Table of Contents

Atari 8-bit Home page

A Error Messages

B Command Summary — Alphabetical

C Command Summary — By Function

D Miscellaneous Notes

E Glossary

Error Message Summary

C h a p t e r 4 Index

The Command Processor — Commands

The description of command processor is broken down into the two chapters. The first (“The Command Processor — Commands") is a listing of SpartaDOS X commands in alphabetical order. The description of each includes the purpose, syntax, and type of command. The second (“The Command Processor — Advanced Features") discusses batch files and I/O redirection, and contains more detailed information on some of its more complex features.

All the commands which follow are documented in alphabetical order. Command names and parameters represent their function or purpose so they should be easy to remember. For each command we briefly define the **"Purpose"** so you can quickly get an idea of what it is used for. We then show the **"Syntax"**, which shows the proper usage of the command along with its options if applicable. The following conventions are used in the **"Syntax”** section:

[...]	The parameters in the brackets is optional.
a|b|...|z	One or more of these options may be selected. Refer to the specific command's remarks for details.
d:	Drive number or letter (A:...I:, 1:...9:, D1:...D9:, etc.).
path	The path from the current or root directory to the desired one, such as TELECOM>EXPRESS> or \DOS\
fname	The one to 8 character filename. With many commands, wildcards (* and ?) are allowed. Refer to the remarks for specifics for each command.
.ext	A 0 to 3 character file extension. Wildcards are often allowed.
+, -, /	The characters should be used as shown.

If there is an **"Alias"** for the command, it is shown next. We tend to have an alias when there is a shorter command which seems logical, to remain compatible with older SpartaDOS versions, or to try and maintain command similarities for people who use MSDOS.

The **"Type"** will either be "internal" or "external". Internal commands are internal to the command processor itself — they require no other program to perform the command. External commands are found in the "CAR:" directory or may reference one of those files. 48K (6 banks) of the SpartaDOS X cartridge is devoted to these external commands.

"Related" commands are shown next. These may be in the same class or family of commands, or may include

other ways of accomplishing the same function.

"Remarks" include all the details and special rules of command usage. The remarks may also show usage examples. A good way to learn SpartaDOS is to read through each command thoroughly and then try typing in examples of the command including its options. This will help you to understand SpartaDOS, which is important if you wish to be a SpartaDOS user.

If you have used a prior version of SpartaDOS, you will find the command processor similar in feel and will recognize most of the commands. Also, you will notice that the command processor has been greatly enhanced with more sophisticated batch files, command line I/O redirection, user definable prompts, command search paths, and more.

Now, on to the commands . . .

Previous page **Next page**

ARC (Archive Files) Command

Index

Purpose Create and maintain file archives.

Syntax ARC command[option] [d:][path]arcfname[.ext] {filelist}

Type External — on device CAR:

Remarks SpartaDOS X brings a full featured ARC utility to Atari 8-bit computers. ARC is based on and compatible with ARC.EXE by System Enhancement Associates which was written for the IBM PC. It is also compatible with ARC versions running on the Atari ST and other computers. ARC will take a group of files and quickly combine and compress them into a single archive file, taking up far less disk space. It will also add or extract files to or from this archive, show a directory of the archived files, display the contents of an archived file, show the compression method used, encrypt/decrypt files, and more. "ARC" with no parameters will display the syntax, command list, and options.

The **"arcfname"** is the file name of the archive. The **"filelist"** is the list of files to be added, deleted, updated, extracted, etc., to or from the archive. Leave a space between each filename in the filelist. Wildcards are perfectly legal. If no filelist is entered, "*.*" is assumed.

"Command" is one of the following:

A Add file(s) to the archive. Add all files from the file list to the archive.

M Move file(s) to the archive. Move deletes the source file once it has been added to the archive.

U Update file(s) in the archive. Update will look at the date of the files in the archive, replacing files with a newer date, and add all files (from filelist) which do not currently exist in the archive.

F Freshen file(s) in archive. This is the same as update but without the "add" feature. Freshen will replace the older files in the archive with any newer files of the same name.

D Delete file(s) from the archive. Delete will remove the files listed in the filelist from the archive.

X,E Extract file(s) from the archive. Both allow you to extract files from an archive. The method(s) of file compression used when creating the archive is reversed and the files specified in the filelist are restored to their original state.

P Print file(s) to the screen. This allows you to examine the contents of files within an archive without extracting them. Of course this can be diverted to other devices with redirection; for example,

```
ARC P MYARC READ.ME >>PRN:
```

will divert the contents of "READ.ME" from the archive "MYARC" to your printer.

L List file(s) in archive. This shows the filename, original file length, and date/time created of each file in the archive as well as the number of files and total size of files if extracted.

V Verbose list of file(s) in archive. This command shows the filename, original file length, number of files, and total size, just as the L command does. Instead of date and time created, however, the V command shows stowage method, stowage factor (percent of space saved), the file size now, and the total size now.

Valid options are:

B Retain a backup copy of the archive. This is a safety option for the A, M, U, F, and D commands. The B option will result in a backup of the old archive with the extension of ".BAK" as well as the new archive.

S Suppress compression. This will archive files without compressing them. Most people will not use this option but it is faster than using compression.

W Suppress warning messages. Use this command sparingly if at all. This will prevent those unsightly errors from being displayed but will also prevent mistakes from being discovered and avoided.

N Suppress notes and comments. This will suppress the display of the standard ARC screen output which shows the current file being compressed or extracted, the compression method used, etc.

H High speed. With the screen off on the Atari, processing speed is increased 20% to 30%. If you wish to go faster but don't need to see the screen, use this option. The screen display will return when finished.

G Encrypt/decrypt an archive entry. This prevents others from reading your files. G must be the last option and must be followed by a password. If you forget your password, you will not have a useful archive. For example,

ARC AHGICD STUFF WASTE.DOC WASTE.COM READ.ME

In the preceding example the three files in the file list would be added into the archive called "STUFF.ARC" under the password of "ICD" with the screen off.

Archive entries are always saved in alphabetical order. This sorting function puts a practical limit of about 80 files per archive on 64K machines (USE OSRAM) and 180 files per archive on computers which use the extended memory mode (USE BANKED). ARC will not run on 48K machines unless they have an AXLON compatible memory upgrade installed. Archive entries do not save pathnames which means duplicate file names are not allowed (one will replace the other).

ARC is very useful for saving time while uploading/downloading files with a MODEM and saving space for archival storage. ARC uses four stowage methods and automatically determines the best method(s) suited to each file. Our SpartaDOS X version of ARC is also fully compatible with ALFcrunched files, but it is highly recommended that you unARC an ALFed file and then ARC it before adding or updating. This will assure the most compact compression and arrange all files alphabetically within the archive. The four stowage methods used in ARC are as follows:

Stored — no compression used. This is mainly used with very short files.

Packed — Strings of repeated values are collapsed. All files are packed before other compression methods are attempted.

Squeezed — Huffman encoding compression. This is usually only effective with larger machine language files. Huffman encoding uses a weighted binary tree method assigning the lowest bit representations to the most commonly used characters.

Crunched — Dynamic Lempel-Ziv compression with adaptive reset. This is created on the fly and is stored as a series of bit codes which represent character strings. Crunched is one of the more effective methods used. ALFcrunch exclusively uses a variant of this method.

NOTICE: The name ARC, compatibility, and all other similarities to the ARC.EXE program by SEA (for MSDOS computers) are intentional. This trademark and the "look and feel" of the program have been licensed for SpartaDOS X by ICD, Inc. from SEA (System Enhancement Associates).

Previous page

Next page

ATR (Attributes) Command

Index

Purpose Sets/clears file attributes in the directory. Replaces the Protect and Unprotect functions from older SpartaDOS versions.

Syntax ATR [+A|H|P] [-A|H|P] [d:][path]fname[.ext]

Alias ATTRIB

Type Internal

Related DIR

Remarks SpartaDOS X adds two new attributes to the standard SpartaDOS directory entry — these are the hidden and archived bits. The old commands PROTECT and UNPROTECT were used to set or clear the protection bit. With SpartaDOS X, the ATR command replaces the old commands and works with the new attributes.

Although many other commands allow the usage of the "S" (subdirectory) attribute, it is illegal to attempt to change this status bit as it would corrupt the subdirectory integrity. Therefore, ATR does not affect this.

Note that although the syntax of the ATR command looks similar to that of the DIRectory or TYPE commands, the attributes here are not the scan mode, but describe the set(+)/clear(-) attributes operation to be performed on the directory entry that matches the given filespec. This means that the scope of the ATR command is all files matching the filespec (including those files which are hidden).

The directory entry attributes are as follows:

A Archived file. This attribute is cleared whenever a file is created or updated. The archive bit is set when the file is backed up by a program such as FlashBack!. This attribute is *not* related to the ARC command.

H Hidden file. You may hide files and/or subdirectories. If a file is hidden, you may load it as a command only — commands such as TYPE and COPY will not see hidden files (unless you specify attributes with those commands). The file is hidden when this bit is set.

P Protected file. You may not ERASE, or update protected files. Use the ATR command to protect or unprotect files. The file is protected when this bit is set.

S Subdirectory. This attribute is unchangeable — thus not legal in the ATR command! This bit is set to indicate a subdirectory. If cleared, it would be seen as a file which could cause significant damage.

For example, to set the archived status and clear the protection bit of all ".COM" files, type the command

```
ATR +A -P *.COM
```

For further information about which status bits in the directory entry are affected by these new attributes, refer to the "Technical Information" chapter.

Previous page **Next page**

DIR (Directory) Command & DIRS (Short Directory) Command

Index

Purpose Lists either all the directory entries, or only those matching a specified filespec. DIR will optionally give you a count of files listed.

Syntax DIR [+A|H|P|S] [-A|H|P|S] [d:][path][fname][.ext] [/PC]

DIRS [+A|H|P|S] [-A|H|P|S] [d:][path][fname][.ext] [/PC]

Type Internal

Related ATR, FIND, MENU, PATH, PAUSE, PROMPT

Remarks DIR displays the SpartaDOS file directory showing filename, extension, file size in bytes, date, and time created. It also shows a <DIR> in the size field when it sees a subdirectory, displays the Volume and Directory name at the top of the listing, and shows the Free Sectors count at the end of the listing. If you include a "/P" parameter, the DIR command will wait for a key press after displaying each directory screen (23 lines). The "/C" parameter will give a count of the number of entries displayed in that directory.

When reading an Atari DOS 2 type diskette, the date and time are omitted for obvious reasons and the file size is roughly converted to bytes. (Atari DOS 2 and clones use sector lengths instead of bytes in the directories so this can not be an exact file size representation.) All AtariDOS 2 type diskettes will have a volume name of "AtariDOS" and a directory name of "ROOT".

You may specify the attributes of the files you wish to display, for example

```
DIR +S
```

will display only subdirectories. Note that the default directory attributes (no attributes specified) is "-H' (do not show hidden files). If you wish to see all files (including hidden files), simply enter

```
DIR +
```

Using a '+' with no attribute listed will match all files, regardless of attribute. This will work with any command that allows attribute selection.

The DIRS command has exactly the same syntax, but it displays the directory in Atari DOS 2

"compatibility mode" — with no time/date, and with the file size displayed in sectors rather than in bytes. Since the Free Sectors count in DIRS is limited to three digits, the maximum size displayed will always be 999. It also displays the "protected" status (+P) as "*" before each protected filename.

The attributes are as follows:

A Archived file. This attribute is cleared (-) whenever a file is created or updated. It is set when the file is backed by a program such as FlashBack!

H Hidden file. You may hide files and/or subdirectories. If a file is hidden, you may load it as a command only. Commands such as TYPE and COPY will not see hidden files (unless you specify attributes with those commands).

P Protected file. You may not ERASE or update protected files. Use the ATR command to protect or unprotect files.

S Subdirectory. This attribute is unchangeable.

If you do not specify a filespec, "*.*" is assumed as in the following examples:

```
DIR MYSUB>
DIR +P
DIR ..\
```

Note that you must follow a subdirectory with a ">" or "\" character if you wish to see the contents of that directory.

Previous page **Next page**

FIND (Find Files) Command

Index

Purpose This command searches all directories on all drives for files matching the given filespec. If you enter a drive number, FIND will only look on that particular drive.

Syntax FIND [d:]fname[.ext]

Type External — on device CAR:

Related DIR

Remarks FIND will quickly find a file anywhere on your drives. This becomes very useful when you start using subdirectories and multiple drives. FIND works like WHEREIS.COM (from SpartaDOS Tool Kit) with a few exceptions. FIND's search is slightly different and it does not have a parameter to display filesize, time, and date, for each file found.

The filename may include wildcards as desired. All filename matches found will be displayed with the full path from the root directory to the filename match. The number of matches found will be displayed at the end of the search. All drives will be searched unless a drive number is specified. FIND will also find and display hidden filenames.

Previous page

Next page

ERASE Command

Index

Purpose This command deletes the file in the specified directory on the designated drive, or deletes the file from the default drive if no drive is specified. If no path is specified, the file is deleted from the current directory.

Syntax ERASE [d:][path]fname[.ext]

Alias DEL & DELETE

Type Internal

Related MENU, UNERASE

Remarks You can use wildcards such as '*' and '?' to delete multiple files, but use caution since no warning is usually given. If you do enter *.* as the specifier, SpartaDOS X will prompt you with:

Erase ALL: Are you sure?

Any other combination of wildcards and characters will assume you know what you want. Use the VDEL command (in SpartaDOS ToolKit) for a prompted delete or use MENU for tagging files to delete.

Previous page

Next page

MENU Command

Index

Purpose This program allows you to select files and then perform COPY, ERASE, RENAME, etc. commands on all selected files. It is similar to other SpartaDOS menu programs, but provides many new features.

Syntax MENU

Type External — on device CAR:

Related COPY, ERASE, RENAME, TYPE

Remarks MENU is useful for operations that include more than one file and is required for single drive copies. It includes three main windows with the commands and prompts displayed below the windows. The "upper left window" is for directories. It will display the subdirectories along with the tree structure showing how they are related. The "upper right window" shows statistics on the area logged. This includes: filespecs, total files, total bytes, tagged files, and tagged bytes. The "lower window" shows the files.

The command menus are broken up into three major classifications: file (**File**), directory (**Dir**), and extra (**Xtra**). The classification is indicated at the lower left of the screen. Toggle between the file and directory command menus by pressing <RETURN>. You can go to the extra command menu by pressing <ESC>. To exit from the MENU you press <ESC> then "Q". The "^" character before a menu selection means to hold down the CONTROL key while pressing the selection key.

File command menus include: COPY, DELETE, FILESPEC, LOG, PRINT, RENAME, TAG, UNTAG, and VIEW. **Dir** command menus include: AVAILABLE, DELETE DIRECTORY, FILESPEC, LOG, MAKE DIRECTORY, PRINT, TAG DIRECTORY, and UNTAG DIRECTORY. **Xtra** command menus include: DISPLAY, QUIT, and SORT.

File Commands While in the file command menu, use the "↑↓" keys to move the file selector up or down one file at a time or use the "← →" keys to move the file selector up or down one screen at a time. The files shown in the file window are sorted alphabetically by name. They represent the current directory shown in the directory window under the directory selector. (See "Dir Cmnds".)

C **Copy** — will copy the file under the file selector. You will be prompted for a destination drive, then a destination path. If copying to the same drive you will also be prompted to insert the destination disk, and then to insert the source disk.

^C **^Copy** — will copy all tagged files. Prompts are the same as with Copy.

F **Filespec** — allows you to enter a filespec with wildcards to narrow down the logged (and displayed) files. Only legal filename characters and wildcards are allowed. Do not enter drive number or path here; instead use Log for that.

L **Log** — will allow you to change the drive number logged and/or path.

P **Print** — will print the file currently under the file selector. This is only useful for ASCII text files unless you have a printer driver installed which will print ATASCII graphics characters.

^P **^Print** — will print the files currently tagged. It will send a form feed in between files.

R **Rename** — will allow you to rename the file under the file selector. As a reference, rename prompts you with the present drive number, path, and filename. You can then enter the new filename directly under the old.

T **Tag** — will tag (mark) the file under the file selector then move the file selector down one filename. A small tag character "♦" will appear to the right of the filename showing it as tagged.

^T **^Tag** — will tag all files currently logged (in the current directory).

U **Untag** — will untag the file under the file selector. The tag character will disappear and the file selector will move down to the next file.

^U **^Untag** — will untag all files currently logged (in the current directory).

V **View** — will display the contents of the file under the file selector.

Dir Commands The directory selector indicates the current directory. While in the directory command menu, use the "↑↓" keys to move the directory selector up or down one directory at a time. When finished with the "Dir Cmnds", press <RETURN> to go back to "File Cmnds" or <ESC> for "Xtra Cmnds".

A **Avail** — will give you the amount of free space available on a drive. You are prompted to enter a drive number and will be shown the free space in bytes.

D **Del Dir** — use "↑↓" to move the directory selector. If the directory selected is empty (as shown in the file window), it can be deleted.

F **Filespec** — allows you to enter a filespec with wildcards to narrow down the logged (and displayed) files. Only legal filename characters and wildcards are allowed. Do not enter drive number or path here; instead use Log for that. You must go to the "File Cmnds" if you want to do any file operations other than Tagging or Untagging full directories.

L **Log** — will allow you to change the drive number logged and/or path.

M **Make Dir** — will create a new subdirectory in the current directory selected. After the new directory is created, the system will relog and you will be back at the root of what was previously logged (always indicated by ">").

P **Print** — will prompt you with two choices: Directory or Tree. Directory will print the list of the files as displayed in the file window. (If the display is set to show the short form files, they will be printed in one long list, not side by side as displayed.) Tree will print the directory map (tree structure) as displayed in the directory window.

T **Tag Dir** — will tag all files in the current directory (under the directory selector).

^T **^Tag Dir** — will tag all files in all directories currently logged.

U **Untag** — will untag all files in the current directory (under the directory selector).

^U **^Untag Dir** — will untag all files in all directories currently logged.

Xtra Commands The "Xtra Cmnds" always take you back to the previous command menu when finished (except Quit). You can also press <ESC> to leave this menu.

D **Display** — toggles the display in the file window between two types. The default display shows the filename with extension, the status of the three file attributes, the file size in bytes, along with the date and time created. This display takes all 38 columns in the file window. The optional display shows two columns of filenames (side by side) with extensions, and their attributes. The attribute letter is displayed if set or a dot if cleared.

Q **Quit** — is the correct way to exit the menu back to the DOS command PROMPT. NOTE: Do not quit or stop operations by pressing <RESET>. This is a very bad practice that can lead to unrecoverable files.

S **Sort** — will sort the file display by: Name, Ext, Date, Size. This is a forward sort which defaults to name. To permanently sort directories or reverse sort them, use SORTDIR from the SpartaDOS Tool Kit.

[Previous page](#) [Next page](#)

UNERASE Command

Index

Purpose This command restores files previously erased (if possible).

Syntax UNERASE [d:][path]fname[.ext]

Type External — on device CAR:

Related ERASE

Remarks Wildcards are permitted. You will be asked (for each recoverable file matching the filespec you entered) if you wish it restored or not. If you think that erased files exist in your directory that were not seen by UNERASE, they were not recoverable for one of two reasons:

1) The file's directory entry has been allocated to another file which was copied to the directory after the original file was ERASEd.
2) A sector of the file has been allocated to another file since the original file was ERASEd.

Previous page

Next page

DUMP (Display File in HEX Format) Command

Index

Purpose This command displays a file in HEX and ATASCII form.

Syntax DUMP [d:][path]fname[.ext] [start] [len]

Type External — on device CAR:

Related TYPE

Remarks The parameters "start" and "len" are the start addresses in the file and the number of bytes to dump (respectively). They are assumed to be in decimal format unless preceded by a "$" (in which case they are HEX format).

DUMP is useful to quickly examine the contents of a file. To modify a file or examine and modify disk sectors, use DiskRx from SpartaDOS Tool Kit.

Previous page

Next page

FORMAT Command

Index

Purpose Initializes a disk in either SpartaDOS or Atari DOS 2 format. You may select density, sector skew, tracks, and volume name before formatting. It supports most known hardware configurations for your computer.

Syntax FORMAT

Type Internal

Related BOOT, CHVOL

Remarks The FORMAT command is really a menu driven program which allows you to initialize just about any type of disk that works with an Atari 8-bit computer. It can be called from the command processor by typing FORMAT or from within a program with XIO 254 (see the "Programming with SpartaDOS X" chapter). This allows FORMAT to be used with most programs that support disk initialization like AtariWriter or 850 Express!

When you format a floppy diskette, you are first writing the sector structure to the diskette so the DOS has a place to put the program information. Next the directory structure is written to the diskette which is where the DOS keeps track of sector usage. You can also initialize a RAMDISK or Hard Disk in the FORMAT menu but only with the BUILD DIRECTORY option.

You can exit or quit the FORMAT menu at any time before formatting begins by pressing <ESC>.

After entering the FORMAT menu you choose the following parameters:

U **Unit** is the initial selection that must be made. The formatter needs to know which drive you wish to initialize. Valid choices are: 1 - 9 or A - I. After entering the unit number or letter the program reads the drive to determine what type it is. FORMAT automatically determines whether the drive is a floppy disk drive and if it is configurable or if the drive is a RAMDISK/Hard Drive which appear the same at this point.

Note: FORMAT will only write directories to RAMDISKS and hard drives. An internal RAMDISK must be installed with the RAMDISK.SYS driver. A hard drive partition must be low-level (physical) formatted with with a program that should have been provided with the hardware.

S **Skew** refers to the order in which the sectors are arranged in a given track. The two valid choices are: High Speed and Standard. High Speed will automatically put the correct UltraSpeed skew on a disk using SpartaDOS with the US Doubler or Indus GT drives. It will also put the correct high speed skew on the Atari XF551 drive under Double Density. Standard skew is used on all other floppy drives. If you do select High Speed skew and the drive does not support a high speed mode, the format program will receive an error from the drive and then try to format under Standard skew. The correct skew is required on most drives for the fastest possible reading and writing. Skew is not applicable to RAMDISKS and Hard Drives, since they can not be physically formatted by this program.

NOTE: Skew refers to the order in which the sectors are written on the diskette. The optimum skew will position the sectors so that after sector 1 is read and the drive CPU is ready for the next, sector 2 is directly under the head for reading; after 2 is read, 3 is directly under the head, etc. (Usually 2-8 sectors have passed under the head before the next sector can be read. This varies with drive speed and SIO baud rate.) If the skew is off, it may take a full disk revolution to read or write the next sector each time. No harm is done, the drive just reads and writes slowly.

M **Mode** is either **Sparta** or **Atari**. Sparta is for SpartaDOS disk directory structure and Atari is for all the AtariDOS 2.0 clones and their directory structures.

NOTE: FORMAT does not write a "DOS" file to the diskette. If you want to create a bootable SpartaDOS disk, you must copy a "DOS" file from the SpartaDOS Construction Set to your formatted disk and then use the BOOT command. (See the BOOT command.) If you want to create a bootable AtariDOS diskette, you will need to boot AtariDOS and write the DOS.SYS and DUP.SYS to the drive through its menu. SpartaDOS X will boot with any or no disk.

V **Volume** is a way of naming your diskettes for organizational purposes. Up to eight characters are allowed on SpartaDOS formatted diskettes. The volume name may contain any ATASCII characters except spaces. Volume is used on SpartaDOS diskettes only and is not applicable to other DOS types.

D **Density** may be one of three types used with 8-bit Atari computers. These are: **Single**, 128 bytes per sector FM, **Dual**, 128 bytes per sector MFM, or **Double**, 256 bytes per sector MFM. (FM and MFM refer to bit density where MFM writes twice the number of bits in the same area as FM.) Atari 810s only support Single density. Stock Atari 1050s support Single and Dual. Atari 1050s with the US Doubler (or other OS enhancement), Atari XF551s, and Indus GTs support all three densities. Most other disk drives for the 8-bit Atari support Single and Double density.

T **Tracks** can be **40 SS, 40 DS, 77 SS, 77 DS, 80 SS,** and **80 DS**. SS means Single Sided (1 head writes on one side of the diskette) and DS means Double Sided (2 heads with each writing on opposite sides of the diskette). All Atari brand drives will use 40 SS except for the XF551 which is capable of 40 DS. Most of the other 5 1/4 inch drives will be either 40 SS or 40 DS. (See your drive manual if you are not sure.) 77 Tracks is used for 8 inch disk drives connected with an interface like the ATR8000 or PERCOM controller. 80 Tracks is used for 3 1/2 inch drives and high capacity 5 1/4 inch drives with a similar interface. All drives with two heads will also format in the SS mode.

NOTE: The drive controllers do not provide adequate feedback to the computer when formatting a diskette to determine whether the Tracks selection is wrong for the drive. It is important to enter the correct information or the disk will end up with an incorrect free sectors count.

F **Format Disk** will start the physical format of a floppy diskette assuming you have entered all the other parameters required. It also writes the directory structure selected in Mode after the physical format and verify is completed. (The physical format and verify are functions of the floppy disk controller and not affected by the SpartaDOS VERIFY command.) CAUTION: The Format Disk procedure obviously destroys all previous information stored on the diskette.

B **Build Directory** is the initialization option available for RAMDISKS and Hard Disks, although it will work equally well with floppy disks. The only parameters available for these disks are Unit number and Volume name. The others are predetermined or not applicable. Build Directory writes fresh SpartaDOS directory structure to the drive. Unit selected which means it will destroy all previous information stored in the RAMDISK or Hard Disk partition.

The physical format of a Hard Disk drive must be performed by a special program written for the particular hard drive, interface, and controller. That is considered a low-level format and is beyond the scope of the FORMAT menu. The physical format of the RAMDISK is provided by the RAMDISK handler at installation.

Sectors and Bytes counts are also shown on the FORMAT menu and are determined by what is read from the configuration on RAMDISKS or Hard Disks or by the parameters selected for a Floppy Drive format.

Indus GT NOTES: It should be mentioned here that the Indus GT has a few known quirks or bugs. Because of this, Dual (Enhanced) Density is not supported on this drive. This should not be a problem since Dual Density is really unnecessary on a double density drive. The Indus has also been known to keep spinning indefinitely when used in a system with US Doubler enhanced 1050s. If you experience this problem, the best solution is to stop using mixed drives in your system.

Previous page **Next page**

BOOT Command

Index

Purpose This command tells a SpartaDOS formatted disk to boot a particular program at start up (normally a disk-based DOS).

Syntax BOOT [d:][path]fname[.ext]

Type Internal

Related COLD, FORMAT

Remarks The DOS loader on the first three sectors of each SpartaDOS formatted diskette (version 2 and above), can load and run files in the same manner as a command file. Normally DOS is loaded, but anything could be loaded as long as it avoids the loader memory ($2E00-$3180).

The FORMAT command does not put SpartaDOS on the diskette it formats, so, if you want to create a bootable SpartaDOS diskette, you must COPY SpartaDOS to the diskette and use the BOOT command. If you just use SpartaDOS X, this will never be necessary (since SpartaDOS X boots from cartridge), but if you have SpartaDOS 3.2 (or 2.3), you may wish to create bootable diskettes. Of course you may still use the XINIT command to format and install DOS on your diskette.

This command is the most misunderstood command in SpartaDOS, so here are a few pertinent facts you should know:

- The BOOT command simply writes the starting sector number of the sector map of the file to boot in a specific location on sector 1 of the diskette. (See "Technical Information")
- If the file which is set to boot is either ERASEd or COPYed over, the boot flag is cleared — you will get the message "Error: No DOS" when attempting to boot that diskette until you set a new file (e.g. X32D.DOS) to boot!
- The file you set to boot may reside anywhere on the diskette — even in a subdirectory.
- This command does **not** work with SpartaDOS 1.1 — 1.1 does not contain a 3 sector booter! SpartaDOS 1.1 has a simplistic booting scheme that just loads a number of consecutive sectors. Since you now have SpartaDOS X, we *strongly recommend* that you abandon SpartaDOS 1.1.

Previous page Next page

CHVOL (Change Volume Name) Command

Index

Purpose This command changes the volume name on the specified drive.

Syntax CHVOL [d:]volname

Type External — on device CAR:

Related CHKDSK, FORMAT, DIR

Remarks This command will not change the volume name on Atari DOS 2 diskettes since they physically have no volume name. Up to eight characters are allowed on SpartaDOS formatted diskettes. The volume name may contain any ATASCII characters including spaces and inverse characters.

Previous page

Next page

KEY (Keyboard Buffer) Command

Index

Purpose This command installs a 32 character keyboard buffer and also links an "internal" KEY command into your system (for turning the buffer on and off).

Syntax KEY ON|OFF

Type External — on device CAR:

Remarks The first time you use this command, it installs a keyboard driver into your system. The keyboard buffer will provide a faster key repeat and allow you to type ahead while the system is busy. Then the ON/OFF parameter is interpreted, enabling or disabling the keyboard buffer accordingly.

Once the keyboard buffer has been installed, the global symbol "@KEY" is defined and further KEY commands call this symbol to turn the buffer on and off.

NOTE: The keyboard buffer may be incompatible with some programs but is more compatible with other programs that the SpartaDOS 3.2 buffer (most notably with the Action! cartridge).

Previous page Next page

COPY Command

Index

Purpose Copies one or more files to another drive and, optionally, gives the copy a different name if specified

COPY also copies files to the same disk. In this case, you must give the copies different names unless different directories are specified; otherwise, the copy is not permitted. Concatenation (combining of files) can be performed during the copy process with the "/A" parameter.

You can also use the COPY command to transfer data between any of the system devices. Some applications of this would be to create a batch file or to print a text file.

To copy files with a single disk drive and no RAMDISK, see the MENU command. MENU has a provision for disk swapping, while COPY does not.

Syntax COPY [d:][path][fname][.ext] [d:][path][fname][.ext][/A]

Type Internal

Related MENU, TYPE

Remarks The first filespec specified is the source file name. If none is given, a default filespec of "*.*" is assumed (which will copy all files in that directory). The device for the source file should be given; however, it is possible to omit the source device if you use commas (instead of spaces) to separate parameters, for example:

```
COPY, ,D3:
```

will copy all files from the default drive/directory to the current directory of drive 3. The second filespec is the destination — if no filename is specified, a default filespec of "*.*" is assumed (which will copy the files without changing the names).

You may use wildcards ('*' and '?') in both source and destination filespecs. If used in the pathnames, the first directory match will be used.

When using wildcards with the COPY command, the same renaming convention as in the RENAME command is used. The source filespec is used to find directory matches, and the destination filespec renames them by overriding characters in the source name with the non-wildcard character

in the corresponding position of the destination name.

If you are copying from a device other than "DSK:" (alias "Dn:" or just "n:"), then just one file is copied and the destination filespec is the name that the file will be saved under. For example:

```
COPY CON: B:*
```

is illegal because you may not have wildcard characters in a destination filename when COPYing from a character device (or for that matter SAVEing any file). However, if copying from one character device to another character device, filenames are not used. (Character devices never use filenames.) For example:

```
COPY CON: PRN:
```

As in the above two examples, when COPYing from "CON:" you signal the end of file by pressing a <CTRL-3> after typing the text. Also, a <RETURN> must follow each line you enter, otherwise that line will be lost.

Another use for the COPY command is to list files to the printer or screen, for example:

```
COPY README.DOC CON:
```

will display the contents of "README.DOC" to the screen and:

```
COPY README.DOC PRN:
```

will send it to the printer. Note that both of the above examples could have been performed with the TYPE command as follows:

```
TYPE README.DOC
TYPE README.DOC >>PRN:
```

with the second command sending the contents of the file to the printer.

You may also append files using the COPY command by using a "/A" immediately following (no space) the destination filespec. (SpartaDOS 3.2 allows a "/A" when SAVEing any file to force append mode — SpartaDOS X only supports this feature on the COPY command.)

If you only have one drive and wish to COPY files from one diskette to another, you must either COPY the file from the source diskette to a RAMDISK and then from the RAMDISK to the destination diskette, or use the MENU program as it allows disk swapping during copying.

Previous page **Next page**

TYPE Command

Index

Purpose This command displays the contents of a specified file.

Syntax TYPE [+A|H|P|S] [-A|H|P|S] [d:][path]fname[.ext] [/P]

Type Internal

Related COPY, DUMP, MENU, PAUSE

Remarks You may display any file and are not limited to a maximum line length (as was the case with SpartaDOS 3.2). Press <CTRL-1> to stop and start the display. You may specify attributes as in the DIR command — the default attributes are "-HS". (See the DIR command for a description of the attributes.)

If you include a "/P" parameter, the TYPE command will wait for a keypress after each 23 lines of text.

Previous page

Next page

COMMAND (The Command Processor)

Index

Purpose This program allows you to enter commands and run other programs. It is not entered as a command itself but is automatically invoked when you enter DOS.

Type External — on device CAR:

Related All Commands

Remarks Many of the commands are of type "internal" — this means that the command processor knows how to perform the command without loading any other programs.

"External" commands must load from disk or CARtridge into memory and then perform their function. When you execute these commands, they must reside on the current drive and directory, otherwise you must specify what drive or device they reside on (by preceding the command with a device or drive identifier). The PATH command can add additional drives/paths to search for the file. For example the default PATH (which is)

```
PATH CAR:
```

allows commands such as CHTD or DUMP to run without having to specify the "CAR:". Of course you may add additional directory paths (see the PATH command).

You will notice that the command processor itself is "external". This is to give you more memory (3-4Kbytes) to run your application programs. In fact, whenever you run an "external" command or program, the command processor is unLOADed from memory and replaced by the new program. Then, when that program is finished running, the command processor is reLOADed and awaits your command. The exception to this rule is if you enter the command

```
LOAD COMMAND.COM
```

This actually holds and links the command processor in memory, thus the unLOAD/reLOAD cycle is circumvented.

Previous page

Next page

DATE Command

Index

Purpose This command displays the current date and allows you to set the date.

Syntax DATE

Type Internal

Related CHTD, TD, TIME

Remarks This command produces the following output

```
Current date is  6-24-88
Enter new date:
```

You may enter the new date or press <RETURN> if you don't want to set a new date. Enter the date in the format "mm-dd-yy" where "mm" is the month, "dd" is the day, and "yy" is the year.

This command will produce meaningless results if you do not have a clock driver installed in your system. The two clock drivers are "CLOCK.SYS" and "JIFFY.SYS" — the first using the R-Time 8 en and the second using the jiffy counter to keep its time. By default, one of these drivers will be installed when you boot, but this can be overridden by creating a custom "CONFIG.SYS" file and not including these drivers in the configuration.

Previous page Next page

RENAME Command

Index

Purpose This command allows you to change the name of one or more files.

Syntax RENAME [d:][path]fname[.ext] fname[.ext]

Alias REN

Type Internal

Related MENU

Remarks Wildcards may be used in both filespecs. A device and path may only be specified on the first filename (the old name filespec). Filenames must be specified for both source and destination names, otherwise an error will occur. The rules for wildcarding are the same as for the COPY command. Here are a few examples:

```
RENAME *.BAK *.DOC
```

The above command changes all the extensions to "DOC" of files that previously had extensions of "BAK".

```
RENAME AC*.* *.XX
```

This command changes the extension of all files beginning with "AC" to "XX".

NOTE: Rename your files with caution. There is no check for existing filenames. Careless renaming may result in more than one file with the same name. The only way to separate these files is with a sector editor like DiskRx (from SpartaDOS Tool Kit) or through tedious use of the ERASE and UNERASE commands. To use this second method, ERASE the duplicate files, then UNERASE them. Answer 'Y' to only one of them and 'N' to the rest. Rename this file to something different and repeat the this UNERASE, RENAME cycle until all of the files are restored.

Previous page

Next page

PROMPT (Set System Prompt) Command

Index

Purpose Change the system prompt.

Syntax PROMPT [prompt_string]

Type Internal

Related PATH

Remarks The text in prompt_string is taken by SpartaDOS to be the new system prompt. Special meta-strings can be imbedded in the text in the form "$c" where 'c' is one of the following characters:

L	print current drive letter ('A' through 'I')
N	print current drive number ('1' through '9')
P	print path on current drive
D	print current date
T	print current time
R	print an EOL character (advance to next line)

NOTE: "P" will cause the current drive to be read every time <RETURN> is pressed to detect a disk change. This should be taken out of the path before parking a hard drive, since the "P" will read the disk and unpark the drive. A batch file changing the prompt, then parking the drive, would be very useful for this purpose.

If no parameter is specified, then the current prompt_string is displayed.

For example the command:

```
PROMPT $L:$P>
```

will display a prompt in the form:

```
B:>DOS>
```

assuming the current drive is 2 and the current path is "DOS". Also, the character '_' will display as a space rather an underline. (Thus a prompt can end in a space.)

The PROMPT command is really just a convenient form of the SET command, for example the above command could be performed by:

```
SET PROMPT=$L:$P>
```

just as easily. The default value of the "prompt" variable is "D$N:" which displays the same prompt as older SpartaDOS versions. If the "PROMPT" variable is not defined, SpartaDOS X will prompt with a '>' character — the only way to clear the "PROMPT" variable is with the command:

```
SET PROMPT
```

You can not use non-inverse lower case letters in the prompt because the command processor automatically converts all lower case characters to upper case before processing. Inverse and cursor control keys (preceded by the escape key) may be used in the prompt.

When using the '$P' meta-string in the prompt, the default drive will be read each time the prompt is printed. This will cause an error to be printed within the prompt if there is no disk or a bad disk in the default drive, or if the disk is of a format unrecognized by SpartaDOS X. (To use Atari DOS format disks with SpartaDOS X you must install the ATARIDOS.SYS driver).

Using the '$P" meta_string in the prompt can also cause problems when attempting to park a hard drive, since the drive will be "unparked" to read the path when the prompt is printed. The solution to this is to set the environment variable PROMPT to a value not containing '$P'. Since drives are usually parked only when you are, through using the computer, you may simply clear the PROMPT variable with a

```
SET PROMPT
```

before parking the drive. You could set up a batch file to clear the prompt variable and then park the drive to simplify this operation.

Previous page

Next page

RMDIR (Remove Directory) Command

Index

Purpose This command deletes an empty subdirectory from the specified drive.

Syntax RMDIR [d:]path

Alias RD & DELDIR

Type Internal

Related CHDIR, MKDIR, PATH

Remarks The directory must be empty before it can be removed. The last directory name in the path is the directory to be removed. This function is not supported by the ATARIDOS.SYS driver even though subdirectories are supported by that driver.

```
RD TEST
DELDIR 3:>MODEM>TEST
```

The first example removes a subdirectory on the default drive by the name of "TEST". An error will occur if the directory has files in it. The second example removes a subdirectory on drive D3: by the name of "TEST" in the subdirectory called MODEM which is under the MAIN directory.

For related information see the CHDIR, MKDIR, and PATH commands.

Note: If a file has been opened for write or update but not properly closed (usually by hitting reset or losing power while it is opened) its entry in the directory will not be removed, although it may not show in a listing. A subdirectory containing a "phantom" entry of this type can not be deleted. CLEANUP or DISKRX from the SpartaDOS Tool Kit can be used to mark such an entry as deleted and not open so that the directory may be removed. The status byte of the directory entry will have bit 7 and bit 3 set. These should be cleared and bit 4 set. It is possible that some sectors have been allocated for this file. These should be de-allocated in the bit map.

Previous page

Next page

PAUSE Command

Index

Purpose Suspends system processing and displays the message "Press <RETURN> to continue".

Syntax PAUSE

Type Internal

Related DIR, TYPE

Remarks You can insert PAUSE commands within a batch file to provide the opportunity to change diskettes between commands or to step through a process, giving you time to read instructions, etc.

To resume execution of the batch file, press the <RETURN> key.

NOTE: It is very dangerous to change diskettes (during a PAUSE) on the drive from which the batch file was running. If using PAUSE to change diskettes, run the batch file from a RAMDISK or another drive which will not be changing.

Previous page Next page

TIME Command

Index

Purpose This command displays the current time and allows you to set the time.

Syntax TIME

Type Internal

Related CHTD, DATE, TD

Remarks This command produces the following output

```
Current time is 14:34:30
Enter new time:
```

You may enter the new time or press <RETURN> if you don't want to change it. Enter the time in the format "hh:mm:ss" where "hh" is the hour (24 hour clock), "mm" is the minutes, and "ss" is the seconds. (SpartaDOS 3.2 uses AM/PM — SpartaDOS X uses a 24 hour clock.)

This command will produce meaningless results if you do not have a clock driver installed in your system. The two clock drivers are "CLOCK.SYS" and "JIFFY.SYS" — the first being for the R-Time 8 and the second uses the jiffy counter to keep its time. By default, one of these drivers will be installed when you boot, but this can be overridden by creating a custom "CONFIG.SYS" file and not including these drivers in the configuration.

Previous page

Next page

MEM Command

Index

Purpose This command displays the current low memory limits of your system and the number of available banks of windowed RAM.

Syntax MEM

Type Internal

Related CHKDSK, LOAD

Remarks This command displays the bottom limit of available user and extended memory. Two limits for each memory region are given — the first being the top limit of installed drivers and the second being the top limit of held-in-memory applications.

In the following example,

```
Main: $0F6D,$1456
 Ext: $6123,$6455
7 banks of RAM available
```

the installed drivers use memory from $700-$F6C and $4000-$6122, and the held-in-memory applications used reside in memory from $F6D-$1455 and $6123-$6454.

The held-in-memory applications are LOADed into memory and consist of files such as COMMAND.COM and X.COM. The drivers installed in memory are files such as SPARTA.SYS, ATARIDOS.SYS, RAMDISK.SYS, etc.

Normally the first and second numbers above will be the same. The OS variable MEMLO (at $2E7) contains the second number. If LOAD is executed with no parameters then all in-memory applications are abandoned and the second number is lowered to the first.

NOTE: If a permanently installed driver is installed after LOADing a held-in-memory application, both low memory values will be raised above it and any held-in-memory applications will become permanent.

Although there are two possible extended memory regions, only one may be used by SpartaDOS X.

This is determined at bootup time and is dependent upon the CONFIG.SYS file and/or the computer you are using. (See "Technical Information" for a discussion on extended memory.) Note that although the MEM command does not explicitly say what memory extended memory range is in use, it can be inferred by the addresses listed in the "Ext:" field. The ranges are as follows

- $4000-$7FFF Banked RAM (130XE or extended RAM computer)
- $E400-$FFBF OS RAM (not available on the Atari 800 computer)

The "banks of RAM available" field indicates how many banks of RAM are still available for a RAMDISK driver and/or BASIC XE extended mode. **Note that you must have at least 4 banks available for BASIC XE extended mode.** Failure to pay attention to this fact may cause your system to crash (generally at the very worst time).

Previous page **Next page**

CHKDSK Command

Index

Purpose Show volume, free/total disk space, and sector size of the selected drive (or diskette).

Syntax CHKDSK [d:]

Type Internal

Related FORMAT, MEM, VER

Remarks The CHKDSK command is used to quickly see how much space is available on a drive and the sector size (this information is not available by doing a DIRectory). Note that the volume name of all Atari DOS 2 style diskettes will appear as "AtariDOS".

You will notice that the disk write-lock status is omitted — this feature is no longer supported. We found this to be more of a hassle than it was worth, and it did not protect you from formatting the diskette. (The write-lock feature of the Multi I/O still works and is totally independent — it is a far more secure write-lock.)

The following is a sample output of the CHKDSK command.

```
      Volume: SPARTA_1 0A 25
Bytes/sector: 256
 Total bytes: 184320
  Bytes free: 123390
```

The two numbers following the volume name are used for disk change detection in cases where volume names are the same on both diskettes. The first is a random number generated when the disk was formatted. The second is a sequence number which is incremented each time a file on the disk is opened for write.

Previous page Next page

LOAD Command

Index

Purpose Loads a file (does not run). This is useful for keeping commonly used commands resident in memory, thereby eliminating the need for these commands to load from disk. If no filename is used, all files previously loaded are removed from memory.

Syntax LOAD [d:][path][fname][.ext]

Type Internal

Related MEM, SAVE

Remarks If you LOAD a standard binary load file, the results are identical to those achieved with SpartaDOS 3.2. The file is loaded into memory and not run. There are a few primary uses:

- To load MAC/65 object files into memory and then SAVE them back as continuous non-segmented binary files.
- To load a binary program prior to running a debugger (for testing purposes).

The only difference is that, if the program contains an INITAD segment, that *will* be executed.

One use of LOAD is to temporarily make external commands memory resident. This will only work with special SpartaDOS X relocatable external commands. See the MEM command for more details.

LOAD is used to:

- Keep an external command such as CAR or X, or keep the command processor (COMMAND.COM) resident in memory.
- Remove all non-installed commands or programs from memory (use LOAD with no filename).
- Load a subprogram into memory for use by other commands.

Previous page Next page

MKDIR (Make Directory) Command

Index

Purpose This command creates a subdirectory.

Syntax MKDIR [d:]path

Alias MD & CREDIR

Type Internal

Related CHDIR, RMDIR, PATH

Remarks If you do not specify a drive, the default drive is assumed. This function is not supported by the ATARIDOS.SYS driver even though subdirectories are supported by that driver.

Directories (also called subdirectories or folders) are used like file folders to organize your files. They also keep a large storage area fast. In a file cabinet it is much quicker to go to a file folder and search through a few documents, than a pile of all your documents. Computers work the same way. It is much quicker for DOS to go directly to a subdirectory and search through a few files than it is to search through one long file list.

Directory names are stored like filenames but marked with the +S attribute bit. They may not be renamed or deleted in the normal ways in which files are. To renarne a subdirectory you must copy all files from inside it to another area, then delete all the files in it, use RMDIR to delete the directory, then create a new name with CREDIR, and copy the files back to it. Otherwise, you can use RENDIR.COM from SpartaDOS Tool Kit.

```
MD TEST
MKDIR 3:>MODEM>TEST
```

The first example creates a subdirectory on the default drive called "TEST". The second example creates a subdirectory on D3: by the name of "TEST" in the subdirectory called MODEM which is in the MAIN directory.

Previous page Next page

CHDIR (Change Directory) Command

Index

Purpose This command changes the current (working) directory on the specified drive, or displays the current directory path if no path is given.

Syntax CHDIR [d:][path]

Alias CD & CWD

Type Internal

Related MKDIR, RMDIR, PATH

Remarks Directories (also called subdirectories or folders) are used like file folders to organize your files. They also make searching large storage areas (such as hard drives) much faster. In a file cabinet it is much quicker to go to a file folder and search through a few documents than a pile of all your documents. Computers work the same way. It is much quicker for DOS to go to a subdirectory and search through a few files than it is to search through one long file list. CHDIR allows you to move among your directories.

The current directory is where SpartaDOS looks to find files whose names were entered without specifying a directory. If you do not specify a drive, the default drive is assumed. If you enter the CHDIR command with no parameters, the current directory path of the current drive is displayed. (This mode is the same as the ?DIR command from SpartaDOS 3.2.)

Whenever SpartaDOS is re-initialized (i.e. RESET), the default directory on *every* drive is reset to the MAIN (root) directory. The default directory of a drive is also reset to the MAIN directory if the diskette has been changed.

This command has no effect on MYDOS diskettes even though subdirectories are supported. This is due to the fact that there is no foolproof way to detect a disk change on DOS 2 style diskettes (SpartaDOS diskettes have a volume names, a random number, and a write count for disk change detection — see "Technical Information").

Previous page

Next page

PATH (Set Search Directory) Command

Index

Purpose Causes specified directories to be searched for commands before searching the current directory.

Syntax PATH [path_string]

Type Internal

Related CAR, CHDIR, MKDIR, PROMPT, RMDIR

Remarks You may specify a list of drives and path names separated by semicolons. Then when you enter a command, SpartaDOS searches the named directories in the sequence you entered them (from path_string) before searching the current directory of the drive that was specified (or implied). The current directory is not changed after the search.

Entering PATH with no parameters causes SpartaDOS to display the current setting of the PATH string.

It is recommended that you include "CAR:" as a device in the search path as this device contains many external commands (such as X, CAR, MENU, DUMP, CHTD, etc.) that you may need. It is also good practice to use ">" or "\" at the start of a device path to force a start at the MAIN directory. The command:

```
PATH A:>;1:>DOS;CAR:
```

sets the search to the root directory of drive A (alias drive D1:), the "DOS" directory of drive 1, and the CARtridge directory.

The PATH command is really just a convenient form of the SET command, for example the above command could also be performed by:

```
SET PATH=A:>;1:>DOS;CAR:
```

The only way to clear the search path to search just the current directory (i.e no search path at all) is the command:

SET PATH

Where no path has been specified, the system defaults to:

PATH CAR:

This means search the CAR: device first, then search the current directory. The current directory will always be searched last unless it is included in the path_string. e.g.

PATH ;CAR: or PATH :;CAR:

The previous examples both mean the same thing; search current directory first, then CAR:, then current directory again. (Remember that current directory is always searched last even if it was already searched.) The stand alone ":" or a space, indicate the current directory.

NOTE: While not required, it is strongly recommended that CAR: always be the first entry in the path string. The programs in this directory are called often. If any other devices are listed first, they will always be checked before the CAR: device, slowing system response considerably.

For more information, see the SET command in chapter 5.

Previous page **Next page**

CAR Command

Index

Purpose This command enters the cartridge plugged into the top of the SpartaDOS X cartridge. If a filename is specified, then that **binary** file is loaded and run with the cartridge enabled.

Syntax CAR [/N] [d:][path][fname] [parameters]

Type External — on device CAR:

Related BASIC, COLD, SET

Remarks If no filename is given, then control is given to the cartridge plugged into the SpartaDOS X cartridge. If you do specify a filename, then that **binary** file is loaded and run with the cartridge enabled. This is useful for compiled Action! programs that need to call routines within the cartridge. The optional "parameters" are whatever the program "fname" needs. The "/N" option returns to the cartridge after running fname, instead of to the command processor which is the default.

This command is recognized by the command processor as an internal command that chains to the external program "CAR.COM", so both the CAR and BASIC commands share the same external program. CAR.COM remains memory resident while in the cartridge environment, so MEMLO will be slightly higher during this time. It will return to the lower value when the cartridge is exited.

SpartaDOS X has a MEM.SAV facility somewhat like Atari DOS 2, but with much more power. The environment variable "CAR" should be set to the file you wish to use as the memory-save file for the cartridge. If no such environment variable exists, then the memory-save feature is disabled. If this feature is disabled, then the cartridge will be entered cold (there will be no program in user memory).

The default value of the "CAR" variable is "I:>CAR.SAV". You may change this with the SET command (see the BASIC and SET commands for details on this).

If a problem is encountered when loading or saving the memory file, you will be told the problem and asked if you wish to cancel (i.e. go back to where you came). Press the <ESC> key if you wish to cancel. The two situations that can occur are as follows

- Upon entering the cartridge, the current MEMLO does not match the MEMLO in the memory-save file. This can occur after installing extra drivers since last time you entered the cartridge (such as the keyboard buffer, RAMDISK, etc), or LOADing commands such as X or COMMAND (see the LOAD command). At this point you may press <ESC> and restore the

system to the way it was when you last entered BASIC (By COLD starting and/or LOADing programs), or press <RETURN> and enter the cartridge cold. This will also happen if the memory-save file has somehow been corrupted.

- Upon exiting the cartridge (using the DOS command), the disk fills up or is not online and the memory-save file can't be saved. You have the option to go to DOS (RETURN) and lose the current data in memory, or to go back to the cartridge (ESC) and SAVE whatever you were working on or clear up the disk problem..

In addition to saving the contents of user memory, the memory-save feature saves page 0 (from $80-$FF), and page 4-6. This means that you may alternate between BASIC and CARtridge without ever loosing what you were working on. Whenever you enter the cartridge the memory-save file is loaded, thus you can edit a program in the cartridge, go to DOS, reboot the computer, and enter the cartridge with exactly what you were working on before rebooting the system (as long as the memory-save file is present and valid).

Executing a cold start while in the cartridge will cause SpartaDOS X to be disabled, while leaving the external cartridge enabled. This is the same as typing COLD /C from the command processor.

Previous page

Next page

BASIC Command

Index

Purpose This command enters the ***internal*** BASIC in your XL or XE computer (1200XL does not have internal BASIC).

Syntax BASIC [/N] [d:][path][fname] [parameters]

Type External — uses CAR.COM on device CAR:

Related CAR, SET

Remarks If no filename is given, control is given to the ***internal*** BASIC of your XL/XE computer. If you do specify a filename, the internal BASIC is enabled and the **binary** file you specified is loaded and run. The optional "parameters" are whatever the program "fname" needs. The "/N" option returns to BASIC after running "fname", instead of the command processor which is the default. To automatically load and run a BASIC program from the command processor, read the I/O Redirection Section in the "Advanced Features" chapter.

This command is recognized by the command processor as an internal command that chains to the external program "CAR.COM", so both the CAR and BASIC commands share the same external program. CAR.COM is memory resident while you are in the BASIC environment, so MEMLO will be slightly higher during this period.

SpartaDOS X has a MEM.SAV facility somewhat like that of Atari DOS 2, but much more powerful. The environment variable "BASIC" should be set to the file you wish to use as the memory-save file for BASIC. If no such environment variable exists, then the memory-save feature is disabled. If this feature is disabled, BASIC will be entered cold (there will be no program in user memory).

The default value of the "BASIC" variable is "I:>BAS.SAV". You of course may change this with the SET command, for example:

```
SET BASIC=D8:BASIC.SAV
```

sets the variable to "D8:BASIC.SAV". To see the current value of "BASIC" (and all the other environment variables) just type:

```
SET
```

and to clear the variable (i.e. disable the BASIC memory-save feature) type

SET BASIC

(See the SET command for a further explanation.)

With the memory-save feature enabled, if a problem is encountered when loading or saving the memory file (BASIC.SAV by default), an error message will appear. If this happens while loading the memory file, you will be prompted with the old MEMLO (when the file was saved). If an error exists while saving the memory file, you will be notified of that. In either case, you will have the option to abort and correct the problem or to proceed, deleting the memory file. Press the <ESC> key if you wish to abort or <RETURN> to proceed. More detail of the two situations that can occur follows:

- Upon entering BASIC, the current MEMLO does not match the MEMLO in the memory-save file. This can occur after installing extra drivers since last time you entered BASIC (such as the keyboard buffer, RAMDISK, etc), or LOADing commands such as X or COMMAND (see the LOAD command). At this point you may press <ESC> and restore the system to the way it was when you last entered BASIC (By COLD starting and/or LOADing programs), or press <RETURN> and enter BASIC cold. This will also happen if the memory-save file has somehow been corrupted.
- Upon exiting BASIC (using the DOS command), the disk fills up or is not online and the memory-save file can't be saved. You have the option to go to DOS (RETURN) and lose the current BASIC program in memory, or to go back to BASIC (ESC) and SAVE whatever you were working on or clear up the disk problem.

In addition to saving the contents of user memory, the memory-save feature saves page 0 (from $80-$FF), and pages 4-6. This means that you may alternate between BASIC and CARtridge without losing what you were working on. When you enter BASIC the memory-save file is loaded, allowing you to edit a BASIC program, go to DOS, reboot the computer, and enter BASIC with exactly what you were working on before rebooting the system (as long as the memory-save file is present and valid).

Performing a cold start (a jump to $E477) while in BASIC will cause the SpartaDOS X cartridge and the external cartridge plugged into the SpartaDOS X cartridge, if any, to be disabled. This will have the same effect as typing COLD /N from the command processor.

Previous page **Next page**

COLD Command

Index

Purpose This command reboots the system (by doing a jump through $E477).

Syntax COLD [/CN]

Type Internal

Related BOOT, CAR

Remarks This command is an alternative to switching the computer's power off and back on. The major advantage of using COLD is that the extended banks of RAM will retain their memory, thus the data in your RAMDISKS will still be there. (See the RAMDISK.SYS driver description.) This is equivalent to the SpartaDOS 3.2 command:

```
RUN E477
```

This command has two options, they are:

- C Reboot the computer with SpartaDOS X disabled and the cartridge plugged into SpartaDOS X enabled.
- N Reboot the computer as if there were no cartridges in your computer.

Hold down <OPTION> while pressing <RETURN> to reboot without internal BASIC.

Once SpartaDOS X has been disabled, it will be necessary to turn the computer off and back on to re-enable SpartaDOS X.

Previous page

Next page

SET Command

Index

Purpose To display the values of all environment variables, and optionally to set an environment variable to a specified value.

Syntax SET [var[=env_string]]

Type Internal

Remarks Environment variables are global strings that can be used for one program to communicate to another with. For example the "CAR" variable tells the CAR command where the memory-save file is. There are three forms of this command. For example the command:

```
SET
```

displays the contents of all environment variables, and:

```
SET CAR=A:CAR.SAV
```

sets the variable "CAR" to the value "A:CAR.SAV". The command:

```
SET CAR
```

deletes the environment variable "CAR" from the system. (This will cause the CAR command to not use a memory-save file.)

Previous page

Next page

POKE Command

Index

Purpose To change the contents of a memory location.

Syntax POKE [$]location [$]value

Type Internal

Related PEEK

Remarks POKE allows you to change memory locations from the command processor which can be useful in batch files and other applications. It is very easy to crash the system with this one if you do not understand what you are doing. A few examples of POKE locations and useful values follow:

POKE 77 (atract) 0=attract mode off for a few minutes
POKE 82 (lmargin) n=number from 0 to 39 for left margin
POKE 83 (rmargin) n=number from 0 to 39 for right margin
POKE 559 (sdmctl) 0=screen off, 34=screen back on
POKE 710 (color2) 0=black, 53=red, 148=blue
POKE 730 (keyrep) 1=hyper, 3=fast, 5=normal (XL/XE only)
POKE 731 (noclik) 0=normal, 1=speaker off (XL/XE only)
POKE 752 (crsinh) 0=cursor off, 1=cursor on
POKE 702 (shflok) 0=lower case, 64=upper case
POKE 65 (soundr) 0=SIO sound off, 1=SIO sound on

A good memory map will give many more and is a must for programming the Atari.

Previous page

Next page

RS232 (Load RS232 Driver) Command

Index

Purpose This command loads the RS232 handler from a P:R: Connection or the Atari 850 interface.

Syntax RS232

Type External — on device CAR:

Remarks You need to use this command prior to using a P:R: Connection or Atari 850 interface **unless** the program you are going to use does this automatically. Try your program without RS232 first. You should hear a beep on your monitor (TV) speaker if the handler loads. If not and an error occurs, type this command and run your program again.

Avoid loading the RS232 handler more than once. Your system may crash if you load several copies of the RS232 handler into memory, since MEMLO is raised each time.

Next page

Previous page

C h a p t e r 5 **Index**

The Command Processor — Advanced Features

The SpartaDOS X command processor has been greatly enhanced with more sophisticated batch files, command line I/O redirection, user definable prompts, command search paths, and more.

This chapter discusses these features and gives many examples. Most of these features are either new to SpartaDOS or have been greatly enhanced over prior versions of SpartaDOS.

Previous page **Next page**

X Command

Index

Purpose Execute a program which requires that no cartridges are installed (such as DISKRX, EXPRESS, most binary files, etc.)

Syntax X [d:][path]fname[.ext] [parameters]

Type External — on device CAR:

Remarks There are four possible environments a command can run under. They are:

1. with internal BASIC present (via BASIC command)
2. with a CARtridge present (via CAR command)
3. with the SpartaDOS X library enabled (just type the program or command name)
4. with no cartridges present (via the X command)

The first three modes use the SpartaDOS X library to perform various DOS functions, including loading and running the command. The fourth mode, however, cannot use the library without moving or disabling the screen! Thus, the following features are disabled when commands are run with this command:

- The search path is not used — you must specify the exact location of the file if it is not in the current directory of the current drive.
- The mini-buffers are not used — single byte get/put will be *very* slow (this is extremely ram since most programs that use single get byte and put byte are in BASIC or use a cartridge)
- Since the library is disabled, only standard binary files are loadable — SpartaDOS external commands such as FIND and can not be run.
- I/O redirection is severely hampered because it must use the library. In doing this the screen will flicker rapidly.

The general rule of thumb is: "If a program does not work with a cartridge installed, prefix the command with the X command, otherwise, just type the command."

X.COM remains resident in memory while the program called is running, so MEMLO will be slightly higher until the program is exited to the command processor.

Executing a cold start (a jump to $E477) while using X.COM will disable the SpartaDOS X cartridge and the external cartridge, if present.

Previous page

Next page

The Command Processor — Advanced Features

Index

Batch Files Batch files are simply a list of SpartaDOS commands that may be fed to the command processor from a text file. Parameters may be passed to the batch file by including them on the command line following the batch file name. The syntax is

```
-fname [parm1 parm2 ... parm9]
```

The filename ("fname") is assumed to have an extension of ".BAT" although this assumption may be overridden by including an extension on the command line. The parameters ("parm") are optional.

The following is an example of a batch file which accepts two files on input and creates an output file containing the contents of both source files (we will call this file "TEST.BAT").

```
COPY %1 %3

Copy %2 %3/A
```

Now, the command

```
-TEST FILE1 FILE2 OUTPUT
```

will concatenate the files "FILE1" and "FILE2" to the file "OUTPUT".

You may pass up to 9 parameters (numbered "%1" to "%9") to a batch file. The parameter "%0" is the name of the batch file (in the above case, this would be "TEST"). The '%' parameters may appear anywhere in the batch file and may be surrounded in text (i.e. no spaces need precede the '%' character).

The batch file parameters are automatically saved in environment variables "_x1" where 'x' is the parameter number. Due to the command processor's non-resident nature, the parameters need to be saved somewhere — environment variables are a perfect place. This also means that the number and size of parameters is limited to a total of 256 characters less overhead (the "_x1=" string) and space used by other environment variables.

Previous page Next page

PEEK Command

Index

Purpose To examine a memory location or perform a HEX conversion.

Syntax PEEK [$]location

Type Internal

Related POKE

Remarks PEEK allows examination of a memory location from the command processor. It is also useful as a quick DEC to HEX or HEX to DEC convertor. (DEC means decimal or base 10; HEX means hexadecimal or base 16.) PEEK returns the dec and hex value of the location entered, the contents of the location in both dec and hex, the dec and hex value of the memory word stored in the location and location+1, and the ATASCII character representing the value of the location.

It is a good idea to PEEK a location before POKEing it if you are not sure what you are doing. You can usually recover by POKEing the old value back in (unless the computer has crashed).

Previous page Next page

CHTD (Change Time/Date Stamp) Command

Index

Purpose This command changes the time/date stamp on all files matching the given filespec to the current time and date.

Syntax CHTD [+A|H|P|S] [-A|H|P|S] [d:][path]fname[.ext]

Type External — on device CAR:

Related DATE, TD, TIME

Remarks By default, this command will only change the time/date stamp on non-hidden and non-protected files — this may be overridden. (See ATR command for more information on attributes.) You must enter a filespec since "*.*" is not assumed.

Previous page Next page

TD (Time/Date Display) Command

Index

Purpose This command allows you to turn on and off a time/date display line on top of your screen.

Syntax TD ON|OFF

Type External — on device CAR:

Related CHTD, DATE, TIME

Remarks This command is much like the KEY command in the way it links into your computer.

You must have either the JIFFY.SYS or CLOCK.SYS driver installed before you can use this command. It calls one of these drivers directly (through the I_GETTD symbol) and will not load without one of these drivers. (These drivers are loaded by default unless you are using your own CONFIG.SYS file.)

NOTE: TD ON may be incompatible with some programs. If you are having problems with a program, try TD OFF, or do not install it at all.

Previous page

Next page

SAVE Command

Index

Purpose This command saves binary data from memory to disk.

Syntax SAVE [d:][path]fname[.ext] [$]address [$]address

Type Internal

Related LOAD

Remarks Addresses are assumed to be decimal unless preceded by a '$' (which indicates HEX). This is useful when used in conjunction with the LOAD command for desegmenting MAC/65 files or to save a snapshot of memory for debugging purposes.

Previous page

Next page

SWAP (Swap Drives) Command

Index

Purpose This command allows you to swap (re-map) your drive configuration.

Syntax SWAP [d,d]

Type Internal

Remarks SWAP, without any parameters, will display the drive map list showing drives 1 through 9. The default is 1=1, 2=2, etc.

For example, to interchange drives 1 and 9, type the following command:

```
SWAP 1, 9     (or)
SWAP A, I
```

The order is not important, so 1,9 is equivalent to 9,1.

The drives will stay mapped that way unless remapped or a COLD start occurs. Note that you may use letters or numbers to reference a drive and that no colon (":") follows the drive specifier.

SWAP works in *addition* to Multi I/O drive remapping, so take that into consideration when using the Multi I/O. It is very easy to lose track of which floppy, RAMDISK, or hard drive partition is at which logical drive.

Previous page

Next page

VERIFY Command

Index

Purpose To turn write verify on or off.

Syntax VERIFY ON|OFF

Type Internal

Remarks When ON, SpartaDOS performs a verify operation following each disk write operation, to verify that the data just written can be read without error. This only applies to floppy drives. Because of the extra time required to perform the verification, the system runs much slower when programs write data to disk. The default is OFF — this command is typically used when you are having floppy drive problems.

Previous page

Next page

The Command Processor — Advanced Features

Index

Default Batch File

When the command processor is first entered, it tries to run a batch file called "AUTOEXEC.BAT". You should put any system setup commands that you may need in this file.

The variable "BATCH" will be read by the command processor just before it prints its prompt. If this variable exists and contains a filename, this file will be executed as a batch file. As soon as the command processor reads the variable, it will delete it from the list of environment variables. This is how the system passes the "AUTOEXEC" filename to the command processor when it is loaded for the first time. The variable BATCH, then, can be used to cause a batch file with a different name to be executed on boot by using the line

```
SET BATCH=d:filename.BAT
```

in CONFIG.SYS.

Previous page Next page

The Command Processor — Advanced Features

Index

I/O Redirection

You may redirect the standard input and output of SpartaDOS X commands on the command line. With SpartaDOS 3.2, redirection was done using batch files (for input redirection) and the PRINT command (for output redirection). SpartaDOS X implements I/O redirection in a totally different manner. Batch files are no longer considered as "input redirection" — they are *only* read by the command processor and are **not** system wide (i.e. you may not feed input to BASIC through a batch file). The PRINT command has been eliminated.

With SpartaDOS X, you may divert output of a single command by including ">>d:fname" on the command line. Similarly, input redirection is accomplished by including a "<<d:fname" on the command line. For example, the command

```
DIR >>PRN:
```

redirects the output from the DIR command to the printer (the directory listing will not appear on the screen). An alternate way to copy a file would be

```
TYPE fname >>dest
```

This will be slower than the copy command and will not copy the date to the new file.

```
BASIC <<AUTOGO
```

will run the BASIC program "START.BAS" if the text file "AUTOGO" contains the line

```
RUN "D:START.BAS"
```

As an example of output redirection, the following batch file will allow paged viewing of the output of any program by redirecting it to a temporary file, then TYPEing the temporary file with the pause option:

```
%1 %2 %3 %4 %5 %6 %7 %8 %9 >>TEMP
TYPE TEMP /P
PAUSE
DEL TEMP
```

For example, if you named the above batch file MORE.BAT and wanted to read the directions

printed by the ARC program, use

-MORE ARC

Previous page **Next page**

The Command Processor — Advanced Features

Index

Search Path

Whenever a command is given to the command processor without a drive and/or path being specified, a check is made to see if it is an internal command (such as ERASE). If not, the list of installed external commands (such as TD or KEY after they have been run once) is searched. If the command is not found, then a check is made to see if the environment variable PATH exists. If it does, all of the devices and/or paths named in the variable are checked for the command in the order specified. If the command is still not found, the default directory is searched. The PATH variable provides a high degree of flexibility and power, allowing you to keep often used utilities out of the way in subdirectories. This is particularly useful if you own a hard drive, since the main directory of D1: can get very cluttered.

You can examine the PATH variable by typing

```
PATH
```

with no parameters at the CP prompt. The default PATH is

```
CAR:
```

The search path may be changed by typing

```
PATH path1;path2;...;pathn
```

at the CP prompt. Each device and/or path specified must be separated from the others with a semicolon (;). It is a good idea to leave CAR: as the first entry, since many often used commands are in this device and it can be accessed much more rapidly than the others. Order is important, since the entries will be searched one after the other. For example,

```
PATH CAR:;A:\DOS\;A:\TOOLKIT\;D9:>;A:>;:>
```

will search the CAR: device, the directory DOS on D1:, the directory TOOLKIT on D1:, the RAMDISK at D9:, the main directory of D1:, the main directory of the default drive, and then the current directory of the default drive.

If a path is specified on the command line, this search path will *not* be used.

Previous page

Next page

C h a p t e r 6 **Index**

SpartaDOS X Functions from BASIC

Many features of SpartaDOS may be accessed in BASIC, Action!, machine language, and other programming environments. The following is a list of common BASIC functions and XIO statements that allow the programmer to accomplish a variety of tasks. Conversion to other languages should not be difficult; refer to the reference material for that language for details.

In this list, *IOCB* refers to an Input/Output Control Block (or channel) number from 0 to 7. IOCB #0 is used by the Atari operating system for the screen editor, so it should normally not be used. An *Atari DOS disk* is one initialized in standard Atari DOS 2 format, whether in single, enhanced (dual), or double density, as produced by Atari DOS 2.0S and 2.5, MYDOS, other DOS 2 clones, and the SpartaDOS X Formatter when used in Atari DOS mode. *d:*, *path*, and *fname.ext* refer to any legal SpartaDOS X device identifier, pathname, and filename with extension as defined in chapter 4.

Notes on the Default Drive

Please remember that D: *from BASIC or another language refers to the* default drive, *not necessarily drive #1*. From the command processor, D: refers to drive #4. With most other DOS types including earlier versions of SpartaDOS, *D:* represents D1:.

```
OPEN #1,4,0,"D:TEST.TXT"
```

will open the file TEST.TXT on the *default drive*, not necessarily drive #1, for read under SpartaDOS X.

Accessing the "Kernel" Through CIO

The D: device available through the CIO with SpartaDOS X is not just the disk drive handler; it is the handler for the SpartaDOS "kernel". Any "kernel" device may be accessed through the CIO from any application by preceding its name with D. For example,

```
OPEN #3,8,0,"DPRN:"
```

will open the printer for output. This also means that D4:, DD:, DD4:, DDD:, DDSK4:, and DDSKD: all refer to drive #4. When referring to a device other than a disk drive or the CAR: device, the fname.ext part of the syntax is ignored. If this confuses you just ignore it and use D1: - D9:, E:, P:, R:, and so on as you would with any other DOS.

Previous page **Next page**

Open File

Index

Purpose To open a disk file for access through SpartaDOS X.

Syntax OPEN #IOCB,aux1,aux2,"Dd:[path]fname.ext"

Remarks This command opens a disk file through SpartaDOS X. **Aux1** is the mode (output, input, update, directory, etc.) in which the file will be opened. The following is a list of legal values for **aux1**. Unless otherwise noted, **aux2** should be 0.

4 Open the file in read only mode.

6 Open a formatted directory. Provides a directory listing as do the DIR and DIRS commands from the command processor. **Aux2** is used to determine the style of the directory. If **aux2** is 0, standard DIRS format will be used. If **aux2** is 128, the long DIR format, including size in bytes, date, and time, will be used.

8 Open the file in write only mode.

9 Open the file in append mode. Data will be written to the end of an existing file. If the file does not exist it will be created.

12 Open the file in update mode. This mode allows both reading from and writing to a file. Note: On a SpartaDOS format disk it is possible to position and/or write past the end of a file while in update mode.

An Example This short BASIC program will read the formatted directory of a disk in drive #1 in long format and print it to the screen:

```
10 DIM ENTRY$(40)
20 OPEN #1,6,128,"D1:*.*"
30 REM The TRAP will cause the program to jump to line
40 REM 80 when the end of the directory is reached.
50 TRAP 80
60 INPUT #1,ENTRY$:PRINT ENTRY$
70 GOTO 60
80 CLOSE #1
```

Accessing the Raw Directory Setting bit 4 of **aux1** puts the OPEN in raw or unformatted directory mode. This allows you to read from and/or write to SpartaDOS directories as if they were normal data files. Although this is much faster than reading a formatted directory, there is no easier way to trash a disk and make it unusable than to make a mistake in the raw directory. Unless you feel confident about what you are doing and are using a disk you don't mind losing, stay away from the raw directories! This mode *will* work with Atari DOS disks if the ATARIDOS.SYS driver is installed. The driver translates the Atari directory format into SpartaDOS format and back.

Scan Mode Adding 64 to **aux1** will place the OPEN in attribute scan mode. **Aux2** is used to determine the attributes desired. If a long directory is wanted in scan mode, then 128 should be added to **aux1** instead of to **aux2**.

To determine the file attributes to be scanned, the following values should be added to **aux2**, assuming an initial value of 0:

Protected	add 1	Unprotected	add 16
Hidden	add 2	Not hidden	add 32
Archived	add 4	Not archived	add 64
Subdirectory	add 8	Not a subdirectory	add 128

Only those files that fit the requested description will be referenced. A value of 0 in **aux2** will ignore all attributes, even "Hidden".

For example, to get a long format directory of only the hidden files in the BASIC example on the previous page, simply substitute the following line:

```
20 OPEN #1,6+64+128,2,"D1:*.*"
```

For a short directory listing without subdirectories, use

```
20 OPEN #1,6+64,128,"D1:*.*"
```

and, finally, for a long listing of the unhidden, protected entries that end with a .COM extender, use

```
20 OPEN #1,6+64+128,1+32,"D1:*.COM"
```

It is possible to select contradictory conditions (such as 1+16, protected and not protected) for each of the attributes. This will not produce an error but, since no directory entry can match both conditions, will always select no files.

Previous page **Next page**

Rename File(s) (RENAME)

Index

Purpose To change the name of a file or group of files.

Syntax XIO 32, #IOCB,0,0,"Dd:[path]fname1.ext fname2.ext"

Remarks The name of the file or names of the files specified by **fname1.ext** will be changed to **fname2.ext**, exactly as with the RENAME command from the command processor. The IOCB selected should be closed for this operation. Wildcards may be used in both file name specifications. **WARNING!** SpartaDOS X has an extremely powerful RENAME function. It is possible to give two or more files on one disk the same name. It will then be impossible to refer to one file without referring to the other(s). For a few verbose ways to recover from duplicate file names, refer to the RENAME command remarks in chapter 4.

Previous page Next page

Erase File(s) (ERASE)

Index

Purpose To remove unwanted files from a disk.

Syntax XIO 33, #IOCB,0,0,"Dd:[path]fname.ext"

Remarks The file or files specified will be erased from the disk. The IOCB selected should be closed for this operation. Wildcards may be used. While it is possible to recover erased files in some instances (see the command UNERASE in chapter 4), it is important to be very careful with this command.

Previous page Next page

Protect File(s) (ATR +P)

Index

Purpose To prevent a file or files from being changed or erased.

Syntax XIO 35, #IOCB,0,0,"Dd:[path]fname.ext"

Remarks This will allow the specified files to be opened in read mode only. Wildcards may be used. The IOCB should be closed. Protected files may not be erased, changed, overwritten, or renamed.

Previous page Next page

Unprotect File(s) (ATR -P)

Index

Purpose To allow files previously protected to be changed or erased.

Syntax XIO 36, #IOCB,0,0,"Dd:[path]fname.ext"

Remarks This removes the protected status of selected files previously protected with the ATR command or the Protect Files XIO command above. The file or files may now be erased, renamed, or changed. Wildcards may be used. The IOCB should be closed.

Previous page Next page

Set File Position — POINT

Index

Purpose To allow direct access to specific points within a disk file (or past the end of a file if necessary).

Syntax

```
X=POS
Y=0 (see text)
POINT #IOCB,X,Y
```

or

```
A=INT(POS/65536)
B=INT((POS-A*65536)/256)
C=POS-A*65536-B*256
POKE 844+IOCB*16,C
POKE 845+IOCB*16,B
POKE 846+IOCB*16,A
XIO 37, #IOCB,aux1,aux2,"Dd:"
```

Remarks Unlike Atari DOS and compatibles, which use an absolute physical disk position (sector and offset into sector) for the NOTE and POINT functions, SpartaDOS X uses a relative position within the file. POS is the desired offset into the currently open file. For example, if POS was 612, the next GET from the file would get the 613th byte of the file. This value will refer to the same position in the file even if the file is physically moved to another disk. The file must be open for this operation.

Because of a limitation in Atari BASIC, BASIC XL, and BASIC XE, the first method shown, using the POINT command, will only work with positions up to and including 32767. If a value greater than 32767 is given a BASIC error will occur. To POINT to a greater location with these languages (and possibly others) it is necessary to use the second method. The POINT command is bypassed by poking the three byte file position directly into the IOCB registers and executing the XIO. **Aux1** and **aux2** must be the values used when the file was opened.

Other languages, such as Action! and Turbo BASIC XL, have no such limitation on the POINT command, allowing it to be used instead of the lengthy XIO method. In this case, use the following format:

```
Y=INT(POS/65536)
X=POS-Y*65536
POINT #IOCB,X,Y
```

If you are a user of an earlier version of SpartaDOS, you should notice that NOTE and POINT now work the same way with Atari DOS disks as they do with SpartaDOS disks. POINT will *not* use sector number and offset regardless of disk format.

Using NOTE and POINT with SpartaDOS X and an Atari DOS disk may prove to be time consuming since, to determine the relative offset into the file, it is necessary to read the file from the beginning every time a POINT is used. This also causes segmented binary files to take much longer to load from Atari DOS disks than from SpartaDOS disks. NOTE and POINT tables created by other DOS types (including earlier versions of SpartaDOS for files on *Atari DOS disks* will, of course, no longer be valid.

Sparse Files On a SpartaDOS diskette, it is possible to point past the end of a file opened in append mode. When data is placed in a file past the end, the file is given the new length, but no physical sectors are used for the space between the old and the new data. In the sector map of the file, the unallocated sectors are represented by a sector number of 0. Should you at any time write to a position in this gap, a sector will be allocated. This gap may not be read, and a file containing gaps may not be copied. An error will occur if either of these is attempted.

Previous page **Next page**

Get Current File Position — NOTE

Index

Purpose To determine the current position within a file.

Syntax NOTE #IOCB, X, Y
POS=X+65536*Y

Remarks This will return the relative current position within the currently open file; i. e., the offset into the current file. This will *not* return sector number and offset into the sector, regardless of disk format. The file must be opened for this operation.

For users of SpartaDOS 2.x and 3.x, you may be interested to learn that this method works with those versions. The XIO 38 command described in the SpartaDOS Construction Set manual will still work but is unnecessary.

Previous page Next page

Get File Length

Index

Purpose To determine the length of the currently open file.

Syntax
```
XIO 39, #IOCB,aux1,aux2,"Dd:"
A=PEEK(844+IOCB*16)
B=PEEK(845+IOCB*16)
C=PEEK(846+IOCB*16)
LNGTH=A+B*256+C*65536
```

Remarks This will return the length of the currently open file. **IOCB**, **aux1**, and **aux2** should be the same values used when opening the file.

Previous page Next page

Load a Binary File (LOAD)

Index

Purpose To load and execute a binary file from another program.

Syntax XIO 40, #IOCB,4,0,"Dd:[path]fname.ext"

Remarks This command will load a binary file and execute it using the INIT and/or RUN vectors. The IOCB should be closed. Loading a binary file from an AtariDOS disk will take much longer than loading the same file from a SpartaDOS format disk.

Previous page Next page

Create a Directory (MKDIR)

Index

Purpose To create a new subdirectory.

Syntax XIO 42, #IOCB,0,0,"Dd:[path]newdir"

Remarks The directory **"newdir"** is the directory that will be created. Any path before this must be valid. For example, if

XIO 42,#1,0,0,"D1:LARRY>MOE>CURLY>SHEMP"

is used, then the path "LARRY>MOE>CURLY>'" must already exist from the current directory, and the directory "SHEMP" will be created.

The IOCB should be closed for this operation. This will ONLY work for SpartaDOS format disks.

Previous page Next page

Delete a Directory (RMDIR)

Index

Purpose To remove an existing directory.

Syntax XIO 43, #IOCB,0,0,"Dd:[path]olddir"

Remarks The directory **olddir** will be deleted. A directory must be empty to be deleted. The rules regarding path and IOCB status defined in XIO 42 apply here.

Previous page Next page

Change Current Directory (CHDIR)

Index

Purpose To change the current working directory of a disk.

Syntax XIO 44, #IOCB,0,0,"Dd:path"

Remarks This will change the directory that is used when the specified drive is accessed without reference to a specific directory. The rules regarding path and IOCB status defined in XIO 42 apply here.

Next page

Previous page

Set Boot File (BOOT)

Index

Purpose To establish the file that will be loaded when the computer is initialized when SpartaDOS X is not used.

Syntax XIO 45, #IOCB,0,0,'Dd:[path]fname.ext"

Remarks This will cause the specified file to load when the computer is turned on or cold started and the SpartaDOS X cartridge is not used. With earlier versions of SpartaDOS, the primary use of this was to cause the *.DOS file to be booted. With SpartaDOS X, the uses of this command are limited. The IOCB should be closed, and a SpartaDOS format disk must be used. **Note:** BOOT will not work with all binary files. There are many specific rules that must be followed when loading a file without DOS. The primary purpose of this command is to load a DOS module.

Previous page Next page

Set Attributes (ATR)

Index

Purpose To manipulate the protected, hidden, and archived status of files.

Syntax XIO 49, #IOCB,aux1,aux2,"Dd:fname.ext"

Remarks This is used to modify the attributes of a file or files. Wildcards are allowed in **fname.ext**. **Aux1** is used to select which attributes are to be changed and whether they will be set or cleared. **Aux2** is used to determine which files are to be affected.

To perform the desired attribute modification, add the following values to **aux1**, assuming an initial value of zero:

Protect	add 1	Unprotect	add 16
Hide	add 2	Unhide	add 32
Set archive	add 4	Clear archive	add 64

Aux2 is used exactly as with the scan mode of the open statement. It will select the files to be affected by current attribute status. These values should be added to **aux2**, starting with a base value of 0:

Protected	add 1	Unprotected	add 16
Hidden	add 2	Not hidden	add 32
Archived	add 4	Not archived	add 64
Subdirectory	add 8	Not a subdirectory	add 128

For example, to hide all of the files on drive #1 with a .BAK extender, use

```
XIO 49,#1,2,0,"D1:*.BAK"
```

To protect and set the archive bit for all of the hidden files on drive #1, use

```
XIO 49,#1,1+4,2,"D1:*.*"
```

and to unhide and unprotect all the hidden files with a *.BAK extender on drive #1, use

```
XIO 49,#1,16+32,2,"D1:*.BAK"
```

The IOCB should be closed for this operation.

Previous page **Next page**

Format a Disk (FORMAT)

Index

Purpose To initialize a disk, setting up the appropriate track, sector, and directory data.

Syntax XIO 254, #IOCB,0,0,"Dd:"

Remarks The Dd: specified is irrelevant, since this command will bring up the SpartaDOS X disk formatter menu. From this menu disk number, format, size, skew, etc. may be selected. Once the formatter is exited with the <ESC> key, control will be returned to the program. This allows disks of all types to be formatted from within any program. The IOCB should be closed for this operation. **WARNING!** Formatting a disk will destroy all existing data on the disk. Hiding or protecting a file will not save it from being destroyed during a disk format.

The next two commands are not available through XIO calls. They must be accessed directly through the CIO. An assembly language listing of a routine to access these will follow, along with a BASIC program that demonstrates its use.

Previous page Next page

Get Disk Information (CHKDSK)

Index

Purpose To read information about a disk.

CIO Data

iccom	=	47
icbal	=	low byte of 'Dd:' address
icbah	=	high byte of 'Dd:' address
icbll	=	low byte of buffer address
icblh	=	high byte of buffer address

CIO Output Results

buffer	=	results of CHKDSK operation (17 bytes)
+0	=	version number of disk, 0 if Atari DOS format
+1	=	number of bytes per sector, 0 if 256
+2	=	total number of sectors on disk (2 bytes)
+4	=	Number of free sectors on disk (2 bytes)
+6	=	volume name, always "AtariDOS" for Atari DOS format disks (8 bytes)
+14	=	volume sequence number, 0 if Atari DOS format
+15	=	volume random number, 0 if Atari DOS format

Previous page Next page

Get Current Directory Path (CHDIR)

Index

Purpose To get the path from the root directory to the current directory of a drive.

CIO Data

iccom	=	48
icbal	=	low byte of 'Dd:[path]' address
icbah	=	high byte of 'Dd:[path]' address
icbll	=	low byte of buffer address
icblh	=	high byte of buffer address

An Example The following is a short BASIC program demonstrating the use of the last two CIO calls. It is followed by an assembly language listing of the code contained in the DATA statements and later in the string CIO$.

```
10 DIM CIO$(32),BUFFER$(64),DRIVE$(4),CHKDSK(17)
20 DRIVE$="Dl: ":DRIVE$(4)=CHR$(155)
30 RESTORE 50
40 FOR X=1 TO 32:READ Y:CIO$(X)=CHR$(Y):NEXT X
50 DATA 104,104,104,10,10,10,10,170,104,104,157,66,3
60 DATA 104,157,69,3,104,157,68,3,104,157,73,3,104,157
70 DATA 72,3,76,86,228
80 REM MAIN LOOP
90 BUFFER$(1)=CHR$(0):BUFFER$(64)=CHR$(0)
100 BUFFER$(2)=BUFFER$
110 ?:?"CIO Call Demonstrator"
120 ?:?"1 -> CHKDSK"
130 ?:?"2 -> Path to current directory"
140 INPUT CHOICE
150 IF CHOICE<>1 AND CHOICE<>2 THEN GOTO 120
160 ICCOM=CHOICE+46
170 ?:?"Which drive";:INPUT D
180 D=INT(D):IF D<l OR D>9 THEN GOTO 170
190 DRIVE$(2,2)=STR$(D):IOCB=l
200 X=USR(ADR(CIO$),IOCB,ICCOM,ADR(DRIVE$),ADR (BUFFER$) )
210 IF CHOICE=1 THEN GOTO 270
220 IF BUFFER$(1,1)=CHR$(0) THEN ?"Root directory":GOTO 80
230 FOR X=1 TO LEN(BUFFER$)
```

```
240 IF BUFFER$(X,X)=CHR$(0) THEN
BUFFER$(X)=">":BUFFER$=BUFFER$(1,X):POP:GOTO 260
250 NEXT X
260 ? BUFFER$:GOTO 80
270 FOR X=1 TO 17:Y=ASC(BUFFER$(X,X)):CHKDSK(X-1)-Y:NEXT X
280 ?"      Volume: "; BUFFER$(7,14)
290 ?"Bytes/sector: ";
300 IF CHKDSK(1)=0 THEN CHKDSK(1)=256
310 ? CHKDSK(1)
320 ?" Total bytes: "
330 ? CHKDSK(1)*(CHKDSK(2)+256*CHKDSK(3))
340 ?"  Bytes free: ";
350 ? CHKDSK(1)*(CHKDSK(4)+256*CHKDSK(5))
360 GOTO 80


;origin is arbitrary since it will be in a string
ciov .equ $E456
iccom .equ $0342
icbal .equ $0344
icbah .equ $0345
icbll .equ $0348
icblh .equ $0349
         *=$5000         ; or whatever
         pla             ; number of arguments.
         pla             ; should be 0
         pla             ; iocb channel number
         asl a           ; multiply by 16 for
         asl a           ; proper IOCB form
         asl a
         asl a
         tax             ; in x where it belongs
         pla             ; 0 again
         pla             ; command number
         sta iccom,x
         pla             ; address of "Dx:" string
         sta icbah,x
         pla
         sta icbal,x
         pla             ; buffer address
         sta icblh,x
         pla
         sta icbll,x
         jmp ciov        ; all done.  Jump CIO
```

Previous page Next page

SpartaDOS User Accessible Data Table

Index

Several SpartaDOS variables have been made available to programmers to allow easy access to the command line for applications and utilities. This data table is referred to as COMTAB and is pointed to by the OS variable DOSVEC at memory location 10 ($0A). An assembly language example will follow as an aid. This table is valid with all versions of SpartaDOS except where noted. Locations COMTAB, ZCRNAME, BUFOFF, COMFNAM, and LBUF are also supported by OS/A+ and DOS XL.

DECOUT COMTAB-19

SpartaDOS X only. Contains the right justified, space padded output of the "misc_convdc" routine, an ASCII string representation of the three byte number at DIVEND (see Page Seven "Kernel" Values). (8 bytes)

LSIO COMTAB-10

This is a pointer to the SpartaDOS high speed SIO routine. You can use the address contained here instead of $E459, the OS SIOV, to perform high speed sector I/O with your programs.

DIVEND COMTAB-6

SpartaDOS X only. A three byte number here will be converted by the "misc_convdc" routine to a string at DECOUT (see Page Seven "Kernel" Values).

WRTCMD COMTAB-2

This location contains the SIO write command. A 'W' here indicates write with verify, while a 'P' indicates write without verify.

COMTAB COMTAB+0

This is a 6502 jump instruction followed by the address of the DOS entry routine. A jump here enters DOS.

ZCRNAME COMTAB+3

This is a 6502 jump instruction followed by the address of the file name crunch routine. This location is used to interpret the command line. A jump here will pull the next command from LBUF, translate the drive or device identifier if one is given (i.e., A: to D1:), add the default drive identifier if none is given, and place the result at COMFNAM. Each call will advance BUFOFF to point to the next entry on the command line, so that each call to the crunch routine will get the next entry on the line. If there are no entries remaining, the 6502 zero flag will be SET on return. Since the 6502 has no indirect jsr, it is necessary to use a few lines of code to access this routine. An example will follow this list.

BUFOFF COMTAB+10

The offset into LBUF where the next parameter to be read is located. This can be manipulated to reread the command line.

DATER COMTAB+13

The date in DD/MM/YY format (3 bytes). Updated by VGETTD. Updated continuously while the Time/Date line is on with SpartaDOS X.

TIMER COMTAB+16

The time in HH/MM/SS format (3 bytes). Updated by VGETTD. Updated continuously while the Time/Date line is on.

_800FLG COMTAB+27

SpartaDOS X only. $FF if the computer is an Atari 800. Zero otherwise.

NBANKS COMTAB+29

SpartaDOS X only. The number of expansion memory banks free. This is the same number shown with the MEM command.

BANKFLG COMTAB+30

SpartaDOS X only. $FF if USEing BANKED. Zero otherwise.

OSRMFLG COMTAB+31

SpartaDOS X only. $FF if USEing OSRAM. Zero otherwise.
Note: USE NONE is indicated by both BANKFLG and OSRMFLG being zero.

COMFNAM COMTAB+33

This is the destination buffer for the ZCRNAME routine. It will ALWAYS begin with a Dd: since the default drive is added if none is given. If you are looking for switches or other options, start looking at COMFNAM+3. This buffer is 28 bytes long.

LBUF COMTAB+63

This is the input buffer for the command processor. The entire command line is stored here. LBUF is 64 bytes long.

COPYBUF COMTAB+127

This is the main buffer used by the SpartaDOS X "kernel".

An Example The following assembly language program demonstrates one way to read the SpartaDOS command line. It simply echoes the command line with the drive specifications added or translated as necessary. It resets BUFOFF to 0 so that the name of the command is printed, too.

```
; CIO and IOCB equates
ciov .equ $E456
iccom .equ $0342
icbal .equ $0344
icbah .equ $0345
icbll .equ $0348
icblh .equ $0349
write .equ $09
; SpartaDOS equates
comtab .equ 10
zcrname .equ 3
bufoff .equ 10
comfnam .equ 33
; The program.
        *=$4000         ; or wherever.
init                    ; patches our crunch routine to
        ldy #zcrname+2  ; be the same as the COMTAB one.
        ldx #2
loop1
        lda (comtab),y
        sta crunch,x
        dey
        dex
        bpl loop1
; zero BUFOFF
        lda #0
        ldy #bufoff
        sta (comtab),y
mainloop
        jsr crunch      ; get next command line entry.
        beq exit        ; quit if there are no more.
; Set up for CIO print of data at COMFNAM
        ldx #0          ; IOCB #0 (E:)
        lda #63         ; set buffer length for max
        sta icbll,x
        lda #0
        sta icblh,x
        lda comtab      ; store COMTAB+33 at icba
        clc
        adc #comfnam
        sta icbal,x
        lda comtab+1
        adc #0
        sta icbah,x
```

```
        lda #write      ; 'print string' command
        sta iccom,x
        jsr ciov        ; print it.
        jmp mainloop
exit
        rts
crunch
        jmp $FFFF       ; will be changed by INIT routine
        *=$02E0
        .word init      ;run vector
```

Previous page **Next page**

Vectors Under the OS ROM

Index

The following vectors are only available on XL/XE computers. They reside under the Operating System ROM and will be invalidated by any program using this space, such as Turbo BASIC XL. It would be a good idea to check _800FLG to be sure it is an XL/XE computer and to check each vector before access to be sure that it is still there.

Since these vectors are under the OS ROM, it is necessary to enable the RAM instead of the ROM in this memory area. One possible method follows:

```
lda $D301   ; PIA, responsible for bank
pha         ; selecting. Store status.
and #$FE    ; RESET bit 0.
sta $D301   ; enable RAM under OS ROM.
jsr VGETTD  ; call the vector.
pla
sta $D301   ;restore PIA to previous state.
```

Those functions each contain a jump (JMP) followed by the address of the function. It is a good idea to always check for this JMP before assuming that the vector is still there.

VGETTD $FFC0

Returns the current time and date to COMTAB locations TIMER and DATER.

VSETTD $FFC3

Sets the current time and date to the values held in COMTAB locations TIMER and DATER.

VTDON $FFC6

Turns the time/date display on and off. If the 6502 Y register contains a 1 on entry, the time/date line will be turned on. If it contains a 0 on entry, the time/date line will be turned off. The 6502 carry flag will be set on return if the operation failed (TD.COM had not been run to install this routine).

VFMTTD $FFC9

Returns a formatted time/date line to a specified buffer. On entry, the 6502 x register should hold the high byte of the buffer address. The Y register should hold the low byte of the buffer address. The 6502 carry flag will be set on return if the operation failed (TD.COM had not been run to install this routine).

VXCOMLI $FFD2

Will cause the line contained in COMTAB buffer LBUF to be executed. BUFOFF should be 0 on entry.

VKEYON $FFDB

Turns the key buffer on and off. If the 6502 Y register contains a 1 on entry, the key buffer line will be turned on. If it contains a 0 on entry, the key buffer line win be turned off.

Previous page **Next page**

Page Seven "Kernel" Values

Index

Several page seven locations allow "kernel" operation to be accessed. While it is beyond the scope of this manual to document all of these locations, a few may prove to be of interest.

Name	Address	Function
sparta_flag	$700	'S' if SpartaDOS
sparta_version	$701	version in hex; $32 = 3.2, $40 4.0, etc.
kernel	$703	jump (JMP) to "kernel" entry
misc	$709	jump (JMP) to "misc" entry
sio_index	$70F	"swap" table (9 bytes)
device	$761	"kernel" device number
name	$762	filename and ext (11 bytes)
date	$77B	see below (3 bytes)
time	$77E	see below (3 bytes)
dateset	$781	see below
path	$7A0	path only string (64 bytes)

The "kernel" routine is called by doing a subroutine jump (JSR) to address $703 with the desired command in the 6502 Y register and the desired device number in *device*. For example, with a $10 in *device*, a value of 100 in Y will cause the current time and date to be placed in the variables *time* and *date*. A 101 will cause the current time to be set to the values contained in the variables *time* and *date*.

kd_gettd	100	get current time and date
kd_settd	101	set time and date

The following is a list of valid "misc" vector commands. These should be loaded into the A register before executing a JSR $709. The Y register is used as an index into COPYBUF for those operations using COPYBUF.

misc_initz	0	initialize misc driver
misc_getfina	1	convert path at COPYBUF to *device*, *path*, and *name*
misc_getpath	2	convert path at COPYBUF to *device* and *path*
misc_convdc	4	convert three byte number at DIVEND to a text string at DECOUT

The low nibble of the *device* number is the unit number of the device, such as 2 for D2:. The high nibble is one of the following:

0	SIO block device (SPARTA.SYS, ATARIDOS.SYS, etc.)
1	Clock driver (CLOCK.SYS, JIFFY.SYS)
2	ROM cartridge driver
3	Console driver
4	Printer driver
5	RS232 driver (COM.SYS)
6	reserved
7	reserved

Whenever a file is opened the time and date for that file will be placed in *time* and *date*. When a file is opened for write only and *dateset* equals 0, the current time and date will be read into *time* and *date* and assigned to the file. If *dateset* is -1 ($FF), the file will get the time and date that are in the variables when the open is executed. *Dateset*, unlike the old COMTAB location TDOVER from previous versions of SpartaDOS, will automatically clear after use. This is how a copy of a file can retain the time and date of the original. This is also how a program like ARC assigns stored time/date information to a new file.

Previous page **Next page**

C h a p t e r 7 — Technical Information

Index

SpartaDOS Disk Format

There are four distinct types of sectors on a SpartaDOS format disk. These are boot, bit map, sector map, and data sectors. Data sectors may contain either directory data or file data. The following is a detailed discussion of each type of sector.

Boot Sectors As with most other DOS types for the 8-bit Atari computer, the first three sectors on the disk are boot sectors. These contain a program to load the file designated into the system when booted and other information needed to be able to store and retrieve data to and from the disk. The boot sectors are always single density, regardless of the density of the rest of the disk.

Sector 1 from offset $30 to offset $7F and all of sectors 2 and 3 are the boot code that loads a file under SpartaDOS 2.x and 3.x if specified (with the BOOT command). This code is not used with SpartaDOS X. The first part of sector 1 is a data table containing the values listed below as offsets into the sector. A *disk* can be a floppy disk, a RAMDISK, or a hard drive partition unless otherwise specified. All two or three byte numbers are stored in standard low byte/high byte format.

These are the sector 1 values, given as offsets into the sector:

9 The sector number of the first sector map of the MAIN directory. 2 bytes.

11 The total number of sectors on the disk (2 bytes).

13 The number of free sectors on the disk (2 bytes).

15 The number of bit map sectors on the disk.

16 The sector number of the first bit map sector (2 bytes).

18 The sector number to begin the file data sector allocation search. This is the first sector checked when an unallocated sector is needed. This serves two purposes; it relieves the necessity of searching the bit map from the beginning every time a file is to be allocated, and it allows sectors to be reserved after the main directory for directory expansion (2 bytes).

20 The sector number to begin the directory data sector allocation search. This is the first sector checked when a directory is to be expanded or added. Keeping this separate from the search number above will help keep directory sectors close together to speed searches (2 bytes).

22 The disk volume name. SpartaDOS uses this as part of the disk change identification procedure (8 bytes).

30 The number of tracks on the disk. If the drive is double-sided bit 7 will be set. If it is not a floppy disk (a RAMDISK or hard drive partition, for example) this location will contain a 1.

31 The size of the sectors on this disk (other than the boot sectors). A 0 indicates 256 bytes per sector, while a 128 indicates 128 bytes per sector.

32 The **major** revision number of the disk format. SpartaDOS 1.1 disks will have a $11 here. Disks formatted for SpartaDOS 2.x, 3.x, and SpartaDOS X will all have a $20 here, since they all use identical disk formats.

38 Volume sequence number. This number is incremented by SpartaDOS every time a file is opened for write on the disk. This is used to identify the disk.

39 Volume random number. This is a random number created when the disk is formatted. It is used with volume name and volume sequence number to positively identify a disk, to determine whether or not the data in the disk buffers is still valid.

40 The sector number of the first sector map of the file to be loaded when the disk is booted. This is usually a .DOS file. It is set by XINIT.COM from the SpartaDOS Construction Set and the BOOT command.

Bit Maps A bit map is used to determine the allocation status of each sector on the disk. Each bit in every byte in the bit map shows whether the corresponding sector is in use, so each byte represents the status of eight sectors. Bit 7 represents the first sector of each group and bit 0 represents the eighth sector of each group. The bytes are in sequential order. Byte 0 of the first bit map sector represents sectors 0 through 7 (although sector 0 does not exist), byte 1 represents 8 through 15, and so on. If the bit representing a sector is SET (1), the sector is not in use. If it is CLEAR (0), then the sector is allocated. If more than one bit map sector is needed, any additional bit maps will follow on consecutive sectors.

Sector Maps Sector maps are lists of the sectors that make up a file. The first two entries are the sector numbers of the next and previous sector maps of the file. The rest of the sector is a list of the sector numbers of the data sectors of the file or directory. The following are listed as offsets into the sector map:

0 The sector number of the next sector map of the file or directory. This will be 0 if this is the last sector map (2 bytes).

2 The sector number of the previous sector map of the file or directory. This will be 0 if this is the first sector map (2 bytes).

4 The sector numbers of the data sectors for the file in the proper order. If the sector number is 0, then that portion of the file has not been allocated. All sector numbers are two bytes long. See the Programming With SpartaDOS X chapter under the POINT command for a description of sparse files.

Previous page Next page

C h a p t e r 7 — Technical Information

Index

Directory Structure

The directory is a special file that contains information about a group of files and subdirectories. Each directory entry is 23 bytes in length and contains the file name, time/date, length, the number of the first sector map, and the entry status. The first entry is different from the others; it contains information about the directory itself. The following is a list of this information given as offsets into the first entry:

1 The sector number of the first sector map of the parent directory. A 0 indicated that this is the main (or root) directory of the disk (2 bytes).

3 The length of the directory (in bytes). This is the length of the directory file, not the number of entries (3 bytes).

6 The name of the directory padded with spaces (8 bytes).

When a directory is opened in unformatted or raw mode (see Programming With SpartaDOS X) the file is positioned to the second entry (that of the first file or subdirectory). To read the first entry you must POINT to the beginning of the file after opening it.

The rest of the directory entries are the same. They are 23 bytes long and provide the following information (given as offsets into the entry):

0 Status byte. The bits of this byte, if SET (1), represent the status of the directory entry as follows:

B0 - Entry is protected.
B1 - Entry is hidden.
B2 - Entry is archived.
B3 - Entry is in use.
B4 - Entry is deleted.
B5 - Entry is a subdirectory.
B7 - Entry is open for write.

Notes: bits 1 and 2 are not supported by earlier versions of SpartaDOS. Bits 3 and 4 should always be opposites. Bit 5 should never be changed! A status byte of 0 indicates the end of

the directory. Bits 6 is not used and should not be, since it may be cleared as other operations are performed.

1	The sector number of the first sector map of the file or subdirectory (2 bytes).
3	The length of the file in bytes (3 bytes).
6	The name of the file or subdirectory, padded with spaces if necessary (8 bytes).
14	The extension of the file or subdirectory, padded with spaces if necessary (3 bytes).
17	The date the file or directory was created in DD/MM/YY format (3 bytes).
20	The time the file or directory was created in HH/MM/SS 24 hour military format (3 bytes).

Exploring Disks The best way to become familiar with the SpartaDOS disk format is to use a sector editor and a test floppy to explore. DiskRx, the SpartaDOS disk editor included in the SpartaDOS Toolkit, is an excellent sector editor tailored specifically for SpartaDOS disks. It will identify boot, bit map, sector map, directory, and data sectors. A good understanding of SpartaDOS disk structure and DiskRx can prove to be invaluable for recovering files from disks with bad sectors or damaged directories. Exploring disks can also be a lot of fun.

[Previous page](#) **Next page**