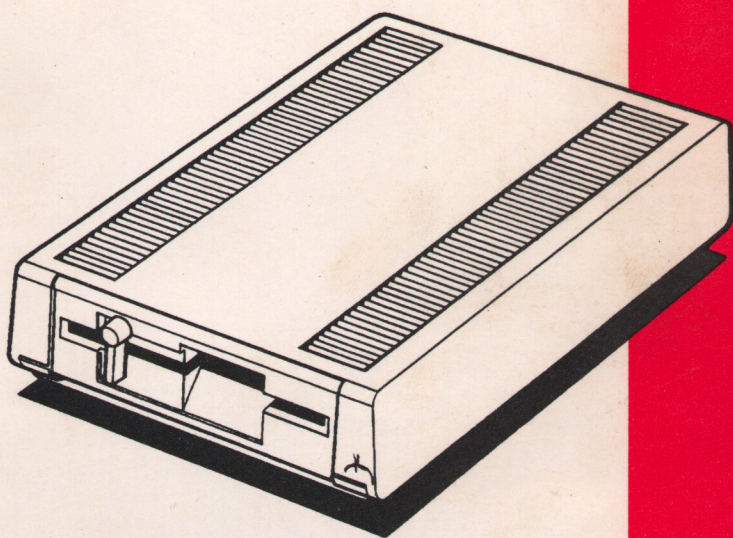


**ATARI®**  
**DOS XE: XF551™**  
DISK DRIVE



Owner's Manual

## IMPORTANT INFORMATION

The Atari XF551 disk drive uses and produces radio frequency energy. If not installed and operated according to the instructions in this manual, the equipment may cause interference with your radio and television reception.

If you experience interference while using the equipment, switch it off. If the interference stops, the equipment is probably at fault. With the equipment switched on, try to correct the the problem using the following measures:

- Adjust the position of the radio or television antenna.
- Reposition the equipment in relation to the radio or TV.
- Move the equipment away from the radio or TV.
- Plug the equipment into a different electrical outlet so the equipment and radio or TV are connected to separate branch circuits.

Consult your Atari dealer or an experienced radio-TV technician for additional suggestions.

A helpful resource is *Interference Handbook*, prepared by the Federal Communications Commission and available from the U.S. Government Printing Office, Washington, DCV 20402, Stock No. 004-000-00450-7.

**WARNING: This equipment is certified to comply with the limits for a class B computing device, pursuant to Subpart J of Part 15 of the FCC rules. These rules are designed to provide reasonable protection from interference when the equipment is used in a residential setting. However, there is no guarantee that interference will not occur in a particular home or residence. Only those computing devices that are certified to comply with the Class B limits may be attached to this equipment.**

In satisfaction of Federal Communications Regulation 15.838(d), this peripheral includes an Atari shielded cable to minimize interference with other electrical devices. Use of this device without an Atari shielded cable or equivalent is prohibited. Replacement cables are available through your Atari dealer.

Every effort has been made to ensure the accuracy of the product documentation in this manual. However, because Atari Corporation is constantly improving and updating its computer hardware and software, it is unable to guarantee the accuracy of printed material after the date of publication and disclaims liability for changes, errors, or omissions.

Atari, Atari BASIC, AtariWriter, AtariWriter Plus, AtariWriter 80, DOS XE, XE, XF551, XL, 400, 800, 810, 1050, 800XL, 1200XL, 65XE, and 130XE are trademarks or registered trademarks of Atari Corporation.

Reproduction of all or any portions of this manual is not allowed without the written consent of Atari Corporation.



Copyright © 1988, Atari Corporation  
Sunnyvale, CA 94086  
All rights reserved.

**DOS XE: XF551**  
DISK DRIVE

**Owner's Manual**

# Table of Contents

<b>INTRODUCTION</b> .....	1
<b>The XF551 Disk Drive</b> .....	1
<b>What DOS XE Does</b> .....	1
<b>Using This Manual</b> .....	2
<b>CHAPTER 1: YOUR ATARI</b> .....	
<b>XF551 DISK DRIVE</b> .....	5
<b>Connecting Your XF551 Disk Drive</b> .....	5
Connecting Multiple Peripherals .....	6
Setting Drive Select Switches .....	7
<b>Taking Care of Your Disks</b> .....	8
<b>CHAPTER 2: GETTING STARTED</b> .....	11
<b>Loading DOS XE</b> .....	11
Boot Errors .....	12
<b>Configuring DOS XE</b> .....	13
COPY3_XE.COM .....	19
<b>Files and Directories</b> .....	19
Naming Files and Directories .....	20
<b>Pathnames</b> .....	21
Working Directories .....	22
<b>Wildcards</b> .....	25
<b>Error Messages</b> .....	25
<b>Duplicating a Disk</b> .....	26
<b>Initializing a Disk</b> .....	27
<b>Copying Files</b> .....	28
<b>Erasing Files and Deleting Directories</b> .....	28
<b>Protecting Files and Directories</b> .....	29
<b>CHAPTER 3: THE MAIN MENU</b> .....	31
<b>Options</b> .....	32
File Access Menu .....	32
Machine Language Access Menu .....	32
System Function Menu .....	32
Exit to Cartridge or Exit to BASIC .....	33
<b>CHAPTER 4: THE FILE ACCESS MENU</b> .....	35
<b>Options</b> .....	36
Files Listing .....	36
Protect Files .....	39
Unprotect Files .....	40
Erase Files .....	41
Rename Files .....	42

View a File.....	44
Working Directory.....	45
New Directory.....	46
Delete Directory.....	47
Copy Files.....	47
Copying a File.....	48
Copying a File to the Same Disk.....	51
Copying a File with Two Disk Drives.....	52
Copying Directories.....	52
Copying a Group of Files.....	53
Copying to and from Devices.....	53
Append to a File.....	55
Initialize Disk.....	56

## **CHAPTER 5: THE MACHINE LANGUAGE**

<b>ACCESS MENU.....</b>	<b>59</b>
<b>Options.....</b>	<b>60</b>
Files Listing.....	60
Working Directory.....	60
Run a Binary File.....	60
Load a Binary File (Do Not Run).....	61
Save Memory to a Binary File.....	62
Append Memory to a Binary File.....	65
Display Memory.....	68
Change Memory.....	70
Go to a Machine Language Program.....	71

## **CHAPTER 6:**

<b>THE SYSTEM FUNCTION MENU.....</b>	<b>73</b>
<b>Options.....</b>	<b>74</b>
Files Listing.....	74
Working Directory.....	74
Run a Batch File.....	74
Set Current Date.....	75
Initialize Disk.....	76
Create DOSXE.SYS File.....	76
Duplicate a Disk.....	77
Allow DOS 2.X Access.....	80

<b>CHAPTER 7: ADVANCED TOPICS.....</b>	<b>83</b>
<b>Command Line Entry.....</b>	<b>83</b>
<b>Batch Files.....</b>	<b>88</b>
<b>Binary Files.....</b>	<b>89</b>

Using DOS XE With Other Disk Drives .....	92
Using DOS XE With Existing Programs .....	92
Changes to Note and Point .....	94
<b>CHAPTER 8: USING DOS XE</b>	
<b>WITH ATARI BASIC .....</b>	<b>95</b>
<b>Tokenized and Untokenized Programs .....</b>	<b>95</b>
<b>DOS XE Pathnames and Atari Basic .....</b>	<b>96</b>
<b>Clearing the Symbol Table .....</b>	<b>96</b>
<b>Using Save, Load, Run, List, and Enter .....</b>	<b>97</b>
Save (S.) .....	97
Load (LO.) .....	98
Run .....	98
List (L.) .....	99
Enter (E.) .....	99
<b>Interactive BASIC Commands .....</b>	<b>100</b>
Open (O.) .....	102
Close (CL.) .....	103
Note (NO.) .....	104
Point (P.) .....	105
Print (PR. or ?) .....	108
Input (I.) .....	109
Put (PU.) .....	111
Get (GE.) .....	111
Status (ST.) .....	112
XIO (X.) .....	113
<b>CHAPTER 9: STRUCTURE OF DOS XE .....</b>	<b>117</b>
<b>Disk Utilization .....</b>	<b>117</b>
<b>Sector Labels .....</b>	<b>117</b>
<b>Sector Organization .....</b>	<b>118</b>
Boot Sectors .....	118
The Volume Table of Contents (VTOC) .....	118
Directory Sectors .....	118
File Map Sectors .....	119
Data Sectors .....	119
<b>Memory Utilization .....</b>	<b>120</b>
<b>Memory Map of DOS XE .....</b>	<b>120</b>
<b>GLOSSARY .....</b>	<b>121</b>

<b>APPENDIX A: ERROR MESSAGES .....</b>	<b>125</b>
<b>APPENDIX B: XF551 DISK DRIVE SPECIFICATIONS .....</b>	<b>129</b>
<b>CUSTOMER SUPPORT .....</b>	<b>131</b>
<b>INDEX.....</b>	<b>133</b>

# INTRODUCTION

## THE XF551 DISK DRIVE

Your Atari personal computer has a large, but not infinite, memory. Furthermore, everything you type into the memory is forgotten when you switch off the computer. The Atari XF551 disk drive solves these problems.

Because the Atari XF551 disk drive stores information on disks, it has an unlimited capacity. Each disk holds almost three times as much information as an Atari 130XE, more than five times as much as an Atari 65XE. There is no limit to the number of disks you can use. The information stored on a disk is permanent. It stays until you change it.

## WHAT DOS XE DOES

A computer cannot directly use the information on a disk. The information must be loaded into the computer's memory first. This requires special software called the Disk Operating System or DOS (pronounced "doss"). DOS enables the computer and the disk drive(s) to work together in storing, retrieving, and managing information.

DOS XE is a DOS for Atari XL and XE computers and disk drives. It organizes information into files and helps you maintain those files. You can group related files into directories, copy them, erase them, and perform other useful tasks.

DOS XE itself is a file on a disk and must be loaded into your computer before it can work. Some programs have a built-in DOS which loads automatically, others require that you load DOS XE separately. (See **Loading DOS XE** in **Chapter 2**.)

The Atari XF551 disk drive and DOS XE are a powerful combination which will greatly enhance your Atari personal computer system.



# USING THIS MANUAL

This manual is designed to serve everyone from the novice computer user to the advanced programmer. It includes two chapters to introduce the Atari XF551 disk drive and DOS XE, four chapters detailing all the capabilities of DOS XE, three chapters of more technical information primarily for programmers, and several appendices. The brief **glossary** of terms used in this manual may be particularly helpful to beginners. As you work with the manual, consult the glossary whenever you are unsure of a term's meaning.

If you have just purchased your first drive, you should start with **Chapter 1: Your Atari XF551 Disk Drive** which provides simple instructions for setting up and using your drive.

**Chapter 2: Getting Started** introduces you to the most frequently used functions of DOS XE. With step-by-step instructions, it explains how to load DOS XE into your computer, prepare disks to store your files, duplicate disks, name and refer to your files, copy files, and erase them. It also shows how to configure DOS XE to fit your system. Most importantly, it explains how to make a back-up copy of your DOS XE master disk.

**Chapters 3, 4, 5, and 6** cover every function of DOS XE and provide detailed instructions on how to use these functions.

**Chapters 7, 8, and 9** contain advanced information of interest mainly to the experienced user or programmer.

The appendices cover both technical and nontechnical information. **Appendix A: Error Messages** will be of interest to everyone.

Paragraphs marked **Note** and **Warning** appear throughout this manual. Notes contain useful hints and information relevant to the topic being discussed. Warnings alert you to potential problems and suggest ways to avoid them.

Many DOS XE functions utilize a series of screen prompts that require the user to perform an action, such as pressing a key. These typically scroll from the bottom of the screen upward. Except for menu screens, all screen displays in this manual represent only a portion of the actual screen display.

In this manual, characters enclosed by square brackets (**[ ]**) represent keys on your computer keyboard. Sometimes a procedure requires you to use two or three keys at the same time. In that case, the keys are listed in order. For example, **[Control] [X]** means to press and hold down the **[Control]** key and then press the **[X]** key.

# CHAPTER 1

## YOUR ATARI XF551 DISK DRIVE

When you unpack your Atari XF551 disk drive, you should have the following items:

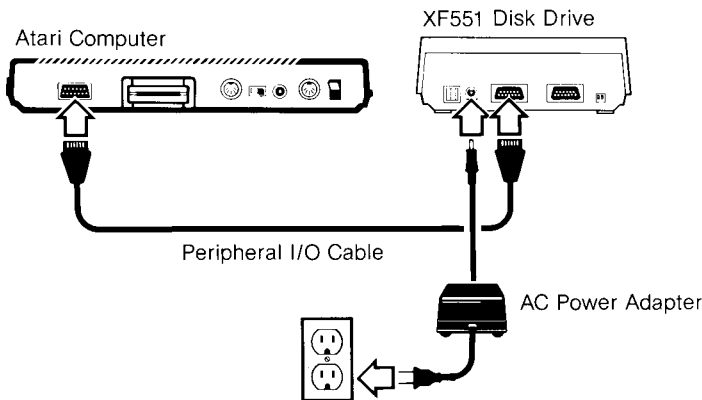
- Atari XF551 disk drive
- Peripheral I/O cable (I/O stands for input/output)
- AC power adapter
- Atari DOS XE: XF551 Owner's Manual
- Warranty/Registration Card

If you are missing any of these items, contact your dealer. Save the packing materials in case you need to transport the disk drive or send it through the mail.

## CONNECTING YOUR XF551 DISK DRIVE

Follow these steps to connect your XF551 to your Atari personal computer:

1. Turn off the power to all components of your computer system.
2. Make sure the power (O I) switch on the back panel of your XF551 disk drive is in the off (O) position.
3. Plug the smaller end of the AC power adapter cord into the jack marked "Power" on the back of the disk drive. Then plug the AC power adapter into the wall socket.



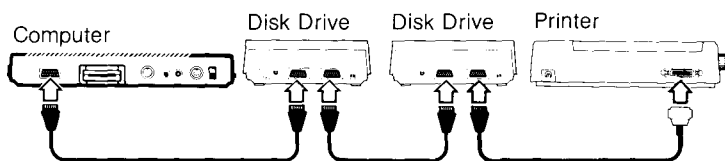
4. Plug one end of the peripheral I/O cable into the jack marked "Peripheral" on the back panel of the computer. Plug the other end of the cable into either of the two jacks marked "Peripheral" on the back of the disk drive.

**Warning:** Your Atari XF551 disk drive should be at least 12 inches away from your television. A television creates strong magnetic fields which could damage the information on the disks.

5. Switch on the disk drive by switching the power (O I) switch on the back of the disk drive to the on (I) position. The light on the front of the disk drive (the busy light) turns on briefly. You may insert a disk when the busy light is not lit.

## Connecting Multiple Peripherals

You can attach up to four disk drives to your Atari computer. You can also connect printers, program recorders, modems, and other components. Peripherals connect to each other in a daisy chain, using the peripheral I/O cables supplied with each component.







There are two peripheral jacks on the back of each disk drive. To install multiple drives, connect a peripheral I/O cable from one peripheral jack on the first disk drive to the peripheral jack on the computer. Connect another peripheral I/O cable from the remaining peripheral jack on the first disk drive to either peripheral jack on the second drive. Connect any additional components in the same way.

## Setting Drive Select Switches

If your system includes more than one disk drive, you must set the device number for each drive. Use the two small Drive Select switches in the window on the back of each drive. The switches identify the device number of each drive.

To set the switches, turn off the power to your computer system. Turn the disk drives around so you can see the Drive Select window on the back of each drive. Inside the window are two small switches.

Drive	Switch Setting
-------	----------------

1	
2	
3	
4	

Use a pen or a small screwdriver to set the switches in the Drive Select window to match the patterns shown here for each drive. You must always have one drive set as Drive 1.

After you set the drive switches, be sure to label each disk drive with its device number so you will not mistake one drive for another when using DOS XE.

## TAKING CARE OF YOUR DISKS

A disk is a round piece of plastic inside a square protective envelope. The disk is covered with a magnetic coating similar to the coating on a cassette tape or video tape. This coating stores your data magnetically. To ensure the long life and reliability of your disks, you must handle them properly and with care.

Most disks have a small write-protect notch on one edge of the square envelope. By covering this notch with one of the adhesive tabs provided with the disk, you can prevent the information on the disk from being erased or changed.

Remember these rules for the proper care and handling of a disk:

- Never switch your disk drive on or off with a disk in the drive, and never store a disk in the drive while the power is turned off.

- Use a soft brush or compressed air from a spray can to remove any dust from the surface of a disk.
- Do not bend or fold your disks; they must turn freely in the protective envelope.
- Store disks in their paper sleeves, standing on edge.
- Keep all disks away from electrical devices, including televisions, telephones, and wires.
- Keep disks away from direct sunlight and excessive heat.
- Do not write on disk labels with a pencil or ballpoint pen. Use a felt-tip pen and press lightly, or write the label before you apply it to the disk.
- Do not use erasers on disk labels. Eraser dust is abrasive and will damage disks.
- Do not attach paper clips to your disks.
- Never touch a disk where it is exposed through the disk envelope. Fingerprints can damage the magnetic surface.

# CHAPTER 2

## GETTING STARTED

To use DOS XE, you must have at least one disk drive, an Atari XL or XE computer, and a monitor.

**Note:** DOS XE will not work on a 400 or 800 computer.

### LOADING DOS XE

Follow these steps to load DOS XE:

1. Make sure that your computer and disk drive are switched off. Remove all cartridges from your computer.
2. Switch on your disk drive(s). Turn the latch on drive one to the open (horizontal) position. Insert the DOS XE master disk in drive number one. Slide it in with the label facing up and the oval read/write windows going in first. Turn the latch to the closed (vertical) position.

**Warning:** Do not turn a drive on or off while there is a disk in it. Do not insert or remove a disk while the disk drive's busy light is on. Doing so could damage the information on the disk.

3. To load DOS XE with BASIC on an XL or XE computer, switch on the computer. On a 1200XL, you must insert the BASIC cartridge, then switch on the computer.
4. To load DOS XE without BASIC on an XL or XE computer, hold down the **[Option]** key while you switch on your computer (this disables BASIC). For the 1200XL, make sure the BASIC cartridge is not in the computer, then switch it on.

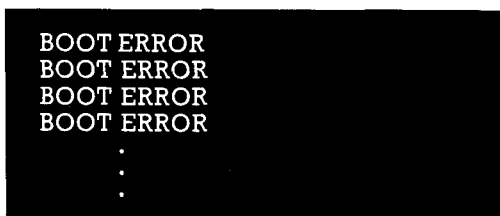


5. The disk drive's busy light goes on and DOS XE loads into your computer automatically. If you turn up the volume on your television or monitor, you can hear a series of beeps as DOS XE loads. When DOS XE is loaded, the busy light goes out and the beeping stops. Loading a program when you first switch on your computer is called booting up.
6. If you boot DOS XE without BASIC, the Main menu appears on the screen. If you boot DOS XE with BASIC, the BASIC READY prompt appears on the screen. To go from BASIC to DOS XE, type the BASIC command **DOS** and press **[Return]**. You then see the DOS XE Main menu.

**Note:** In order to protect the original, it is important that you make a working copy of the DOS XE master disk. It would be a good idea to do so now. (See **Duplicating a Disk** in this chapter for instructions.)

## Boot Errors

If a problem occurs while booting up, the following message may appear on your screen:



The following problems can cause a boot error:

1. The disk drive latch was left open. Close it.
2. DOS XE is not on the disk. Use the correct disk.

3. The disk is inserted incorrectly. Reinsert the disk with the label facing up and the oval read/write windows going in first.
4. The disk is damaged. Use another disk.

The following conditions also prevent successful loading, but no error message will appear on the screen, or it may take several seconds or minutes before the message appears:

1. The disk drive was switched on after the computer. Remove the disk and switch them both off. Switch on the disk drive, insert the disk, then switch on the computer.
2. The disk drive is not properly connected to the computer. Be sure the peripheral cable is securely plugged in at both ends.
3. The disk drive power cord is not properly connected. Make sure it is securely plugged into the wall socket and the disk drive power socket.
4. The drive select switches are not set correctly. (See **Connecting Multiple Peripherals** in **Chapter 1**.) Remove the disk and turn off the disk drive. Turn the switches to the down (drive #1) position and reboot.

## CONFIGURING DOS XE

DOS XE is configurable. Certain features can be customized for your personal setup. Configuration is done with the SETUP.COM program. Once DOS XE is configured, it can be saved to disk using the Create DOSXE.SYS File option in the System Function menu. You can make several configurations and save each of them on a different disk, then use the appropriate disk for each task you will perform.

The DOS XE features which can be configured are:

- The number and types of drives.
- The number of file buffers.

- Installation of the 130XE RAM disk.
- Whether the RS-232 handler should be loaded automatically.
- Whether a BASIC program should be run automatically.

Run SETUP.COM. It is a normal binary file and is run from the Run a Binary File option in the Machine Language Access menu. SETUP.COM displays the following menu:

```
SETUP.COM -- MAIN MENU

1 Set up what will happen when
  DOS XE boots

2 See or change configuration of
  disks as known to DOS XE

3 See or change detailed info
  about a particular drive type
  as it is known to DOS XE

(Any other choice exits this program)

Which option do you want?
```

The first option creates an AUTORUN.SYS file. This file makes several things happen when you boot up DOS XE. A RAM disk can be created, the RS-232 driver can be loaded, and a BASIC program can be run.

The second option allows you to change the number of drives (and their device numbers) and file buffers.

The third option displays detailed information about each drive recognized by DOS XE, and lets you alter its Write Verify status. Examine these options in reverse order.

Type **[3]** and press **[Return]**. You see the following menu:

```
DRIVE TYPE INFORMATION

Available Drive Types:

Drive Type 1 is AT810
Drive Type 2 is AT1050
Drive Type 3 is XF551
Drive Type 4 is 130RAM
Drive Type 5 is SSDD

Choose one of the listed drive types
by number to see more information.
(Other responses exit to main menu)

Drive Type Number >>
```

If other drive types have been incorporated into your DOS XE, they will be listed here. Select a drive type by typing the number and pressing **[Return]**. Look at number 4, the 130 XE RAM disk. The following information is displayed:

```
DETAILED INFORMATION, DRIVE TYPE 4

Type name is 130RAM

Total Sectors on Disk: 257
Usable DOS XE Sectors: 251
Writes Single Density Disks
Writes Single Sided Disks
Disk is set for write WITH verify

Do you want to alter the write mode
for this drive type (Y/N) ?
```

This screen gives basic information about the drive. It also gives you a chance to set the Write Verify mode.

Write Verify means that DOS XE checks a file as it writes it to make sure it is accurate. This is the normal mode and is desirable when using floppy disks. Type **[Y]**. The following prompt is displayed:

```
Do you want future writes to be
WITH verify (Y/N) ?
```

Type **[N]** and press **[Start]** to return to the Option Three menu. You can check the other drive types or just press **[Return]** to return to the Main menu.

When you return to the Main menu, press **[2]** and **[Return]** to get a screen like the following:

```

CONFIGURATION OPTIONS
Current Configuration:
Active Drives:
  Drive 1 is XF551 (configurable)
Maximum of 3 files may be open at once
Is this configuration okay?
Answer Yes or No (Y/N) >>
```

This shows the drives that are hooked to your system. Let's assume you have a 130XE and install the RAM disk. Press **[N]** to get the following screen:

## CHANGE CONFIGURATION

Type in the **NUMBERS** of the drives that you want to be active.

For example, if you have two drives you might type in

1 2

(Spaces are allowed but ignored)

Do **NOT** specify a 130XE RamDisk here!

What drives do you want active?

>>

Enter the numbers of the drives you want to use. Normally these would be numbered consecutively but they do not need to be. You could have drives 1, 5, and 6 if you want. This might be useful, for example, if you have modified drives which can be numbered only 5 to 8. Enter **[1]** if you have one drive, **[1] [Space Bar] [2]** if you have two drives.

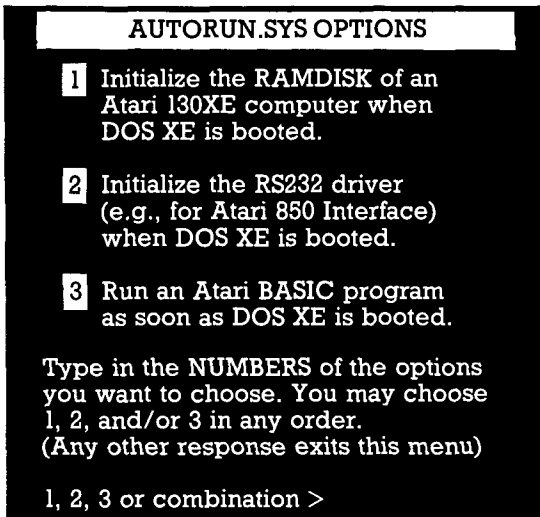
The next prompt asks if you want a RAM disk for your 130XE. Type **[Y]** for yes. You are then asked for the drive number for the RAM disk. Use 8 because it is the most common number, so most software will expect it.

You are asked if you want to change the maximum number of files open at once (file buffers). Type **[Y]** and you see a prompt asking how many. One more than the number of drives is usually sufficient. Press **[Return]** to enter the default value **3**. If your software needs more, the instruction manual will tell you so.

**Note:** For DOS XE menu functions, two files are enough, except when using batch files, where three may be needed for a few commands.

You are returned to the Configuration screen to check the information you entered. If it is correct, enter **[Y]** and go back to the Main menu.

Finally, you can set the boot-up conditions for DOS XE. Press **[1]** and **[Return]**. The following screen is displayed:



The first option will set up the RAM disk when you boot the computer. You want that for this example. The second loads the RDRIVER.SYS program which is the handler for the RS-232 serial ports on the 850 Interface. If you use this option, the file RDRIVER.SYS must be on the boot disk. Ignore the serial handler for now.

The third option makes a BASIC program autorun at boot-up. (BASIC must be installed and the BASIC program must be present when you boot up.) There is a sample BASIC program called WELCOME.BAS on the disk to demonstrate this feature. Type **13** (or **[1] [Space Bar] [3]**, or **[3] [Space Bar] [1]**) and press **[Return]**.

Enter the pathname for the program--**D1:WELCOME.BAS**-- and press **[Return]**. A verification prompt is displayed if the BASIC program does not exist. Press **[Y]**. You will be told that a file named AUTORUN.SYS has been created. That file contains the instructions you have given.

Press **[Start]** to return to the Main menu. Press **[Return]** to leave SETUP.COM. You are asked whether to reinitialize DOS XE. This will put the changes you have made into effect.

Type **[Y]** and you will be returned to the DOS XE Main menu. The DOS XE in the computer's memory is now configured the way you wanted it.

You must write this modified DOS XE to a disk to make it permanent. Use the Create DOSXE.SYS File option in the System Function menu to do this. You can overwrite the DOS XE on your work disk or make a new disk with your customized DOS XE.

## COPY3\_XE.COM

There is one more file on the DOS XE master disk. It is COPY3\_XE.COM. Those of you who have files in the DOS 3 format can use this file to convert them to DOS XE format. Run the file from the Run a Binary File option in the Machine Language Access menu. Follow the prompts to convert the files on your DOS 3 disks.

## FILES AND DIRECTORIES

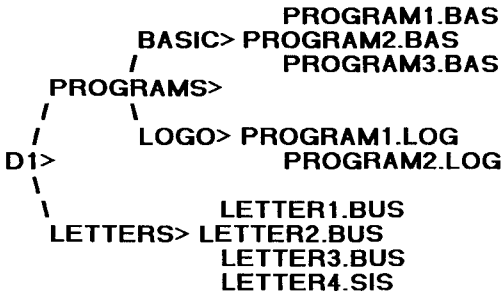
DOS XE organizes information stored on disks into files. Each file contains one specific group of information, perhaps a letter from your word processor, a game program, or DOS XE itself.

These files can be stored randomly on the disk, or they can be grouped together into directories. Every disk has at least one main directory called the root directory, which contains everything on the disk. You can create other directories and put files into them.

A directory can hold up to 1250 files. It can also contain other directories. These subdirectories can contain even more files and directories.



The structure of directories and subdirectories is like a sideways tree, with the root directory at the left. For example, a typical disk may be organized as follows:



There is a limit of 1250 files per directory, but no theoretical limit to the number of directories. However, there are two overriding practical limits: the capacity of the disk, and the 80 character limit on pathnames. (See **Pathnames** in this chapter.)

## Naming Files and Directories

When you create a file or directory, you must give it a name. The name may be up to eight characters in length, followed, if you like, by a period and an extension of up to three characters. Except for the period which separates the name from the extension, all the characters must be numbers, letters, or one of a few special characters, such as `_` or `@`. All letters typed in lowercase are converted to capitals. No punctuation marks or symbols are permitted in the name or the extension.

You can use :

**these names:**

**LETTER1.BUS**

**LETTER1**

**LETT\_1.BUS**

**PROGRAM1.BAS**

**but not these names:**

**LETTER\*1.BUS (illegal character \*)**

**letter1 (will be converted to LETTER1)**

**LETT 1.BUS (illegal space)**

**PROGRAM12.BAS (too many characters)**

These examples use intentionally nondescriptive names. You will want to use more meaningful names for your files. For instance, you may want to use the name of the person you were writing to instead of LETTER1. A game might be called MAZE.BAS, instead of PROGRAM1.BAS.

The optional extension is useful when you name related but distinct files that you might want to manage as a group. (See **Wildcards** in this chapter.) For example, you could use .BAS as an extension for programs which you write in Atari BASIC--PROGRAM1.BAS, PROGRAM2.BAS, and so on. You might use BUSINESS.LET to identify a business letter and SISTER.LET to identify a letter to your sister.

## PATHNAMES

When you refer to a file you must indicate which disk drive it is on and which directory it is in, as well as its name. This is called the pathname because it shows the path that DOS must follow to find your file. Here is what a typical pathname looks like, with an explanation of each part:

**D1>PROGRAMS>BASIC>PROGRAM1.BAS**

**D (Device name).** The device names used by DOS XE are "D" for normal disks and "A" for disks in the alternate DOS 2.0/2.5 format. (See **Allow DOS 2.X Access** in **Chapter 6**.) Other devices may be used, such as "P:" (printer) or "E:" (screen editor).

**1 (Device number).** The device number is the disk drive number.

**> (Delimiter).** These symbols separate the parts of the pathname.

**PROGRAMS (Directory name).** A directory name is used if the file is not in the root directory. The extension must be included if there is one.

**BASIC (Subdirectory name).** Additional subdirectory names are used if the file is at a lower level in the tree.

**PROGRAM1.BAS (Filename and extension).** The filename is the file you want. If the filename has an extension, the extension must be included.

Pathnames cannot exceed 80 characters. This limits the levels of subdirectories you can have. If you need more levels (very unlikely on floppy disks), use shorter directory names.

## Working Directories

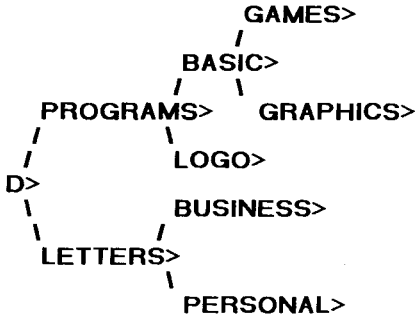
DOS XE provides a short cut when dealing with long pathnames. It allows you to define a working directory. DOS XE remembers the name of your working directory. When you first boot DOS XE, the working directory is D1>, the root directory. You can define a new working directory using the Working Directory option in the menus. (See **Working Directory** in **Chapter 4**.)

DOS XE displays your working directory pathname at the top of the menu screens. There are seven rules for using the working directory:

1. If no pathname is specified, the working directory pathname will be used.
2. A colon (:) is shorthand for the working directory's pathname.
3. A < symbol is shorthand for "move back one level."
4. "D>" is always treated as "D1>."
5. If a device name or number does not match the beginning of the working directory's pathname, then no part of the working directory's pathname is used.
6. If the device name and number are followed by a > symbol, (or a colon **and** a > symbol), then no part of the working directory's pathname is used.

7. If the first character is a > symbol, then only the device name and number from the working directory's pathname are used.

These will be clearer with a few examples. Look at the following directory tree:



This tree only shows directories, but assume there are also files at every level. Also assume you use working directory D1>PROGRAMS>BASIC> as your starting point to explore ways to move around the tree in the following examples.

A file in the working directory:

**If you type:**

**FILE  
:FILE  
D:FILE  
D1:FILE**

**DOS XE uses:**

**D1>PROGRAMS>BASIC>FILE  
D1>PROGRAMS>BASIC>FILE.  
D1>PROGRAMS>BASIC>FILE.  
D1>PROGRAMS>BASIC>FILE.**

A file in a directory to the right of the working directory:

**If you type:**

**GAMES>FILE**

**DOS XE uses:**

**D1>PROGRAMS>BASIC>GAMES  
>FILE.**

A file in a directory to the left of the working directory:

<b>If you type:</b>	<b>DOS XE uses:</b>
<FILE	D1>PROGRAMS>FILE.
<<FILE	D1>FILE.
<<<<<FILE	D1>FILE.

**Note:** DOS XE ignores all extra < symbols when it reaches the root directory.

A file in a directory at the same level as the working directory:

<b>If you type:</b>	<b>DOS XE uses:</b>
<LOGO>FILE	D1>PROGRAMS>LOGO>FILE.

A file in the root directory:

<b>If you type:</b>	<b>DOS XE uses:</b>
>FILE	D1>FILE.
D>FILE	D1>FILE.
D1>FILE	D1>FILE.
D:>FILE	D1>FILE.
D1:>FILE	D1>FILE.

A file in a different path:

<b>If you type:</b>	<b>DOS XE uses:</b>
>LETTERS>FILE	D1>LETTERS>FILE.

A file on a different disk drive:

<b>If you type:</b>	<b>DOS XE uses:</b>
D2>FILE	D2>FILE.
D2:FILE	D2>FILE.

A file in a subdirectory on a different disk:

<b>If you type:</b>	<b>DOS XE uses:</b>
D2>MUSIC>FILE	D2>MUSIC>FILE.
D2:MUSIC>FILE	D2>MUSIC>FILE.
D2:>MUSIC>FILE	D2>MUSIC>FILE.

# WILDCARDS

In a card game, wildcards are valuable because they stand for any card you choose. Similarly, DOS XE recognizes wildcard symbols that stand for any character or combination of characters in a file name. Wildcards allow you to refer to a group of files rather than to each one individually.

The wildcards recognized by DOS XE are the question mark (?) and the asterisk (\*). The question mark stands for any single character. The asterisk stands for the rest of the characters in the name or the extension. Consider a directory with the following files:

```
PROGRAM1.BAS  LETTER1.BUS  
PROGRAM2.BAS  LETTER2.BUS  
PROGRAM1.LOG  LETTER1.SIS
```

As in this example, you could use **LETTER\*.\*** to specify all the letter files. **\*.BAS** specifies all the BASIC programs. **\*.??S** specifies all the files except PROGRAM1.LOG. **\*.\*** specifies all of the files in the directory.

Wildcards can also save you a lot of typing. Rather than typing a whole filename, just type the characters needed to indicate its unique name and use the asterisk for the rest. For example, PROGRAM1.LOG can be specified with **\*.LOG** as long as there are no other files on the disk with .LOG extensions.

# ERROR MESSAGES

While working with DOS XE, you may occasionally see something like this on your screen:



This means that something has gone wrong, perhaps a loose cable. A list of error messages can be found in **Appendix A**. When you have corrected the problem, press **[Start]** to return to the menu and continue.

## DUPLICATING A DISK

It is wise to make backup copies of important disks and store them in a safe place. This protects a disk or file from damage or accidental erasure. Begin by duplicating your DOS XE disk.

**Note:** Some commercial programs are copy protected so that they cannot be duplicated.

Get a blank disk for the copy and follow these steps.

1. Go to the System Function menu by pressing **[S]** and **[Return]** from the Main menu. The System Function menu will appear.
2. Press **[D]** and then **[Return]**. The Message **Caution! This operation destroys program area** appears.
3. Press **[Start]** to continue.
4. Press **[1]**.
5. We will assume you only have one drive, so press **[1]** again. (If you have more than one drive, see **Duplicate A Disk** in **Chapter 6**.)
6. In this case, your original DOS XE master disk is the **[FROM]** disk. Make sure the write-protect notch (if any) is covered and place it your drive. Press **[Start]**.

7. The blank disk is the **[TO]** disk. Make sure the write-protect notch is not covered on it. When DOS XE says:

```
PUT [TO] DISK IN DRIVE 1  
  
PUSH [START] TO CONTINUE  
PUSH [SELECT] TO STOP NOW
```

take the master disk out of the drive and put the blank one in. Press **[Start]**. If the master disk is very full, you may have to repeat steps 2 and 3 several times.

**Note:** Duplicating a disk will initialize the blank disk.

8. When DOS XE says:

```
SELECT ITEM OR [ESC] OR [RETURN]:
```

duplication is done. Press **[Esc]** to return to the Main menu.

9. Put your original in a safe place and work with the copy.

For more details, see **Duplicate a Disk** in **Chapter 6**.

## INITIALIZING A DISK

A new disk must be initialized before it can be used. This procedure organizes the disk so that DOS XE can find its way around. Get another blank disk and follow these steps:

1. Place the blank disk in disk drive one.
2. Go to the System Function menu from the Main menu by pressing **[S]** followed by **[Return]**. The System Function menu appears on the screen. Press **[1]** and then **[Return]**. Press **[1]** for disk drive one.



3. Type **XF551** for your disk drive type and press **[Return]**. Since initializing a disk permanently destroys any information previously stored on that disk, you will get one final chance to change your mind. Press **[Start]** and the disk will be initialized in the XF551 format.

When initialization is finished, you will be able to store files on the disk.

For more details on initializing disks, see **Initialize Disk** in **Chapter 4**.

## COPYING FILES

You will often use DOS XE to copy files from one disk to another. You can make a backup copy of one or several files without having to copy the whole disk. You may also copy files to another directory.

When a file is copied, the original remains unchanged. If a file is copied to a directory which already has a file of the same name, the existing file is overwritten, unless it has been locked.

For more details on copying files, see **Copy Files** in **Chapter 4**.

## ERASING FILES AND DELETING DIRECTORIES

You can erase a file or delete a directory with options in the File Access menu. Erasing outdated files makes space on a disk for more information. Once all files have been erased from a directory and it is empty, the directory can be deleted.

**Warning:** Use the Erase Files option with care; it is permanent.

By using wildcards, you can erase as many files as you wish in one operation. The files must all be in the same directory. Make sure that you want to erase all of the files which match the name; it is easy to make a mistake.

**Note:** We recommend using Files Listing with the same filenames first to be sure that only the desired files will be affected.

For more details on erasing files and deleting directories, see **Erase Files** and **Delete Directory** in **Chapter 4**.

## PROTECTING FILES AND DIRECTORIES

You can protect your files and directories so they cannot be changed or erased. Use the Protect Files option in the File Access menu to protect them and the Unprotect Files option to release them so that they can be changed or erased.

Protected files and directories have an asterisk (\*) before their names in the file listing.

A protected file can be loaded into the computer and used, but nothing can be saved back to that file. This is the suggested way to store all your permanent files. It will prevent loss through mistakes with wildcards.

A protected directory can have files saved to it or erased from it, but the directory itself cannot be deleted.

For more details, see **Protect Files** and **Unprotect Files** in **Chapter 4**.

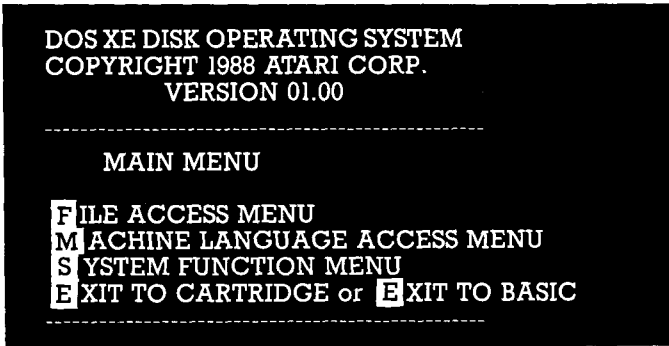
# CHAPTER 3

## THE MAIN MENU

After you load DOS XE, the Main menu appears on the screen. The menu is a list of options you can choose from. To choose an option, press the key which corresponds to the letter highlighted in inverse video, then press **[Return]**.

The Main menu usually contains four options. The first three are other menus. The fourth option is either EXIT TO CARTRIDGE or EXIT TO BASIC. However, if there is no cartridge in the computer and BASIC has been disabled, the Exit option is not displayed. Each of the first three menu options has further options. For instance, press **[F]** and **[Return]** to look at the File Access menu.

The first option in the File Access menu provides a list of files on the disk. Press **[F]** and then press **[Return]** three times. DOS XE gives you the listing. The individual options in the menu will be explained in detail in **Chapters 4, 5, and 6**. For now, press **[Esc]** to return to the Main menu.



# OPTIONS

The Main menu is the gateway to the other menus. Each of the other menus has commands to execute the various DOS XE options. The options are selected by typing the letter highlighted with inverse video and then pressing **[Return]**.

All three menus have options for listing files and changing the working directory. The File Access menu and the System Function menu both have options for initializing new disks. Beyond that, each menu covers a different range of functions.

## File Access Menu

Type **[F]** and press **[Return]** for the File Access menu. This menu has options for file and directory maintenance. You can copy files and create directories. You can also rename, protect, or erase the files and directories.

## Machine Language Access Menu

Type **[M]** and press **[Return]** for this menu. The Machine Language Access menu has options to manipulate the computer's memory. You can display and change memory locations, save blocks of memory into binary files and load the binary files back into memory, and run binary files.

## System Function Menu

Type **[S]** and press **[Return]** for this menu. The System Function menu provides options for maintaining your disk operating system. You can duplicate disks, run batch files, reset the date, create a new DOS XE file, or work with the older Atari DOS 2.0/2.5.

## Exit to Cartridge or Exit to BASIC

DOS XE senses if you have a cartridge installed or have the built-in BASIC enabled. Typing **[E]** and pressing **[Return]** returns you to the cartridge or BASIC. If neither BASIC nor a cartridge is present this option is blank and you cannot exit DOS XE. You can, however, run a binary file or go to a machine language program. (See **The Machine Language Access Menu** in **Chapter 5**.)

# CHAPTER 4

## THE FILE ACCESS MENU

To enter the File Access menu from the Main menu, type **[F]** then press **[Return]**. The File Access menu appears.

```
DIRECTORY          TODAY: 20SE88
DI>

-----

          DOS XE FILE ACCESS MENU

F ILES LISTING    W ORKING DIRECTORY
P ROTECT FILES   N EW DIRECTORY
U NPROTECT FILES D ELETE DIRECTORY
E RASE FILES     C OPY FILES
R ENAME FILES    A PPEND A FILE
V IEW A FILE     I NITIALIZE DISK

-----

SELECT ITEM OR [ESC] APE FOR MAIN MENU:
```

The pathname of the working directory is at the top left of the screen, under the word "DIRECTORY." The current date is at the top right. Below the working directory pathname is the list of options available in this menu. Press the key which corresponds to the highlighted letter and press **[Return]**. At the bottom is a reminder that you can return to the Main menu at any time by pressing **[Esc]**.

After you have executed an option, DOS XE displays the following line:

```
SELECT A LETTER, [ESC] APE, OR [RETURN]
```

This is a reminder that, although the current menu is not displayed, you can press a letter and select an option from it. You can also press **[Esc]** to return to the Main menu, or press **[Return]** to display the File Access menu again.

# OPTIONS

The following sections give detailed descriptions of each option available from the File Access menu. Be sure to read each one before selecting an option.

## Files Listing

This option gives a listing of the contents of a directory. A directory may contain files, other directories, or both.

Type **[F]** and press **[Return]**. DOS XE displays this prompt:

```
FILES LISTING
LIST WHAT FILES?:*.*
```

This prompt asks which files you want listed. Type in the pathname you want. You may use wildcards to select groups of files. Then press **[Return]**. If you do not enter a pathname, DOS XE will provide a listing of all files in the working directory (:\*.\*).

DOS XE then displays:

```
LIST TO WHERE?
```

This prompt asks where you want the listing sent. It may be any legal device or file. If you do not enter a name, DOS XE lists the files on the screen. Press **[Return]**. DOS XE displays a listing like the one on the next page.

D1>\*.\*

```
* DOSXE.SYS      20SE88 20SE88 14224
  DOS2.SYS       20SE88 20SE88  2776
  LETTERS.>      20SE88 20SE88   >
  PROGRAMS.>     20SE88 20SE88   >
00344 K BYTES FREE
```

SELECT ITEM OR **ESC** APE OR **RETURN**:

The pathname of the directory listed is in the upper left of the screen. Below it is a listing of the files and directories it contains, one entry per line.

Each entry begins with the file or directory name. If this name is preceded by an asterisk (\*), it means that the file or directory is protected. (See **Protect Files** later in this chapter.) The name is followed by the extension (if there is one). Directory names are followed by a > symbol.

There are two dates after each name in the directory: the creation date and the revision date. The first is when the file was created. The second date reflects the last time the files was updated, copied, or had material appended to it.

Finally, the size of the file is displayed. If the entry is a file, DOS XE displays the number of bytes in the file. If the item is a directory, DOS XE displays a > symbol.

After the files and directories are listed, DOS XE displays the amount of free space on the disk. Each kilobyte equals 1000 bytes, not 10245 bytes as in other computer uses. The number is rounded to the nearest 1000 bytes.

If the listing is too large to fit on the screen, the list will scroll. You can pause the scroll by holding **[Control]** and pressing **[1]**. It can be restarted by typing **[Control] [1]** again. Interrupt the scroll by pressing **[Break]**.



**Example 1:** List the contents of the working directory to the screen.

1. Select **[F]** from the menu. This prompt appears:

```
FILES LISTING
LIST WHAT FILES? *.*
```

2. Press **[Return]**. The computer asks:

```
LIST TO WHERE?
```

3. Press **[Return]** to list to the screen. After a brief pause, the contents of the directory appear on the screen.

**Example 2:** List the contents of the root directory to the printer.

1. Make sure your printer is connected and power is on.
2. Select **[F]** from the menu. The following message appears on the screen:

```
FILES LISTING
LIST WHAT FILES? *.*
```

3. Enter **D>** and press **[Return]**. The computer asks:

```
LIST TO WHERE?
```

4. Enter **P:** and press **[Return]**. The directory is printed on your printer.

**Example 3:** Create a file with a list of your business letters.

1. Select **[F]** from the menu. The following appears on the screen:

```
FILES LISTING
LIST WHAT FILES? *.*
```

2. Enter **D>LETTERS>\*.\*BUS** and press **[Return]**. The computer asks:

```
LIST TO WHERE?
```

3. Enter **D>LETTERS>LISTING.BUS** and press **[Return]**. The information on the screen is listed to the disk in a file called LISTING.BUS.

## Protect Files

This option protects a file or a directory so that it cannot be changed or erased. Protected files and directories have an asterisk in front of their names in file listings. The protection can be removed with the Unprotect Files option.

Type **[P]** and press **[Return]**. DOS XE displays this prompt:

```
PROTECT FILES
PROTECT WHAT FILES?
```

This prompt asks for the pathname of the file or directory to protect. Wildcards may be used for groups of files. Type in the pathname and press **[Return]**.

**Example:** Protect all your business letter files.

1. Select **[P]** from the menu. This prompt appears:



2. Enter **D>LETTER>\*.BUS** and press **[Return]**. Those files are now protected.

## Unprotect Files

This option removes the protection created by the Protect Files option. Type **[U]** and press **[Return]**. DOS XE displays this prompt:



The prompt asks for the pathname of the file or directory to be unprotected. Wildcards may be used for groups of files. Type in the pathname and press **[Return]**.

**Example:** Unprotect your business letter files.

1. Select **[U]** from the menu. The following prompt appears:



2. Type **D>LETTERS>\*.BUS** and press **[Return]**. Those files are no longer protected.

## Erase Files

This option erases files. It does not erase directories. Files cannot be erased if they are protected. Erased files can't be recovered.

Type **[E]** and press **[Return]**. DOS XE displays this prompt:



This prompt asks for the pathname of the file to be erased. Wildcards may be used for groups of files. Type in the pathname and press **[Return]**.

It is a good idea to use the Files Listing command (with the same wildcard name) to verify the files you will be erasing. Make sure you type in the correct pathname and filename; there is no verification prompt.

**Example:** Erase all the letter files.

1. Select **[E]** from the menu. This prompt appears:



2. Type **D>LETTERS>LETTER?.\*** and press **[Return]**. Your letter files are erased from the disk.

## Rename Files

This option changes the names of files and directories. You cannot rename protected files.

Type **[R]** and press **[Return]**. DOS displays this prompt:

```
RENAME FILES
RENAME WHAT FILES?
```

This prompt asks for the pathname of the file, or directory, to be changed. You can use wildcards for groups of files. If a wildcard is used, the part of the name designated by the wildcard will not be changed. Type in the pathname and press **[Return]**. DOS XE displays the following prompt:

```
DO NOT GIVE DRIVE (DN:) IN NEW NAME
TO WHAT NEW NAME?
```

This prompt asks for the new name of the file or directory. Do not include the pathname, just the name (and extension, if any). If wildcards were used in the name of the file(s) to be changed, the same wildcards should be used in the same places in the new name. Type in the new name and press **[Return]**.

**Example 1:** Change the name of a game file from BLASTER to BLAZER.

1. Select **[R]** from the menu. This message appears:

```
RENAME FILES
RENAME WHAT FILES?
```

2. Enter **D>PROGRAMS>GAMES>BLASTER.BAS** and press **[Return]**. The following message appears:

```
DO NOT GIVE DRIVE (DN:) IN NEW NAME  
TO WHAT NEW NAME?
```

3. Enter **BLAZER.BAS** and press **[Return]**. The filename is now changed.

**Example 2:** Make the same change using wildcards.

1. Select **[R]** from the menu. The following prompt appears:

```
RENAME FILES  
RENAME WHAT FILES?
```

2. Enter **D>PROGRAMS>GAMES>???STER.\*** and press **[Return]**. The following message appears:

```
DO NOT GIVE DRIVE (DN:) IN NEW NAME  
TO WHAT NEW NAME?
```

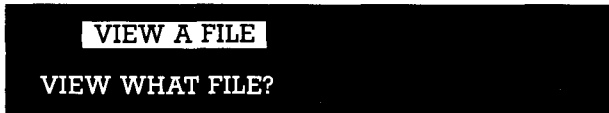
3. Enter **???ZER.\*** and press **[Return]**. The name change is complete.

You cannot rename using wildcards if the result would be two or more files with the same name. When wildcards are not used, no such restrictions apply. It is possible, therefore, to get two files with the same name. If this happens, use Rename again without wildcards. Only the first file with a given name will be renamed.

# View A File

This option displays a file on the screen. This is designed for reading text files. Any file can be viewed, although a nontext file may look like gibberish and the screen may jump around.

Type **[V]** and press **[Return]**. DOS displays the following prompt:



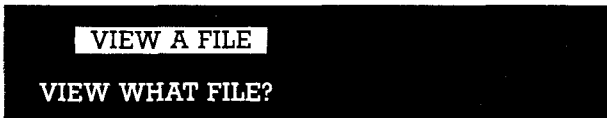
This prompt asks for the pathname of the file to be viewed. Type in the name and press **[Return]**.

Large files will scroll down the screen. You can pause the scroll by holding down **[Control]** and pressing **[1]**. It can be restarted by typing **[Control] [1]** again. The scroll can be interrupted by pressing **[Break]**.

**Example:** Read a business letter without using your word processor.

**Note:** Printing control codes used by your word processor may put some strange characters on the screen.

1. Select **[V]** from the menu. This message appears:



2. Type **D>LETTERS>LETTER1.BUS** and press **[Return]**. The contents of the file appear on the screen.

# Working Directory

This option changes the pathname of the working directory. The working directory is the one DOS XE uses when you do not enter a pathname. (See **Pathnames** in **Chapter 2**.)

Type **[w]** and then press **[Return]**. DOS XE displays this prompt:



This prompt asks for the pathname of your new working directory. Type in the pathname and then press **[Return]**. The new pathname is displayed in the top left corner of the menus.

**Example:** Change the working directory to LETTERS.

1. Select **[w]** from the menu. This message appears:



2. Enter **D>LETTERS** and press **[Return]**. You have just changed the working directory.



# New Directory

To create a new directory, type **[N]** and press **[Return]**. DOS XE displays this prompt:

```
NEW DIRECTORY
GIVE NAME FOR NEW DIRECTORY
NAME?
```

Type in the pathname and press **[Return]**.

**Example:** Create a games directory in the PROGRAMS directory.

1. Select **[N]** from the menu. This prompt appears:

```
NEW DIRECTORY
GIVE NAME FOR NEW DIRECTORY
NAME?
```

2. Enter **D>PROGRAMS>GAMES** and press **[Return]**. The directory has been created.

**Note:** This will not work under the following conditions:

- If the **D1>PROGRAMS>** directory does not exist.
- If a file named **D1>PROGRAMS>GAMES** already exists and is locked or for any other reason cannot be erased or deleted.

# Delete Directory

This option deletes a directory from the disk. It does not erase files. Directories which contain files or other directories cannot be deleted.

Type **[D]** and press **[Return]**. DOS XE displays this prompt:



This prompt asks for the pathname of the directory to be deleted. Wildcards are not permitted. Enter the pathname and press **[Return]**.

**Example:** Delete the games directory.

1. Select **[D]** from the menu. This prompt appears:



2. Type **D>PROGRAMS>GAMES** and press **[Return]**. The directory is erased.

# Copy Files

The Copy File option can be used in the following ways:

- Copy a file to the same disk or to a different disk.
- Copy a directory to another directory on the same disk or to a different disk.
- Copy a group of files using wildcards..
- Copy the contents of the screen editor to a disk file.
- Copy a file to a device, such as a printer.

Each of these options is discussed in detail below.

**Warning:** The Copy Files option overwrites a part of memory normally reserved for programs. If you have a program in memory, and do not want to lose it, be sure to save the program before continuing the copy process.

## Copying a File

The following process details the steps necessary to make a copy of a file using one disk drive. This option copies a file from one disk to a different disk, prompting you to change disks during the process.

If you are familiar with the Copy Files option, use the short steps in the left column as reminders. Read the descriptions on the right side if you want details about a specific step.

For this example, you will use the defaults provided by the system. Optional changes to defaults, such as copying directories or using two disk drives, will be discussed later.

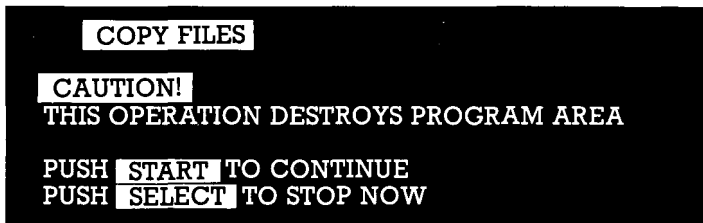
**Note:** You will need a formatted disk with available space to complete this operation.

### Action:

1. Press **C** and **[Return]**

### Description:

Selects the Copy Files option from the File Access menu.



**Note:** This prompt appears the first time you use Copy Files during a session with DOS XE. If the program area has already been destroyed, the prompt will not appear.

2. Press **[Start]**.

This begins the copying procedure. If you do not wish to use the Copy Files operation, press **[Select]** instead.

**CONTINUING**

**COPY FROM WHAT FILE?**

3. Enter the name of the file you want to copy and press **[Return]**.

Tells DOS XE which file to copy. For example, enter **WELCOME.BAS**.

**COPY TO WHAT FILE?**

4. Enter the name you want to give the copy, then press **[Return]**.

You must give the backup copy a name. For this example, **WELCOME.BAS** is the filename.

**COPY SUBDIRECTORIES OR ONLY FILES?**

**SUBDIRECTORIES/  FILES (S OR F)? F**

5. Press **[Return]**.

Copies files only. You do not need to type **F**, since that selection is provided by default.

**VERIFY EACH NAME BEFORE PROCEEDING?**

**VERIFY/  NO VERIFY (V OR N)? V**

6. Press **[Return]**.

The verify prompt enables you to confirm the filenames to be copied.

**ONE DRIVE, FILES ON SAME DISK?**

**SAME DISK/  NOT SAME (S OR N)? N**

7. Press **[Return]**.

Tells DOS XE that you are copying files from one disk to another disk.

```
PUT FROM DISK IN DRIVE 1  
PUSH START TO CONTINUE  
PUSH SELECT TO STOP NOW
```

8. Insert the disk with the file to be copied and press **[Start]**.

Make sure you insert the source disk correctly.

```
COPYING : WELCOME.BAS  
PUSH START TO COPY THIS FILE  
PUSH SELECT TO BYPASS IT
```

**Note:** This verification prompt is especially useful when copying with wildcards (discussed later), because it allows you to bypass files which you may not want to copy but are included within the group of files selected with the wildcards.

9. Press **[Start]**.

Confirms the name of the file you wish to copy.

```
CONTINUING  
PUSH TO DISK IN DRIVE 1  
PUSH START TO CONTINUE  
PUSH SELECT TO STOP NOW
```

**Warning:** This is your last chance to bail out before the new file is written to disk. If you wish to stop the procedure, press **[Select]**.

10. Insert the disk you are copying to and press **[Start]**.

Make sure the destination disk is not write-protected, otherwise it cannot receive the file.

```
PUT FROM DISK IN DRIVE 1
PUSH START TO CONTINUE
PUSH SELECT TO STOP NOW
```

11. Press **[Start]**.

Because **WELCOME.BAS** is a short file, you do not need to reinsert the original source disk. If you were copying a longer file, you might have had to repeat steps 8 to 10 several times before the entire file is copied to the destination disk.

```
SELECT ITEM OR ESC APE OR RETURN :
```

This indicates that the copy procedure is finished. You can select one of the following choices to continue:

- To copy another file, enter **[C]**.
- To return to the Main menu, press **[ESC]**.
- To return to the File Access menu, press **[Return]**.

### Copying a File to the Same Disk

If you want to make a backup copy of a file, but you want to keep it on the same disk, follow the above procedure, with a couple of important exceptions.

You cannot give the destination file the same filename as the source. If you are copying **WELCOME.BAS**, for example, you could name the destination file **HELLO.BAS**.

The only other change will be in step 7, where you must enter **[S]** (same disk) instead of accepting the default value **[N]** (not the same disk).

The copy procedure will end following step 9.

## Copying a File with Two Disk Drives

To copy a file with two disk drives, follow the procedure outlined in the **Copying a File** section, with the exception of steps 3 and 4. When you enter the filenames of the source and destination files, you must add a pathname.

**Example:** Copy a file called WELCOME.BAS from a disk in drive 1 to a disk in drive 2.

1. Type **D1>WELCOME.BAS** then press **[Return]**.
2. Change your response in step 4 to **D2>WELCOME.BAS**, then press **[Return]**.
3. Continue to respond to the prompts as you did in the section on **Copying a File to the Same Disk**. You will be told to insert the FROM disk (the source) in drive 1 and the TO disk (the destination) in drive 2.
4. Press **[Start]** and DOS XE will copy the file in drive 1 to drive 2. You will not need to swap disks.
5. When DOS XE finishes, you can choose another option, press **[ESC]**, or press **[Return]**.

## Copying Directories

You can use the Copy Files option to copy directories and subdirectories, as well as files. This allows you to move a directory into a subdirectory.

In other words, you could copy a directory containing business letter files into a directory containing other letter files. Your letter directory might already contain separate subdirectories with letters to your sister and letters to a friend named Ann.

**Note:** When you copy a directory, you copy the directory and all directories and files that it contains.

When prompted to enter filenames, you must enter the pathname of the directory you wish to copy, in this case maybe **D1>LETTERS.BUS** and the destination pathname **D2>LETTERS>**. Then you need to change the default in step 5 to **[S]**, telling DOS XE to copy subdirectories.

You can change the defaults for the number of disks and disk drives as you would if you were copying files.

The result would be a directory called **LETTERS** which would contain subdirectories called **LETTERS.SIS**, **LETTERS.ANN**, and the newly-copied **LETTERS.BUS**.

## Copying a Group of Files

By using wildcards in your filenames, you can copy a group of files at the same time. For example, you could use **LETTERS.\*** to copy all of your letter files. (For more information about wildcards, see **Wildcards** in **Chapter 2**.)

Specify the wildcards when prompted for the filename, then follow the procedure outlined in this chapter, making alterations for number of disks, disk drives, and so on.

**Note:** When copying with wildcards, keep Verify mode enabled. This allows you to select the files you want to copy by pressing **[Start]** and reject those you do not by pressing **[Select]**. Rejected files will remain on the source disk or directory, but will not be copied to the destination.

## Copying To and From Devices

DOS XE also makes it possible to copy files and directories to devices such as a printer or serial port, as well as copy to and from the screen editor. The procedure is much the same as that outlined above. You must, however, change the device name.



In other words, instead of entering the pathname, you can answer the **COPY TO WHAT FILE?** prompt by typing **P:**. This copies the file to a printer and prints it out.

The valid device names include

- **P:** (printer)
- **E:** (screen editor--display a file on screen)
- **R:** (serial port--send the file to a peripheral connected to the serial port on your Atari computer, such as a serial printer)

Although you cannot copy information from a printer or serial port (except for possible specific cases outlined in your peripheral's manual), you *can* copy from the screen editor. Entering **E:** as the FROM device name allows you to copy the contents of the screen editor to a disk file with the filename you specify.

For example, you can copy the contents of the screen editor to a disk file named **SCREEN** by entering **E:** as the FROM file and **D1>SCREEN** as the TO file. Press **[Return]**.

**Warning:** You must give the destination disk file a name, otherwise the data you save from the screen editor will be irretrievable.

The screen goes blank, except for the cursor in the top left corner. Simply type the information you want to save. You could, for example, type the words, **THIS IS THE INFORMATION FOR THE SCREEN FILE**. When you finish typing, press **[Control] [3]**. The data you typed will be saved to the disk with the filename **SCREEN**.

To copy information from the screen editor to a printer, follow the above steps, but instead of typing **D1>SCREEN** as the destination file, enter **P:**. Type what you want to print, then enter **[Control] [3]**. The information will print out on your printer.

**Note:** You can type more than one screenful of data, but you can only see the text that is currently onscreen. You cannot return to a previous section. However, everything you typed will be copied to the disk or printer.

## Append To A File

This option adds a copy of one file (the source) onto the end of another file (the destination). For example, start with two files, one containing "DEF" and the other "ABC". Append "DEF" (the source) onto "ABC" (the destination). The result will be an unchanged source file, "DEF", and a destination file which now has "ABCDEF".

If you only have one disk drive, both the source and destination files must be on the same disk.

You cannot append to a protected destination file.

Type **[A]** and press **[Return]**. DOS XE displays this prompt:



```
APPEND TO A FILE
APPEND FROM WHAT FILE?
```

This prompt asks for the pathname of the source file. Type in the pathname and press **[Return]**. DOS XE displays the following prompt:



```
APPEND TO WHAT FILE?
```

This prompt asks for the pathname of the destination file. Type in the pathname and press **[Return]**.

Append does not destroy the program area and can be used to copy small files, if desired. If the destination file does not exist, it is created and the append becomes a copy. When used this way, Append creates a copy of a file but gives it a new creation date.

**Example:** Append letter 2 onto letter 1.

1. Select **[A]** from the menu. This prompt appears:

```
APPEND TO A FILE
APPEND FROM WHAT FILE?
```

2. Type **D>LETTERS>LETTER2.BUS** and press **[RETURN]**. You will see the prompt:

```
APPEND TO WHAT FILE?
```

3. Enter **D>LETTERS>LETTER1.BUS** and press **[RETURN]**. The computer does its work.

## Initialize Disk

Before a new disk can be used by your disk drive, it must be initialized. This option performs that procedure.

1. Type **[I]** and press **[Return]**. DOS displays this prompt:

```
INITIALIZE DISK
WHAT DRIVE NUMBER?
```

2. This prompt asks which disk drive has the disk which is to be initialized. Enter the drive number (no "D") and press **[Return]**. DOS XE displays the following prompt:

DRIVE IS A(N) XF551

AVAILABLE DEVICE TYPES ARE

AT810  
AT1050  
XF551  
SSDD

CHOOSE ONE OF ABOVE DEVICE TYPES  
GIVE ITS NAME >

3. If the drive number is the RAM disk on a 130XE (see **Configuring DOS XE** in **Chapter Two**), DOS XE displays this prompt instead of the one above:

DRIVE IS A(N) 130RAM

AVAILABLE DEVICE TYPES ARE

130RAM

CHOOSE ONE OF ABOVE DEVICE TYPES  
GIVE ITS NAME >

4. You can select any drive type shown in the list below:

**AT810** Single density, single sided disk drives.  
**AT1050** Dual density, single sided disk drives.  
**XF551** Double density, double sided disk drives.  
**SSDD** 5 1/4 inch single sided, double density disk drives.  
**130RAM** The RAM disk in the Atari 130XE computer.

Type in the name and press **[Return]**. DOS XE displays this prompt:

READY TO INITIALIZE DISK

PUSH **START** TO CONTINUE  
PUSH **SELECT** TO STOP NOW

5. This prompt is a safety check. Initializing a disk will destroy any information on it. If you are certain the disk is blank and you want to initialize it, press **[Start]**.

After a short wait while DOS XE does its work, the disk will be initialized and ready to use in disk drive you specified.

# CHAPTER 5

## THE MACHINE LANGUAGE ACCESS MENU

To enter the Machine Language Access menu from the Main menu, type **[M]** then press **[Return]**. The Machine Language Access menu appears on the screen.

```
DIRECTORY          TODAY: 20SE88
DI>
-----

      DOS XE MACHINE LANGUAGE ACCESS MENU

F ILES LISTING
W ORKING DIRECTORY

R UN A BINARY FILE
L OAD A BINARY FILE (DO NOT RUN)
S AVE MEMORY TO A BINARY FILE
A PPEND MEMORY TO A BINARY FILE
D ISPLAY MEMORY
C HANGE MEMORY
G O TO A MACHINE LANGUAGE PROGRAM
-----

SELECT ITEM OR [ESC] APE FOR MAIN MENU:
```

Below the pathname under the word **DIRECTORY** is the list of options available in this menu. To select an option, press the key corresponding to the highlighted letter, then press **[Return]**. The last line reminds you that you can return to the Main menu by pressing **[Esc]**.

After you execute an option, DOS XE displays this line:

```
SELECT ITEM OR [ESC] APE OR [RETURN] :
```

This is a reminder that, although the current menu is not displayed, you can press a letter and select an option from it. You can also press **[Esc]** to return to the Main menu, or press **[Return]** to display the current menu again.

## OPTIONS

This section provides details on how to use each option available in the Machine Language Access menu. Be sure to read section before attempting to execute an option.

### Files Listing

This option gives a listing of the contents of a directory. It is available in all three menus. Detailed instructions are contained in the File Access menu section of **Chapter 4**.

### Working Directory

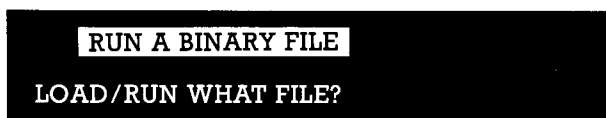
This option lets you change your working directory. It is available in all three menus. Detailed instructions are contained in the File Access menu section of **Chapter 4**.

### Run a Binary File

A binary file is a special file format. This option loads a binary file from a disk and runs it (if it has a run address).

**Note:** This option does not work with programs which require a language such as BASIC. You can only run BASIC programs from BASIC, not from this option. (See **Binary Files** in **Chapter 7**.)

Type **[R]** and press **[Return]**. DOS XE displays this prompt:



Enter the pathname of the file you wish to run.

**Example:** Run a game called MAZE.BIN in the main directory of the disk in drive 1.

1. Select **[R]** from the Machine Language Access menu. The following prompt appears on your screen:

```
RUN A BINARY FILE
LOAD/RUN WHAT FILE?
```

2. Enter **D1>MAZE.BIN** and press **[Return]**. The file loads and runs automatically.

## Load a Binary File (Do Not Run)

This option loads a binary file, but does not run it.

**Note:** This option does not work with programs which require a language, as BASIC programs do.

Type **[L]** and press **[R]return**. DOS XE displays this prompt:

```
LOAD A BINARY FILE
LOAD/RUN WHAT FILE?
```

The prompt asks for the pathname of the file you wish to load into memory.

**Example:** Load a binary file called MAZE.BIN from the current working directory, but do not run it.

1. Select **[L]** from the menu. This prompt appears:

```
LOAD A BINARY FILE
LOAD/RUN WHAT FILE?
```



2. Enter **MAZE.BIN** and press **[Return]**. The file is loaded into memory.

## Save Memory To A Binary File

This option saves the information from a block of memory into a binary file. It allows you to define an initialize address and a run address.

A binary file can have a run address and/or an initialize address. These addresses tell DOS XE how to run the program. (See **Binary Files** in **Chapter 7**.)

Enter the addresses in hexadecimal or, by preceding the number with a number sign (**#**), in decimal notation.

Type **[S]** and press **[Return]**. DOS XE displays this prompt:

```
SAVE MEMORY TO A BINARY FILE
SAVE TO WHAT FILE?
```

This prompt asks for the pathname of the file to which the memory data will be saved. Type in the pathname and press **[Return]**. DOS XE displays the following prompt:

```
START ADDRESS?
```

This prompt asks for the address of the beginning of the block of memory to save. Enter the address and press **[Return]**. DOS XE displays the following prompt:

```
END ADDRESS?
```

This prompt asks for the address of the end of the block of memory to save. Enter the address and press **[Return]**. DOS XE displays the following prompt:

```
INITIALIZE ADDRESS?
```

This prompt asks for the initialize address of the program. If there is no initialize address, press **[Return]**. If there is an initialize address, enter it and press **[Return]**. DOS XE displays the following prompt:

```
RUN ADDRESS?
```

This prompt asks for the run address of the program. Enter the address, if there is one, and press **[Return]**. If there is no run address, just press **[Return]**.

**Example 1:** Page 6 (0600-06FF, #1536-#1791) is an area of memory used for many utilities. Save a routine in page 6 to a file called UTILITY.BIN. Assume that the routine has no initialize address and that the run address is the beginning of the routine. We will use hexadecimal notation.

1. Select **[S]** from the menu. You see the following:

```
SAVE MEMORY TO A BINARY FILE
```

```
SAVE TO WHAT FILE?
```

2. Enter **D1>UTILITY.BIN** and press **[Return]**. This prompt appears on the screen:

```
START ADDRESS?
```

3. Enter **0600** and press **[Return]**.

```
END ADDRESS?
```

4. Enter **06FF** and press **[Return]**.

INITIALIZE ADDRESS?

5. Press **[Return]**.

RUN ADDRESS?

6. Enter **0600** and press **[Return]**.

**Example 2:** This is the same thing done in decimal notation:

1. Select **[S]** from the menu. This prompts appears:

SAVE MEMORY TO A BINARY FILE

SAVE TO WHAT FILE?

2. Enter **D1>UTILITY.BIN** and press **[Return]**. The screen displays this prompt:

START ADDRESS?

3. Enter **\*1536** and press **[Return]**.

END ADDRESS?

4. Enter **\*1791** and press **[Return]**.

INITIALIZE ADDRESS?

5. Press **[Return]**.

RUN ADDRESS?

6. Enter **\*1536** and press **[Return]**.

# Append Memory To A Binary File

This option adds information from a block of memory onto the end of an existing binary file. (See **Binary Files** in **Chapter 7**.) It lets you define both an initialize address and a run address. Enter the addresses in hexadecimal or, by preceding the number with a number sign (#), in decimal.

Type **[A]** and press **[Return]**. DOS XE displays this prompt:

```
APPEND MEMORY TO A BINARY FILE
APPEND TO WHAT FILE?
```

This prompt asks for the pathname of the file to which data will be appended. Type in the pathname and press **[Return]**. DOS XE displays the following prompt:

```
START ADDRESS?
```

This prompt asks for the address of the beginning of the block of memory to be appended. Enter the address and press **[Return]**.

If you want to append an initialize or run address and not a block of memory, just press **[Return]**.

If you do not enter a start address, DOS XE skips the next prompt, otherwise it displays the following prompt:

```
END ADDRESS?
```

This prompt asks for the address of the end of the block of memory to be appended. Enter the address and press **[Return]**. DOS XE displays the following prompt:

```
INITIALIZE ADDRESS?
```

This prompt asks for the initialize address of the program. If there is an initialize address, enter it and press **[Return]**. If there is no initialize address, just press **[Return]**. DOS XE displays the following prompt:

```
RUN ADDRESS?
```

This prompt asks for the run address of the program. Enter the run address and press **[Return]**. If there is no run address, just press **[Return]**.

**Example 1:** LMARGN and RMARGN (52-53, #82-#83) set the left and right margins for the screen display. By appending the values in these locations to a program, the margins will be set automatically when the program loads. We will append the margins to a utility program named UTILITY.BIN. Since this is not a program, it does not have initialization or run addresses. We will use hexadecimal notation.

1. Select **[A]** from the menu. You see this prompt:

```
APPEND MEMORY TO A BINARY FILE
```

```
APPEND TO WHAT FILE?
```

2. Enter **D1>UTILITY.BIN** and press **[Return]**.

```
START ADDRESS?
```

3. Enter **52** and press **[Return]**.

```
END ADDRESS?
```

4. Enter **53** and press **[Return]**.

```
INITIALIZE ADDRESS?
```

5. Press **[Return]**.

RUN ADDRESS?

6. Press **[Return]**. The computer appends the memory to the file.

**Example 2:** This is the same thing, this time done in decimal notation:

1. Select **[A]** from the menu. The following sequence of prompts is displayed:

APPEND MEMORY TO A BINARY FILE

APPEND TO WHAT FILE?

2. Enter **D1>UTILITY.BIN** and press **[Return]**.

START ADDRESS?

3. Enter **\*82** and press **[Return]**.

END ADDRESS?

4. Enter **\*83** and press **[Return]**.

INITIALIZE ADDRESS?

5. Press **[Return]**.

RUN ADDRESS?

6. Press **[Return]** and the computer does its job.

# Display Memory

This option displays the contents of memory locations in your computer. The contents are displayed in hexadecimal numbers and ATASCII characters. You can enter the addresses in hexadecimal or, by preceding the number with a number sign (#), in decimal notation. (See examples.)

Type **[D]** and press **[Return]**. DOS XE displays the following prompt:

```
DISPLAY MEMORY
START ADDRESS?
```

This prompt asks the start address of the block of memory you want to see. Enter the address and press **[Return]**. DOS XE displays the following prompt:

```
END ADDRESS?
```

This prompt asks the address of the end of the block of memory you want to see. Enter the address and press **[Return]**. If you want to see fewer than eight address locations, you can press **[Return]** without entering an address. DOS XE displays a line like the following:

```
0006 00 FF 00 01 3B 27 24 27 . . . . ; '$'
```

There will be more lines for larger blocks of memory. Each line displays the contents of eight memory locations. All numbers are displayed in hexadecimal.

The first number is the starting address for the line. The next eight numbers are the values in the next eight memory locations. The last eight figures are the contents of the same eight addresses displayed as ATASCII characters. Inverse video characters are shown in normal video and editing characters are shown as periods.

If the block of memory is too large to fit on the screen, the screen will scroll. To pause the scroll, hold down **[Control]** and press **[1]**. It can be restarted by typing **[Control] [1]** again. The scroll can be interrupted by pressing **[Break]**.

**Example 1:** Display the value of MEMLO (02E7-02E8, #743-#744) using hexadecimal addresses.

1. Select **[D]** from the menu. You will see this prompt:

```
DISPLAY MEMORY
START ADDRESS?
```

2. Enter **02E7** and press **[Return]**.

```
END ADDRESS?
```

3. Press **[Return]**. The memory is displayed on the screen.

**Example 2:** Display the values in the Atari color registers (02C0-02C8, #704 -#712) using decimal addresses.

1. Select **[D]**. The following prompt appears:

```
DISPLAY MEMORY
START ADDRESS?
```

2. Enter **\*704** and press **[Return]**.

```
END ADDRESS?
```

3. Enter **\*712** and press **[Return]**. The memory is displayed on the screen.



# Change Memory

This option permits you to change the values in your computer's memory. You can enter the address of the location to change or the new value in hexadecimal or, if you precede the number with a number sign (\*), in decimal notation.

**Warning:** Incorrect memory changes can cause the computer to lock up or crash. You may need to reboot to recover.

Type **[C]** and press **[Return]**. DOS XE displays this prompt:

```
CHANGE MEMORY
START ADDRESS?
```

This prompt asks for the address of the first memory location you want to change. Enter the address and press **[Return]**. DOS XE displays a line similar to the following:

```
006A A0 NEW DATA?
```

The first number is the address you entered in hexadecimal notation; the next is the contents of that address.

The prompt "NEW DATA?" asks what you want to change the value to. Enter the new value and press **[Return]**. DOS XE displays a line similar to the following:

```
006B 28 NEW DATA?
```

This line is for the next address location. You can change it, as above, and DOS XE will display the next location. If you do not want to change it, but want to go on to the next location, type an asterisk (\*) and press **[Return]**. If you want to return to the menu, just press **[Return]**.

**Example:** Change the value in LMARGN (0052, #82) to set the left margin of the screen to 0.

1. Select **[C]** from the menu. You will see the following prompt:

```
CHANGE MEMORY
START ADDRESS?
```

2. Enter **52** and press **[Return]**.

```
0052 02 NEW DATA?
```

3. Enter **0** and press **[Return]**.

```
0053 27 NEW DATA?
```

4. Press **[Return]**. The left margin on your screen shifts to the edge of the display area.

**Note:** To return to normal, follow the above process, but enter **02** in response to the **0052 00 NEW DATA?** prompt. All other values remain the same.

## Go to a Machine Language Program

This option runs a program which is already in memory.

Type **[G]** and press **[Return]**. DOS XE displays the following prompt:

```
GO TO A MACHINE LANGUAGE PROGRAM
START ADDRESS?
```

This prompt asks for the start address of the program. This is usually the run address, but you may start at another point. Enter the address and press **[Return]**. If you enter an asterisk instead of an address, DOS XE uses the run address in RUNAD (02E0-02E1, #736-#737). If the run address has not been corrupted, the last program loaded by Load a Binary File (Do Not Run) will run.

**Warning:** An incorrect start address can cause the computer to lock up or crash. You may need to reboot to recover.

**Example:** Run BASIC (start address = A000, #40960). This will only work if BASIC is enabled.

1. Select **[G]** from the menu. This prompt appears:

```
GO TO A MACHINE LANGUAGE PROGRAM
START ADDRESS?
```

2. Enter **A000** and press **[Return]**. The **Ready** prompt appears. You are now in BASIC.
3. To return to DOS XE, type **DOS**. You are back at the Main menu.

# CHAPTER 6

## THE SYSTEM FUNCTION MENU

To enter the System Function menu from the Main menu, type **[S]** and press **[Return]**. The System Function menu appears on the screen.

```
DIRECTORY          TODAY: 20SE88
DI>
-----
      DOS XE SYSTEM FUNCTION MENU

F ILES LISTING
W ORKING DIRECTORY

R UN A BATCH FILE
S ET CURRENT DATE

I NITIALIZE DISK
C REATE DOSXE.SYS FILE
D UPLICATE A DISK
A LLOW DOS 2.X ACCESS
-----

SELECT ITEM OR [ESC] APE FOR MAIN MENU:
```

Below the pathname of the working directory, shown under the word "DIRECTORY" in the top left corner of the screen, is the list of options available in this menu. Select an option by pressing the key corresponding to the highlighted letter, then press **[Return]**. At the bottom is a reminder that you can return to the Main menu at any time by pressing **[Esc]**.

After you execute an option, DOS XE displays the line:

```
SELECT ITEM OR [ESC] APE OR [RETURN] :
```

This is a reminder that you can press a letter and select an option from the current menu, even though it is not displayed. You can also press **[Esc]** to return to the Main menu, or **[Return]** to display the current menu again.

# OPTIONS

Following are detailed descriptions of each option available in the System Function menu. We recommend that you read each section before executing an option.

## Files Listing

This option gives a listing of the contents of a directory. It is available in all three menus. Detailed instructions are contained in **Chapter 4**.

## Working Directory

This option lets you change your working directory. It is available in all three menus. Detailed instructions are contained in **Chapter 4**.

## Run A Batch File

A batch file is a text file of DOS XE commands. It can be used to automate DOS XE functions. A batch file can be no longer than 511 bytes, but it can invoke another batch file. (See **Command Line Entry** and **Batch Files** in **Chapter 7**.) This option runs a batch file.

Type **[R]** and press **[Return]**. DOS XE displays this prompt:



RUN A BATCH FILE

This prompt asks for the pathname of the batch file to run. Enter the pathname and press **[Return]**.

**Example:** Run a batch file named BACKUP.BAT.

1. Select **[R]** from the menu. This prompt appears:

```
RUN A BATCH FILE
RUN WHAT FILE?
```

2. Enter **D1>BACKUP.BAT** and press **[Return]**. The file loads and runs.

## Set Current Date

DOS XE datestamps each file, but you must tell DOS XE what the date is. This option lets you set the correct date.

Type **[S]** and press **[Return]**. DOS XE displays the prompt:

```
SET CURRENT DATE
GIVE DATE IN FORM DDMMYY
VALID MONTHS (MM) ARE:
JA FB MR AP MY JN JL AG SE OC NO DE
SET TO WHAT DATE?
```

This prompt asks what the correct date is. It also is a reminder of the form in which the date must be entered: two characters each for day, month, and year, with no spaces between them. A single character day may be entered as one character or may be preceded by a zero. The month must be entered as in the example. Enter the date and press **[Return]**.

**Example:** Set the date to January 1, 1989.

1. Select **[S]** from the menu and this message will appear:

```
SET CURRENT DATE
GIVE DATE IN FORM DDMMYY
VALID MONTHS (MM) ARE:
JA FB MR AP MY JN JL AG SE OC NO DE
SET TO WHAT DATE?
```

2. Enter **01JA89** or **1JA89** and press **[Return]**. Any files you work on today will be stamped with that date.

## Initialize Disk

This option prepares a new disk for use with DOS XE. It is also available in the File Access menu. Detailed instructions can be found in **Chapter 4**, under **Initialize Disk**.

## Create DOSXE.SYS File

This option writes DOS XE from the computer memory back into the disk file DOSXE.SYS, the method used to save a reconfigured version of DOS XE. (See **Configuring DOS XE** in **Chapter 2**.) Type **[C]** and press **[Return]**. DOS XE displays the following prompt:

```
CREATE DOSXE.SYS FILE
WHAT DRIVE NUMBER?
```

Enter **[1]** and press **[Return]**. The file is written to the disk.

**Warning:** Do not use this option unless a files listing for the disk shows at least 15k bytes free. If a DOSXE.SYS file already exists, this command will overwrite it, unless it is protected.

# Duplicate a Disk

This option copies the contents of one disk onto another disk. The original disk is called the source disk and the other disk is the destination disk. It is a good idea to cover the write-protect notch on the source disk so it will not accidentally be erased. The destination disk is automatically initialized during the duplication process. Type **[D]** and press **[Return]**. DOS XE displays the following prompt:

```
DUPLICATE A DISK
DUPLICATE FROM WHAT DRIVE NUMBER?
```

This prompt asks which disk drive contains the source disk. This will usually be drive one. Type the number (without the "D") and press **[Return]**. DOS XE displays the following prompt:

```
DUPLICATE TO WHAT DRIVE NUMBER?
```

This prompt asks which disk drive will contain the destination disk. If you only have one drive, this will be drive one. If you have more than one drive, this will usually be drive two. If the source drive number is different from the destination drive number, DOS XE displays this prompt:

```
PUT TO DISK IN DRIVE 2
PUT FROM DISK IN DRIVE 1

PUSH START TO CONTINUE
PUSH SELECT TO STOP NOW
```



If the source drive number is the same as the destination drive number, DOS XE displays the following prompt:

```
PUT FROM DISK IN DRIVE 1  
PUSH START TO CONTINUE  
PUSH SELECT TO STOP NOW
```

These prompts remind you to put the disk(s) in the proper drive(s) and press **[Start]** to begin the duplication or press **[Select]** to abort it. If you want to continue the duplication, press **[Start]**.

If you are using two drives, the duplication will be completed. If you are using one drive, DOS XE does part of the duplication and then displays this prompt:

```
PUT TO DISK IN DRIVE 1  
PUSH START TO CONTINUE  
PUSH SELECT TO STOP NOW
```

This prompt tells you to swap the source and destination disks. Swap the disks and press **[Start]**. Depending on how much information is on the source disk, you may have to swap disks several times.

**Example 1:** Duplicate a disk with one drive.

1. Select **[D]** from the menu and follow this sequence of prompts:

```
DUPLICATE A DISK  
DUPLICATE FROM WHAT DRIVE NUMBER?
```

2. Enter **[1]** and press **[Return]**.

```
DUPLICATE TO WHAT DRIVE NUMBER?
```

3. Enter **[1]** and press **[Return]**.

```
PUT FROM DISK IN DRIVE 1  
PUSH START TO CONTINUE  
PUSH SELECT TO STOP NOW
```

4. Press **[Start]**. The word **CONTINUING** appears, while the disk drive reads the first disk. After a moment, this prompt is displayed:

```
PUT TO DISK IN DRIVE 1  
PUSH START TO CONTINUE  
PUSH SELECT TO STOP NOW
```

5. Press **[Start]**. The computer displays **CONTINUING** while information is written to the second disk.
6. Switch disks according to the prompts until the process is complete.

**Example 2:** Duplicate a disk with two drives:

1. Select **[D]** from the System Function menu. The following prompt appears:

```
DUPLICATE DISK  
DUPLICATE FROM WHAT DRIVE NUMBER?
```

2. Enter **[1]** and press **[Return]**.

```
DUPLICATE TO WHAT DRIVE NUMBER?
```

3. Enter **[2]** and press **[Return]**. You will see this prompt:

```
PUT TO DISK IN DRIVE 2  
PUT FROM DISK IN DRIVE 1  
  
PUSH START TO CONTINUE  
PUSH SELECT TO STOP NOW
```

4. Press **[Start]** The computer displays **CONTINUING** while the disk is being copied.

## Allow DOS 2.X Access

This option allows you to access disks which are in DOS 2.0 or DOS 2.5 formats. (DOS 2.0 and DOS 2.5 are previous disk operating systems for Atari computers.)

The extra program code to accomplish this access takes up valuable memory space; therefore, it is not loaded unless you specifically ask for it with this option.

**Note:** The DOS2.SYS file must be on the disk in drive one when you select this option.

Once the code has been loaded, you can address any DOS 2.0 or 2.5 disk by calling it A: in the pathname (instead of D:). All DOS read functions except Initialize Disk can be performed on DOS 2.0 and DOS 2.5 disks.

Type **[A]** and press **[Return]**. DOS XE displays this prompt:

```
ALLOW DOS 2.X ACCESS
CAUTION
THIS OPTION DESTROYS PROGRAM AREA
PUSH START TO CONTINUE
PUSH SELECT TO STOP NOW
```

This prompt warns you that loading the DOS 2.X code overwrites the program area of memory. If you have valuable information in memory which has not been saved, you can bail out now. This warning is not shown if the program area has already been overwritten. Press **[Start]** to load the DOS 2.X code. DOS XE displays the **CONTINUING** prompt while the process is completed.

# CHAPTER 7

## ADVANCED TOPICS

### COMMAND LINE ENTRY

Once you become familiar with the DOS XE menus, you can speed things up with command line entries. Most DOS XE options require additional information to execute.

For example, the Working Directory option asks you the pathname of the new working directory. The answer to this question is called a parameter. DOS XE permits you to enter parameters at the same time you select an option. This is called a command line entry. For example, from the File Access menu, use the menus as follows:

1. Type **[W]** and press **[Return]**.



2. Type **D1>LETTERS** and press **[Return]**.

You can do the same thing by typing this command line:

**W D1>LETTERS [Return]**

DOS XE displays each of the prompts and fills in the responses from the parameters in the command line. This speeds things up and gives a visual reminder of what is happening.

A command can be the first letter of an option, or, to be more readable, it can be a whole word. DOS XE only looks at the first letter, so the word and the spelling don't matter.

You could use WORKINGDIRECTORY or WORKDIR instead of "W" in the above example, . This is not normally used for command line entry, but it is good practice for batch files. (See **Batch Files** in this chapter.)

DOS XE treats each space (or group of spaces) in a command as a **[Return]**. After it accepts the W, it sees the space and enters a **[Return]** so the Working Directory option is executed.

Some commands can be destructive (like initializing a disk). They have prompts that must be answered by pressing **[Start]**. Those prompts cannot be answered in command lines. DOS XE will wait for you to actually press **[Start]**. This protects your files.

Similarly, error messages require that you press **[Start]**. Regardless of the structure of a command line, DOS XE will not pass an error message. DOS XE will pause until you have seen the message and pressed **[Start]**. Errors abort the rest of the command line, as well as any active batch file.

In addition to accepting parameters for menu options, the command line will let you move between menus. A period at the beginning of a command always takes you to the Main menu, like an **[Esc]** does in menu prompts.

Once inside the Main menu, other menus are selected in the usual manner. For example, if you wanted to view a file named D1>FILE and you are in the Machine Language Access menu, you could use the menus as shown on the next page.

Press **[Esc]**. You will see the following menu:

```
DOS XE DISK OPERATING SYSTEM
COPYRIGHT 1988 ATARI CORP.
VERSION 01.00
-----
MAIN MENU
FILE ACCESS MENU
MACHINE LANGUAGE ACCESS MENU
SYSTEM FUNCTION MENU
EXIT TO BASIC
-----
```

Type **[F]** then **[Return]** to go to the File Access menu.

```
DIRECTORY          TODAY: 20SE88
DI>
-----
DOS XE FILE ACCESS MENU
FILES LISTING     WORKING DIRECTORY
PROTECT FILES    NEW DIRECTORY
UNPROTECT FILES  DELETE DIRECTORY
ERASE FILES      COPY FILES
RENAME FILES     APPEND TO A FILE
VIEW A FILE      INITIALIZE DISK
-----
SELECT ITEM OR [ESC] APE FOR MAIN MENU:
```

Type **[V]** and press **[Return]**. The following prompt appears on the screen:



Type **D1>FILE** and press **[Return]**. The file is displayed.

Or, you could do the same thing with the following command line:

```
. F V D1>FILE [Return]
```

DOS XE will flip through the menus instantly and begin displaying the file.

Enter multiple commands in a command line by separating them with semicolons. Selecting a menu is not a separate command and does not need to be followed by a semicolon, though it will not hurt to do so. The semicolon answers any remaining prompts in the current command with line returns and moves on to the next command.

This gives you a quick way to select menu options with default parameters. For example, **F; [Return]** gives you a listing of the working directory on the screen.

You can also use a command to print everything that appears on the screen to your printer. Type **[Control] [P]** and everything will be printed as it appears on the screen. This command helps you keep track of your work. This feature can be turned off by typing **[Control] [P]** again.

A command line cannot be more than two screen lines (80 characters) long. Consider this example. You have booted up and want to set the date, create a new directory for some temporary files, and exit to BASIC. The following command line would do it:

```
. S S 31OC88;. F N D1>TEMP;. E [Return]
```



This command line works like this:

- The period insures that we are in the Main menu. We already were, but it does not hurt to make sure whenever you change menus.
- The first S selects the System Function menu.
- The second S selects the Set date option.
- 31OC88 is the date you want to enter.
- The semicolon indicates the end of the command.
- The period returns you to the Main menu.
- The F selects the File Access menu.
- The N selects the New Directory option.
- D1>TEMP is the pathname of the new directory.
- The semicolon ends the command.
- The period starts the next command and returns to the Main menu.
- The E exits to BASIC.

The workings of the spaces, semicolons, and **[Returns]** can be a little confusing, so here are the rules:

1. A space, or group of spaces, are treated as a **[Return]**.
2. A semicolon answers all the remaining prompts in a command with **[Returns]**.
3. Spaces adjacent to a semicolon are ignored.
4. Spaces adjacent to a **[Return]** are ignored.
5. A semicolon followed by a **[Return]** (for example, at the end of a line) is treated as a semicolon alone.

# BATCH FILES

Batch files are command lines which are saved as text files. When a batch file is run with the Run a Batch File option in the System Function menu, the command lines are executed. A batch file named AUTORUN.BAT runs automatically when DOS XE is booted.

Commands may be separated by semicolons as in command lines, or they may be on individual lines separated by **[Returns]**. A batch file is limited to 511 bytes, but it can run another batch file if the last command is . S R FILE-NAME.

The ! symbol can be used for a comment line which can be read on the screen but will be ignored by DOS XE.

Since batch files can be saved for a long time, it is a good idea to use descriptive words for commands instead of initials. It is also important to remember that batch files are run from the System Function menu. Since batch files are text files, it is easy to review the contents of one with the View File option in the File Access menu. Batch files are usually identified by the extender ".BAT".

Here is an example of a batch file which creates a new directory for temporary files and selects it as the working directory, returns to the Main menu, and exits to BASIC.

```
. FILEMENU  
NEWDIR D1>TEMP; WORKDIR D1>TEMP  
. E
```

Batch files can be created with any text editor or word processor which produces ASCII text files (AtariWriter Plus and AtariWriter 80 do this with the "Save ASCII" command).

Batch files also can be created directly from DOS XE by copying from the screen to a file. To do so, use the following command line:

```
. FILEMENU COPY E: D1>SAMPLE.BAT
```

This takes you to the File Access menu, executes the Copy Files option, then gives the screen editor (E:) as the source and SAMPLE.BAT as the destination. The screen blacks out briefly, then comes up empty with the cursor in the upper left corner. You can now type your batch file using the Atari editing features.

As you end each line with a **[Return]**, it is sent to DOS XE. After you have entered the last line and the **[Return]**, hold down **[Control]** and type **[3]**. This is the code for the end of a file. DOS XE closes the file and returns you to the File Access menu. Your batch file is ready.

## BINARY FILES

Binary files are the standard DOS XE file format. They are created by the Save Memory To Binary File option in the Machine Language menu. They are also created by most programming assemblers and compilers.

A binary file begins with a 6 byte header which contains the following information:

Byte #	Decimal	Hexadecimal	Description
1	255	FF	Identification code
2	255	FF	for binary file
3	0	00	LSB Starting address
4	60	3C	MSB 3C00 (#15360)
5	255	FF	LSB Ending address
6	91	5B	MSB 5BFF (#23551)

The first two bytes are always FF FF (#255 #255). They are the identifying code for binary files. The next two bytes are the starting address in Least Significant Byte, Most Significant Byte format. This is where DOS XE will begin loading the data into memory. The final two bytes are the ending address, again in the LSB, MSB format.

The data comes after the header. There must be the same amount of data as memory between (and including) the starting and ending addresses.

In a compound binary file, the data does not load into one continuous block of memory. It has two or more blocks of data which load into different areas. In the header for additional blocks, the FF FF identification code is optional.

A compound binary file has the following format:

<b>Byte #</b>	<b>Decimal</b>	<b>Hexadecimal</b>	<b>Description</b>
1	255	FF	Identification code
2	255	FF	for binary file
3	0	00	LSB Starting address
4	60	3C	MSB 3C00 (#15360)
5	255	FF	LSB Ending address
6	91	5B	MSB 5BFF (#23551)
.	.	.	.
.	.	.	Data, 2000 (#8192) bytes
8199	255	FF	ID code for binary file (optional)
8200	255	FF	file (optional)
8201	0	00	LSB Starting address
8202	6	06	MSB 0600 (#1536)
8203	128	80	LSB Ending address
8204	6	06	MSB 0680 (#1664)
.	.	.	.
.	.	.	Data, 0081 (#129) bytes
.	.	.	.

Additional blocks follow the same format as the second block. Compound binary files can be produced by assemblers and compilers or they can be created with the Append Memory To A Binary File option in the Machine Language Access menu or the Append To A File option in the File Access menu.

Memory locations 02E2-02E3 (#738-#739) are called INITAD. An optional initialization address goes into them. If, during a binary load, DOS XE loads an address into INITAD, it immediately jumps (JSR) to the code pointed to by that address. The code is run and, if the code returns control to DOS XE, the binary load continues. The code to be run should be loaded before INITAD is loaded.

A compound binary file can have many initialization addresses and each will be run as it is loaded. DOS XE will append an initialization address to the end of a binary file if you give the address during Save Memory To Binary File or Append Memory To A Binary File.

**Note:** DOS XE leaves the loading file and the keyboard device (K:) open during the call to the initialization code. Thus, such a code should not alter or use IOCBs (channels) 1 or 2.

Memory locations 02E0-02E1 (#736-#737) are called RUNAD. An optional run address goes into them. If a run address is loaded into RUNAD while DOS XE is executing the Run A Binary File option, DOS XE finishes the binary load and then jumps (JSR) to the code at that address.

Since the code is not run until the binary load is complete, it can be loaded after RUNAD. If a compound binary file has more than one run address, only the last one loaded will be executed. DOS XE will append a run address to the end of a binary file if you give the address during the Save Memory To Binary File or the Append Memory To A Binary File options.

The Load A Binary File option will load RUNAD along with the rest of the file, but will not jump to that address. When executing the Go To A Machine Language Program option, if you answer the START ADDRESS? prompt with an asterisk (\*), DOS XE will go to the address in RUNAD.

If a DOS XE disk has a binary file named AUTORUN.SYS in the root directory, that file will run automatically when the disk is booted up. If a batch file named AUTORUN.BAT is on the disk, it will be run after AUTORUN.SYS.

# USING DOS XE WITH OTHER DISK DRIVES

DOS XE works with all Atari-compatible disk drives. It will, however, be limited to the capacities of those disk drives. When initializing a disk, it must be formatted so that your disk drive can read it.

Disk drives can read disk formats as follows:

<b>Atari 810</b>	<b>AT810 only</b>
<b>Atari 1050</b>	<b>AT810 or AT1050</b>
<b>Atari XF551</b>	<b>AT810, AT1050 SSDD, or XF551</b>
<b>5 1/4" SSDD</b>	<b>AT810 or SSDD (some can also read AT1050)</b>
<b>RAM disk</b>	<b>130RAM only</b>

# USING DOS XE WITH EXISTING PROGRAMS

Every effort has been made to make DOS XE's power available to existing programs. Programs which perform DOS access through legal CIO calls should work with DOS XE. Most programs will need to be copied onto DOS XE disks. Memory conflicts should not be a problem unless the program uses the RAM under the operating system.

Pathnames and subdirectories are available to any program which allows long filenames (such as programming languages). Other programs will be able to use at least one subdirectory, the working directory.

Most programs send filenames to DOS in the following form:

**D1:FILE.EXT**

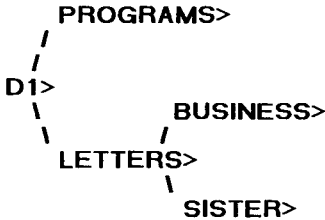
DOS XE will interpret that to mean:

**D1> *Working directory*>FILE.EXT**

Select the working directory before you run your program. Thanks to DOS XE's ability to run AUTORUN.BAT, this can be done automatically, even if the file boots without giving you access to DOS XE. (See **Batch Files** in this chapter.)

Most programs can change the working directory. One feature of DOS XE is that if you append an equal sign when you try to load a file which is actually a directory, it becomes the working directory and DOS XE returns an EOF (end of file).

If your program can digest an empty file (no data, just an EOF), then it should work. For example, if the working directory is D1> and the disk is organized as follows:



your program gets a directory listing as follows:

```
PROGRAM >
LETTERS >
```

Load "D1:LETTERS=" into your program. Your working directory is now D1>LETTERS. Your program gets a directory listing as follows:

```
BUSINESS >
SISTER >
```

Load "D1:BUSINESS=" into your program. The working directory becomes D1>LETTERS>BUSINESS.

You can move around the tree by loading the < or > symbols as a filename. The < symbol moves you one step to the left in the tree. The > symbol returns you to the root directory. The < and > symbols can be combined with a short directory name.

Load "D1:<SISTER=". The working directory changes to D1>LETTERS>SISTER

Load "D1:>PROGRAM=". The working directory changes to D1>PROGRAM

## CHANGES TO NOTE AND POINT

NOTE and POINT are CIO programming commands. This section will be of interest only to programmers.

In DOS XE, NOTE and POINT refer to a position within a file. In DOS 2.X, they refer to absolute physical locations on the disk. By making them relative to the beginning of the file, DOS XE NOTE and POINT do not change when a file is copied to another disk. Also, NOTE and POINT can now refer to a position at or beyond the end of a file.

This permits you to use POINT to append to files and to build files with holes in them. Space for the holes is not allocated on the disk until the holes are filled. So it is possible to have logical files larger than the actual disk space.

DOS 2.X uses two numbers for NOTE and POINT, a two byte **sector** number and a one byte **byte** number. DOS XE uses one three byte number for the position within a file. These three bytes are passed in the same three byte space used by DOS 2.X. Most languages will want to interpret these three bytes as the two DOS 2.X numbers.

You may have to massage the results to get the correct values. For example, Atari BASIC expects two numbers, "sector" and "byte". You can substitute high and low and proceed as follows (where OFFSET is the position within the file):

```
NOTE*3 HIGH,LOW,  
OFFSET=LOW+256*HIGH
```

or

```
HIGH=INT(OFFSET/256)  
LOW=OFFSET-(HIGH*256)  
POINT*3 HIGH,LOW
```



## CHAPTER 8

# USING DOS XE WITH ATARI BASIC

Because this section is about programming, it will not be of interest to nonprogrammers. Although it concerns Atari BASIC, much of the information will be useful to those who program in other languages.

Atari BASIC has many commands designed to be used with disk drives. All of them can be used with the Atari XF551 and DOS XE. This chapter will explore how each command works with DOS XE. Most of the commands also have other uses.

For example, the command RUN (without a pathname) will RUN the program in memory, PRINT can address any device, including the screen and a printer. In the interests of clarity, we will limit ourselves to describing the uses of the commands with DOS XE. The **Atari BASIC Reference Manual**, **Inside Atari BASIC**, and other books cover these other uses in detail. Before we discuss the commands, we will look at how Atari BASIC programs are stored on disks.

## TOKENIZED AND UNTOKENIZED PROGRAMS

As each line of a BASIC program is typed in, it is translated into a tokenized form and stored in memory. This makes the program smaller and helps it run faster. When you LIST a program, it is translated from the tokens back into words so you can read it.

BASIC programs are usually stored on disk in tokenized form. Tokenized programs are stored with the SAVE command. They are reloaded with the LOAD command. Files of tokenized BASIC programs are usually called BASIC files. They are often identified with the ".BAS" extension.

BASIC programs can also be stored in untokenized form, exactly as they appear on the screen. Files of untokenized BASIC programs are simple ATASCII text files; if you look at them with the View a File option, they look just like they do on the screen. Untokenized BASIC programs are stored with the LIST command. They are reloaded with the ENTER command.

Files of untokenized BASIC programs are usually called LISTed files. They are often identified with the .LST extension. LISTing and ENTERing programs in untokenized form is slower than SAVEing and LOADing them in tokenized form. The untokenized versions are larger and must be translated as they are LISTed or ENTERed.

## DOS XE PATHNAMES AND ATARI BASIC

Atari BASIC permits pathnames to be of unlimited length. You should have no problem with DOS XE pathnames. (See **Pathnames** in **Chapter 2** for the proper uses of the colon and the < and > symbols.)

If your program has file access routines, you should plan the user interface carefully to allow efficient use of sub-directories and pathnames. It is recommended that you provide a distinct method of changing the working directory.

## CLEARING THE SYMBOL TABLE

The tokenized form of programs maintains a table of symbols. During program development, this table often becomes cluttered with unused or deleted variable names. A significant amount of memory may be wasted on these dead names. Use the following procedure to remove the unused names.

1. **SAVE** a copy of your program. This is always a good safety procedure. (See **Save** in this chapter.)
2. **LIST** an untokenized version of the program to the disk. Use a different filename. (See **List** in this chapter.)
3. Type **NEW [Return]**. This erases the copy of the program in the computer's memory.
4. **ENTER** the untokenized version of the program. (See **Enter** in this chapter.) As each line of the program is loaded it is retokenized, just as if you typed it in. A new symbol table is built containing only the current variable names.
5. **SAVE** this new, more efficient version of your program.

## USING SAVE, LOAD, RUN, LIST, AND ENTER

### Save (S.)

Format:     **SAVE pathname**

pathname    a string with the pathname of the file to which you want to **SAVE** your program.

#### Examples:

```
SAVE "D1>PROGRAMS>BASIC>FILE.BAS"  
SAVE NAME$
```

**SAVE** stores a tokenized program on the disk. The program is stored in the directory, with the filename, specified in the pathname. **SAVEd** programs can be loaded with the **LOAD** and **RUN** commands.

Do not confuse this command with DOS XE's **SAVE MEMORY TO A BINARY FILE** command. They are not compatible.

## Load (LO.)

Format:       **LOAD pathname**

pathname      A string with the pathname of the file which contains the program you want to LOAD.

### Examples:

```
LOAD "D1>PROGRAMS>BASIC>FILE.BAS"  
LOAD NAME$
```

LOAD loads a tokenized program from disk. When a program is LOAded, the program in memory is erased. LOAD only loads programs stored with the SAVE command.

## Run

Format:       **RUN pathname**

pathname      A string with the pathname of the file which contains the program you want to RUN.

### Examples:

```
RUN "D1>PROGRAMS>BASIC>FILE.BAS"  
RUN NAME$
```

RUN loads a tokenized program from a disk and runs it. When a program on a disk is RUN, the existing program in memory (if any) is erased. RUN only loads and runs programs which were stored with the SAVE command.

RUN can be used for chaining programs. If a program is too big for memory, it can be broken into pieces. Each piece can use RUN commands to load and run the other parts. Since RUN erases the existing part with all of its data as it loads the next, each piece must be able to stand alone.

For example, a word processor might have the printing section in a separate file. The text in the word processor must be stored on a disk before the printing section is RUN. If the text is not stored, it will be lost.

## List (L.)

Format:     **LIST pathname[,start[,end]]**

pathname    A string with the pathname of the file to which you want to LIST your program.

start        An optional line number for the beginning of the block to be LISTed. If only a starting line number is given, only that line is LISTed.

end          An optional line number for the ending of the block to be LISTed

### Examples:

```
LIST "D1>PROGRAMS>BASIC>FILE.LST"  
LIST NAME$,100  
LIST NAME$,100,2000
```

LIST writes an untokenized program to disk. The program is stored in the directory, with the filename specified in the pathname. If a starting line number is included, the file is LISTed starting from that line number. If no ending line number is included, only the starting line is LISTed. If an ending line number is included, the listing will stop after that line. LISTed files can be loaded with the ENTER command.

## Enter (E.)

Format:     **ENTER pathname**

pathname    A string with the pathname of the file which contains the program you want to ENTER.

### Examples:

```
ENTER "D1>PROGRAMS>BASIC>FILE.LST"  
ENTER NAME$
```

ENTER loads an untokenized program from a disk. Unlike LOAD and RUN, it will merge with, not erase, the existing file. If there are duplicate line numbers, the incoming lines replace the existing lines.

ENTER only loads programs stored with the LIST command.

ENTER can be used for chaining programs. If a program is too big to fit in memory, it can be broken up into smaller pieces. With careful programming, a block of line numbers can be reserved for the pieces. Each piece must duplicate the line numbers of the others. A master piece with different line numbers controls the shuffling.

As each piece is brought into the program, it overwrites the piece no longer needed. All of the data are still available and the program keeps running. An example of this might be a word processor in which the printing section overwrites the text editor (and vice versa). Note that the text in the word processor is preserved as you go from one section to the other.

## INTERACTIVE BASIC COMMANDS

The commands in this section require more information from you than the commands in the previous section do. To use them properly, you need some understanding of how DOS XE and the computer talk to each other.

The Atari computer has eight communications channels, called Input/Output Control Blocks (IOCB). Each device that wants to talk to the computer must use an IOCB.

Only eight devices can have channels to the computer at one time. DOS XE needs a separate channel for each file you want to access.

In order to use NOTE, POINT, PRINT, INPUT, PUT, GET, STATUS, or XIO with a file, you must OPEN a channel to the file. When you finish, CLOSE the channel so that it can be used by other devices. SAVE, LOAD, RUN, LIST, and ENTER also use an IOCB, but BASIC takes care of it automatically.

IOCBs are memory areas set up by the computer. They store the following information:

- Device name (1 byte)
- Device number (1 byte)
- I/O command (1 byte)
- The most recent status of the device (1 byte)
- The buffer or filename address (2 bytes)
- Put byte routine address-1 (2 bytes)
- The buffer length (2 bytes)
- Auxillary control information (6 bytes).

There are eight IOCBs in the Atari, numbered 0 to 7. Atari BASIC uses them as follows:

- IOCB 0 for E: (the screen editor)
- IOCB 6 for S: (screen)
- IOCB 7 for SAVE, LOAD, RUN, LIST, ENTER, CSAVE, CLOAD, and LPRINT

You will have to choose an IOCB channel each time you want to use NOTE, POINT, INPUT, PRINT, PUT or GET. Choosing a channel is done with the OPEN command. Channels 1 through 5 can be used freely. Channel 6 can be used if you do not use any graphics commands. Channel 7 can be used if you do not use any of the commands listed for IOCB 7. Channel 0 should not be used in a BASIC program.

To use PRINT, PUT, INPUT, and GET, you must know your location in a file. DOS XE maintains a file pointer to keep track of your position. It points to the spot in the file where the next PRINT, PUT, INPUT, or GET will occur. You can find the location of the file pointer with the NOTE command and change that location with the POINT command.

# Open (O.)

Format:       **OPEN #channel,aux1,aux2,pathname**

#               Required symbol. It must always precede the channel number.

channel        The number of the IOCB you have chosen to use.

aux1           Tells the computer what operation to perform. Possible aux1 numbers for DOS XE are:

**4 Input.** Permits you only to get information from the file in the pathname. The file pointer is set to the start of the file.

**5 Input.** Like 4 above, but file pointer is set to the end of the file. Follow with a NOTE command to find the size of a file.

**6 Files listing.** Permits you to read the directories of DOS 2.x format disks or see a DOS 2.0-like directory listing of DOS XE disks.

**7 Files listing.** Permits you to read the files listing of the directory in the pathname. Shows more information than **6**.

**8 Output.** Lets you send information to the file in the pathname. The file pointer is set to the start of the file.

**Caution:** if an existing file is OPENed for output, it will be erased. If you do not want to erase it, use update instead.

**9 Append.** Lets you add information to the end of the file in the pathname. The file pointer is set to the end of the file.

**12 Update.** Permits you to get information from and send it to the file in the pathname. The file pointer is set to the start of the file.



**13 Update.** Like 12 above, but the pointer is set to the end of the file.

- aux2      Required character. Always 0 for DOS XE disk operations.
- pathname    String with the pathname of the file or directory you want to use.

**Examples:**

```
OPEN #2,12,0,"D1>PROGRAMS>BASIC>FILE.DAT"  
OPEN #N,READ,NAME$
```

OPEN chooses an IOCB and passes the necessary information to it. Most of this is done automatically. You need to provide the IOCB number, the aux1 number, and the pathname of the file you want.

## Close (CL.)

Format:      **CLOSE #channel**

#            Required symbol. It must always precede the channel number.

channel      The number of the IOCB you have chosen to CLOSE.

**Examples:**

```
CLOSE #2  
CLOSE #N
```

CLOSE releases an IOCB which has been previously OPENed. It is good programming procedure to CLOSE a channel as soon as you are through using it. CLOSEing a CLOSEd channel does not cause an error, so it is good practice to CLOSE a channel immediately before you OPEN it. The END command CLOSEs all OPEN channels.

## Note (NO.)

Format:	<b>NOTE #channel,high,low</b>
#	Required symbol. It must always precede the channel number.
channel	The number of the IOCB which you OPENed to the file
high	an arithmetic variable into which DOS XE will place the two high bytes of the location
low	an arithmetic variable into which DOS XE will place the low byte of the location

**Example:** **NOTE #2,HI,LO**

NOTE is the complement of POINT. It returns the location of the file pointer, which specifies the location in a file where the next byte will be read or written. The location is the offset from the beginning file. This is different from DOS 2.0/2.5. (See **Changes to NOTE and POINT in Chapter 7**). The location is returned in two variables. The first is the two high bytes and the second is the low byte. The actual location can be computed with the following formula:

$$\text{location}=(256*\text{high})+\text{low}$$

You will seldom use this formula. The location is generally used only with POINT, which uses the same two variables.

NOTE records the location of information within a file. As the information is written, NOTE its location. Use POINT to recover the information later.

The following program stores information from the keyboard in a disk file, D1>INFO.DAT. It NOTEs the locations and stores them in a file called D1>POINTERS.DAT. The program accepts up to 120 characters (three screen lines) and stores it when you press **[Return]**.

```

1 REM NOTE DEMO
2 REM This program reads information
3 REM from the keyboard and stores
4 REM it in the file D1>NOTE.DAT.
5 REM Locations within that file are
6 REM NOTEd and the pointers are
7 REM stored in file D1>POINTERS.DAT
8 REM
10 REM Set things up
20 DIM INFO$(120)
30 OPEN #1,8,0,"D1>NOTE.DAT"
40 OPEN #2,8,0,"D1>POINTERS.DAT"
50 REM Get info from keyboard
60 INPUT INFO$
70 REM If info is blank, then stop
80 IF LEN(INFO$)=0 THEN GOTO180
90 REM NOTE record location
100 NOTE #1,HI,LO
110 REM Store info
120 PRINT #1;INFO$
130 REM Store pointers
140 PRINT #2;HI
150 PRINT #2;LO
160 REM Do it again
170 GOTO 50
180 REM Mark end of pointers file
190 PRINT #2;255
200 PRINT #2;255
210 REM Get out gracefully
220 CLOSE #1
230 CLOSE #2
240 END

```

## Point (P.)

Format:        **POINT #channel,high,low**

- |         |  |
|---------|--|
| #       | Required symbol. Must always precede the channel number.               |
| channel | The number of the IOCB which you OPENed to the file                    |
| high    | an arithmetic expression containing the two high bytes of the location |
| low     | an arithmetic expression containing the low byte of the location       |

## Examples:

```
POINT #2,HI,LO  
POINT #3,651,23
```

POINT, the complement of NOTE, It moves the file pointer to a new location. The file pointer specifies the place in a file where the next byte will be read or written. The location is the offset from the beginning file. (See **Changes to NOTE and POINT in Chapter 7.**)

The new location is specified in two arithmetic expressions: first the two high bytes and second the low byte. The two expressions can be calculated from the actual location with the following formulas:

```
high=INT(location/256)  
low =location-(high*256)
```

Since the location specified by POINT is often identified by NOTE (which uses the same variables), you will not need the formulas in those cases. In some applications, such as files with fixed length records, you may use the formulas extensively.

The following program retrieves information stored by the sample program for NOTE. It recovers the locations stored in D1>POINTERS.DAT and uses them to get the information in D>INFO.DAT. The information is printed on screen.

```
1 REM POINT DEMO  
2 REM This program retrieves the  
3 REM information stored in the NOTE  
4 REM DEMO. It retrieves the locations  
5 REM from file D1>POINTERS.DAT and  
6 REM uses them to retrieve the info  
7 REM from file D1>NOTE.DAT  
8 REM  
10 REM Set things up  
20 DIM INFO$(120),P(100,1):I=0  
30 OPEN #1,4,0,"D1>NOTE.DAT"  
40 OPEN #2,4,0,"D1>POINTERS.DAT"  
50 REM Read in pointers  
60 INPUT #2,HI,LO  
70 REM If it's the end of file, go to next step  
80 IF HI=255 AND LO=255 THEN GOTO 140  
90 REM Put pointers in the array  
100 P(I,0)=HI:P(I,1)=LO  
110 REM Do it again
```

```

120 I=I+1
130 GOTO 50
140 REM Get number from keyboard
150 TRAP 250
160 INPUT N:N=N-1
170 REM Get pointers
180 HI=P(N,0):LO=P(N,1)
190 REM Get info
200 POINT #1,HI,LO
210 INPUT #1,INFO$
220 PRINT INFO$
230 REM Do it again
240 GOTO 140
250 REM Get out gracefully
260 CLOSE #1
270 CLOSE #2
280 END

```

The following sample program shows how to calculate the POINT values to retrieve fixed length records. It assumes the records are all 134 bytes long.

**Note:** This sample program will not run by itself. You must first create the SAMPLE.DAT file.

```

1 REM POINT DEMO 2
2 REM Program to retrieve 134 byte
3 REM fixed length records from a
4 REM file named "SAMPLE.DAT"
5 REM
10 REM Set things up
20 DIM RECORD$(134)
30 OPEN #1,4,0,"D>SAMPLE.DAT"
40 REM Get the record number
50 PRINT "Which record do you want?";
60 INPUT NUMBER
70 REM Calculate POINT values
80 OFFSET=NUMBER*134
90 HIGH=INT(OFFSET/256)
100 LOW=OFFSET-(HIGH*256)
110 REM Get the record
120 POINT #1,HIGH,LOW
130 INPUT #1,RECORD$
140 PRINT RECORD$
150 REM Get out gracefully
160 CLOSE #1
170 END

```

## Print (PR. or ?)

Format:       **PRINT #channel[[:]expression]...**

**#**            Required symbol. It must always precede the channel number.

channel       The number of the IOCB which you OPENed to the file

Separator, required if there is an expression. If a comma is used instead of the semicolon, a number of spaces (determined by the TAB value) are inserted.

expression   String or arithmetic, if no expression is given, only an [EOL] is sent. If additional expressions are given, they will be joined (concatenated).

### Examples:

```
PRINT #3,NUM  
PRINT #2;A$,B$,C$  
? #4;X,"SAMPLE";
```

PRINT is the complement of INPUT. It is used with line oriented data. Lines are terminated by an end of line character (EOL = 155, \$9B). They may be single numbers or strings.

Although there is no limit to the size of PRINT data, INPUT data are limited to 255 characters and INPUT data longer than 127 characters overwrite part of page six, possibly as much as \$600 to \$67F. It is wise to restrict the length of data which must be INPUTed later.

Multiple expressions can be stored with the same PRINT command, but they will be joined (concatenated). Multiple strings are concatenated into one string. Multiple numbers are joined end to end; for example, 5 and 26 become 526. This makes multiple numbers in a PRINT command impractical.

If the separators between expressions are semicolons, the expressions are concatenated directly. For example, abc and def become abcdef.

If the separators are commas, a number of spaces, determined by the TAB value, are inserted between the expressions. For example, abc and def would become abc     def.

If the last expression in a PRINT command does not end with an EOL, one is added. There is an exception if the last expression is followed by a semicolon or a comma. In that case the EOL is suppressed.

Use one PRINT # for each data item that is sent to a file.

```
1 REM PRINT DEMO
2 REM This program PRINTs information
3 REM to a data file named
4 REM D1>PRINT.DAT
5 REM
10 REM Set things up
20 OPEN #1,8,0,"D1>PRINT.DAT"
30 PRINT #1;"First a sentence."
40 PRINT #1;"Now a number:"
50 PRINT #1;125
60 REM CLOSE things up and stop
70 CLOSE #1
80 END
```

## Input (I.)

Format:     **INPUT #channel,variable[,variable]...**

#            Required symbol. It must always precede the channel number.

channel     The number of the IOCB which you OPENed to the file

, or ;       Required separator.

variable    String or arithmetic, it should match the type of the incoming data. Additional variables may be used to get additional data. Variables must be separated by commas.

## Examples:

```
INPUT #3,INFO$  
INPUT #2;X,Y,Z  
INPUT #4;X,INFO$
```

INPUT is the complement of PRINT. It is used with line-oriented data. Lines are terminated by an end of line character (EOL = 155, \$9B). They may be single numbers or strings. INPUT data are limited to 255 characters and INPUT data longer than 127 characters overwrite the first 128 bytes of page six (memory locations 1536-1663 \$600-\$67F).

The first variable is separated from the channel by a comma or semicolon.

You can get more than one line with an INPUT command. Use additional variables for the additional lines. Each type of variable must match the type of the incoming data. The additional variables are always separated by commas.

```
1 REM INPUT DEMO  
2 REM This program INPUTs information  
3 REM from the file named D1>PRINT.DAT  
4 REM (which was created by the PRINT  
5 REM DEMO) and PRINTs it to the screen  
6 REM  
10 REM Set things up  
20 DIM INFO1$(20),INFO2$(20)  
30 OPEN #1,4,0,"D1>PRINT.DAT"  
40 REM INPUT each piece of information  
50 INPUT #1,INFO1$  
60 INPUT #1,INFO2$  
70 INPUT #1,X  
80 REM PRINT them to the screen  
90 PRINT INFO1$  
100 PRINT INFO2$  
110 PRINT X  
120 REM CLOSE things up and stop  
130 CLOSE #1  
140 END
```



## Put (PU.)

Format:     **PUT #channel,byte**

#            Required symbol. It must always precede the channel number.

channel     The number of the IOCB which you OPENed to the file.

byte        An expression that evaluates to a single byte.

### Examples:

```
PUT #3,ASC("A")  
PUT #2,PEEK(764)
```

PUT is used to send a single byte to the disk.

```
1 REM PUT DEMO  
2 REM This program PUTs single bytes  
3 REM of information to a data file  
4 REM named D1>PUT.DAT  
5 REM  
10 REM Set things up  
20 OPEN #1,8,0,"D1>PUT.DAT"  
30 REM PUT two bytes to the file  
40 PUT #1,125  
50 PUT #1,ASC("A")  
60 REM CLOSE things up and stop  
70 CLOSE #1  
80 END
```

## Get (GE.)

Format:     **GET #channel,variable**

#            Required symbol. It must always precede the channel number.

channel     The number of the IOCB which you OPENed to the file.

variable    An arithmetic variable that receives the byte.

### Example: GET #3,X

GET is used to get a single byte from the disk.

```
1 REM GET DEMO
2 REM This program GETs single bytes
3 REM of information from the data
4 REM file named D1>PUT.DAT (which was
5 REM created by the PUT DEMO) and
6 REM displays them on the screen
7 REM
10 REM Set things up
20 OPEN #1,4,0,"D1>PUT.DAT"
30 REM GET two bytes from the file
40 GET #1,X
50 GET #1,Y
60 REM PRINT them to the screen
70 PRINT X
80 PRINT CHR$(Y)
90 REM CLOSE things up and stop
100 CLOSE #1
110 END
```

## Status (ST.)

Format:       **STATUS #channel,variable**

- #               Required symbol. It must always precede the channel number.
- channel        The number of the IOCB which you OPENed to the file.
- variable       An arithmetic variable which will receive the status number.

**Example: STATUS #2,ERROR**

The STATUS command is used to determine the condition of a file. It checks for several ways an error might occur. It checks for the following disk errors:

<b>Sector Buffer available?</b>	<b>If no, then ERROR=161</b>
<b>Legal device number?</b>	<b>If no, then ERROR=20</b>
<b>Legal pathname</b>	<b>If no, then ERROR=170</b>
<b>File on diskette?</b>	<b>If no, then ERROR=170</b>
<b>File locked</b>	<b>If yes, then ERROR=167</b>

The STATUS command also checks for the following serial bus errors:

<b>Device timeout</b>	<b>ERROR=138</b>
<b>Device not acknowledging</b>	<b>ERROR=139</b>
<b>Serial bus error</b>	<b>ERROR=140</b>
<b>Serial bus data frame overrun</b>	<b>ERROR=141</b>
<b>Serial bus checksum error</b>	<b>ERROR=142</b>
<b>Device done error</b>	<b>ERROR=144</b>

BASIC does not allow you to use a filename with the STATUS command. It is strongly recommended that you use the XIO 13 command which allows filenames instead.

## XIO (X.)

Format: **XIO command,\*channel,aux1,aux2,info**

command XIO command number.

# Required symbol. It must always precede the channel number.

channel The number of the IOCB which you OPENed to the file.

aux1 Auxiliary information number, 0 except for OPEN and INITIALIZE DISK.

aux2 Auxiliary information number, always 0 with DOS XE.

info String containing information required by the individual XIO command.

**Example: XIO 33, \*2,0,0,"D1>TEMP.BAS"**

The XIO command is an extended Input/Output statement used for special operations. It can be used when you want to perform some of the functions the would otherwise be performed using the DOS XE menu options.

COMMAND	OPERATION	FORMAT
13	STATUS	XIO 13,#Channel,0,0,pathname
32	RENAME	XIO 32,#channel,0,0,pathname,new filename (do not use path in new name, e.g., XIO 32,#2,0,0, "D>PATH>FILENAME.EXT,NEWNAME.EXT")
33	ERASE FILE	XIO 33,#channel,0,0,pathname
35	PROTECT	XIO 35,#channel,0,0,pathname
36	UNPROTECT	XIO 36,#channel,0,0,pathname
37	NOTE	XIO 37,#channel,0,0,pathname
38	POINT	XIO 38,#channel,0,0,pathname
42	NEW DIRECTORY	XIO 42,#channel,0,0,pathname
43	DELETE DIRECTORY	XIO 43,#channel,0,0,pathname
44	WORKING DIRECTORY	XIO 44,#channel,0,0,pathname
252	INITIALIZE DISK	XIO 252,#channel,0,0,drive type Drive types are the same ones available in the INITIALIZE DISK option. In a standard configuration, that would be: AT810 (Atari 810 drive) AT1050 (Atari 1050 drive) XF551 (Atari XF551 drive) 130RAM (RAMdisk on an Atari 130XE) SSDD (5.25 inch single sided, double density)
253	INITIALIZE DISK	XIO 253,#channel,drive type,0,drive number (Example: XIO 253,#2,3,0,"D2:") If drive = 0, AT810 is used. Otherwise, drive type is the position of the type in the Drive Table (DT). In a standard configuration, that would be: 1 AT810 2 AT1050 3 XF551 4 130RAM 5 SSDD
254	INITIALIZE DISK	XIO 254,#channel,0,0,drive number (Example: XIO 254,#2,0,0,"D2:") Initialize disk in current format of disk drive. Caution — uses the format that the drive thinks it is, not necessarily the same as what DOS XE thinks it is.

The following is a mini-DOS program that lets you manipulate your disk files from BASIC.

```
1 REM Mini DOS to access DOS XE
2 REM functions directly from
3 REM BASIC using XIO commands
4 REM
10 REM Set things up
20 DIM FILE$(80),NAME$(20)
30 REM Menu
40 GRAPHICS 0:PRINT :PRINT ,"Mini-DOS":PRINT
50 PRINT "(1) DIRECTORY LISTING"
60 PRINT "(2) CHANGE WORKING DIRECTORY"
70 PRINT "(3) RENAME A FILE"
80 PRINT "(4) ERASE A FILE"
90 PRINT "(5) UNPROTECT A FILE"
100 PRINT "(6) PROTECT A FILE"
110 PRINT "(7) CREATE A NEW DIRECTORY"
120 PRINT "(8) INITIALIZE A DISK"
130 PRINT "(9) DELETE A DIRECTORY"
140 PRINT :PRINT "Enter a number (1)-(9)"
150 INPUT N
160 IF N<1 OR N>9 THEN GOTO 40
170 GRAPHICS 0:PRINT :PRINT :PRINT
180 ON N GOSUB 200,300,400,500,600,700,800,900,1100
190 GOTO 40
200 REM Get a Directory Listing
210 TRAP 260
220 OPEN #1,7,0,"D:*.*)"
230 INPUT #1,FILE$
240 PRINT FILE$
250 GOTO 230
260 CLOSE #1
270 PRINT :PRINT "Press RETURN"
280 INPUT FILE$
290 RETURN
300 REM New Working Directory
310 PRINT "What is the new Pathname"
320 INPUT FILE$
330 XIO 44,#1,0,0,FILE$
340 RETURN
400 REM RENAME a file
410 PRINT "Which file do you want to RENAME"
420 PRINT "(include Pathname)"
430 INPUT FILE$
440 PRINT :PRINT "What is the new Filename"
450 PRINT "(do not include Pathname)"
460 INPUT NAME$
470 FILE$(LEN(FILE$)+1)=","
480 FILE$(LEN(FILE$)+1)=NAME$
490 XIO 32,#1,0,0,FILE$:RETURN
500 REM ERASE a file
510 PRINT "Which file do you want to ERASE"
```

```

520 PRINT "(include Pathname)"
530 INPUT FILE$
540 PRINT :PRINT :PRINT "Are you sure you want to"
550 PRINT " ERASE ";:PRINT FILE$;" (Y/N)"
560 INPUT NAME$
570 IF NAME$(1,1)<>"Y" THEN RETURN
580 XIO 33,#1,0,0,FILE$
590 RETURN
600 REM UNPROTECT a file
610 PRINT "Which file do you want to UNPROTECT"
620 PRINT "(include Pathname)"
630 INPUT FILE$
640 XIO 36,#1,0,0,FILE$
650 RETURN
700 REM PROTECT a file
710 PRINT "Which file do you want to PROTECT"
720 PRINT "(include Pathname)"
730 INPUT FILE$
740 XIO 35,#1,0,0,FILE$
750 RETURN
800 REM Create a new directory
810 PRINT "What's the name of your new directory"
820 PRINT "(include pathname)"
830 INPUT FILE$
840 XIO 42,#1,0,0,FILE$
850 RETURN
900 REM INITIALIZE a disk
910 PRINT "INITIALIZE a disk on which drive"
920 INPUT X
930 IF X<1 OR X>8 THEN GOTO 920
940 FILE$="D":FILE$(LEN(FILE$)+1)=STR$(X):FILE$(L
    EN(FILE$)+1)=":"
950 PRINT "INITIALIZE disk drive ";FILE$
960 PRINT "in which format":PRINT
970 PRINT "(1) AT810"
980 PRINT "(2) AT1050"
990 PRINT "(3) XF551"
1000 PRINT "(4) 130XE RAM disk"
1010 INPUT N
1020 IF N<1 OR N>4 THEN GOTO 990
1030 PRINT "INITIALIZING disk will erase it."
1040 PRINT "Are you sure you want to go ahead (Y/N)"
1050 INPUT NAME$
1060 IF NAME$(1,1)<>"Y" THEN RETURN
1070 XIO 253,#1,N,0,FILE$
1080 RETURN
1100 REM DELETE a directory
1110 PRINT "Which directory do you want to DELETE"
1120 PRINT "(include Pathname)"
1130 INPUT FILE$
1140 XIO 43,#1,0,0,FILE$
1150 RETURN

```

# CHAPTER 9

## STRUCTURE OF DOS XE

### DISK UTILIZATION

DOS XE supports disk drive sizes up to 16 Megabytes. Larger drives must be partitioned so logical drives are less than 16 Megabytes. Files can be up to 8 Megabytes long.

All disks are addressed in 256 byte sectors. DOS XE simulates 256 byte sectors on 810 and 1050 disk drives, which have 128 byte sectors, by reading and writing sector pairs. Up to 64K sectors can exist on a single disk (64K x 256 bytes = 16 Megabytes).

### SECTOR LABELS

DOS XE uses a file structure with built-in features to help restore damaged files. When a disk is initiated, DOS XE generates a random volume number. The number is two bytes long and the two bytes can never match.

Directory, file map, and data sectors have six byte sector labels. A label contains the data needed to reconstruct damaged files. DOS XE uses sector labels to help identify bad sectors. Sector labels are organized as follows:

<b>Byte</b>	<b>Description</b>
1-2	File ID number.
3-4	Volume number.
5-6	If byte 6=255 (\$FF), it is a directory sector and byte 5 is a sequence number* within the directory. If byte 6 is between 128 (\$80) and 254 (\$FE) then byte 5 is the sequence number* of a file map sector. If byte 6 is 127 (\$7F) or less, bytes 5 and 6 form a standard 6502 word-sized sequence number*.

\* The sequence number indicates the relative position of that sector within its series (that is, third directory sector, second file map sector, seventy-fifth data sector, and so on.).

# SECTOR ORGANIZATION

A DOS XE disk has five different types of sectors: Boot sectors, the Volume Table of Contents (VTOC), Directory, File map, and data. Each is described below.

## Boot Sectors

Sectors 1-3 contain information which allows the computer to find and load the boot program (the boot program is usually DOS XE). The boot sectors also contain a 32 byte Drive Table describing the physical and logical layout of the disk. The RAM disk does not have boot sectors.

## The Volume Table of Contents (VTOC)

The VTOC starts in sector 4 and extends across as many sectors as needed to map the disk (one sector only on all drives discussed in this manual). The first 10 bytes give information about the current status of the disk and the rest of the VTOC is a bit map of the blocks on the disk. A bit set to 0 indicates a block which is in use. A bit set to 1 indicates a block which is free.

## Directory Sectors

The first directory block immediately follows the VTOC sectors. Additional directory blocks are allocated as needed and may be scattered throughout the disk. They are linked by a two-byte pointer at the end of each block. Each directory entry contains the file name, information about the file, and up to 12 two-byte pointers which point to the file map blocks for the file. A typical XF551 directory sector would be organized like the following one.



Byte No.	No. of Bytes	Description
1	1	File type
2-9	8	Filename
10-12	3	Filename extension
13-14	2	Number of sectors in file
15	1	Number of bytes in last sector
16-19	4	First 4 bytes of sector label of this file
20-43	24	12 pointers to file map blocks
44-45	2	Creation date*
46-47	2	Last modified date*
48-49	2	Reserved
50-245	196	4 more entries, 49 blocks each
246-248	3	Unused
249-250	2	Link to next sector in directory
251-256	6	Sector label

\* Date format:

First byte, first seven bits: Year (00-99)

First byte, last bit and second byte first three bits: Month (1-12)

Second byte, last five bits: Day(1-31)

## File Map Sectors

File map sectors are two bytes long and hold pointers to the data blocks. There is a pointer for each data block in the file. A typical file map sector would be as follows:

Byte No.	No. of Bytes	Description
1-2	2	First data block pointer
3-250	248	124 more data block pointers
251-256	6	Sector label

## Data Sectors

A typical XF551 data sector would be as follows:

Byte No.	No. of Bytes	Description
1-250	250	File data
251-256	6	Sector label

# MEMORY UTILIZATION

DOS XE uses the same Zero Page memory locations and low memory area as DOS 2.0 and 2.5. It also uses RAM area "under" the OS ROM. The following memory will not change in future versions of DOS XE:

<b>Address</b>	<b>Label</b>	<b>Description</b>
\$709	MAXFILES	Maximum # of concurrently open files. Same as DOS 2.0/2.5
\$70A	DVBYT	Active drive map (1 bit per drive). Same as DOS 2.0/2.5
\$70C	BUFFERS	Address of start of system buffers. Same as DOS 2.0/2.5
\$0710	CDT	Type of drive currently in use
\$0CC0	DTT	Type numbers of all drives (0 and 9 reserved)

## MEMORY MAP OF DOS XE

**\$41- \$47 DOS XE Zero Page area**

.....  
\$700-\$1483           DOS XE core code  
\$1484-\$1CFB\*       DOS XE buffers  
\$1CFC\*               Application program area  
                      (LOMEM)  
.....

### In RAM Parallel with OS ROMS

\$C000-\$CBFF       DOS XE code  
\$CC00-\$CFFF       Copy of International  
                      Character set  
\$D800-\$DFFF       DOS XE code  
\$E000-\$E3FF       Copy of US character set  
\$E400-\$FFFF       DOS XE code

\* LOMEM value varies with number of drives and file buffers. \$1CFC is for 2 drives and 3 files, the same as DOS 2.0S.

# GLOSSARY

**Address** A location in the computer's memory. Addresses in your Atari range from 0 (\$0000) to 65,535 (\$FFFF).

**Append** To add on to the end of a file.

**Binary file** A disk file which does not require a special language (for example, BASIC) to use. Binary files include machine language programs, compiled programs, and memory images.

**Bit** Short hand for "binary digit." The smallest unit of information in a computer. Can be either "0" or "1."

**Boot** When a computer is turned on, it goes through a number of initialization procedures. This is called "booting" or "booting up."

**Buffer** An area of memory reserved for special uses.

**Byte** Eight bits. In your Atari, most operations process eight bits (one byte) at a time.

**Daisy Chain** Two or more peripherals connected to each other by Serial I/O cables.

**Data** Information of any kind.

**Decimal** Our common number system. Uses base 10 and characters 0,1,2,3,4,5,6,7,8,9.

**Default** The condition or value which exists unless you have specifically changed it.

**Delimiter** A character that marks the beginning or end of a data item, but is not part of it. For example, BASIC uses quotation marks (") as delimiters for strings.

**Density** The denseness of data stored on the disk. The XF551 supports three densities: Single density stores 90K bytes, Dual density stores 127K, and Double density stores 180K per disk side.

**Destination** The device, file, or address which receives data during a transfer. See **Source**.

**Directory** A subgroup of files on a disk.

**Disk** (Also called a diskette). A round flat plastic disk inside a square envelope. The disk is coated with a magnetic recording/playback medium. Disks are used to store information for computers.

**DOS** Abbreviation for Disk Operating System. The program that allows the use of a disk drive system.

**DOSXE.SYS** Filename reserved for DOS XE.

**DOS XE** The Atari DOS for the XF551 disk drive.

**Drive number** An integer from 1 to 8 which specifies the drive to be used.

**EOF** End Of File. The end of a given file.

**EOL** End Of Line. The marker for the end of a data string. On the Atari it is the ATASCII character 155 (\$9B).

**Extension** An extension of a filename. It is separated from the filename by a period (.) and is composed of up to three characters. The characters can be capital letters or numbers. Extensions are often used to indicate the type of file, for example, .DAT for data files, .BAS for BASIC files, and so on.

**File** A collection of data stored under one name.

**Filename** The name used to identify a file. DOS XE filenames can be up to eight characters long. Can be followed by an extension of up to three characters (see **Extension**).

**File pointer** Pointers maintained by DOS XE to keep track of where a file was last accessed. Each OPEN file has its own file pointer.

**Format** The arrangement of tracks and sectors on the disk. Single density (AT810 Format) has 40 tracks, 18 sectors per track, 128 bytes per sector. Dual density (AT1050 Format) has 40 tracks, 26 sectors per track, 128 bytes per sector. Double density (SSDD Format) has 40 tracks, 18 sectors per track, 256 bytes per sector. Double density, double sided (XF551 Format) has 80 tracks (40 per side), 18 sectors per track, 256 bytes per sector. (Format is also sometimes used interchangeably with Initialize - see **Initialize**).

**Hexadecimal** (Usually called Hex). A number system. Uses base 16 and characters 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.

**Initialize** To prepare a disk for use with the disk drive. Initializing a disk erases any old information on the disk and sets out the tracks and sectors for the proper density.

**Kilobyte** (Usually called K or Kbyte). 1024 bytes. For example, 16K is actually 16,384 bytes (16 X 1024). Note that DOS XE uses **K** to mean 1000 for file sizes and disk free space.

**Parameter** A variable in a command or a function.

**Record** A block of data delimited by EOL characters.

**Sector** The basic organizational unit for disks. In single- and dual densities a sector is 128 bytes. In double density a sector is 256 bytes.

**Source** The device, file, or address which contains the data to transfer. See **Destination**.

**String** A sequence of characters usually delimited by quotation marks ("").

**Track** A circular path around the surface of a disk. There are 40 tracks per side of an Atari disk.

**Write-Protect** To prevent a disk drive from changing the information on a disk. Atari disks are write-protected by covering a notch on the side of the disk with an opaque sticker.

# APPENDIX A

## ERROR MESSAGES

- 001 INVALID NUMBER OR DATE**  
The number or date was entered incorrectly.
- 002 INVALID LOAD FILE**  
The file you have tried to load or run is not a binary file.
- 003 INVALID ADDRESS**  
DOS XE does not recognize the address entered. Make sure you have entered it correctly (decimal or hexadecimal).
- 004 NUMBER NEEDED**  
A required number was left out.
- 128 BREAK**  
The **[Break]** key was pressed.
- 129 IOCB ALREADY OPEN**  
The I/O channel you tried to OPEN is already open. CLOSE it first, then reOPEN it.
- 130 INVALID FILE/DEVICE NAME**  
DOS XE does not recognize the file name or device name. Make sure the pathname is correct.
- 131 FILE NOT OPEN FOR READ**  
A READ operation was attempted on a file which was not OPENed for READ. CLOSE the file and OPEN it for READ (example: OPEN #1,4,0,"D:FILENAME")
- 132 ILLEGAL DEVICE HANDLER COMMAND**  
DOS XE did not recognize the command passed to it by the device handler.
- 133 DEVICE/FILE NOT OPEN**  
An operation was attempted on a file which was not OPENed. OPEN the file.
- 134 BAD IOCB NUMBER**  
The IOCB number was not in the range 0 - 7.
- 135 IOCB READ ONLY**  
A WRITE operation was attempted on a file which was not OPENed for WRITE. CLOSE the file and OPEN it for WRITE (example: OPEN #1,8,0,"D:FILENAME")
- 136 UNEXPECTED END OF FILE**  
A POINT command pointed beyond the end of a file.
- 137 TRUNCATED RECORD**  
The file ended before it was supposed to. The file may be damaged.

- 138    **HARDWARE CONNECTION PROBLEMS****  
Device timeout. DOS XE did not receive an acknowledgement from the device, usually because the device is connected improperly or turned off.
- 139    **HARDWARE CONNECTION PROBLEMS****  
Device NAK. DOS XE did not receive an acknowledgement from the device, usually because the device is connected improperly or turned off.
- 140    **SERIAL FRAME ERROR****  
Communication between the computer and other devices is garbled. You must reboot the computer. This is a very rare error. If it happens more than once, have your equipment checked for hardware problems.
- 141    **CURSOR OUT OF RANGE****  
The cursor is off the screen for the graphics mode you are in. Change the cursor location. Not a DOS XE-specific error.
- 142    **SERIAL BUS OVERRUN****  
Timing problems on the serial bus. Try the operation again. This is a very rare error. If it happens more than once, have your equipment checked for hardware problems.
- 143    **CHECKSUM ERROR****  
Communication between the computer and other devices is garbled. The checksum sent by the device does not match the one calculated by the computer. There is no standard recovery procedure because it could be either a hardware or software problem.
- 144    **DISK PROBABLY WRITE PROTECTED****  
The disk drive is unable to execute a valid command. It may be an attempt to format or write to a write protected disk. It may be an attempt to read a disk formatted in a density your drive doesn't support. It may also be caused by incorrect disk speed or an unformatted disk..
- 145    **ILLEGAL SCREEN MODE****  
The Screen Editor was OPENed with an illegal graphics mode number. Not a DOS XE-specific error.
- 146    **FUNCTION NOT IMPLEMENTED****  
The device does not support the operation requested, for example, WRITEing to the keyboard.
- 147    **INSUFFICIENT RAM****  
Not enough available memory for the graphics mode selected. Not a DOS XE-specific error.
- 160    **DRIVE NUMBER ERROR****  
The number was not between 1 and 8, or the disk drive was turned off when the computer was booted.
- 161    **NO MORE FILE BUFFERS****  
All the file transfer buffers are in use. Reconfigure your system (see **Configuring DOS XE, Chapter 2**). Also caused by a batch file longer than 511 bytes.

- 162 DISK FULL**  
No more room on the disk. Delete a file from your disk or use another disk.
- 163 UNRECOVERABLE SYSTEM I/O ERROR**  
The DOS XE disk is damaged and has loaded a bad DOS. Use another copy.
- 164 FILE NUMBER MISMATCH**  
The file is damaged. Reboot your system and try to access the file again. If it still fails, the file is lost.
- 165 INVALID FILE/DEVICE NAME**  
The pathname has illegal characters in it.
- 166 UNEXPECTED END OF FILE**  
A POINT command pointed beyond the end of a file in read mode. (Not an error when writing a file unless the point goes past the last possible legal file position--374999 for most drive types.)
- 167 CAN'T WRITE/ERASE PROTECTED FILES**  
The file is protected. UNPROTECT it before ERASEing or WRITEing to it.
- 168 INVALID COMMAND FOR DEVICE**  
You tried to copy a DOS XE file to a DOS 2.x disk.
- 170 FILE DOES NOT EXIST**  
File is not in the specified directory.
- 171 POINT INVALID**  
You tried to POINT to a file which was not OPENed for update (example: OPEN #1,12,0,"D:PATHNAME").
- 172 BAD DEVICE OR DISK TYPE**  
There was an attempt to access a disk of the wrong format, for example, a DOS 2.5 format disk.
- 173 CAN'T INITIALIZE DISK**  
The drive has no disk, the latch was not closed, or the disk has a damaged surface.
- 176 BAD DEVICE OR DISK TYPE**  
The wrong disk format type was given during the initialize process.
- 177 INVALID SUBDIRECTORY REQUEST**  
You attempted to create a new directory with the same pathname as one which already exists.
- 178 FILE NUMBER/MAP/LINK MISMATCH**  
The file is damaged. Reboot your system and try to access the file again. If it still fails, the file is lost.
- 179 INVALID SUBDIRECTORY REQUEST**  
You tried to use a directory as an ordinary file.



**180 INVALID SUBDIRECTORY REQUEST**

You tried to erase a directory with the Erase Files command.  
Use the Delete Directory command instead.

**253 SYSTEM ERROR**

Unrecoverable DOS XE system problem. For example, you tried to  
erase a damaged file. A damaged file cannot be erased.

# APPENDIX B

## XF551 DISK DRIVE

### SPECIFICATIONS

Processor:	8040/8050 at 18.3333 MHz
Disk Controller:	WD 1772
Interface:	Atari serial interface
SIO Data Rate:	
Normal:	19,040 BPS (NTSC) 18,866 BPS (PAL)
High Speed:	38,908 BPS (NTSC) 38,553 BPS (PAL)
Rotation Rate:	300 RPM
Indicator Light:	LED on when drive is accessed
Toggle Switch:	Power ON/OFF
Dip Switches:	Drive select setting (0 to 3)
Power Source:	External power adapter 9V AC, 31VA, 60 Hz (NTSC), 50 Hz (PAL)
Dimensions:	
Height:	2.75 in. (70 mm)
Width:	7.50 in. (190 mm)
Length:	11.80 in. (300 mm)
Weight:	6 lb. (2.7 kg)
Temperature:	
Operating:	50 to 95°F (10 to 35°C)
Storage:	25 to 160°F (-5 to 72°C)
Relative Humidity:	
Operational:	20 to 80%
Storage:	5 to 95%
Operating Modes:	Single, dual, or double density

	<b>Single</b>	<b>Dual</b>	<b>Double</b>
Encoding Method	FM	MFM	MFM
Number of Sides	1	1	2
Total Tracks	40	40	80
Sectors Per Track	18	26	18
Bytes Per Sector	128	128	256
Total Bytes	92,160	133,120	368,640

# CUSTOMER SUPPORT

Atari Corporation welcomes inquiries about your Atari computer products. We also provide technical assistance. Write to **Customer Relations** at the address below.

Atari user groups also provide outstanding assistance. To receive a list of Atari user groups in your area, send a self-addressed, stamped envelope to an address below.

In the United States, write to:

Atari Corporation  
Customer Relations  
P.O. Box 61657  
Sunnyvale, CA 94088

In Canada, write to:

Atari (Canada) Corp.  
90 Gough Road  
Markham, Ontario  
Canada L3R 5V5

In the United Kingdom, write to:

Atari Corp. (UK) Ltd.  
P.O. Box 555  
Slough  
Berkshire SL2 5BZ

Please indicate **User Group List**, **Technical Assistance**, or the subject of your letter on the outside of the envelope.

# INDEX

- A**
  - Allowing DOS 2.X access, 80-81
  - Appending
    - Files to files, 55-56
    - Memory to files, 65-67
  - AUTORUN.SYS, 14-18
- B**
  - BASIC, exit to, 33
  - BASIC files, 95-96
  - Batch files, 88-89
    - Creating batch files, 88-89
    - Running batch files, 74-75
  - Binary files, 89-91
    - File structure, 89-91
    - Loading, 61-62
    - Running, 60-61
    - Saving memory to, 62-64
  - Boot errors, 12-13
  - Boot sectors, 118
- C**
  - Care of disks, 8-9, 26
  - Cartridge, exit to, 33
  - Changing memory, 70-71
  - CLOSE, 103
  - Command line entry, 83-87
  - Compatibility
    - with Atari DOS 2.0/2.5, 80-81
    - with existing programs, 92-94
  - Configuring DOS XE, 13-19
  - Connecting the drive, 5-6
  - Copying
    - Directories, 52-53
    - Disks, 26-27, 77-80
    - Files, 28, 47-54
    - Groups of files, 53
    - To and from devices, 53
  - COPY3\_XE.COM, 19
  - Customer support, 131
- D**
  - Data sectors, 119
  - Date
    - Creation date, 37
    - Revision date, 37
    - Setting the date, 75-76
  - Decimal, 62, 64
  - Deleting directories, 28, 47
  - Device name, 21, 38, 54
  - Directories, 19-25
    - Copying directories, 52-53
    - Creating new directories, 46
    - Deleting directories, 28, 47
    - Listing Directories, 36-39
    - Naming directories, 20-21, 46
    - Protecting directories, 29, 39-40
    - Root directory, 39
    - Structure of directory, 20
  - Directory sectors, 118-119
  - Disk drives
    - XF551, 1, 5-6, 129
    - Other disk drives, 15-17, 57, 92
    - Using multiple drives, 6-8, 16-17, 52
  - Displaying memory, 68-69
  - Disks, 1
    - Care of disks, 8-9, 26
    - Duplicating disks, 26-27, 77-80
    - Initializing disks, 27-28, 56-58, 76
    - Utilization of disks, 117
  - DOS 2.5/2.0 compatibility, 80-81
  - DOS 3, 19
  - DOSXE.SYS, 13, 19, 76
  - Drive Select switches, 7-8, 13
  - Duplicating
    - Disks, 26-27, 77-80
    - Files, 28, 47-54
- E**
  - ENTER, 99-100
  - EOL, 110
  - Erasing
    - Files, 28-29, 41
  - Error messages, 25-26, 84, 125
  - Extensions, 20-21, 22

## F

### Files, 19-25

- Append to a file, 55-56
- Binary files, 89-91
- Copying files, 28, 47-54
- Creating files, 39
- Erasing files, 28-29
- Protecting files, 29, 39-40
- Listing files, 36-39, 60, 74
- Naming files, 20-21, 39, 51
- Renaming files, 42-43
- Structure of files, 117
- Unprotecting files, 40
- View text files, 44

### File Map sectors, 119

### Filenames, 20-21, 22, 23-24, 51

- Extensions, 20-21, 22

### Formats

- Single density/single sided, 92
- Dual density/single sided, 92
- Double density/single sided, 92
- Double density/double sided, 92

## G

### GET, 111-112

### Glossary of terms, 121-123

### Going to machine language programs, 71-72

## H

### Hexadecimal, 62, 63-64

## I

### Initialize address, 63

### Initializing disks, 27-28, 56-58, 76

### INPUT, 109-110

### Interactive BASIC commands, 100

### IOCB, 100-101

## K

### Kilobyte, 37

## L

### LIST, 99

### LOAD, 98

### Loading binary files, 61-62

### Loading DOS XE, 11-13

## M

### Memory

#### Appending memory to

- binary files, 65-67

#### Changing memory contents, 70-71

#### Displaying memory contents, 68-69

#### Saving memory to binary files, 62-64

#### Utilization, 120

### Memory map, 120

### Menus

#### File Access menu, 32, 35-58

#### Machine Language Access menu, 32, 59-72

#### Main menu, 31-33

#### Systems Functions menu, 32, 73-81

## N

### New directory, 46

### NOTE, 94, 104-105

## O

### OPEN, 102-103

## P

### Pathnames, 21-24, 45, 96

### POINT, 94, 105-107

### PRINT, 108-109

### Printing files, 54

### Protecting files and directories, 29, 39-40

### Programs

- Sample BASIC programs, 105, 106, 107, 109, 110, 111, 112, 115-116

#### Using DOS XE

- with Existing Programs, 92-94

### PUT, 111

## R

### RAM disks, 15-18, 57

### RDRIVER.SYS, 18

### Renaming files, 42-43

### Root directory, 39

### RUN, 98

### Run address, 63

### Running batch files, 74-75

### Running binary files, 60-61

## **S**

SAVE, 97  
Saving memory to a binary file, 62-64  
Screen editor, 54  
Sector, 117-119  
Setting boot up conditions, 18  
SETUP.COM, 14-18  
STATUS, 112-113  
Symbol table, 96, 97

## **T**

Tokenized files, 95-96

## **U**

Unprotecting files, 40  
Untokenized files, 95-96

## **V**

Viewing text files, 44  
Volume Table of Contents, 118

## **W**

Wildcards, 25, 29, 36, 41, 43  
Working directories, 22-24, 38, 45,  
60, 74  
Write Verify mode, 15-16  
Writing DOS XE to disks, 76

## **X**

XIO, 113-116

# Notes

---

# Notes

---





Copyright © 1988, Atari Corporation.  
Sunnyvale, CA 94086  
All rights reserved.

Printed in USA.

C300557-001 Rev. A