Atari Dos 4 (aka ANTIC Dos aka QDOS)#

Table of Contents

Disk#

- Atari DOS 4/DOS40-SD.ATR

Documentation#

Disk Operating System 4.0 On-Line Help System

From ANTIC - The Atari Resource

HOW TO USE THE HELP SYSTEM#

The information in the help system is organized into "screens". To move from screen to screen, you press the space bar, the letter keys A-Z, the RETURN key, and the ESC key. Most screens contain text for you to read. Press the space bar to move from one text screen to the next. When you reach the last text screen in a sequence, press the space bar to return to the previous menu. Some screens contain menus of topics. To select a topic, press one one of the letter keys A-Z. At any time, you may:

- Press the RETURN key to go back to the previous menu.
- Press the ESC key to go back to the main menu.
- Press the BREAK key to exit from the help system and return to DOS.

GENERAL INFORMATION ON DOS 4#

The DOS 4 Disk Operating System is a program which controls disk drives on Atari computers. DOS 4 can be configured to run a variety of different disk drives, single-density or double-density, and single-sided or double-sided. DOS 4 consists of three separate parts:

1. The File Management System (FMS)
2. The Command Processor (CP)
3. The Disk Utility Package (DUP)

The FMS remains in computer memory all the time, while the CP and DUP remain on the disk and are loaded into memory only when needed. The File Management System maintains the file structure on the disk. It performs disk input/output (I/O) commands like OPEN, CLOSE, INPUT, and PRINT. The DOS 4 FMS supports all the commands accepted by Atari DOS 2.0. Virtually all software written for DOS 2.0 or DOS 3 will also work on DOS 4.

The Command Processor displays a menu of all the programs on a disk and runs any of them with a single keystroke. The CP also lets you examine the disk directory (a list of all the files on the disk) or run the cartridge (if a cartridge is installed). The Disk Utility Package performs a wide variety of manipulations on disk files, such as copying, locking and unlocking, renaming, and deleting. Among the most useful commands in the DUP are those which copy files from one disk to another or duplicate an entire disk even if you have only one disk drive.

DISK DRIVE NUMBERS#

There are three different types of disk drive numbers:

1. Physical drive numbers
2. Logical drive numbers
3. Unit numbers

Physical drive numbers correspond to the actual pieces of hardware. They range from 1 to 8. Physical drive numbers are assigned first to disk drives on the parallel bus and second to disk drives on the serial bus. For example, if you have two parallel bus drives and three serial bus drives then the parallel bus drives are 1 and 2 and the serial bus drives are 3, 4, and 5. Logical drive numbers are numbers which appear in the names of disk files. There are ten different logical drive numbers, ranging from D0: to D9:. Any time the computer asks you to specify a disk drive, give the logical number of the drive. Normally, logical drive numbers D1: through D8: refer to physical drives 1 through 8 respectively, while logical drive numbers D0: and D9: are not supported.

You can change the assignment of logical drive numbers to physical drive numbers by using the REDIRECT utility program. Unit numbers are numbers that the computer uses to communicate with disk drives. They range from P1 to P8 for drives on the parallel bus, and from S1 to S8 for drives on the serial bus. On each bus, unit numbers are assigned beginning with # For example, if you have two parallel bus drives and three serial bus drives, the parallel bus drives are units P1 and P2, and the serial bus drives are units S1, S2, and S3. Most serial bus drives have switches you must set to indicate the unit number.

Parallel bus drives determine their own unit numbers automatically.

FILE SPECIFICATIONS AND WILDCARDS#

The data on a disk is organized into files. Each file has a name. To access a disk file, you must give its name so that DOS 4 will know which file you are referring to. The name of a disk file is always in the form of a file specification (or "filespec" for short). A filespec consists of three parts:

1. A device specification
2. A primary file name
3. An optional extender

The device specification indicates on which disk drive the file is locat. It consists of the uppercase letter "D", followed by a digit (0-9), followed by a colon.

Normally, device specifications "D1:" through "D8:" refer to disk drives 1 through 8, while "D0:" and "D9:" are not supported.

The digit in the device specification is optional; if it is omitted (as in "D:") then 1 is assumed. The primary file name consists of one to eight characters. Each character must be either an uppercase letter (A-Z), a lowercase letter (a-z), or a digit (0-9). The first character must be a letter. Lowercase letters are automatically converted to uppercase by DOS 4.

The extender, which is optional, consists of a period followed by zero to three characters. Each character must be either a letter or a digit; lowercase letters are automatically converted to uppercase.

Examples of valid filespecs are "D:MYPROG.BAS", "D:Chapter4.Txt", "D2:paper.7", and "D1:r1245".

WILDCARDS#

There are two wildcard characters which may be used within file names (both the primary and the extender): the question mark "?" and the star "*".

The question mark "?" matches any character (including a blank).

Example, "D:WH?" matches "D:WHO" and "D:WHY" and "D:WH", but not "D:WHEN".

The star "*" is equivalent to padding the rest of the field (primary or extender) with question marks.

Example, "D:WH*" will match "D:WHO" and "D:WHEN" but not "D:WAVE".

filespec "D:*.BAS" will match all file names with extender ".BAS", and "D:*.*" will match anything.

If a wildcard is used in a filespec in an OPEN or STATUS command, the first file in the disk directory which matches the filespec is used. In a RENAME, DELETE, LOCK, or UNLOCK command, the indicated operation is done on all files that match.

THE DOS 4 FILE MANAGEMENT SYSTEM#

```
C. The CLOSE command
D. The GET and INPUT commands
E. The PUT and PRINT commands
F. The STATUS command
G. The RENAME command
H. The DELETE command
I. The LOCK command
J. The UNLOCK command
K. The POINT command
L. The NOTE command
M. The FORMAT command
N. AUTORUN.SYS files
```

Input/output control blocks (IOCBs)#

An input/output control block (IOCB) is a block of memory used to control an input/output operation. Every I/O operation in progress must have an IOCB associated with it. There are eight IOCBs, numbered 0 through 7. Atari BASIC uses IOCB 0, so in BASIC programs you may use only IOCBs 1 through 7. In assembly language programs, you may use all eight IOCBs.

The OPEN command#

The OPEN command establishes a line of communication to a disk file. must OPEN a disk file before you can read data from it or write data to it. In BASIC, the command takes the form `OPEN #iocb,aux1,aux2,filespec`. Here "iocb" is a number from 1 to 7 which indicates the IOCB to be used for this file, and "filespec" is a string which contains the file specification.

"Aux1" is a code number with the following meanings. Select the desired topic for further information:

value	IO direction	
4	INPUT	
9	APPEND	
5	BAD FILE INPUT	
12	UPDATE	
6	DIRECTORY READ	
14	VERIFY	
8	OUTPUT	

The OPEN INPUT command takes the form `OPEN #iocb,4,0,filespec`. This command establishes a line of communication to the specified file (which must already exist) and allows you to read data from the file, beginning with the first byte of the file. A file open for input may be accessed with the GET, INPUT, NOTE, and POINT commands.

Open Bad File Input#

The OPEN BAD FILE INPUT command takes the form `OPEN #iocb,5,fill,filespec` where "fill" is the numerical code of a character to be used to fill in bad sectors of the file. OPEN BAD FILE INPUT is the same as OPEN INPUT except that handling of errors is different. If a bad sector is encountered while reading a file open for input then DOS 4 issues an error message and freezes the file. If a bad sector is encountered while reading a file open for bad file input then DOS 4 replaces it with a sector of fill characters, does not issue an error message, and continues reading the file. The OPEN BAD FILE INPUT command enables you to recover all the readable portions of a file which contains bad sectors.

A file open for bad file input may be accessed by the GET, INPUT, and NOTE commands.

Open Directory Read#

The OPEN DIRECTORY READ command takes the form `OPEN #iocb,6,0,filespec`. This command allows you to read the disk directory using the INPUT and GET commands. For each directory entry which matches the given filespec, DOS 4 returns an eighteen-character record of the following form:

1. A star if the file is locked, or a blank if the file is not locked.
2. A blank.
3. Eight characters giving the primary file name, left justified and padded with blanks.
4. Three characters giving the extender, left justified and padded with blanks.
5. A blank.
6. Three digits giving the number of sectors allocated to the file.
7. A carriage return.

Following the last such record, DOS 4 sends this 17-character record:

1. Three digits giving the number of free sectors on the disk.
2. The phrase " FREE SECTORS".
3. A carriage return.

If the disk has more than 999 sectors, then the numbers appearing in the directory are a fraction of the true sector counts. For Atari 1050 drives in dual-density mode and for Percom-compatible drives in double-sided double-density mode, the numbers in the directory are half the true sector counts.

Open Output

The OPEN OUTPUT command takes the form `OPEN #iocb,8,0,filespec`. This command establishes a line of communication to the specified file and allows you to write data to the file, beginning with the first byte of the file. If the file does not already exist then it is created; if the file does exist then it is erased (provided that it is not locked). A file open for output may be accessed by the PUT, PRINT, and NOTE commands.

Open Append#

The OPEN APPEND command takes the form `OPEN #iocb,9,0,filespec`. This command establishes a line of communication to the specified file (which must already exist and be unlocked) and allows you to write data to the file. Data written to the file is added on to the end of the file. A file open for append may be accessed by the PUT, PRINT, and NOTE commands.

Open Update#

The OPEN UPDATE command takes the form `OPEN #iocb,12,0,filespec`. This command establishes a line of communication to the specified file (which must already exist and be unlocked) and allows you to both read data from the file and write data to the file, beginning with the first byte of the file. Data written to the file overwrites the previous contents of the file. A file open for update may not be extended. A file open for update may be accessed by the GET, INPUT, PUT, PRINT, POINT, and NOTE commands.

Open Verify#

The OPEN VERIFY command takes the form `OPEN #iocb,14,0,filespec`. This command establishes a line of communication to the specified file (which must already exist) and allows you to both read data from the file and write data to the file, beginning with the first byte of the file. However, data written to the file is not sent to the disk. Instead, DOS 4 reads from the disk the data already in the file and compares it to the data you write; if the two do not agree, DOS 4 returns ERROR 174. Contents of the file is unchanged. A file open for verify may be accessed by the GET, INPUT, PUT, PRINT, POINT, and NOTE commands.

Close#

The CLOSE command takes the form `CLOSE #iocb`. This command causes DOS 4 to write out to the file any data stored in its internal buffers, update the disk directory, release all internal resources used to support the file, and free the IOCB. A file opened for output will not appear in the directory until it has been closed. Failure to close a file opened for output causes all data written to the file to be lost (however, sectors allocated to the file are not lost; they remain available for use by other files). If a file opened for append is not closed, the appended data is lost. The original contents of the file remains intact. If a file opened for update is not closed, one sector of data may be lost.

Failure to close a file opened for input, bad file input, or verify will not cause any harm.

Get

The GET command reads one byte from the disk file. In BASIC, this command takes the form `GET #iocb,var` which places into the arithmetic variable "var" the numerical value of the byte.

Input

The INPUT command reads one or more records from the disk file. It takes the form `INPUT #iocb,var_1,...,var_n` where each "var" is either an arithmetic variable or a string variable.

Put#

The PUT command writes one byte of data to the disk file. In BASIC it takes the form `PUT #iocb,exp` where "exp" is an arithmetic expression which gives the numerical value of the byte to be written.

Print#

The PRINT command writes a record to the disk file. It takes the form `PRINT #iocb;exp_1;...;exp_n` where each "exp" is either an arithmetic or a string expression. After "exp_1" through "exp_n" are written, a carriage return (which indicates end-of-record) is written; the carriage return can be suppressed by placing a semicolon after exp_n.

Status

The STATUS command takes two different forms. Which form you use depends on whether the file whose status you want to check is open or closed. If the file is open, use the form `STATUS #iocb,var`. This stores a status code into the arithmetic variable "var". If the file has been frozen due to an error, "var" contains the error code. If the file has not been frozen, "var" contains the value # If the file is closed, use the form `XIO 13,#iocb,0,0,filespec`.

Rename#

In BASIC, the RENAME command takes the form `XIO 32,#iocb,0,0,filespec`. This command changes the name of a disk file. In the RENAME command, the filespec has a special form: a device specification, followed by the old filename and extender, followed by a comma, followed by the new filename and extender. For example, use `"D:BLUE.BAS,GREEN.BAS"` to change the name of file BLUE.BAS to GREEN.BAS. If wildcards are used in the filespec, all files which match the filespec are renamed. When you use the RENAME command, be careful not to create two or more disk files with the same name.

Delete#

In BASIC, the DELETE command takes the form `XIO 33,#iocb,0,0,filespec`. The DELETE command deletes a disk file. Once a file is deleted, there is no way to get it back. If wildcards are used in the filespec, all files which match the filespec are deleted.

Lock

In BASIC, the LOCK command takes the form `XIO 35,#iocb,0,0,filespec`. The LOCK command locks a disk file. Locked disk file cannot be overwritten or deleted. If wildcards are used in the filespec, all files which match the filespec are locked.

Unlock#

In BASIC, the UNLOCK command takes the form `XIO 36,#iocb,0,0,filespec`. The UNLOCK command unlocks a disk file. If wildcards are used in the filespec, all files which match the filespec are unlocked.

Point and Note#

The POINT command allows you to perform random access by skipping from place to place within a file without reading all the data in between. In Atari BASIC, the POINT command should be given as the following three-line sequence:

```
Q=INT(P/256)
R=P-Q*256
POINT #iocb,Q,R
```

This tells DOS 4 that the next byte to be read or written is the byte in position P. If P was 0, the next byte read or written will be the first byte of the file; if P was 1, the next byte read or written will be the second byte of the file; and so on.

The NOTE command tells the position within the file of the next byte to be read or written. In Atari BASIC, the NOTE command should be given as the following twoline sequence:

```
NOTE #iocb,Q,R
P=Q*256+R
```

This gives the variable P a value equal to the position within the file of the next byte to be read or written. If the next byte to be read or written is the first byte of the file, P is given the value 0; if the next byte to be read or written is the second byte of the file, P is given the value 1; and so on.

Format#

In BASIC, the FORMAT command takes the form `XIO 254,#iocb,0,0,devspec` where "devspec" is a device specification ("D0:" through "D9:"). The FORMAT command prepares a blank disk to receive data. A new disk must be formatted before it can be used to store disk files. Warning: Formatting a disk erases all the files on the disk, regardless of whether or not they are locked!

Boot#

When the computer is turned on, the DOS 4 File Management System automatically loads in from the disk. Once it is loaded, it runs the program in file CONFIG.SYS. CONFIG.SYS configures all the disk drives and then checks disk drive #1 to see if there is a file named D1:AUTORUN.SYS. so, then the program contained in AUTORUN.SYS is automatically loaded and run.

The contents of the AUTORUN.SYS file must be a machine-language program in the standard binary load format.

THE DOS 4 COMMAND PROCESSOR#

The DOS 4 Command Processor is a non-resident program from which you select a program to run, run a cartridge, or examine the disk directory.

If there is no cartridge installed, the Command Processor loads automatically when the computer is turned on. From within BASIC, you load the Command Processor by typing DOS. This erases any BASIC program in memory, so be sure to save your program to disk before typing DOS.

To load the Command Processor from within other software, give the command equivalent to the BASIC "DOS" command, usually something like "EXIT" or "RETURN TO DOS". Before you load the Command Processor, insert a disk that contains the Command Processor program into disk drive #1 (if using the disk drive number redirection feature of DOS 4 then the CP program must go into logical drive #1, not physical drive #1). If you attempt to load the Command Processor without first inserting such a disk, the computer will "freeze". If this happens, press BREAK and another attempt will be made to load the Command Processor. The CP menu is constructed by reading the directory of disk drive #1 and listing all files whose names end in the extender ".COM". (If disk drive number redirection is being used, the CP reads the directory of logical drive #1, not physical drive ##) maximum of sixteen such files can be listed.

After all ".COM" files are listed, the CP lists the RUN CARTRIDGE and DISK DIRECTORY items and prompts you to make a selection. You can add new items to the CP menu by creating disk files whose names end in ".COM". Each such file must be a machine-language program in standard binary load format. To select an item from the CP menu, press the appropriate key:

- A letter key from A to P runs the program listed on the menu next to that letter.
- The Z key runs the cartridge installed in the computer.
- A digit key from 0 to 9 displays the directory of the corresponding disk drive.
- The RETURN key reconstructs the CP menu.

A menu selection can be made while the CP is loading.

THE DOS 4 DISK UTILITY PACKAGE#

Commands selected using letter keys:

```
A. Com Processor    I. Format Data Disk
B. Run Cartridge    J. Duplicate Disk
C. Copy File        K. Binary Save
D. Delete File      L. Binary Load
E. Rename File      M. Run At Address
F. Lock File        N. Configure Drive
G. Unlock File      O. Duplicate File
H. Write DOS        P. Merge DCF
            Q. Identify Mode
```

R. The Directory commands (0-9) S. Loading the Disk Utility Package T. How to make a menu selection U. How to respond to prompts V. How to make bootable disks

The Disk Utility Package is loaded from the DOS 4 Command Processor menu. Press the letter key listed next to "DISKUTIL" (this is usually A). To make a menu selection, type one of the letters A-Q or one of the digits 0-9, and then press RETURN. Note that only uppercase letters are recognized as valid selections. Menu selections can be made whenever the prompt "SELECT ITEM OR FOR MENU" appears. Press RETURN to re-display the Disk Utility Package menu. Most commands require some additional information, such as the name of a file.

The DUP requests the information necessary. You may give the name of a file with or without a device specification. Device "D1:" is assumed if the device specification is omitted. To select the DIRECTORY command, press the digit key 0-9 corresponding to the number of the drive whose directory you want to examine. DUP responds by asking for a search specification. Press RETURN to list all the files on the selected disk drive. Alternatively, enter a search specification (do not include a device specification) and only files whose names match the search specification are listed. If the disk has more than 999 sectors, then the numbers appearing in the directory are a fraction of the true sector counts. For Atari 1050 drives in dual-density mode and for Percom-compatible drives in double-sided double-density mode, the numbers in the directory are half the true sector counts. The directory is always displayed on the screen. There is no provision for sending the directory to a disk file or printer.

The A. COM PROCESSOR command returns you to the DOS 4 Command Processor.

The B. RUN CARTRIDGE command runs the cartridge installed in the computer.

The C. COPY FILE command copies the contents of one file into another file. The DUP asks for the source file and the destination file. Type the name of the source file, followed by a comma, followed by the name of the destination file. Add "/A" at the end of the name of the destination file to append the contents of the source file onto the end of the destination file. Wildcards may be used in the file names, but only the first file matching the filespec is copied. To copy several files at once, use the DUPLICATE FILE command. Specify "P:" as the destination file to have the source file printed by the printer. Specify "E:" as the destination file to have the source file displayed on the screen. Specify "E:" as the source file to type in a file. You may use the cursor control keys normally while doing so; data is entered only when you press RETURN. Press CTRL 3 when you have finished typing.

The D. DELETE FILE command deletes a disk file. If wildcards are used in the file name, all files matching the filespec are deleted. Before deleting a file, the DUP asks for permission. Type "Y" if you want to delete the file, or "N" if you don't want to delete the file. If you add "/N" at the end of the file name, the DUP automatically deletes all matching files without asking permission. Once a file has been deleted, there is no way to get it back.

The E. RENAME FILE command changes the name of a disk file. The DUP asks for the old and new names. Type the current name of the file, followed by a comma, followed by the desired new name. You can include a device specification in the current name, but not in the new name. If wildcards are used in the file name, all files whose names match the filespec are renamed.

The F. LOCK FILE command locks a disk file. A locked disk file cannot be overwritten or deleted. If wildcards are used in the file name, all files whose names match the filespec are locked.

The G. UNLOCK FILE command unlocks a disk file. If wildcards are used in the file name, all files whose names match the filespec are unlocked.

The H. WRITE DOS command writes the three disk files which make up DOS 4 - QDOS.SYS (containing the File Management System), QDUP.SYS containing the Command Processor), and DISKUTIL.COM (containing the Disk Utility Package). This command also writes a file called CONFIG.SYS which automatically configures all the disk drives when you turn on the computer. On disk drives which require an invisible boot file, this command writes the boot file so that the resulting disk is bootable. In order for a DOS 4 diskette to be bootable, it is necessary that the file manager (QDOS.SYS) be the first file written onto the diskette. This means that the best time to give the WRITE DOS command is immediately after the diskette has been formatted. Note that if you WRITE DOS to a drive other than drive #1, the configuration of the drive to which you write DOS must be the same as the configuration of drive ##

The I. FORMAT DATA DISK command formats a blank disk. You must format a disk before you can use it to store disk files. Warning: Formatting a disk erases all the files on the disk, regardless of whether or not they are locked!

The J. DUPLICATE DISK command makes a copy of an entire disk. It works with one or more disk drives. command can be used to copy the entire contents of a disk from one kind of disk drive to another (say, from a single-density drive to a double-density drive or vice versa, or from a drive made by one manufacturer to a drive made by a different manufacturer). The DUP asks for the numbers of the source drive and of the destination drive (which default to "1,1" if you press RETURN). If you specify two different drives, the DUP asks you to insert both disks and then the duplicate disk operation will proceed. If you specify the same drive as both source and destination, the DUP informs you that swapping is required and asks how many copies you want to make. Respond with a number from 1 to (The number defaults to 1 if you press RETURN.) When you specify the number of copies, you may place the option "/R" after the number (for example, you may type "3/R"). The /R option specifies that the drive is to be reconfigured each time disks are swapped. This makes it possible to move files between disks of different types (e.g., between single- and double-density disks) even if you have only one disk drive. If you choose the /R option, the DUP asks you to specify the drive configuration to be used for the source disk and the configuration to be used for the destination disk(s). Each time, DUP displays a menu of configurations to choose from. Next, the DUP asks you to insert the source disk. When you have done so, it reads in as much of the source disk as will fit into memory and then asks you insert the destination disk. Once you've done that, the data in memory is written out. If you requested more than one copy, the DUP asks you to insert the second destination disk and again the data in memory is written out. The DUP continues to ask for destination disks until it has written onto each of them. At this point, if the entire contents of the source disk has been written then the operation ends. If not, the DUP asks you to reinsert the source disk, and the above steps are repeated until the entire contents of the source disk has been written onto each destination disk. Whenever the destination drive of a DUPLICATE DISK command requires an invisible boot file, the DUP writes the boot file onto the destination disk(s). Thus, if you use DUPLICATE DISK to duplicate a bootable disk, the resulting duplicates are also bootable.

The K. BINARY SAVE command saves a block of memory into a disk file in the standard binary load format. Use this command, give the name of the file and the addresses of the first and last bytes of the block of memory to be saved. The addresses must be in hexadecimal. In addition, you may

specify the address at which the program is to be run and the address of an initialization routine in the program. If you place "/A" at the end of the file name, the saved data is appended onto the end of the file.

The L. BINARY LOAD command loads a disk file into memory and runs it. The file must be in the standard binary load format. If you place "/N" at the end of the file name, the file is loaded and initialized but not run.

The M. RUN AT ADDRESS command runs a program beginning at an address you specify. The address must be given in hexadecimal.

The O. DUPLICATE FILE command copies one or more files from one disk to another. It works with one or more disk drives. The command can be used to copy files from one kind of disk drive to another (say, from a single-density drive to a double-density drive or vice versa, or from a drive made by one manufacturer to a drive made by a different manufacturer). When you select this item, the DUP asks for the numbers of the source drive and of the destination drive (which default to "1,1" if you press RETURN). What happens next depends on whether or not you specify the same drive as both source and destination. If you specify two different drives then the DUP asks you for the names of the files to be duplicated. Respond by entering the names of one or more disk files, separated by commas (do not include device specifications). If wildcards are used in the file names, all files which match the given filespecs are copied. There is, however, one exception to the wildcard rules: an extender which contains a wildcard will not match an extender of ".SYS" or ".COM". For example, "*.*" refers to all files except those ending in ".SYS" or ".COM", while "*.COM" refers to all files whose names end in ".COM". When you have done this, the duplicate operation will proceed. If you specify the same drive as both source and destination, the DUP informs you that swapping is required and asks how many copies you want to make. Respond with a number from 1 to (The number defaults to 1 if you press RETURN.) When you specify the number of copies, you may place the option "/R" after the number (for example, you may type "3/R"). The /R option specifies that the drive is to be reconfigured each time disks are swapped. This makes it possible to move files between disks of different types (e.g., between single- and double-density disks) even if you have only one disk drive. If you choose the /R option, the DUP asks you to specify the drive configuration to be used for the source disk and the configuration to be used for the destination disk(s). Each time, DUP displays a menu of configurations to choose from. Next, the DUP asks for the names of the files to be copied. Respond by entering one or more file names, separated by commas, as previously described. The DUP then asks you to insert the source disk. When you have done so, it reads in as many of the specified source files as will fit into memory and then asks you insert the destination disk. Once you've done that, the data in memory is written out. If you requested more than one copy, the DUP asks you to insert the second destination disk and again the data in memory is written out. The DUP continues to ask for destination disks until it has written onto each of them. At this point, if all of the specified source disk files have been written then the operation ends. If not, the DUP asks you to reinsert the source disk, and the above steps are repeated until all of the specified files have been written onto each destination disk.

The N. CONFIGURE DRIVE command does two things: it tells DOS 4 what kind of disk drive is in your system, and it tells the disk drive how it should configure itself (e.g., single-density mode or double-density mode). When you select this item, the DUP asks you which drive you want to configure. Respond by typing a number from 0 to 9 (if you are using the disk drive number redirection feature of DOS 4, this is a logical drive number, not a physical drive number). After you have specified the drive number, the DUP displays a menu of disk drive configurations, with a star placed next to the drive's current configuration (actually, the star indicates what DOS 4 thinks the current configuration is; the DUP does not interrogate the drive to find out what its actual configuration is). Respond by typing the letter listed next to the configuration that you want the drive to become. If you press RETURN without typing a letter, the DUP assumes that you want the drive to remain in its current configuration. For an Atari 810 disk drive, you must select configuration A. For an Atari 1050-compatible drive, you may select either configuration A for single-density operation, or configuration

B for dual-density operation. For a Percom-compatible drive, you may select configuration C for single-sided single-density operation, or configuration D for single-sided double-density oparation. If the drive is double-sided, you may also select configuration E for double-sided double-density operation. Note that configurations A and C are not the same. You must select the correct one for your drive.

The P. MERGE DCF command adds new disk drives to the menu of disk drive configurations. This menu is used by the CONFIGURE DRIVE, IDENTIFY MODE, DUPLICATE DISK, and DUPLICATE FILE commands. When you select the MERGE DCF command, the DUP asks for the name of a "Disk Configuration File" (DCF). Type the name of the DCF for the disk drive model that you want to add to the menu. After you have merged a DCF into the DUP, use the WRITE DOS command to write out the new DUP.

The Q. IDENTIFY MODE command tells you the mode in which a disk was formatted (e.g., it tells you whether a disk was formatted in single-density mode or in double-density mode). When you select this item, the DUP asks you which disk drive contains the disk. Respond by typing a number from 0 to 9 (if you are using the disk drive number redirection feature of DOS 4, this is a logical drive number, not a physical drive number). After you have specified the drive number, the DUP attempts to identify the mode in which the disk was formatted. If the DUP successfully identifies the mode, it configures the drive into the correct mode and displays the menu of disk drive configurations with a star placed next to the correct mode. If the DUP is unsuccessful in identifying the mode, it leaves the disk drive configuration unchanged and displays the message "UNABLE TO IDENTIFY MODE". A disk is "bootable" if you can load in DOS 4 from it when the computer is turned on. In order to use the disk drives, you must place a bootable disk in drive #1 before turning on the computer. In order for a disk to be bootable, QDOS.SYS must be the first file written onto it. Also, the disk drive configuration program CONFIG.SYS must be on the disk. In addition, some disk drives require that a special "boot file" be present on the disk. The boot file is not listed in the directory and is invisible to DOS 4. The DUP automatically writes the boot file on the destination disk whenever you select the WRITE DOS command or the DUPLICATE DISK command.

ADVANCED INFORMATION#

INTRODUCTION#

The advanced information provided by the help system describes some of the fixed memory locations within DOS 4 and gives a DOS 4 memory map. This information makes it possible for you to redirect the disk drive numbers D0: through D9: to whichever drives you choose, to control the allocation of sector buffers, and to use the resident binary loader. WARNING: Use extreme caution if you modify any of the memory locations within DOS 4, because a mistake might destroy your disk files.

DISK DRIVE NUMBER INDIRECTION#

DOS 4 has the ability to support up to eight physical drives and up to ten logical drives. The physical drives are the actual pieces of hardware, and are numbered from 1 to 8. The logical drives are the drives referred to in file specifications, and are numbered from D0: to D9:. Whenever you give a file specification, DOS 4 must read the logical drive number and decide which physical drive you are referring Normally, D1: through D8: refer to physical drives 1 through 8 respectively, while D0: and D9: are not supported. However, by modifying certain memory locations within DOS 4, you can make each logical drive refer to whichever physical drive you wish. Drive number indirection is controlled by DTYPE at $73F,$A. four low-order bits of memory location DTYPE+n gives the physical drive number to be associated with logical drive number Dn:. If the four low-order bits of DTYPE+n are zero then logical drive Dn: is not supported. You may change the contents of DTYPE at any time, even if there are open files to the drives in question. The physical drive associated with a file is determined when the OPEN statement is executed, so that changing DTYPE will not cause an open file to start reading

from a different drive. To change DTYPE, use AND and OR operations to alter the four low-order bits without altering the four high-order bits.

BUFFER ALLOCATION#

Buffer allocation is controlled by the contents of memory locations BUFMAX at $710,1 and BUFSIZ at $711,1. BUFMAX contains the number of sector buffers to be allocated; it must be between 2 and 16 (decimal) inclusive. If you want to have N open files on a system with D disk drives then the minimum number of buffers required is N+MIN(N,D). The standard number of buffers is 5. BUFSIZ determines the size of the buffers. It must contain either 0 (for 256-byte buffers) or $80 (for 128-byte buffers). If you have any double-density drives (or single-density drives with a two-sector VTOC), BUFSIZ must contain 0. The standard value of BUFSIZ is 0. It is recommended that you use a value of 0 even if you have only single-density drives. The contents of BUFMAX and BUFSIZ may be changed only when there are no open disk files. The recommended procedure is:

- close all disk files,
- store new values into BUFMAX and/or BUFSIZ,
- load the Disk Utility Package,
- use the WRITE DOS command to write out the new version of DOS 4, and
- reboot the system.

THE RESIDENT BINARY LOADER#

The resident File Management System contains a program which loads and runs a machine-language program in the standard binary load file format. loader has two entry points: LOADER at $70A and KERNEL at $70D. Use entry point LOADER if (a) you know that the loaded program is not going to overwrite the calling program, or (b) you know that the loaded program is not going to return, or (c) you are chaining programs and the loaded program is the next one in the chain. The calling sequence for LOADER is:

```
LDY #FILE&255 ;FILE contains the
LDA #FILE/256 ;filespec
LDX #$FF ;Use #$FF to load and run,
;#0 to load and not run
JSR LOADER
CPY #0 ;Error status returned in Y
BMI ERROR ;Branch if error
```

If you are chaining programs, precede the above code by two PLA instructions and replace the JSR LOADER instruction with JMP LOADER. Entry point KERNEL is used if the loaded program may overwrite the calling program and then return. Should this happen, KERNEL will automatically load and run the Command Processor when the loaded program returns. The calling sequence for KERNEL is the same as for LOADER, except that before calling KERNEL you must store a value into DUPFLG at $736,1 and optionally into DUPLO at $732,2 and DUPHI at $734,2. Storing zero into DUPFLG forces KERNEL to load the Command Processor when the loaded program returns. Storing any non-zero value into DUPFLG makes KERNEL load the CP only if the calling program was overwritten during the load. In this case, the calling program stores the address of its first byte into DUPLO and the address of its last byte into DUPHI. If the calling program is not overwritten during the load then KERNEL will return to the calling program instead of loading the CP. If KERNEL decides that it must load and run the Command Processor then, before it does so, it stores the error status resulting from the binary load into BLDFLG at $737,1.

The CP will examine BLDFLG and issue an error message if the contents of BLDFLG indicates that an error occurred during the binary load process. Note that both LOADER and KERNEL use IOCB

#1 to perform the load. Therefore, before calling either of these routines, make sure that IOCB #1 is closed.

THE STANDARD BINARY LOAD FILE FORMAT#

DOS 4 uses the same format for binary load files as Atari DOS 2.0 and DOS 3. A binary load file consists of one or more "segments", each of which gives the data to be loaded into a contiguous block of memory. A segment consists of three parts:

1. A two-byte file type code, in which each byte contains $FF. This is required on the first segment in the file, but optional on subsequent segments.
2. A four-byte header in which the first two bytes give the address where the first byte of the data block goes, and the last two bytes give the address where the last byte of the data block goes.
3. A data block which contains one or more bytes of data to be loaded into memory.

There are two memory locations which have special significance to the loader: INIVEC at $2E2,2 and RUNVEC at $2E0,2. Every time the loader finishes loading a segment, it checks to see if a non-zero address was loaded into INIVEC. If so, the loader immediately executes a subroutine call (JSR statement) to the address in INIVEC.

When the program returns, the loader will continue the load. When the entire file has been loaded, the loader checks to see if a nonzero address is in RUNVEC. If so, the loader executes a subroutine call (JSR statement) to that address.

COMMAND PROCESSOR FILENAME#

DUPSPC at $722,$10 contains the name of the file which contains the Command Processor. This is normally "D1:QDUP.SYS". The filename must end with a carriage return. If you wish to replace the Command Processor with a program of your own, place the name of the file containing your program into DUPSPC. The program must be in the standard binary load file format. The H. WRITE DOS command can be used to write out a version of DOS 4 containing the new value of DUPSPC. Before it loads the CP, the resident FMS closes all IOCB's. The FMS does not clear the screen, so if a screen clear is desired then the CP must do it. If the CP is going to use the resident screen editor "E:", it should begin with an initialization routine which:

- sets the screen margins,
- opens IOCB #0 to E:, and then
- delays long enough to allow vertical blank to bring up the screen.

There are two entry points which load the CP: indirectly through DOSVEC at $0A, or directly through KERNEL at $70D. The CP can determine which entry point was used by examining BLDFLG at $737,1. The contents of BLDFLG is zero if entry DOSVEC was used, and non-zero if entry KERNEL was used. In the latter case, the contents of BLDFLG is the error status code resulting from the binary load process; a value greater than or equal to $80 indicates that an error occurred.

SIO/PIO COMMANDS AND INTERCEPTION#

The following locations contain the bus commands used by DOS 4 to communicate with the disk drives:

WRCOMD $73A,1 write RDCOMD $73B,1 read DWCOMD $73C,1 write STCOMD $73D,1 status

WRCOMD is the command used to write data sectors, and DWCOMD is the command used to write directory and VTOC sectors. Use $50 for fast write (write without verify) and $57 for slow write (write with verify).

Memory location RRVECT at $7D1,3 contains a JMP instruction. The FMS executes a subroutine call (JSR statement) to RRVECT immediately prior to each call to SIO or PIO (the call takes place after setting up the DCB). If you store an address into RRVECT+1 and RRVECT+2, you can effectively intercept all calls to SIO or PIO. The intended use of RRVECT is to allow programs to remain responsive to the user during long disk operations. The contents of RRVECT+1 and RRVECT+2 are initialized to point to an RTS during both coldstart and warmstart. If an RRVECT routine wishes to prevent the FMS from calling SIO or PIO, it can do so by pulling the return address off the stack, adding 3, and then pushing the result back on the stack. If this is done, the RRVECT routine must store an error status code into DSTATS at $303,1 and into the 6502 Y-register before it returns.

DOS 4 MEMORY MAP#

Memory Loc.	Function	
$0A	CP Load And Run Vector	
$0C	FMS Init Vector	
$1A-	DUP Zero Page	
$43-	FMS Zero Page	
$2E0- $2E1	Binary Load Run Vector	
$2E2- $2E3	Binary Load Init Vector	
$701-$17FB	File Management System	
$17FC-	Sector Buffers	
$1BFC-$20FB	Command Processor	
$20FC-$4FFB	Disk Utility Package	

Note 1: DOS 4 assumes that the FMS Zero Page is not altered in between calls to the FMS.

Note 2: The actual top address of DUP varies depending on the DCF's merged into it.

The top of DUP will never be higher than $4FFB.

ENVIRONMENT CONTROL#

CTBOOT at $7D5,1 determines whether control goes to the CP or to the cartridge when the computer is turned on. If CTBOOT is zero then control goes to the CP. If CTBOOT is non-zero then control goes to the cartridge (provided that a cartridge is installed). Built-in BASIC acts just like a cartridge.

CRTENV at $7D4,1 determines whether control goes to the CP or to the cartridge when SYSTEM RESET is pressed: zero for the CP, non-zero for the cartridge. Any program loaded from the CP that passes control to a cartridge should store a non-zero value into CRTENV.

RELAXATION OF FILESPEC RULES#

REQEOL at $7D6,1 determines how strictly the FMS enforces the rules for filespecs. A value of $FF means that the rules are strictly enforced. In particular, each filespec must end with a carriage return. A value of $0B is the same as $FF, except that extra characters in the extender are ignored. A value of $00 relaxes the rules, which means that any invalid character is interpreted to mean end-of-filespec. The default value of REQEOL is $0B. Values other than $00, $0B, or $FF are not allowed.

I/O ERROR CODES#

Code	Message	Description	
128	BREAK ABORT	You pressed the BREAK key during an I/O operation. This stops the operation.	
129	IOCB ALREADY OPEN	You tried to give an OPEN command to an IOCB which is already open.	
130	NONEXISTENT DEVICE	You did not give a correct device specification (D0: through D9:).	
131	IOCB WRITE ONLY	You tried to do a GET or INPUT command to an IOCB which was opened for output only.	
132	INVALID COMMAND	You gave an I/O command which is not recognized by the Central Input/Output System (CIO).	
133	DEVICE OR FILE NOT OPEN	You tried to GET, INPUT, PUT, or PRINT to an IOCB which is not open.	
134	BAD IOCB NUMBER	You gave an invalid IOCB number.	
135	IOCB READ ONLY	You tried to PUT or PRINT to an IOCB which was opened for input only.	
136	END OF FILE	You tried to read or write past the end of the file.	
137	TRUNCATED RECORD	During an INPUT (or GET RECORD) operation, the input buffer filled before end-of-record was found.	
138	DEVICE TIMEOUT	The device does not respond. indicate a drive which is not turned on or not connected. Also may indicate an incorrect disk drive configuration.	
139	DEVICE DOES NOT ACKNOWLEDGE	An error in the serial I/O system. May indicate a malfunctioning piece of hardware. Also may indicate an incorrect disk drive configuration.	
140	SERIAL BUS INPUT FRAMING ERROR	Hardware malfunction.	

142	SERIAL BUS OVERRUN	Hardware malfunction.	
143	CHECKSUM ERROR	Serial bus communications are garbled. May indicate an incorrect disk drive configuration. May also indicate a hardware malfunction.	
144	DEVICE DONE ERROR	You attempted to write on a write-protected disk. Also may indicate that the disk drive needs adjustment. Also may indicate an incorrect disk drive configuration.	
146	FUNCTION NOT IMPLEMENTED	You gave DOS 4 an invalid command (for example, attempting to give a POINT command to a file open for output).	
160	DRIVE NUMBER ERROR	You specified a disk drive number which is not supported.	
161	TOO MANY OPEN FILES	There are no free sector buffers available.	
162	DISK FULL	There are no more free sectors on the disk. Use another disk.	
164	DISK MAP ERROR	The control information on the disk is messed up. May indicate an incorrect configuration, or an attempt to read a disk that was written by another DOS.	
165	FILE NAME ERROR	You gave an invalid file name.	
166	POINT DATA ERROR	The data you gave in a POINT statement is too big (i.e., you tried to point past the end of the file).	
167	FILE LOCKED	You tried to overwrite or delete a file which is currently locked.	
169	DIRECTORY FULL	You tried to create a new file on a disk whose directory is already full.	
170	FILE NOT FOUND	You tried to access a disk file which does not exist.	
173			

	BAD SECTORS AT FORMAT TIME	Bad sectors were found on the disk while it was being formatted.	
174	SOFTWARE VERIFY ERROR	Output sent to a disk file which is open for verify did not match the contents of the file.	
175	BAD LOAD FILE	An attempt was made to load a disk file which is not in the standard binary load format.	
176	UNKNOWN DRIVE CONFIGURATION	The Disk Utility Package encountered a disk drive configuration it does not understand.	
177	UNABLE TO MERGE	The Disk Utility Package was unable to merge a Disk Configuration File because the file is not in the correct format, or there is insufficient memory, or the disk drive menu is already full.	

CONFIGURING DOS 4 TO YOUR SYSTEM#

To configure DOS 4 to your system, perform steps 1-8 below:

1. Load in the Disk Utility Packageby selecting DISKUTIL on the DOS 4 Command Processor menu.
2. Select item N. CONFIGURE DRIVE.
3. In response to the prompt "CONFIGURE DRIVE - DRIVE NUMBER?", type "1" to indicate drive ##
4. A menu of disk drive configurations will appear on the screen. Type a letter to select the type of drive you have installed as drive #1 and the density in which you want to use it.
5. For an Atari 810 (or equivalent) drive, you must select A.
6. For an Atari 1050-compatible drive, you may select A for single-density operation, or B for dual-density operation.
7. For a Percom-compatible drive, you may select C for single-sided single-density operation, or D for single-sided double-density operation. If the drive is double-sided, you may also select E for double-sided double-density operation.
8. If you have more than one disk drive, use the N. CONFIGURE DRIVE command to configure each drive in your system.
9. Insert a blank diskette in drive #1 and use the I. FORMAT DATA DISK command to format it.
10. Use the H. WRITE DOS command to write out to drive #1 a version of DOS 4 which is configured to your system.
11. Try to boot the system using the new system disk you just created.

If it works, you're done. If not, then go back and try again. You now have a disk with a properly configured version of DOS 4. It contains the files QDOS.SYS, QDUP.SYS, CONFIG.SYS, and DISKUTIL.COM. If you want to copy the remaining DOS 4 files onto your new diskette, perform the steps below:

1. Load the Disk Utility Package and select the O. DUPLICATE FILE command.
2. When the prompt "DUP FILE - SOURCE_DRIVE,DEST_DRIVE" appears, press RETURN.
3. When the prompt "SWAPPING REQUIRED - HOW MANY COPIES" appears, type "1/R" and press RETURN.
4. The disk drive configuration menu will appear. Select the single-sided single-density choice (A or C) for the drive you have installed as drive ##
5. The configuration menu will appear again. Press RETURN. (No choice is necessary.)
6. In response to the prompt "GIVE NAME(S) OF FILE(S) TO MOVE", type "*.COM,*.*" and press RETURN.
7. When "INSERT SOURCE DISK" appears, insert your original DOS 4 program disk in drive #1 and press RETURN.
8. When "INSERT DESTINATION DISK" appears, insert your new system disk in drive #1 and press RETURN.
9. Repeat steps 15 and 16 until the DUPLICATE FILE operation is finished.

You now have a system disk which contains a version of DOS 4 that is correctly configured for your system, together with all the DOS 4 files. If you ever add a new disk drive to your system, you will have to repeat steps 1-8 above to create a version of DOS 4 which is customized to your new system configuration. By this time, you will probably have lots of disks containing your old version of DOS 4; it is safe to use the H. WRITE DOS command to overwrite the old version of DOS 4 with the new version.

Dos 4 System Equates#

```
;FILE QDOSEQU
;9/15/83
;COPYRIGHT 1983 MICHAEL BARALL
;
;This file contains equates for the
;fixed memory locations in QDOS.
;
;
DOSORG EQU $701 ;FMS origin
LOADER EQU $70A ;Binary loader
KERNEL EQU $70D ;Kernel binary loader
BUFMAX EQU $710 ;# of sector buffers
BUFSIZ EQU $711 ;Size of buffers
AUTSPC EQU $712 ;Disk config filespec
DUPSPC EQU $722 ;CP filespec
DUPLO  EQU $732 ;Kernel low address
DUPHI  EQU $734 ;Kernel high address
DUPFLG EQU $736 ;Overwrite flag
BLDFLG EQU $737 ;Bin load error stat
DUPRES EQU $738 ;Reserved for CP
WRCOMD EQU $73A ;SIO data write cmd
RDCOMD EQU $73B ;SIO read cmd
DWCOMD EQU $73C ;SIO map write cmd
STCOMD EQU $73D ;SIO status cmd
SDTYPE EQU $73E ;Shadow for DTYPE+1
DTYPE  EQU $73F ;Drive # indirection
MAPOFF EQU $748 ;Drive density
CONTYP EQU $751 ;Drive configuration
DENSTY EQU $751 ;Flags & type code
SCPERB EQU $752 ;# sectors per block
VTCSEC EQU $753 ;VTOC sector
FSTSEC EQU $755 ;First sector
LOBLK  EQU $757 ;First block
```

```
HIBLK  EQU $758 ;Last block
DIRBLK EQU $759 ;Directory block
DIRSEC EQU $75A ;# sectors in direct.
DIRCNT EQU $75B ;# blocks in direct.
DSPERB EQU $75C ;# logical sec/block
FMCOMD EQU $75D ;SIO format cmd
MODEID EQU $75E ;Disk mode ID code
DRVRES EQU $75F ;Reserved
UNIT    EQU $760 ;Unit #, PIO bypass
RRVECT EQU $7D1 ;SIO/PIO interception
CRTENV EQU $7D4 ;Cart. environ. flag
CTBOOT EQU $7D5 ;Boot to cart. flag
REQEOL EQU $7D6 ;Filespec rule relax.
BUFFER EQU $17FC ;Sector buffers
DUPORG EQU $1BFC ;CP origin
VMENU  EQU $1BFC ;CP run vector
VMENLO EQU $1BFE ;CP run low address
VMENHI EQU $1C00 ;CP run high address
UTLORG EQU $20FC ;DUP origin
UTLTOP EQU $2103 ;Ptr. to top of DUP
MODNUM EQU $2105 ;# of drive modes
MDLNUM EQU $2106 ;# of drive models
MODTAB EQU $2107 ;Drive mode table
MDLTAB EQU $2137 ;Drive model table
AUTOSP EQU $2147 ;Autorun filespec
```