

```
?Memory allocation failed
```

You used the monitor call GETMEM from within an assembly language program to allocate space for a memory module within a memory partition, and there wasn't enough room in the partition to perform the allocation. Make sure that no unnecessary modules are in the memory partition, or see the System Operator about increasing your memory.

```
?Memory map destroyed
```

Each module in your memory partition maintains a pointer to the address of the next module in memory; if these connecting links become confused or broken, AMOS is not sure where your memory modules are in your partition.

You may not need to reset the computer, but you may want to delete the modules from memory and reload them just to be sure that your memory map is intact.

```
?Memory parity error
```

A memory parity error occurred whil your job was running. After this error occurs on an AM-100/L system, you should examine the memory boards to determine which one the error occurred on. On both AM-100/L and AM-1000 systems, the System Operator may want to perform memory diagnostics to isolate the cause of the parity error.

```
?No memory available
```

There is no more free memory available on the system. Consult with the System Operator. The solution may be to change the amounts of memory allocated to each job on the system.

```
?Privilege violation at nnnnnnnn
```

Your job tried to execute a privileged instruction while in user mode. Certain privileged instructions, such as STOP, require that you be in supervisor mode.

```
?Privileged program - must be logged into OPR:
```

The program that you called can only be run from account DSK0:[1,2]. This is the System Operator's account, and you should have the System Operator run the command or program for you.

```
?Privileged program - must be logged into [1,2]
```

The program can only be run from account [1,2] (on any disk). Log into [1,2] and run it again.

```
?Trace return at nnnnnnnn
```

Your job returned from an instruction trace, but your job's trace trap address (JOBTRC) was not set up properly.

```
?TRAPV instruction at nnnnnnn
```

Your job executed a TRAPV instruction which caused the trap to be taken. This instruction is not currently supported by AMOS/L.

```
?     ^ specification error
```

The format of your command line was confusing. For example:

```
VUE RND.COMD RETURN
?           ^ specification error
```

In the case above, you entered too many characters for the file extension. Retype the command line, making sure that the syntax you use is the correct form for that particular command.

APPENDIX B

RESERVED FILE EXTENSIONS

You can use file extensions to categorize your files and to indicate the types of information they contain. The ones listed below have become standardized to represent specific types of files that you use frequently.

What we mean when we say that a file extension is "reserved" is that commands on the system will often work in different ways depending upon the file extension. Also, many of the commands and systems (such as AlphaVUE) have defaults built in or set in the system initialization command file.

An example is when you use the "G" command to exit from an AlphaVUE file. the "G" command is normally used on files with .TXT extensions to format them with TXTFMT and store the resulting formatted version on your disk in a file with a .LST extension. However, if you used a .CMD extension instead of .TXT on your text file, G would try to execute the file as a command, and you would get an error message.

Also, certain programs and commands create intermediate files with extensions such as .TMP or .IPF that are automatically deleted. If any of your own files has one of these extensions, it is in danger of being written over and subsequently erased from your disk.

If you use Alpha Micro Assembly language, you'll find additional reserved file extensions that those languages recognize. These additional extensions are explained in the appropriate programmer's manual.

.ALC	An AlphaCALC file.
.AMX	A line editor function key translation file.
.BAK	A backup file created automatically when you use AlphaVUE. It ensures that you have a copy of each file in case something goes wrong while you're working on the current version.
.BAS	An AlphaBASIC source file.
.BP	An AlphaBASIC PLUS source file.
.C	An AlphaC source file.

.CAX	An AlphaCALC function key translation file.
.CBX	An AlphaCOBOL function key translation file.
.CMD	A command file. A type of text file that contains a series of system commands. You can make AMOS execute the series of commands by entering tthe name of the command file without the extension.
.COB	An AlphaCOBOL source file.
.DAT	A data file.
.DO	A "DO" file. A type of command file that allows you to send variable data to the command file for it to process.
.FOR	An AlphaFORTRAN source file.
.FWD	A file forwarded with an AlphaMAIL message.
.HLP	A help file. A text file that contains information on a specific system related topic. (Type HELP and a RETURN to see a list of these topics.)
.HLV	An AlphaVUE help file.
.IDX	An ISAM index file.
.IDY	An ISAM index file.
.IPF	An intermediate program file. Files with .IPF extensions are used in several Alpha Micro software packages to pass data from one program to another. When processing is completed, these files are automatically erased from your disk.
.LIT	An executable object module that has been processed by the linkage editor.
.LSP	An AlphaLISP source file.
.LST	A list file. This extension is created automatically when .TXT files are processed by TXTFMT, when you use DIR to create a directory listing file, and when you use M68 to create a listing of an assembly language program.
.MAL	An AlphaMAIL message file.
.MAX	An AlphaMAIL function key translation file.

.MLX	A Multi function key translation file.
.M68	An assembly language source file.
.OBJ	An assembly language object file.
.PAS	A program written in AlphaPASCAL.
.PHN	A Multi phone list.
.PSB	An AlphaPASCAL assembly language subroutine.
.RMX	An AlphaCOBOL (RM) function key translation file.
.RP	A compiled AlphaBASIC PLUS program.
.RUN	A compiled AlphaBASIC program.
.SBR	An AlphaBASIC subroutine, usually written in assembly language, that performs some frequently performed function within an AlphaBASIC program.
.SYM	A symbol file. Contains a symbol table for a .LIT file. Created by using the SYMBOL or SYMLIT program on an .OBJ file.
.SYS	A system file.
.TMP	A temporary work file. Many applications create .TMP files during processing and erase them during end of job processing. However, if the application is cancelled or terminates abnormally, the .TMP files might remain on the disk. They should disappear then next time the job is run to completion.
.TXT	A text file. A memo, a letter, or a report created using AlphaVUE.
.WRT	An AlphaWRITE file.
.WRX	An AlphaWRITE function key translation file.

APPENDIX C

CONTROL CHARACTERS

Char.	Monitor function	Function in AlphaVUE	ASCII Definition
@	Null	Null	Null
A		Last Word	Start of heading
B		EOL* to next Line	Start of Text
C	End run		End of Text
D		Delete Character	End of Transmission
E		End of File	Enquiry
F		Insert Character	Acknowledge
G		Next Character=Text	Bell Code
H		Cursor Left	Back Space
I		Horizontal Tab	Horizontal Tab
J	Line Feed	Cursor Down	Line Feed
K		Cursor Up	Vertical Tab
L		Cursor Right	Form Feed
M	Carriage Return	Carriage Return	Carriage Return
N		Cursor to EOL*	Shift Out
O		Concatenate	Shift In
P		Mark Block of Text	Data Link Escape
Q	Resume Display	Insert Mode	Device Control 1
R	Previous Command	Previous Page	Device Control 2
S	Suspend Display	Center Line	Device Control 3
T	Next Stored Cmd	Next Page	Device Control 4
U	Cursor to Start of Line	Cursor to Start of Line	Negative Acknowledgement
V		Delete Word	Synchronous Idle
W		Cursor to Next Word	End of Transmission blocks
X		Cursor to Next Marker	Cancel
Y		Erase to EOL*	End of Medium
Z		Delete Line	Special Sequence
[		Escape	Escape
^		Cursor Home	Group Separator

*NOTE: EOL stands for "End Of Line" -- the line the cursor is currently on.

APPENDIX D

OPTIMIZING THE FILE RESTORE PROCESS

This section contains special notes on optimizing your system to increase the reliability of your file restores.

Normally, you can easily restore files without worrying about special restrictions or considerations. However, when restoring files from video cassette tapes, there are certain timing requirements that must be met. The following paragraphs discuss special situations that can slow down the restore so much that timing problems occur, resulting in some data being lost.

Even if your VCR is remotely controlled, the computer cannot control the speed at which the VCR records and plays back your data. That means that the computer cannot communicate with the VCR to tell it, "Hold it! Slow down, I'm busy and can't catch all that data just now." Instead, no matter what happens on the computer's end, it must be able to keep up with the steady stream of data coming from the VCR.

Anything that "ties up" the computer, preventing it from catching the data streaming in from the VCR, can result in a bad file restore (incomplete files, for example). If this should happen, you will see a list of those files from which data was lost. Then you can try again, restoring just those files that were bad.

However, our intention in this section is to tell you how to prevent such problems from occurring in the first place.

Of course, the first thing to avoid if you are having restore problems is the use of the /TRANSFER switch. Any restore problems will simply be compounded if the monitor is sharing time with other users.

D.1∞IF YOUR WINCHESTER CONTAINS MULTIPLE LOGICAL DEVICES

If you are using the VCRRES command to restore traditional format files, it is always a good idea to log into the device you are restoring to. This is because VCRRES creates a special control file on the disk and account you are logged into. VCRRES consults this file once for every 15 files restored. If you restore to devices other than the one you are logged into, VCRRES must switch back and forth from the device it is restoring to and the device that contains the control file. This slows down the restore.

However, in the case of Winchester disks that contain multiple logical devices, it is even more important to log into the device you are restoring to. Because of the unique way Winchester physical units are configured to contain multiple logical devices, switching frequently between logical devices when restoring files takes a fair amount of time.

To avoid these timing problems, log into the Winchester logical device you will be restoring to before using VCRRES to perform the restore.

If you are restoring to more than one logical device, log into the "middle" logical device. For example, if you are restoring files to DSK1:, DSK2:, DSK3:, DSK4:, and DSK5:, log into DSK3:. This will help to reduce the amount of time used by the disk in switching back and forth between the logical devices.

D.2∞IF YOU RESTORE TO A NON-DSK DEVICE

If you are going to restore files to a device that is not designated as "DSK", the driver for that device should be loaded into system memory by the System Operator before restoring to that device.

The reason for this is that when the monitor accesses any non-DSK devices, the monitor must have access to the device driver. If it is not in either system memory or the user partition, it must be loaded from the disk, which takes additional time. This process can slow the file restore down, causing data to be lost. If you do not want to load the driver into system memory, you can put it into your own memory partition by using the LOAD command.

D.3∞IF YOUR DISKS SHARE BITMAPS

If the disks on your system share bitmaps, that is, if the bitmaps share common memory space, there is an important restriction you should know about. (The System Operator can tell you whether or not the disks on your system share bitmaps by looking at the system initialization command file.) If a cassette contains files from different logical devices, when you restore the tape you should restore the files from each logical device separately.

For example, if your system contains DSK0:, DSK1:, and DSK2:, instead of saying:

```
RESTOR=ALL:[]*.DAT,*.BAS RETURN
```

you should say:

```
RESTOR=DSK0:[]*.DAT,*.BAS,DSK1:[]*.DAT,*.BAS,DSK2:[]*.DAT,*.BAS RETURN
```

The first example tells RESTOR to first restore all .DAT files on every device on the system and then to restore all .BAS files on every device on the system. The second example tells RESTOR to restore all .DAT files and then all .BAS files on DSK0:, then all .DAT files and .BAS files on DSK1:, and finally, all .DAT files and .BAS files on DSK2:

The difference between the two examples is that in the first case, RESTOR must "jump" between disks every time it restores a file. This causes bitmaps to be read every time a file is restored. If your disks share bitmaps, the bitmaps must be switched in and out of memory for every file restored. The time it takes to do this can cause the computer to lose data, resulting in a bad restore.

In the second case, all files for each logical device are restored before another logical device's bitmap must be consulted. This keeps bitmap switching down to a minimum.

If using the RESTOR command in the second form is too inconvenient for your particular application, there are two things you can do, each of which will free you from worrying about the situation:

1. Ask the System Operator to change the system initialization command file so that bitmaps do not share the same memory locations.

2. Specify extra copies when using BACKUP. By specifying enough extra copies, you can compensate for the time used to switch bitmaps. (The exact number of extra copies needed will vary depending on your particular disk and bitmap arrangement.)

D.4 IF YOU TRANSFER DATA BETWEEN DIFFERENT TYPES OF DISKS

Under certain circumstances, data can get lost when you restore to a disk that is of a different type than the disk the data was saved from. The reason for this is the difference between the speeds of the two disk types.

If you restore to a disk that is much slower than the disk from which the data was saved, the computer may not be able to write to the disk fast enough to catch all of the data streaming in from the VCR. The cure for this situation is to specify extra copies when you originally use the BACKUP command to save the files. For a more detailed discussion, see Section 9.3.

D.5 RESERVED FILE NAMES - TRADITIONAL FORMAT

Both the VCRSAV and the VCRRES programs create a temporary directory file named VCRTOC.IPF in the disk and account you are currently logged into. This file is erased and recreated each time VCRSAV or VCRRES is used within the account, therefore, the disk you are logged into must **not** be write protected. Do not use this filename for one of your own files since VCRSAV and VCRRES will write over it the next time you use one of these commands. Also, if you specify VCRTOC.IPF in a VCRSAV or VCRRES command line, VCRSAV and VCRRES will automatically bypass it.

There is also a file called FRGTBL.IPF that VCRRES creates and uses throughout its operation. Therefore, do not use this name for any of your files.

APPENDIX E

VERIFYING AND CHECKING A VCR CASSETTE

After a video cassette has been used for quite a while, you may start to encounter errors of one sort or another. To determine if the tape cassette is the source of the problem, you can verify or check the cassette. A special program, CRT610, does this job for you and also enables you to prepare a warm boot monitor cassette.

Building the warm boot monitor and transferring it to a video cassette are explained in the next chapter. The CRT610 command does not lock other users out of the system, nor do any of the optional switches. This means that you can verify your cassettes without inconveniencing other users. In addition, CRT610 can also generate a disk file containing certification information, allowing a job to run CRT610 without being attached to a terminal.

You should always check a cassette after you have created it, using CRT610 with the /CHECK option or /FILE option.

E.1∞CRT610 SWITCHES

You may use one of the optional switches below to choose the CRT610 function you want to perform.

/BOOT or /B	Create a warm boot cassette using the warm boot monitor already created on disk by WRMGEN. This is the default if a filespec is given or if no other switches are specified (operation switch).
/VERIFY or /V	Write a data pattern to the video cassette, read it back, and print the status of the tape (operation switch). NOTE: Any data already recorded on the tape will be destroyed.
/CHECK or /C	Read a pre-recorded video cassette, and print the status of the tape (operation switch). Does not destroy data. Disables the /BOOT and /VERIFY switches.

/FILE or /F	Read a pre-recorded video cassette, and send the status of the cassette to the file VCRSTS.LST instead of to the terminal. Otherwise, it functions exactly like the /CHECK switch (operation switch). Creates the VCRSTS.LST file on the device and account you are logged into, after deleting any existing file named VCRSTS.LST.

A "warm boot" is the procedure that you use to restore your Winchester System Disk when it is erased or written over accidently. A warm boot from cassette requires a special monitor file that you generate using the WRMGEN command and which is preinitialized with just enough information to get your system up and running on one terminal and in one memory partition.

You can then restore the damaged files (with the RESTOR command) to your System Disk from other cassette backups you have previously created using the BACKUP command. (Refer to the reference sheets for these commands in the *System Commands Reference Manual* for further information.) We will discuss the warm boot procedure more fully in Appendix F.

E.2∞COMMAND FORMAT

To use the CRT610 program, type CRT610 followed by an optional Filespec and an optional switch.

```
        CRT610 {Filespec}{/switch}
```

where Filespec is the specification of the warm boot monitor file that you want to copy onto a video cassette, and /switch indicates one of the various processing options you may select. The default Filespec for the warm boot monitor file is DSK0:AMOSL.WRM[1,4], and the default switch is /BOOT.

E.3∞HOW TO VERIFY A CASSETTE

(The /VERIFY Switch)

This function of CRT610 writes a specific data pattern onto the tape, and after asking you to rewind the tape, reads back the tape to make sure the data was recorded accurately. Statistics printed at the end of this process indicate the error rate for the cassette. NOTE: Any data previously recorded on the tape cassette will be destroyed.

This data pattern can be checked visually by connecting a video monitor to the VCR. This will show you directly whether data has been written to the tape. The verification pattern spells out "Alpha Micro" in large block letters.

The /VERIFY procedure is:

1. Load a blank or scratch cassette into your video cassette recorder. (A scratch tape is one that has data on it you no longer want to save.) Make sure it's rewound, and type CRT610 followed by the /VERIFY switch and a RETURN. For example:

   ```
   CRT610/VERIFY RETURN
   ```

2. **CRT610 acknowledges your request by displaying:**

   ```
   VERIFY
   ```

3. Now CRT610 asks you for more information.

   a. `Enter cassette size (hours):`

      You can specify a value of one to six hours. The default is two. Press RETURN to continue.

   b. `Enter number of copies:`

      The allowed values are from four to 255, and the default value is four. (This is the actual number of copies that will be written onto the cassette; it is not an additional number of copies as when you use the /COPIES: switch with the BACKUP command.)

      The VCR cassettes were not originally designed to record data, and so are not as reliable as other types of magnetic tape; in order to ensure that your data can be recovered from the cassette, you must write multiple copies of each record to the cassette. (As a point of interest, CRT610 automatically generates sixteen copies of your warm boot monitor when you create a warm boot tape.) When you have entered the number of copies you need, press RETURN.

4. If you have a VIDEOTRAX Video Tape Recorder, you will now see messages explaining the steps CRT610 is taking as it certifies your tape. If you have a manually operated VCR, CRT610 prompts you to ready the VCR unit:

   ```
   Place cassette at load point.
   Press RETURN on CRT when ready.
   ```

Make sure the cassette is rewound to the beginning of the tape and is ready to record.

5. When you have done this, you will see:

```
Press RECORD on VCR.
Press RETURN on CRT when ready.
```

There is a ten second wait before CRT610 begins writing its data patterns to the tape.

6. The actual writing of data onto the tape may take quite a while—several hours, in fact. The length of time will depend on how long you said the video tape cassette was.

   While it is writing, CRT610 displays:

```
nnnn blocks written
```

and the display is updated every two seconds.

7. When CRT610 is finished writing, it displays:

```
The transfer is now complete.
Press STOP on the VCR.
```

8. CRT610 now tells you:

```
Reading entire cassette.

Place cassette at load point.
Press RETURN on CRT when ready.
```

Make sure the cassette is completely rewound and is ready to be read.

9. When you have done this, CRT610 displays:

```
Press PLAY on VCR.
Press RETURN on CRT when ready.
```

CRT610 now begins to read each block of data on the tape and check it for accuracy. Again, this may take a long time.

10. As it reads the tape, CRT610 displays the following statistics and updates them every two seconds:

```
Total Blocks Read - nnnn
Total Copies Read - nnnn
Total  CRC Errors - nnnn
Total Hard Errors - nnnn
Reliability Ratio - nnn:1
```

Total Blocks Read is the number of unique blocks of data read from the tape. Total Copies Read is the total number of blocks read. Total CRC Errors is the number of bad copies read back. This number can vary widely depending on the recording speed and quality of the cassette and recorder unit. The Error ratio shows you how many CRC or "soft" errors occurred in relation to the number of blocks read. For example, a Reliability Ratio of 1000:1 means that one soft error occurred for each 1000 blocks read.

11. CRC errors are acceptable as long as the Reliability Ratio is greater than 100:1. (that is, if more than 100 blocks are read per error). If the ratio should drop below 100:1, a warning message will print out:

```
%Warning - Possible equipment/media problems.
```

This indicates either that the tape is bad or your VCR equipment has a problem. First check the VCR. One of the most common problems is simply that the recording heads are dirty. Also be sure that you have the right kind of connecting cable, and that the cable is not loose. If you determine that you have no technical problems, it probably is the tape. In that case, you probably want to discard the cassette, and use a new one.

A Hard Error means that a data block could not be recovered at all—this usually means that the tape is bad and should be discarded. As with a soft error, check your VCR equipment for problems before discarding the tape.

12. If all four totals are zero, either the tape is completely bad, or the VCR unit is not connected to the system properly.

NOTE: You should write these totals on the outside of the cassette, to give you an evolving picture of how the tape is wearing. A high proportion of CRC errors compared to the number of copies indicate that hard errors might occur unless the number of copies is increased the next time the cassette is used.

13. Rewind and remove the cassette, and return it to its storage location.

E.4 HOW TO CHECK A PREVIOUSLY RECORDED CASSETTE

(The /CHECK Switch)

This function of CRT610 reads the data already recorded on a video cassette and displays statistics upon completion that indicate the error rate of the data. It overrides the /BOOT and /VERIFY switches.

The /CHECK procedure is:

1. Load the cassette you want to check into your VCR unit.

2. Type CRT610 followed by the /CHECK switch, and press RETURN:

```
CRT610/CHECK RETURN
```

3. CRT610 acknowledges your request by displaying:

```
CHECK
Reading entire cassette
```

4. If you have a VIDEOTRAX Video Tape Recorder, you will now see messages as CRT610 checks your cassette. If you have a manually operated VCR, follow CRT610's prompts, and shown below. First CRT610 will display the tape label, or `%No tape label`.

5. Then it will tell you:

```
Place cassette at load point.
Press RETURN on CRT when ready.
```

   Make sure the cassette is rewound and is ready to be read.

6. When you have done this, CRT610 displays:

```
Press PLAY on VCR.
Press RETURN on CRT when ready.
```

CRT610 now begins to read each block of data on the tape and check it for accuracy. This might take quite a while, depending on how much data has been written on the tape.

7. As it reads the tape, CRT610 displays the following statistics which are updated every two seconds:

```
Total Blocks Read - nnnn
Total Copies Read - nnnn
Total  CRC Errors - nnnn
Total Hard Errors - nnnn
Reliability Ratio - nnn:1
```

Total Blocks Read is the number of unique blocks of data read from the tape. Total Copies Read is the total number of blocks read. Total CRC Errors is the number of bad copies read back. This number can vary widely depending on the recording speed and quality of the cassette and recorder unit. The Error ratio shows you how many CRC or "soft" errors occurred in relation to the number of blocks read. For example, a Reliability Ratio of 1000:1 means that one soft error occurred for each 1000 blocks read.

8. CRC errors are acceptable as long as the Reliability Ratio is greater than 100:1. (that is, if more than 100 blocks are read per error). If the ratio should drop below 100:1, a warning message will print out:

```
%Warning - Possible equipment/media problems.
```

This indicates that either the tape is bad, or your VCR equipment has a problem. First check the VCR. One of the most common problems is that the recording heads are dirty. Also be sure that you have the right kind of connecting cable, and that the cable is not loose. If you determine that you have no technical problems, it probably is the tape. Recover the data on the cassette and discard the tape.

A Hard Error means that a data block could not be recovered at all—this usually means that the tape is bad and should be discarded. If a Hard Error should occur in the table of contents, nothing on the tape can be recovered. If the hard error did not occur in the table of contents, recover what files you can and discard the tape. As with a soft error, check your VCR equipment for problems before discarding the tape.

9. If all four totals are zero, either the tape is completely bad, or the VCR unit is not connected to the system properly.

NOTE: You should write these totals on the outside of the cassette, to give you an evolving picture of how the tape is wearing. A high proportion of CRC errors compared to the number of copies indicate that hard errors might occur unless the number of copies is increased the next time the cassette is used.

10. If all of the files listed in the tape directory file were recorded (that is, the tape did not run out before the SAVE was completed), you will see the message:

```
Certification is complete.
```

If you do not see this message, it means that the tape ran out before all of your files were transferred. You will probably want to back up again, using a longer tape or copying less files per tape.

The system then prompts you to rewind the cassette. Remove the cassette and return it to its storage location.

E.5 HOW TO USE THE /FILE SWITCH TO CHECK A CASSETTE

(The /FILE Switch)

The /FILE switch performs exactly the same function as the /CHECK switch except that it sends the status display to a file named VCRSTS.LST instead of to your terminal display. (The VCRSTS.LST file is created in the account and disk you are logged into.)

Checking a video cassette is a time consuming process, so you may want to use the FORCE command to force another job to run the CRT610 program so you can use your own terminal to run other programs. (Of course, you can only use FORCE if your system has at least one job other than your own.)

To force another job to run CRT610, you will use the ATTACH, FORCE, LOG and MEMORY commands. See the reference sheets for these commands in the *System Commands Reference Manual*.

The paragraphs below discuss the steps in the procedure of forcing another job to run CRT610. If you will often force another job to run CRT610, you may want to ask the System Operator to modify the system initialization command file so that steps #1-4 are done automatically at the time of system bootup, so that all you need do is perform Step #5.

1. Make sure that a job other than your own is available for use. (Don’t use any jobs that are being used to control printer spoolers.) If there is no extra job, ask the System Operator to create one.

2. Although the job that runs CRT610 does not need to be attached to a "real" terminal (such as an actual CRT or printer terminal), in order to use the FORCE command, the job does need to be attached at least to a pseudo-terminal. (A pseudo-terminal simulates an actual terminal. You probably already have at least one pseudo-terminal defined on your system if your system uses a printer spooler.)

   A pseudo-terminal is one which is defined as using the PSEUDO interface driver. Use the ATTACH command to attach the job to the pseudo-terminal. For example, if the job is named EXTRA, and the pseudo-terminal is named NULL:

   ```
   ATTACH NULL,EXTRA RETURN
   ```

3. Now you need to give the job some memory, using the MEMORY command. For information on using this command, refer to the *System Commands Reference Manual*. For example:

   ```
   FORCE EXTRA MEMORY 6K RETURN
   ```

   You should allocate at least 6K bytes of memory to the job.

4. Now that the job is attached to a terminal and has memory, you need to log the job into a disk account. Choose the disk account in which you want the VCRSTS.LST file to be created. Use the FORCE command to log the job in. For example, to log the job into DSK2:[100,1]:

   ```
   FORCE EXTRA LOG DSK2:[100,1] RETURN
   ```

5. Now you are ready to force the job to run CRT610. Enter:

   ```
   FORCE EXTRA CRT610/FILE RETURN
   ```

   where EXTRA is the name of the job.

6. Once the job has been forced, you can then start your VCR by pressing PLAY. If you press PLAY before the job has been forced, you may miss some of the tape, and any part of the tape that runs before CRT610 begins will not be checked. You may also get some hard or soft errors reported.

When you use the /FILE switch, the statistics are updated periodically as they are when you use the /CHECK switch. You can use the TYPE command to display the contents of the VCRSTS.LST file at any time.

For example, if the job is logged into DSK2:[100,1], you can look at the VCRSTS.LST file at any time by entering:

```
TYPE DSK2:VCRSTS.LST[100,1]RETURN
```

The entire sequence of commands would look like this:

```
ATTACH NULL,EXTRARETURN
FORCE MEMORY 6KRETURN
FORCE EXTRA LOG DSK2:[100,1]RETURN
FORCE EXTRA CRT610/FILERETURN

TYPE DSK2:VCRSTS[100,1]RETURN

Total Blocks Read - nnnn
Total Copies Read - nnnn
Total CRC Errors  - nnnn
Total Hard Errors - nnnn
Reliability Ratio - nnn:1

Certification is complete
```

You may also run the /FILE option without forcing the CRT610 command to another job, if you do not need your terminal while the certification is running.

Remember that the /FILE switch will cause any existing VCRSTS.LST file to be overwritten. Be careful not to have any valuable data in a file named VCRSTS.LST in the account under which you run CRT610/FILE.

APPENDIX F

THE WARM BOOT

A warm boot monitor is an abbreviated version of your system monitor that contains just enough information to get your system up and running in one memory partition and with one terminal when your System Disk has been accidently erased or written over.

It was designed specifically for the use of systems that contain only Winchester disks—such systems must be able to boot from something other than a disk in case something happens to one of the Winchester disks. ***If you have only Winchester disks on your system, a warm boot tape is a necessity***.

After you have performed a warm boot, you can restore other files to your system from backup tapes, and perform a normal boot to restore your entire system to its former configuration.

This chapter dicusses how to create, test, and save the warm boot monitor. We also talk about how to determine what files to load into the warm boot monitor you create, and how to use the warm boot monitor to perform an actual disk recovery.

F.1∞CREATING THE WARM BOOT MONITOR FILE (WRMGEN)

You will use the WRMGEN program to create the warm boot system monitor file on the disk. Then you will use CRT610 or VCRSAV to copy this monitor file onto a video cassette tape.

Before you run WRMGEN, you should run the DEVTBL and BITMAP programs at AMOS command level. Simply enter the command name and press the RETURN key. Then make a note of the devices on your system, and whether they are sharable or not (DEVTBL displays this), and the BITMAP size for your system. You will need this information when you use WRMGEN.

Then, at AMOS command level, enter:

```
WRMGEN {Filename}RETURN
```

where Filename is an optional name for your warm boot monitor file. The Filename default is AMOSL.WRM.

WRMGEN then asks you for each piece of information it requires to build the monitor file. When it is finished, WRMGEN displays a message telling you how large the warm boot monitor file is.

Each statement you are about to see requires a response that has default values built in. You may enter the complete specification for each response, if you wish; but more likely than not, your system is set up so that you can take advantage of many of these defaults. The defaults are listed after each statement. Press RETURN after each response:

a. `Input monitor:` **`Filespec`**

   where the Filespec of your system monitor must be specified. You must enter at least the file name, such as AMOSL. The rest of the filespec defaults to DSK0:filename.MON[1,4].

b. `System disk driver:` **`Filespec`**

   where Filespec identifies the disk driver of your system. You must enter at least the file name, such as PLD or PMD. The rest of the filespec defaults to DSK0:Filename.DVR[1,6]. It MUST be a 3-character name, and cannot be "DSK".

c. `Number of logical devices:` **`Number`**

   where Number is the number of logical devices your system has. You will only see this question if you have a self-configuring disk.

d. `Bitmap size:` **`Number`**

   where Number is the size of the bitmap on your system. You will only see this question if you have a self-configuring disk.

e. `System terminal interface driver:` **`Filespec`**

   Enter at least the filename of your system terminal interface driver, such as AM355. The rest of the filespec defaults to DSK0:filename.IDV[1,6].

f. `System terminal interface port number:` **`n`**

   where n is the port number of the system terminal interface for the terminal you will be using to boot up. The default value is 0 (zero).

g. `System terminal interface baud rate:` **`n...n`**

   where n...n is the baud rate of your system terminal interface. You must enter the baud rate, such as 9600. These baud rates are documented in the "System Initialization Command File" portion of the *System Operator's Guide*. The default is the maximum baud rate of 19200.

h. `System terminal driver:` **`Filespec`**

Enter at least the file name of your system terminal driver, such as AM62. The rest of the filespec will default to DSK0:filename.TDV[1,6].

After you have entered this basic information, you will see a message asking you to enter the names of Secondary Devices to be defined into system memory (if you have self-configuring secondary devices, you will see the `Number of logical devices:` and `Bitmap size:` questions for those devices). You must enter the device name, one device name per line (for example, MIN0). Notice that there is no colon after MIN0—in this case, it should be written without the colon. Enter a blank line and press the RETURN key to end this device specification.

The DEVTBL command can tell you what devices are defined on your system. If a device is nonsharable, put a slash, /, in front of the device name (1/4" streaming tape drives, video tape recorders, and some printers are nonsharable devices). Bitmaps, etc., are automatically set up as needed.

This only DEFINES the devices into system memory; it does not load the actual device driver programs (such as VCR.DVR). You must do that when WRMGEN asks you to list the programs you want preloaded into system memory.

This is the next step. Enter only one program name at a time. To terminate loading, enter a blank line. The programs to be preloaded into SYSTEM MEMORY should be only those programs that are both necessary and needed for the entire boot procedure. Once in the SYSTEM MEMORY, these programs cannot be deleted during the boot. For Example:

```
Program to load: VCR.DVR[1,6]RETURN
Program to load: VCRRESRETURN
Program to load: RESTORRETURN
Program to load: CMDLIN.SYSRETURN
Program to load: SCNWLD.SYSRETURN
Program to load:RETURN
```

You may enter only the program name, if you wish. The rest of the program specification defaults to DSK0:filename.LIT[1,4].

IMPORTANT NOTE: If you have a self-configuring disk (such as an AM-515), you must enter the micro-code file (such as AM515.MIC) along with the driver program. If the micro-code file is not in system memory, the disk will not function. This applies to any device that has an accompanying micro-code file.

Now you will be asked to enter the names of the programs to be preloaded into the USER PARTITION. As before, enter only one program name at a time, and enter a blank line to stop. The same defaults apply.

The programs preloaded into the USER PARTITION should be programs that are necessary to the warm boot, but that can be deleted after they have been run. Programs that can be deleted can make room in memory for other needed programs, since you will only have one memory partition to work with. See the example above for the format.

Since there are so many types of systems, and so many ways to configure systems, it is not possible for us to tell you what to put into your own warm boot monitor. Exactly how you build your warm boot monitor, and the number of programs that you load into your system memory and user partition, will depend on your own system.

Here is a sample of what an entire sequence of commands might be:

```
Warm boot monitor generator - Vxx.x

Input monitor: AMOSL RETURN
        System disk driver: PLD.DVR RETURN
        Number of logical units: 8 RETURN
        Bitmap size: 3889 RETURN

System terminal interface driver: AM355 RETURN
        System terminal interface port number: 0 RETURN
        System terminal interface baud rate: 19200 RETURN
        System terminal driver: AM62 RETURN

Enter name of SECONDARY DEVICE(S) to be defined into system,
 one per line.  Enter blank line to terminate loading.
        Device to define: DSK1 RETURN
        Device to define: /VCR0 RETURN
        Device to define: RETURN

Enter name of program(s) to be preloaded into SYSTEM MEMORY,
 one per line.  Enter blank line to terminate loading.
        Program to load: VCR.DVR[1,6] RETURN
        Program to load: VCRRES RETURN
        Program to load: RESTOR RETURN
        Program to load: SYSMSG.USA RETURN
        Program to load: SCNWLD.SYS RETURN
        Program to load: CMDLIN.SYS RETURN
        Program to load: 515DVR.DVR[1,6] RETURN
        Program to load: AM515.MIC[1,6] RETURN
        Program to load: RETURN
```

```
   Enter name of program(s) to be preloaded into USER PARTITION,
    one per line.  Enter blank line to terminate loading.
           Program to load: SYSACT⏎
           Program to load: CRT420⏎
           Program to load: BADBLK⏎
           Program to load:⏎

* * Warm boot generation complete. Warm boot monitor is xx bytes * *
```

F.2∞TESTING THE WARM BOOT MONITOR

Now you have a warm boot monitor file on the disk. Before you transfer it to the video cassette tape, you should test it to make sure that it works as you would like it to. You do not want to wait for an emergency to see if your warm boot tape works! The program you use to test a warm boot monitor is called MONTST.

You must log into the System Operators account to run MONTST. Then enter the MONTST command, specifying the name of your warm boot monitor. For example:

```
LOG OPR:⏎
Logged into OPR:
MONTST AMOSL.WRM[1,4]⏎
```

NOTE: Enter the MONTST command exactly as shown above. Do NOT enter the name of a system initialization command file. Warn any other users on your system before running MONTST.

After booting successfully, you will see:

```
AMOS/xx       Version xx.x
```

Now you are ready to try some of the commands included in the warm boot monitor (the non-destructive ones). Remember that for a true test you can only use the commands you loaded into the monitor. If the warm boot monitor doesn't work correctly, just push the RESET button to boot from the System Disk and try building a new warm boot monitor.

F.3 TRANSFERRING THE WARM BOOT MONITOR TO THE VCR (CRT610/BOOT)

Now that your warm boot monitor file is finished and tested, you can transfer it to a video cassette tape.

Both the BACKUP command and the CRT610 command have a /BOOT switch that enables you to copy a warm boot monitor file to a video cassette. If you wish to copy other files onto the same cassette at the same time, you can use the BACKUP command which is explained in Chapter 9. Otherwise, you can use the /BOOT switch with the CRT610 command, as shown in the following example.

1. Load a blank video cassette into your VCR unit.

2. Type CRT610 followed by the optional file specification of the warm boot monitor. Then press the RETURN key.

```
CRT610 WARMB.OOT RETURN
```

NOTE: Since /BOOT is the default switch if you specify a filename, we didn't add the /B here. If you do not specify a filename, you must add the /B. For example:

```
CRT610/B RETURN
```

If you enter a filespec, the default device and account are DSK0:[1,4]. If you don't, CRT610 first tries DSK0:AMOSL.WRM[1,4]. If it doesn't find it there, it looks for AMOSL.WRM in the device and account you are logged into.

3. Now CRT610 acknowledges your request by displaying:

```
BOOT RETURN
Creating Boot Cassette
```

And it asks you for more information for the cassette label.

a. `Volume Name:`

Enter a verbal description of the tape. It can be up to forty characters long.

b. `Volume Id:`

Type any name or serial number you wish to give this tape as long as it is no more than ten characters.

c. `Installation:`

Enter the name of your company. Maximum length is thirty characters.

d. `System:`

Enter the name of the computer system this tape is created on. This thirty character field is useful if your installation has more than one computer system.

e. `Creator:`

Enter the name of the person creating the tape. This field may be up to thirty characters long.

4. If you have a VIDEOTRAX Video Tape Recorder, you will now see messages as CRT610 creates the warm boot tape. If you have a standard VCR, CRT610 then tells you to:

```
Place cassette at load point.
Press RETURN on CRT when ready.
```

That is, make sure the cassette is completely rewound to the beginning of the tape, and is ready to record.

5. When CRT610 is ready to transfer the monitor file, it displays:

```
Press Record on VCR.
Press RETURN on CRT when ready.
```

CRT610 waits about ten seconds and then begins transferring the warm boot monitor file from disk to tape.

6. The last message you see is this:

```
The transfer is now complete.
Press STOP on the VCR.
```

7. Rewind the cassette. You should now use CRT610 with the /CHECK switch to make sure that the data on the tape is good. When you are finished, remove the cassette, label its container, and return it to its storage location.

IMPORTANT NOTE: If you change the configuration of your system, you may need to create a new warm boot cassette.

F.4∞WARM BOOTING FROM THE VIDEO CASSETTE RECORDER

If everything is fine, try booting for real: load the tape containing the warm boot monitor into the VCR, push the PLAY button, and push the RESET button on the computer front panel.

Since you already know that your warm boot monitor works fine, this test just tries out the rest of the system—the tape itself, the VCR, the AM-610 and AM-610 cables (or the AM-1200).

The above is the procedure you will use if you ever have a problem booting normally. Once you are successfully booted, you will see:

```
AMOS/xx      Version xx.x
```

Then press STOP on the VCR, rewind the warm boot tape, and return it to its storage case. If you have data files stored on the same cassette as the warm boot monitor, you can leave the tape in the VCR and use the VCRRES command to restore those files.

F.5∞HINTS ON BUILDING AND USING WARM BOOT MONITORS

This section contains some hints on determining what files to include in a warm boot monitor, and gives some procedures for recovering from disk problems once you have booted up under a warm boot monitor.

Of course, we hope that you never have a hardware, software, or user problem that forces you to rebuild your System Disk, but it is good practice to prepare for problems ahead of time, just in case.

The guidelines below are only suggestions—you know your system best, and you are in the best position to decide what to include. Look in account DSK0:[1,4] to see what programs might be most useful if you were suddenly unable to load anything in off of DSK0:. You might want to try out several different warm boot cassettes before an error actually occurs. That way you can be sure that each warm boot monitor really does what you expect it to do.

You will want to prepare several different warm boot cassettes, each aimed at a specific problem. For example, one warm boot monitor might be aimed at a total disaster situation where nothing is retrievable off the disk, while another might be aimed at a situation where the file structure of the disk is fine, but the system initialization command file has accidently been erased. Make sure you label each cassette with the programs that it contains. The next sections discuss some of the possible types of warm boot monitors.

First, some general guidelines:

1. If you are going to use RESTOR to restore extended format files to the disk, your warm boot monitor MUST contain CMDLIN.SYS and RESTOR.LIT in user or system memory. If you are going to use VCRRES to restore traditional format files to the disk, your warm boot monitor MUST contain SCNWLD.SYS and VCRRES.LIT in user or system memory.

   Remember that files backed up with the BACKUP command can only be restored by the RESTOR command. If your backups were created by the VCRSAV command, they can only be restored with the VCRRES command.

   It is also a good idea to have the file SYSMSG.USA on your warm boot tape so that you can understand error messages printed on the terminal.

2. The warm boot monitor automatically logs the operator into account DSK0:[1,2]. Therefore, you don't need to load LOG into your warm boot monitor, because from account [1,2] you can write to any disk account.

3. Be careful about adding too many programs to the warm boot monitor—you need to have enough memory once booted to run the programs or you will not be able to perform the recovery procedures you have chosen. When in doubt, create another warm boot monitor to perform extra functions.

4. In addition to making different warm boot cassettes, you will probably want to make several copies of each type of warm boot cassette; this ensures that you always have at least one good copy.

In all of the examples that follow, we assume a hypothetical system that has two Winchester disk drives that contain two logical devices each. For your own system, of course, you will need to use the appropriate programs instead of the specific examples given below.

Each one of the sections below define a specific problem for which you should create a warm boot monitor. We discuss what should go into the warm boot monitor, and how to use that monitor when the need arises.

F.5.1 Reinitializing the Disks

Suppose a situation occurs when, through a software or hardware error (or even through a user's mistake), most or all of the data is destroyed on your disks.

There are two possible ways to recover from this situation: if you have a resident wizard, he or she can try to use a warm boot monitor containing DSKDDT to repair the disks. However, this requires a thorough understanding of the Alpha Micro disk structure, is a very complex procedure, and is a lot of work. We don't recommend it. Or, you can use a warm boot monitor that contains SYSACT and you can re-initialize the disks and start over.

Remember that every program you need must be in the warm boot monitor, since nothing is available from the disk.

NOTE: The driver you specified to WRMGEN as the System Device driver must have previously been configured to handle the proper number of logical devices per physical unit (in our example, two). And, don't forget to define the DSK devices other than DSK0: as secondary devices if you need to use them initially.

When you run the WRMGEN program, load the following programs into system memory:

```
VCR.DVR
SCNWLD.SYS
CMDLIN.SYS
```

VCR.DVR is the driver for the AM-610 and the AM-1200 Video Cassette Recorder Interfaces, and must be in memory if you are to access that device. SCNWLD.SYS and CMDLIN.SYS are wildcard specification handlers—SCNWLD.SYS is required by the VCR software; CMDLIN.SYS is required by the BACKUP/RESTOR software. You may want to load SYSMSG.USA, also.

Load the following programs into user memory:

```
MOUNT
SYSACT
VCRRES
RESTOR
```

MOUNT is used to access the logical devices on the physical units. SYSACT is used to initialize all logical devices on the two physical units.

RESTOR will be used to restore extended format files, and VCRRES will be used to restore traditional format files from a cassette to the newly prepared disks. You may also want to include one or more of the following programs in user memory:

```
VCRDIR
BAKDIR
SET
DEL
```

BAKDIR or VCRDIR can be used to determine the contents of the tapes you want to restore from, SET can be used to determine that the parameters of your system are correct, and DEL can be used to delete programs from memory (making room for other programs to run).

CAUTION: The procedure below clears all of the files on all of the DSK devices. Make sure that you have multiple tape backups, and that you have tested your warm boot monitors.

Let's say that you've now booted up with the monitor defined above. The procedure below is just one possible use of the warm boot monitor to recreate a system with two Winchester disks each of which contain two logical devices:

```
DEL SET.LITRETURN
DEL VCRDIR.LITRETURN
DEL BAKDIR.LITRETURN
```

After this is done, do the following to restore your data. (In this example the AMOS prompt is shown as a dot.)

```
.SYSACT DSK0:RETURN
*I RETURN
Create extended directory structure? YRETURN
Initializing the disk clears all files - enter Y to continue: YRETURN
Reserve space for how many accounts? RETURN
*E RETURN
.DSKANA DSK0:RETURN
.MOUNT DSK1:RETURN
.SYSACT DSK1:RETURN
*I RETURN
Create extended directory structure? YRETURN
Initializing the disk clears all files - enter Y to continue: YRETURN
Reserve space for how many accounts?RETURN
*E RETURN
.DSKANA DSK1:RETURN
.MOUNT DSK2:RETURN
.SYSACT DSK2:RETURN
*I RETURN
```

```
Create extended directory structure? Y RETURN
Initializing the disk clears all files - enter Y to continue: Y RETURN
Reserve space for how many accounts? RETURN
*E RETURN
.DSKANA DSK2: RETURN
.MOUNT DSK3: RETURN
.SYSACT DSK3: RETURN
*I RETURN
Create extended directory structure? Y RETURN
Initializing the disk clears all files - enter Y to continue: Y RETURN
Reserve space for how many accounts? RETURN
*E RETURN
.DSKANA DSK3: RETURN
.RESTOR ALL:[] = ALL:*.*[] RETURN
```

Now your system is fully restored. For information on SYSACT and MOUNT, see the *System Commands Reference Manual*.

F.5.2∞Recovering Individual Files

A situation can arise in which the monitor or system initialization command files are lost or destroyed. Although the contents of the rest of the disk may be valid, you cannot boot because these special files are gone.

You can usually assume that all the files except the monitor and system initialization comand file are still available. However, it is safest to preload, with the warm boot monitor, the programs you will need if any specific files are not available.

Load the following programs into system memory:

```
VCR.DVR
SCNWLD.SYS
CMDLIN.SYS
```

VCR.DVR is the device driver for the AM-1200 and AM-610 Video Cassette Recorder Interfaces, and is needed to access those devices. CMDLIN.SYS and SCNWLD.SYS are wildcard file specification handlers—SCNWLD.SYS is needed by the VCR software; CMDLIN.SYS is needed by the BACKUP/RESTOR software.

Load the following programs into user memory:

```
MOUNT.LIT
VCRRES.LIT
RESTOR.LIT
PPN.LIT
DIR.LIT
LOG.LIT
```

MOUNT is required to access the logical devices on a physical unit. RESTOR is needed to restore the missing extended format files from a backup cassette, and VCRRES is needed to restore the missing traditional format files from a backup cassette. PPN is needed to verify that the proper accounts still exist on the disk. DIR is used to see which files are missing or, by using the /H option, to see which files have been corrupted.

You may also want to include BAKDIR, VCRDIR, SYSMSG.USA, and/or SET. BAKDIR and VCRDIR can be used to look at the contents of your backup tapes, and SET can be used to determine that your system parameters are correct.

For example, to restore your monitor and system initialization command files from a backup cassette created by the BACKUP command, do the following:

```
PPN DSK0:RETURN
DIR DSK0:AMOSL.*[ ]RETURN
RESTOR = DSK0:AMOSL.*[ ]RETURN
```

You should now be able to reboot your system and run normally.

F.5.3∞Miscellaneous Situations

Use your imagination in determining what other warm boot tapes might be needed. For example, you might want to have a diagnostic warm boot tape that contains REDALL, DSKDDT, ERASE, CMDLIN.SYS, SCNWLD.SYS, and DSKANA. This tape would help you correct disk file structure problems and "bitmap kaput" errors.

Or, you might want another tape for assessing the damage on the disk that just contains DIR, SCNWLD.SYS, PPN, and TYPE. And, of course, you probably want a boot monitor that contains CRT420, in case your BADBLK.SYS file should get erased or you have hard errors that prevent you from booting from the drive.

AMOS User's Guide

Document History

Revision A00 - AMOS Release 4.1 - (Printed 4/79)

New Document, part number DWM-00100-35.

Revision A01 - AMOS Release 4.3 - (Printed 10/79)

This edition included an enhanced version of the PRINT command documentation, changes to the LOAD command, and changes to the SET command to support magnetic tape. Also, we incorporated information in support of the Phoenix disk drives, new system error messages, and miscellaneous corrections.

Revision B00 - AMOS Release 5.0 and AMOS/L Release 1.1 - (Printed 1/83)

The manual was completely rewritten to incorporate the latest features of both AMOS and AMOS/L systems.

Revision B01 - AMOS/32 Release 1.0 - (Printed 7/86)

Added new information about the User Names feature and the Line Editor. Chapter 10 made more general because of the creation of the *Command File User's Manual*.

Revision 00 - AMOS Release 2.0 - (Printed 3/88)

New part number assigned: DSO-00042-00. This document was extensively revised to include information on new backup software, extended directory format, and file protection.

Revision 01 - AMOS Release 2.0A - (Printed 8/88)

Took out section from Chapter 6 describing the removal of leading zeroes in file protection codes for extended format files. (Although removing leftmost zeroes from the full ten digit code is permitted, leading zeroes cannot be omitted from any of the individual two digit group codes to the right of the first non-zero digit.) Removing this section of text encourages use of the complete ten digit code.

Revision 02 - AMOS Release 2.2 - (Printed 4/91)

Added section to Chapter 4 about input/output redirection.

INDEX