# SpartaDOS™

Advanced Disk Operating System
for Atari Home Computers

## from ICD, Inc.

SpartaDOS v 1.1

by Mike Gustafson

©1984 ICD, Inc.

Rev. 11/2/84

All Rights Reserved

## In the Beginning....

SpartaDOS was written because we felt there was a void in Disk Operating Systems for the Atari Computer. After using systems like MS-DOS, PC-DOS, and CP/M, we at ICD felt there were certain features missing on the Atari.

So we set some goals for the ideal DOS for the Atari,

A. It should be compatible with all software and systems.
B. It should be memory resident.
C. It should support all densities and switch config- urations intelligently.
D. It should be able to handle hard disk drives if they become available.
E. It should allow file storage only limited by the capacity of the media.
F. This created the need for subdirectories.
G. It should allow time/date stamping of files in the directories.
H. MEMLO should be below $2000.
I. It should support batch file processing.
J. It should support full redirection of I/O.
K. It should support relative file access.
L. It should be command driven for entry speed and easy expansion.
M. It should have provisions for UltraSpeed I/O.
N. It should be 'disk compatible' with DOS 2.

Almost immediately it was obvious we couldn't support all these features and still maintain 'disk compatibility'. There was just to much waste and overhead in Atari DOS 2. To keep expanding on that framework would eventually evolve into a total 'kluge'. We decided to dismiss that goal as long as Mike could develop a copy program that would transfer both to and from SpartaDOS with automatic configuration; thus SPCOPY was born.

We believe the end result of our efforts is a superior product with more power than can even be used by the present systems available. We will continue to support SpartaDOS and develop more utilities as time goes on. I would be glad to hear from you with comments and suggestions for future revisions and utility programs, but please don't be dismayed if I don't return your letters. They are all read but we are just too swamped with mail to answer all letters and still be a productive company.

sincerely,

Tom Harker

Note - throughout this manual:

SpartaDOS, UltraSpeed, U S Doubler, and Archiver II, are trademarks of IC D, Inc.
CP/M is a trademark of Digital Research
MS-DOS is a trademark of Microsoft Corp.
PC-DOS is a trademark of IBM, Corp.
Atari is a trademark of Atari, Corp.
OS/A+ is a trademark of OSS, Inc.
ATR8000 is a trademark of SWP, Inc.

# Table of Contents

| Contents | Section |
|----------|---------|

**Section 1**     **SpartaDOS Overview**

## 1.1        SpartaDOS Overview

The Disk Operating System (DOS) is a special program which directs the internal operation of your Atari computer and disk drive. The DOS oversees many important functions including allocation of memory and management of files including storage and retrieval, etc.

SpartaDOS is primarily a memory resident DOS. This means that you can go from an applications program running under DOS, to DOS and use most internal commands, then back to the applications program without reloading from disk each time.

The commands in DOS are of two types; INTERNAL or EXTERNAL. Internal commands are memory resident and include: APPEND, Batch Processing, BUFS, CAR, COPY (as long as Page 6 is not used, otherwise it's external), CREDIR, CWD, DELDIR, DIR, ERASE, LOAD, PRINT, RENAME, RUN, SAVE, and TYPE.

The external commands include: AT_RS232, CLS, COPY (if using Page 6), DIS_BAT, DUPDSK, FORMAT, INIT, MEMLO, PAUSE, PORT, RS232, SET, SPCOPY, TIME, TREE, and UNERASE. The external commands TSET and TD replace SET and TIME when using our optional R-TIME Clock cartridge. To use the external commands, files of the same name (such as FORMAT.COM) must reside on the disk from where they are called. To run the FORMAT.COM program you would just type FORMAT and then return. To provide more disk storage space, copy only the command files that you will be using to each disk and keep a utilities disk (with all the command files) near by.

When formatted, diskettes are given a 'volume' name; density and other options are selected. Each disk must be given a unique volume name for your diskette library such as Games 1, Games 2, WP 1, 000243, etc.

The disk is then broken into directories. Each directory is limited to 128 files. The initial directory is called 'MAIN'. Other directories (subdirectories) can be created under MAIN. The same rules apply to subdirectory names as filenames except they show up in the directory listing with <DIR> after the name and they have special commands to create, move, delete and display. Subdirectories can be 'nested' under other directories with no limits restricting the number of directories other than disk space and filenames per directory. Pathnames are used to connect from one directory to another.

With SpartaDOS there are many format options available but your drive must have the specific hardware in order for the options to work, e.g. you cannot use double density format with the standard 810 disk drive, etc. Once the format is written on a given diskette, the drive will automatically configure for that format type when trying to read or write to it.

## 1.2      Getting Started

Turn your disk drive on, insert the SpartaDOS Master Diskette and turn on your Atari Computer. After loading, the computer display will show the SpartaDOS name and version number at the top of the screen and then run the file called STARTUP.BAT if it exists on disk. To bypass the batch file press the 'SYSTEM RESET' key.

## DISK DUPLICATION

One of the first operations you should do is backup your SpartaDOS MASTER diskette:

Bootup the diskette.

Type FORMAT

Answer 'Y' to write SpartaDOS?

Press RETURN after 'Insert diskette to copy DOS' message.

'1' will be the drive to format.

'1' will be the tracks/sides selection.

'1' will be the correct density.

'Master1' will be the volume name.

To UltraSpeed sector skew answer 'Y' if your drive is modified with U S Hardware (U S Doubler, Archiver II, etc.). Answer 'N' if you are using standard SpartaDOS (sold without U S Hardware).

Now remove the MASTER diskette and insert the destination diskette for Master1 then press RETURN. The drive is now writing your format along with SpartaDOS.

When the disk stops press 'ESC' to escape from the format program. Replace the MASTER diskette into D1: and type DUPDSK and then RETURN.

Source drive number is '1' and so is Destination. Press RETURN after 'Insert source diskette' and swap disks as prompted by the screen until the disk is copied.

The MASTER should then be put away in a safe place.

## 1.3        Some Conventions

**THINGS YOU SHOULD KNOW**

'ESC' will allow you to exit most programs.

'<CR>' means pressing the Return Key.

To change default drives type 'Dn:<CR>' where 'n' is the new drive number.

## VOLUME NAME

A unique (different) volume name must be given to each of your diskettes. Otherwise you will have problems with the system crashing. A volume name can be from 1 to 8 characters long and can include any of the 256 possible numbers, characters or symbols available on the Atari keyboard.

## FILENAME CONVENTIONS (fname.ext).

The general syntax of a filename is:

[dn:][path>]fname[.ext]

Fnames use from 1-8 of the letters A-Z, the numbers 0-9, or the underscore (_). The optional extender can be from 0-3 of the same characters that apply to fname and is proceeded by a period (.).

Here are some commonly used file extender names (.ext):

.COM - command file (a load and go file)
.BAS - a basic file
.TXT - an ASCII text file
.OBJ - an object code file
.SYS - a system file
.EXE - executable file
.ASM - an assembler file
.BIN - a binary file
.DAT - a data file
.PRN - a printer file
.BAT - a batch file
.HEX - a hexidecimal file
.FNT - a font file (character set)
.LST - a listed basic file
.SRC - a source file
.MUS - a music file
.DOS - a DOS module for the INIT program

## COMMAND CONVENTIONS

The general syntax of a Command is:

[dn:][path>]fname [parameters]

The extender '.COM' is reserved for command files. These can be run by typing 'fname<CR>'. The extender .BAT is reserved for Batch files. These can be run by typing '-fname<CR>'. Do **not** type out the extender in either of these cases.

NOTE: The 'command file' in SpartaDOS replaces the 'L' command (load binary file) used in Atari DOS 2. All binary files to be used with SpartaDOS should be renamed with the .COM extender so they can be run by typing the fname.

## DOS MODULES

There are currently four DOS modules on your Master SpartaDOS diskette. These are used with the INIT Command to install DOS on a freshly formatted diskette. The modules can be identified by the extender '.DOS' and more will be added as new modules become available.

'RO' stands for read only. This DOS cannot be used to write to the disk but RO makes a great game loading DOS due to its extremely low MEMLO. Once this has been installed on a disk you cannot write over or erase the files unless you first boot a disk with a Standard R/W DOS.

'STD' is our standard R/W (read/write) DOS. The normal speed version has a MEMLO of $2000 and is used for most DOS applications.

'HS' means high speed and is used with our UltraSpeed Hardware. This works with non-modified drives in normal speed but pushes MEMLO up by about $300 bytes. There are HS versions of both RO and STD SpartaDOS.

## BUFFERS

The BUFS Command is used to change the number of DOS buffers (see BUFS in section 2). Buffers are 128 byte blocks of memory (RAM) that a DOS will reserve for its temporary storage area when reading and writing files. A lower number of buffers gives a lower MEMLO but a higher number of buffers means less disk access especially when doing disk I/O from BASIC.

## FILE SIZE

The file size is given as the actual number of bytes the file occupies. This is a 'true' indication of file size where number of sectors can become misleading as the sector size changes with different density formats.

## DATE/TIME

The date and time when the file was created is shown in the directory. This date and time is true only to the value set when using the SET or TSET Commands. Also, the clock must have been installed with TIME or TD. If the clock is not used, the default of 1-01-84 3:59p is assigned.

## WILD CARDS

Two wild cards (* and ?) can be used to take the place of 'fname.ext' characters in order to select a range of filenames, to find an unknown filename or to speed up entering command files.

The asterisk (*) can be used as a substitute for any combination or number of characters.

Example:    ERASE *.*

Will erase all filenames found on the default drive and directory.

Example:      ERASE *

Will only erase the filenames with no extenders.

The question mark (?) can be used as a substitute for a single character.

Example:      DIR TEST?.PRN

Will display a directory of all filenames TESTA.PRN through TESTZ.PRN and TEST0.PRN through TEST9.PRN.  TEST12.PRN will not be displayed since the sixth character did not match.

Wild cards can be used with most commands that use 'fname' in the command line.  One of the most common and time saving uses of wild cards is to run .COM files.  Consider the following example:

DONKEY.COM
SPACE_IN.COM
GI_JOE.COM
MISSION.COM
FILE_MGR.COM
TELEPHON.COM

Any of the above files in a directory could be run by simply typing the first letter, *, and <CR>.

If DISMAL.COM was included in the above example, then DI*<CR> would run it and you should use DO*<CR> for DONKEY.COM.

CAUTION: Wild Cards are great time savers but can be very dangerous.  Read warnings using COPY and ERASE.

---

## THE MAIN DIRECTORY

The main (or root) directory is a special directory. It can never be deleted whereas subdirectories can be deleted. Whenever DOS is reinitialized (RESET, error, a new diskette is placed in the disk drive, etc), DOS makes MAIN the current directory.

It is good practice to keep any .COM (EXTERNAL command) files in the MAIN directory. When an EXTERNAL command is called from a subdirectory, DOS first scans the sub-directory for the .COM file. If it is not found DOS then checks the MAIN directory, runs it, and returns to the subdirectory.

## SUBDIRECTORIES

All directories other than MAIN, can be thought of as subdirectories. SpartaDOS uses the 'tree' type directory structure, where the MAIN directory is the trunk and the subdirectories can be thought of as branches from that trunk with more subdirectories attached (at nodes) to those branches.

The 'path' can be used in various DOS commands to specify which directories act as source and/or destination for the command. '>' at the beginning of the path starts at the trunk (MAIN) and '<' moves back a node i.e. one directory. The directory name (dname) within the path, selects a branch using '>' as a place holder between dnames.

## THE CURRENT DIRECTORY

The current directory is the directory that you are presently in. If no path is given with a command or file-name, then the current directory is used. The CWD (change working/current directory) command makes a new directory current.

## PATH(s)

A path is basically the connection or route between two places or points. Since we can have more than one directory on each disk, we use 'path' to describe the route. For our use, the path is the list of directories from where we are (the current directory) to where we are going (the destination directory). We use '>' as a delimiter (place holder) between each directory name in our path. When we are not in the MAIN directory, we can also use one '<' to move backwards each directory in our path.

For ease of further explanation we will call directory names 'dname' and consider that dname has the same syntax including extender as 'fname.ext' e.g. TEST, TEST1, and TEST.1, will show up in the directory as:

```
        TEST          <DIR>
        TEST1         <DIR>
        TEST    1     <DIR>
```

'>' used at the start of a path moves the reference directly to the MAIN directory.

The general syntax of a path is:

[>]dname[>dname>dname...>dname]

where the optional starting '>' references the MAIN dir- ectory and the dnames each move one directory along the path with '...' meaning repeat until.

An optional syntax of a path to start moving back in dir- ection is:

<[<<<...<]dname[>dname>dname...>dname]

where each '<' moves backwards in the path one directory. The rest of the syntax is the same as the previous example.

## 1.4        Direct File Access

**NOTE and POINT**

When doing direct access on the diskette in a nonsequential manner, SpartaDOS uses pointers relative to the beginning of each file. This is unlike all other Atari DOS's with the exception of OS/A v.4 and the much unused Atari DOS 3. Atari DOS 2 uses a sector and byte number which works OK until you try to move the file and the sector and byte number in your program no longer match the byte desired. For information on the NOTE and POINT Commands as well as DOS Commands from BASIC see the Technical Notes in Appendix C.

## Section 2     SpartaDOS Command Set

## 2.1    APPEND Command

**Purpose:**    To save a binary block of data at the end of an existing binary file.

**Format:**    APPEND [Dn:][path>]fname[.ext] address address

**Type:**    Internal

**Remarks:**    The format is the same as SAVE. The file specified should already exist since the $FF $FF header is NOT written. Also the file is opened for append/write rather than just write as in the SAVE command. Remember that 'address' is in HEX notation.

**Example:**    APPEND D1:GAMES>GHOST.COM 4000 47FF

The above command appends the block of memory from $4000 to $47FF onto the end of the file called GHOST.COM on the disk in drive #1 under the existing subdirectory called GAMES. This is a command primarily for advanced users working in assembly language.

## 2.2      Batch Commands

**Purpose:**      To retrieve and execute a file (fname.BAT) which instructs DOS to go perform specific operations in a specific order. STARTUP.BAT is a special batch file which is automatically executed upon bootup.

**Format:**      -fname

**Type:**      Internal

**Remarks:**      A batch file contains executable DOS instructions. It can be created with a word processing program or with the Screen Editor using the COPY command. You can use the TYPE command to view the contents of a batch file. A typical example of a batch file could be to load an RS232 handler, then go to the BASIC cartridge, then RUN a communications program. All batch files must end with the filename extension of .BAT.

Comments can be added to your batch files by typing a semicolon (;) at the beginning of the command line. To execute a batch file type a dash (-) then the filename (do not type the extender) and RETURN. Pressing SYSTEM RESET while a batch file is running aborts the batch operation and goes directly to DOS or the cartridge if present.

There are several special command files for use in batch files. CLS.COM will clear the screen, PAUSE.COM will stop execution of the batch file until another key is pressed, and DIS_BAT.COM will disable batch file processing.

NOTE: This is the input redirection ('PRINT' is the output redirection) allowing it to work from BASIC. While the command is in effect, IOCB #5 may NOT be used, since this is the IOCB the output goes through. Also, this IOCB acts as if it were closed, meaning that it could be opened though this may have bad side effects on the system and cause unpredictable things to happen. The reason for making the IOCB appear closed, is to prevent the system from closing the file; e.g. BASIC when entered, closes all IOCBs.

SPCOPY, FORMAT, INIT, and other commands, may reinitialize the DOS which will terminate batch execution. Batch files cannot be used to call up other batch files (linked).

**Example:**     -MODEM

This command will execute the set of in-structions saved under the filename MODEM.BAT on the default drive under the current directory. This file might look like the following example:

    RS232
    CAR
    RUN"D:AMODEM4"

This batch file when executed will run a file called RS232.COM then go to the BASIC cartridge, then finally RUN the Basic program called AMODEM4 from the default disk drive.

---

**2.3**          **BUFS Command**

**Purpose:**     To set or check the number of buffers cur-
                 rently in use.

**Format:**      BUFS [n]

**Type:**        Internal

**Remarks:**     BUFS will display the number of 128 byte
                 blocks of memory (buffers) currently re-
                 served for DOS use.  This display is a DECIMAL
                 value between 2 and 16.  BUFS n will set
                 the number of buffers to be reserved for
                 DOS use, where 'n' is a HEX value between
                 $2 and $10.  The boot up default is 4 under
                 single density unless reset by a batch file
                 or configured by the INIT Command.

                 NOTE:  More buffers require more memory
                 moving MEMLO up.  The minimum requirement
                 for single density both read and write (R/W)
                 is 3.  The minimum for double density R/W
                 is 6.  2 buffers is sufficient if only reading
                 in any density.  In general, if the program
                 requires R/W in random fashion, the more
                 buffers you have, the faster operation will
                 be, and the less wear on your disk and drive.

**Example:**     BUFS F

                 The above command sets the number of DOS
                 buffers to decimal 15.

                 BUFS

                 The screen shows 'BUFS = n' where n = the
                 current decimal number of buffers in use.

---

**2.4**          **CAR    Command**

**Purpose:**    Exit from DOS to a language cartridge.

**Format:**     CAR

**Type:**       Internal

**Remarks:**    The cartridge program will be intact unless an external command was executed in which case the program will be erased upon entry of the cartridge. Also the copy command will cause the contents of memory to be erased.

NOTE: The CAR command with no cartridge present will cause a system crash.

**Example:**    CAR

Sends program control to the language cartridge or the internal BASIC chip in the case of the XL series computer.

**2.5**      **CLS Command**

**Purpose:**      To clear the screen display.

**Format:**      CLS

**Type:**      External

**Remarks:**      The CLS Command is primarily used to clear the screen display during batch file execution which can improve readability on comment lines.

**Example:**      Consider the following batch file:

          ;Good Morning!
          ;
          ;Welcome to Atari!
          CLS
          ;Please turn on your 850 interface.
          ;
          PAUSE
          RS232

          Upon execution this batch will clear the screen after welcoming the operator, display an instruction and wait for keyboard input, then proceed.

## 2.6          COPY    Command

**Purpose:**     Copy one or more files from one device to another and, if specified gives the copy a different name.

COPY can also copy files to the same disk, however the copy must have a different name unless the destination is another directory. Note that a file may NOT be copied to the same disk drive with a different diskette. There is no provision to switch diskettes in the middle of the COPY process. If a single drive copy is desired, see the SPCOPY or DUPDSK commands.

You may also use COPY to transfer data between any of the other system devices, i.e. the Screen Editor, Printer, Keyboard, etc. (See examples to follow).

**Format:**      COPY  d[n]:[path>][fname[.ext]] [dn:][path>][fname[.ext]]

**Type:**        Internal

**Remarks:**     The first file specified is the source file name. If none is given, a default file spec. of '*.*' is assumed which will copy all files. The device for the source must be given. The second file is the destination. If no file name is specified, a default file spec of '*.*' is assumed, which will copy without changing names.

You may use wild cards in both the source and destination file names as well as the extensions. If wild cards are used in the path names, the first directory match will be used. Multiple directories cannot be copied with one COPY command.

When using wild cards with the COPY command, the same renaming convention as in the RENAME command is used. The source file spec is used to find directory matches, and the destination file name renames them by over-riding characters in the source name when the destination name has characters other than '?' or '*' in it.

**IMPORTANT:** ONLY the device ID of 'D' follows this convention, since this device has di-rectories. If a device other than 'D' is used as the source file, then only one file is copied and the source file spec is the source file name. On the 'D' device, the source file name is the filename from the directory.

In the example:

COPY d:*.* Dn:*.*

If the 'd' is an 'E' or any device other than a 'D', the source file name will be '????????.???' and it <u>will</u> be copied to the destination diskette <u>replacing the FIRST file of the destination diskette.</u>

The COPY command, aside from the obvious ability to COPY disk files can also create batch files, print files on the printer, or allow typing directly to the printer.

The internal COPY command resides in PAGE 6 of memory. Occasionally another program might wipe this out and take PAGE 6 for its own use. To continue use of the COPY command an external file provision was built into SpartaDOS. Use the SAVE command to write the file COPY.COM onto the disk with the offending programs. When the COPY command is called, a checksum is done to determine whether COPY is still intact. If not the external file will replace it. The format to create this COPY.COM file is:

        SAVE COPY.COM 600 6FF

**Example:**        COPY D:*.PRN P:

This command copies all files from disk with an extension of .PRN to the printer.

        COPY E: D:INPUT.BAT

This command creates a batch file called 'INPUT'. When this command is entered, the screen will clear and you may begin typing lines of text. When done, a CTRL 3 will signal an end of file for the Editor, and the data will be saved in the disk file.

COPY E: P:

This example may be used for sending ini-
tialization sequences to the printer.

**Caution:** <u>Never</u> type a command like this:

COPY E:

This will copy from the default file
????????.??? to the default device (usually
the disk drive) and find the first match,
then write over it. The first file on your
destination drive will be lost forever and
replaced with ????????.???. <u>Always</u> specify
a destination name when using the COPY command
from anything other than the 'D' device!

**2.7**          **CREDIR Command**

**Purpose:**      Creates a subdirectory on the specified disk.

**Format:**       CREDIR [Dn:]path

**Type:**         Internal

**Remarks:**      The directory to be created is the last directory in the path name. If no path is given, an error will occur. Path is in the format of 'NAME1>NAME2>NAME3' etc. and indicates the route from the current working directory to the destination directory.

**Example:**      CREDIR D2:UTILITY

                Creates a subdirectory on Drive #2 called UTILITY.

                CREDIR GAMES>ARCADE

                Creates a subdirectory on the default drive under the preexisting subdirectory called GAMES.

**2.8**        **CWD Command**

**Purpose:**        Change the working (current) directory on the specified disk.

**Format:**        CWD [Dn:]path

**Type:**        Internal

**Remarks:**        The current directory is where DOS looks to find files whose names were entered without specifying which directory they were in. Also, the current directory is the base directory for relative path names.

IMPORTANT: when a file is opened for read, the current directory is the first to be scanned for the file, but if it is not there, the main (root) directory is then scanned for the file. This is so that one may keep '.COM' files in the main directory and be able to access them from a subdirectory.

During DOS initialization, the current directory is reset to point to the main directory. Initialization occurs when the RESET key is pressed, when a fatal I/O error occurs i.e. bad sector, or when some application causes an initialization when it loads.

Remember that the current directory is in the header of the expanded directory listing.

NOTE: The 'path' can be substituted with '<' to move backwards in the path one directory.

**Example:**      CWD <

Takes you backwards to the previous direc-
tory in the path.

CWD D3:GAMES>ARCADE

Takes you to the subdirectory called arcade
on drive #3 under the subdirectory of GAMES.

## 2.9      DELDIR Command

**Purpose:**     Deletes a subdirectory from the specified disk.

**Format:**     DELDIR [Dn:]path

**Type:**     Internal

**Remarks:**     The directory to be deleted must be totally empty before it can be deleted. The directory to be deleted must be the last directory in the path name. Note that the main (root) directory may not be deleted.

**Example:**     DELDIR GAMES>ARCADE

Removes the subdirectory called ARCADE under directory GAMES only if it is empty ; otherwise an ERROR results.

---

**2.10**    **DIR  Command**

**Purpose:**    To display the volume name and the specified directory name, to list files and subdirectories in the directory, the file size in bytes, the date and time the files were created, and the number of free sectors left on disk.  DIR may be used to list all files matching a file spec pattern by using wild cards.

**Format:**    DIR  [Dn:][path>][fname[.ext]]

**Type:**    Internal

**Remarks:**    If no file spec is specified, all files will be listed i.e. a default file spec of '*.*' is used.  If no path is specified, the current directory is listed.

**Example:**    DIR

Displays the entire current directory of the default drive.

DIR  D2:MODEM>XM*.*

Displays the directory range of XM??????.??? under subdirectory MODEM on drive #2.

**2.11**          **DIS_BAT Command**

**Purpose:**     The DIS_BAT Command is used to disable batch processing within SpartaDOS which may be necessary in order to run certain programs.

**Format:**      DIS_BAT

**Type:**        External

**Remarks:**     Certain programs will not run under SpartaDOS unless the batch file processing is removed. DIS_BAT will disable this and allow most of those programs to run correctly. DIS_BAT can be run as the last command of a batch file. 'System Reset' reenables batch processing.

**Example:**     The STARTUP.BAT file on a disk could read:

                 DIR
                 DIS_BAT

                 Under execution, the directory will be displayed then DIS_BAT will be patched in to disable batch processing.

**2.12**     **DUPDSK**

**Purpose:**     To duplicate an entire SpartaDOS diskette using one or two disk drives.

**Format:**     DUPDSK

**Type:**     External

**Remarks:**     DUPDSK is a disk copy program which will duplicate an entire SpartaDOS disk including subdirectories while using one or two disk drives.   This command will not format or transfer the diskette volume name.   These must be created with the format command since there are many possible variations in format.   Also, the destination format must be the same type as the source format and the destination disk should not have any files on it as they will be overwritten!

**Example:**     DUPDSK

Comes up with prompts for source and destination drives with 1 through 4 being valid drive numbers.   You are then prompted to 'Insert Source diskette ?' if a single drive copy or 'Insert Source & Dest. Diskettes ?' if a two drive copy.  Press any key (except ESC) to start the copy and repeat as necessary if swapping disks in a single drive.

---

**2.13**      **ERASE Command**

**Purpose:**      ERASE deletes the file or files from the specified file name from the specified directory. If no path is specified, then the file is deleted from the current directory.

**Format:**      ERASE [Dn:][path>][fname[.ext]]

**Type:**      Internal

**Remarks:**      You may use wild cards in the file spec, however, use caution as one command can erase multiple files. If no file spec is given, an error will occur. Also, if a file spec of '*.*' is given, the all files will be erased and NO warning will be given. Note that only files will be erased; any subdirectories will be left intact. Also see the 'UNERASE Command'.

**Example:**      ERASE FILE.TXT

Will erase the file named FILE.TXT with no warning.

**CAUTION:** Be very careful when using wild cards with this command. ERASE *.* will wipe out all files under the current directory with no warning! Some other DOS's always ask 'ARE YOU SURE?' We assume you know what you are doing with this command so we don't bog you down with unnecessary keystrokes and questions.

**2.14**          **FORMAT Command**

**Purpose:**      This command is used to format the diskette,
                  create the directory structure, and option-
                  ally put DOS on the diskette.

**Format:**       FORMAT

**Type:**         External

**Remarks:**      The FORMAT program allows many format den-
                  sities, gives the user the option to put
                  DOS on the diskette and to give the diskette
                  a unique volume name. Once a diskette has
                  been formatted, DOS cannot be put on the
                  diskette without reformatting. The FORMAT
                  program does not allow you to change boot
                  defaults or choose many different SpartaDOS
                  types rather it reads the SpartaDOS with
                  defaults from the disk you use as the SOURCE.

**Example:**      FORMAT

                  The first question is whether to write DOS.
                  If answered 'Y' then you must insert a
                  SpartaDOS Source disk of your choice into
                  Drive #1. After pressing 'Return,' SpartaDOS
                  is read into memory. The Source can be
                  any of the versions of SpartaDOS and the
                  newly formatted disk will retain the same
                  defaults as the Source.

The rest of the prompts are the same as the latter part of the INIT prompts.

Drive to format?   (1-4 are valid)
Select number of tracks (usually #1)
Select Density?
Volume name? (1 to 8 characters)
UltraSpeed sector skew? (requires US hardware modification for high speed)

**2.15**      **INIT Command**

**Purpose:**    This is the master formatting program and allows selection of certain default parameters.

**Format:**    INIT

**Type:**    External

**Remarks:**    The INIT program is necessary since SpartaDOS can support many different drive configurations. This program loads SpartaDOS from '.DOS' modules which must also be on the diskette. INIT allows selection of all possible parameters and boot defaults. Since the '.DOS' modules together with the INIT file require a large amount of disk space, we created the FORMAT Command for general everyday use. (See FORMAT)

The FORMAT Command reads SpartaDOS from diskette (not from modules) and is smaller in size. It does not have the flexibility of INIT.

NOTE: There are four .DOS modules included on the present Master diskette. These are:

RO_HS
RO
STD
STD_HS

'RO' means read only. This is a stripped down DOS with a very low MEMLO and short load time. Anytime you attempt to write with an RO version you will get an error (usually 170). The main use of an RO DOS is for loading game files.

'STD' (standard) is the general purpose DOS to use most of the time.

These two DOS versions also have 'HS' (high speed) counter parts which give UltraSpeed I/O when used with the appropriate hardware (US Doubler, Archiver II, etc.). The HS versions are about $300 bytes larger (higher MEMLO).

**Example:**     INIT

The INIT program will load a menu of the possible SpartaDOS versions (fname.DOS) available on the disk along with N) for NO DOS. This is selected if you don't want SpartaDOS on the diskette but want it for-matted. Assuming you selected a DOS, the correct module then loads into memory and gives you the 'modify default parameters' choice. You can select whether to write with verify (slows the write down), the default drive, and the number of buffers. Next you will be asked which drive you want to format in; valid selections are 1 - 4.

INIT then gives a menu of track selection. Most Atari Dives will use 1) 40 tracks/SS, unless you are using an ATR8000 interface or double sided drive.

The next choice, density, allows Single density (128 byte sectors), Double density (256 byte sectors), and 1050 double density (128 byte sectors).

Volume name? is the next selection. You must enter a Volume name and this should be unique to this particular diskette.

UltraSpeed (US) sector skew? is a Y or N question. Answer 'N' unless your drive is equipped with the necessary U S hardware and you are using an HS type SpartaDOS. U S sector skew will make your drive read and write slower than normal unless it is properly equipped.

Now insert the diskette and press return. 'Diskette initialized...' will appear when finished. To format more diskettes press RETURN or to leave this program press ESC.

## 2.16        LOAD Command

**Purpose:**     To load any binary file and not run the file. The standard DOS run and init. vectors are NOT used.

**Format:**     LOAD [Dn:][path>]fname[.ext]

**Type:**      Internal

**Remarks:**    This command is useful for loading character sets, binary data, or files that do not want to be run at the time. Note that a load can only be done from the 'D' device, since load is now an XIO function of the 'D' handler.

**Example:**    LOAD MYFILE.OBJ

This loads a file called MYFILE.OBJ into the memory locations specified in its boot header but does not RUN the file.

NOTE: Don't get this confused with the LOAD command from BASIC. Its function is similar but it loads only binary files (with a header of $FFFF). BASIC programs are generally relocatable while many LOAD type files are not and can cause system crash.

## 2.17     MEMLO Command

**Purpose:**     To display MEMLO for user information.

**Format:**     MEMLO

**Type:**     External

**Remarks:**     The MEMLO Command displays the contents of $2E7 and $2E8. This will tell the user where the top of SpartaDOS resides in case he needs to change to a more memory efficient version.

**Example:**     MEMLO

The display will show Memlo = $$$$ where $ = any hexidecimal digit. RO SpartaDOS with 2 buffers has the lowest MEMLO with $1980. STD_HS SpartaDOS with 16 buffers has the highest with $2800. Loading an RS232 Handler or the TIME Command will move MEMLO up in memory.

## 2.18      PAUSE Command

**Purpose:**    To temporarily halt execution of a batch
file and to prompt the user for a response
to continue.

**Format:**     PAUSE

**Type:**       External

**Remarks:**    This is a convenient way to stop the screen
while displaying instructions from a batch
file.

**Example:**    Consider execution of the following batch
file:

> RS232
> ;Please insert your communications
> ;program diskette into drive #2
> ;
> Pause
> CAR
> RUN 'D2:AMODEM4.2'

This file will first load the RS232 Handler
from the 850 interface then display the
next 3 comment lines and stop with the display
'Press any key to continue'

After the user follows the instructions
and presses a key this program will continue
and go to the Basic cartridge then load
and run the modem program specified.

**2.19**      **PORT Command**

**Purpose:**      To set parameters for RS232 communications.

**Format:**      PORT [path>]fname[.ext]

**Type:**      External

**Remarks:**      PORT sends a two byte configuration file to the RS232 port. The first byte is for XIO 36, Aux1 and the second byte is for XIO 38, Aux1. The first byte will set baud rate, word size, number of stop bits to transmit. The second byte will set parity checking in and parity checking out, translation mode and allow LF after CR.

                These configuration files can be created with the COPY K: D: command but first you must look up the Aux 1 code and figure out what the bytes are.

**Example:**      PORT P_4800.RC

                This will send the configuration file P_4800.RC to the RS232 port.

                P_4800.RC is an example of a configuration file which is included on the Master disk. Its two bytes set the port at 4800 baud, no parity, 8 data bits, 1 stop bit, and send LF after CR on output.

**2.20**  **PRINT Command**

**Purpose:** To send all output, that normally gets written to screen editor through IOCB #0, to a specified output file.

**Format:** PRINT [dn:][path>]fname[.ext]

**Type:** Internal

**Remarks:** This command is normally used to send everything that gets printed on the screen to the printer. However, the output may go anywhere the user desires, including a disk file. This feature is very useful if one wants to have the output of a BASIC program go to the printer or an editing session, etc.

The PRINT command acts like a toggle; the first time a device and file are specified and the output goes to that device, and the second time, the command closes the file and output returns to normal.

NOTE: This is the output redirection (Batch file processing is the input redirection) allowing it to work from BASIC. While the command is in effect, IOCB #4 may NOT be used, since this is the IOCB the output goes through. Also, this IOCB acts as if it were closed, meaning that it could be opened though this may have bad side effects on the system and cause unpredictable things to happen. The reason for making the IOCB appear closed, is to prevent the system from closing the file; e.g. BASIC when entered, closes all IOCBs.

**Example:**     PRINT P:

Sends all future screen display to the printer until 'PRINT' <CR> toggles this off.

PRINT D1:SAVIT.NOW

Sends all future screen display to a file on drive #1 called SAVIT.NOW. The file stays open and closes when PRINT<CR> is entered.

**2.21**       **RENAME Command**

**Purpose:**   Change the name of an existing file or files.

**Format:**    RENAME [Dn:][path>]fname[.ext] fname[.ext]

**Type:**      Internal

**Remarks:**   Wild cards may be used in the file specs.
               A path and device may only be specified
               on the first file name.  A file name must
               be specified for both source and destination
               names.  The rules for wildcarding are the
               same as in Atari DOS.

**Example:**   RENAME FILEZ FILES

               Changes the name of the file on the default
               drive and default directory from FILEZ to
               FILES.

## 2.22 RS232 Command

**Purpose:** To load the RS232 Handler for communications.

**Format:** RS232 or AT_RS232

**Type:** External

**Remarks:** This command is used with the Atari 850 interface module to boot the RS232 Handler. This command can be used as part of a batch file for automatic loading. Unlike Atari DOS 2, you can go to Basic then SpartaDOS and back to Basic without rebooting RS232.

AT_RS232 is the handler for the ATR8000. No 850 interface is needed with it. AT_RS232 is a concurrent only handler though it is intelligent in that it enables and disables concurrent mode automatically as needed for disk access. It is possible to do a

PRINT R:-R:

and take complete control of the Atari Computer from a remote terminal.

**Example:** RS232

With the 850 Module connected properly and powered up you will hear the familiar beep over your monitor or TV speaker which tells you the handler was successfully booted.

---

**2.23**        **RUN Command**

**Purpose:**    To re-execute the last .COM file or execute
                at a given address.

**Format:**     RUN [address]

**Type:**       Internal

**Remarks:**    If address is not specified, then the last
                .COM file is executed. RUNLOC contains
                the default address (see technical notes).
                If you specify an address, execution begins
                there, and RUNLOC will be updated with that
                address. Note that 'address' is in HEX no-
                tation.

**Example:**    RUN 4000

                Runs the file starting at $4000.

                RUN

                Runs the last file executed. If SPCOPY
                was run and you use the OPTION key to get
                back to DOS then typing RUN will take you
                back into SPCOPY as long as the file was
                not destroyed in memory. (This can be a
                great time saving feature).

                NOTE: Don't get this confused with the RUN
                Command from BASIC. It is meant to RUN
                binary (machine language) files, not tokenized
                BASIC programs.

## 2.24 SAVE Command

**Purpose:** To save binary data from memory to disk. To append data, see the append command.

**Format:** SAVE [Dn:][path>]fname[.ext] address address

**Type:** Internal

**Remarks:** This command saves a block of data with the first address being the start memory address and the second address being the ending memory address. The file is saved in the same format as all binary files on the Atari. A $FF $FF header is written first followed by the data header and then the data. Remember that 'address' is a number in HEX notation.

**Example:** SAVE D1:CODE.OBJ 8000 9FFF

Saves the memory from $8000 to $9FFF in a file called CODE.OBJ.

**2.25**          **SET Command**

**Purpose:**      This command allows the user to set the
                  time and date.

**Format:**       SET [mm/dd/yy] [hh/mm/ss]

**Type:**         External

**Remarks:**      If no parameters are specified, then the
                  program will ask for the time and date,
                  otherwise, the time and date specified on
                  the command line will be used.  If using
                  our real time calendar with battery backup
                  the TSET Command must be used.

**Example:**      SET

                  Since no parameters were specified, the
                  prompt showing the current date and asking
                  for a new date appears.  Type in the new
                  date using slashes as delimiters (5/12/84).
                  When asked to enter the time, repeat the
                  above steps using 24 hour time (13/01 results
                  in  1:01:xxpm ,  1  results in  1:xx:xxam).


                  NOTE: 'xx' in time/date indicates the standard
                  default meaning the same number as before.

                  SET  12/10/84  21/12

                  This command line sets the date at 12/10/84
                  and the time to 9:12:xxpm.  Notice we use
                  slashes as delimiters in the command line.
                  Do **not** ever use colons in a SET or TSET
                  command line or unpredictable things will
                  happen!

## 2.26    SPCOPY Command

**Purpose:**    This command is used for single or dual drive file transfers from any Atari compatible source format and density to any compatible destination format. This is the way to convert Atari DOS 2 files to SpartaDOS or the reverse of this.

**Format:**    SPCOPY

**Type:**    External

**Remarks:**    This utility program allows single or dual drive file transfers to or from SpartaDOS. SPCOPY is menu driven and the screen format is easy to follow.

The screen is divided into four 'windows' - TOP, UPPER RIGHT, LOWER RIGHT and LEFT.

TOP - displays your choice for filename range on both SOURCE and DESTINATION diskettes.

UPPER RIGHT - displays the drive numbers selected for the source and destination diskettes.

LOWER RIGHT - displays the command keys and prompts used for this utility.

LEFT - displays the selected directory from the range of the diskette currently being read from or written to.

**Example:** SPCOPY

The menu appears on the screen. The default setting is a single drive copy to and from the main directory. With the source disk in the drive, press 'Start'. The directory is then displayed at the left with the arrow pointing to the current file. Press the 'Space Bar' to tag the file or 'Select' to move on to the next. Once all desired files have been tagged press 'Start' to copy the files. You will be prompted to swap disks as necessary.

NOTE: It is very important to have different or unique volume names for each diskette since SPCOPY reads the volume name to determine a different diskette.

SYSTEM I/O ERROR: this is a general purpose error given by SPCOPY when something goes wrong e.g. inserting the wrong disk when swapping source and destination, copying between two disks with the same volume name, etc.

## 2.27 TD Command

**Purpose:** Used with R-TIME Cartridge to display the time and date on the first line or turn the time and date line off.

**Format:** TD [X]

**Type:** External

**Remarks:** If TD is entered and the R-TIME Cartridge is present in the right or left slot, the current time and date will appear on the top display line. TD uses the interrupt vectors to read the R-TIME Cartridge 60 times a second and update the display every second. If the R-TIME Cartridge is not installed then garbage is displayed on the top line. If the 'X' parameter is entered, the time and date is turned off. To change the time or date in the R-TIME Cartridge use the TSET Command.

The R-TIME Cartridge is our real time clock calendar cartridge with battery backup. It can be used in either cartridge slot and will bootup with the correct time and date without operator input.

NOTE: This command patches itself in the initialization vector and is reinitialized with every RESET. The time/date routine stays in memory and moves MEMLO up. IF TD X is entered, the display is turned off but the module still resides in memory.

NOTE: TD patches itself into the deferred VBLANK vector. During disk I/O the time will move sluggishly since it is doing CRITICAL I/O, but the time will quickly catch up when the disk operations are done.

**Example:**     TD

Turns the time/date display on.

TD X

Turns off the time/date display.

**2.28**     **TIME Command**

**Purpose:**   To display the time and date on the first line or turn the time and date line off. Same as the TD Command but for use without the R-TIME Cartridge.

**Format:**    TIME [X]

**Type:**      External

**Remarks:**   If only TIME is entered, the time/date line will appear with the current time according to DOS. If TIME is already on, then nothing happens. If the 'X' parameter is entered, the time and date is turned off. To change the time and date see the SET Command.

NOTE: This command patches itself in the initialization vector and is reinitialized with every RESET. The time/date routine stays in memory and moves MEMLO up. IF TIME X is entered, the display is turned off but the module still resides in memory.

NOTE: TIME patches itself into the deferred VBLANK vector. During disk I/O the time will move sluggishly since it is doing CRITICAL I/O, but the time will quickly catch up when the disk operations are done.

**Example:**    TIME

Turns the time/date display on.

TIME X

Turns off the time/date display.

---

**2.29**      **TREE Command**

**Purpose:**      To display all the directory paths found on the disk or under the specified directory, and optionally lists the files found in each directory in alphabetical order.

**Format:**      TREE [Dn:][path] [/F]

**Type:**      External

**Remarks:**      The TREE Command displays all path names found on the disk when called from the main directory. If a path is specified then all path names under that directory will be displayed. When called from a subdirectory, TREE will display all path names from that directory on. If the /F is specified then all filenames in each directory will be displayed in alphabetical order.

**Example:**      TREE D1:MODEM /F

Subdirectory MODEM is displayed as the root directory and all filenames under that are displayed then any subdirectories under MODEM are displayed along with the filenames under each of those. This continues until the last subdirectory and filenames are displayed.

---

**2.30**      **TSET Command**

**Purpose:**      This command allows the user to set the time and date in the R-TIME Cartridge.

**Format:**      TSET [mm/dd/yy] [hh/mm/ss]

**Type:**      External

**Remarks:**      If no parameters are specified, then the program will ask for the time and date, otherwise, the time and date specified on the command line will be used.

**Example:**      TSET

Since no parameters were specified, the prompt showing the current date and asking for a new date appears. Type in the new date using slashes as delimiters (5/12/84). When asked to enter the time, repeat the above steps using 24 hour time (13/01 results in 1:01:xxpm , 1 results in 1:xx:xxam).

NOTE: 'xx' in time/date indicates the standard default meaning the same number as before.

TSET 12/10/84 21/12

This command line sets the R-TIME Cartridge date to 12/10/84 and the time to 9:12:xxpm. Notice we use slashes as delimiters in the command line. Do **not** ever use colons in a SET or TSET command line or unpredictable things will happen!

**2.31**   **TYPE Command**

**Purpose:**  To display the contents of an ASCII file.

**Format:**  TYPE [Dn:][path>]fname[.ext]

**Type:**   Internal

**Remarks:**  The file is read, line by line, and printed to the screen editor. If a line is longer than 64 characters, an error will occur (truncated record). This command will only print one file. This same function could also be done with the COPY command, however the COPY command, will erase the contents of program memory.

**Example:**  TYPE STARTUP.BAT

     Displays the contents of the batch file used for initialization.

**2.32**          **UNERASE Command**

**Purpose:**     To restore a file that has been erased.

**Format:**      UNERASE [Dn:][path>][fname[.ext]]

**Type:**        External

**Remarks:**     This command will restore files that have
                 been accidentally erased but only if they
                 are still intact.  If new files have been
                 created since the desired file was last
                 erased then part of the erased file might
                 have been overwritten and is lost!

**Example:**     UNERASE *.*

                 The program will map the current directory
                 and then display the names of the files
                 being unerased as it restores them.

                 NOTE: Occasionally the free sector count
                 is decreased by one after an UNERASE.  The
                 reason for this is that the UNERASE Command
                 will increase the size of the directory
                 map if the last map sector is close to being
                 full.

| **Appendix** | **Section** |
|---|---|
| DOS 2 Comparison . . . . A |
| Error Messages . . . . . B |
| Technical Notes . . . . . C |
| Clock Access. . . . . . . D |
| Diskette Info . . . . . . E |
| Glossary . . . . . . . . . F |

ATARI DOS 2.0S       SpartaDOS  Semi  Equivalent

A. DISK DIRECTORY    DIR
B. RUN CARTRIDGE     CAR
C. COPY FILE         COPY
D. DELETE FILE       ERASE
E. RENAME FILE       RENAME
F. LOCK FILE         SEE UNERASE
G. UNLOCK FILE       SEE UNERASE
H. WRITE DOS FILES   SEE FORMAT/INIT

I. FORMAT DISK       FORMAT
J. DUPLICATE DISK    DUPDSK
K. BINARY SAVE       SAVE, APPEND
L. BINARY LOAD       TYPE FILE NAME OF '.COM' FILE/LOAD
M. RUN AT ADDRESS    RUN
N. CREATE MEM.SAV    NOT NEEDED WITH RESIDENT DOS
O. DUPLICATE FILE    SPCOPY


SpartaDOS COMMANDS NOT AVAILABLE IN ATARI DOS 2.0S

BATCH FILES
BUFS
CREDIR
CLS
CWD
DELDIR
DIS_BAT
INIT
MEMLO
PAUSE
PORT
SET
TIME
TREE
UNERASE

## OTHER MAJOR ADVANTAGES

SpartaDOS supports all densities and possible configurations for the Atari Computer line. There is no need to configure your drive for a particular density as it automatically checks format when reading.

Time and date stamping is available for all files including support of a hardware real time clock.

UltraSpeed I/O is supported with appropriate drive hardware.

RS-232 Handlers are included for communications using the Atari 850 interface or the ATR8000 Computer interface.

SPCOPY Command allows file transfer in batches from any density to any density using one or two drives and automatically translates in both directions to or from SpartaDOS.

## CODE # ERROR CODE MESSAGE

### Atari Basic Messages

| | |
|---|---|
| 2 | Insufficient Memory |
| 3 | Value Error |
| 4 | Too Many Variables |
| 5 | String Length Error |
| 6 | Out of Data Error |
| 7 | Number > 32767 |
| 8 | Input Statement Error |
| 9 | Array or String DIM Error |
| 11 | Floating Point Overflow/Underflow Error |
| 12 | Line Not Found |
| 13 | No Matching FOR Statement |
| 15 | GOSUB or FOR Line Deleted |
| 16 | RETURN Error |
| 17 | Garbage Error |
| 18 | Invalid String Character |
| 19 | LOAD Program Too Long |
| 20 | Device Number >7 or =0 |
| 21 | LOAD File Error |

## SpartaDOS Error Messages

| | |
|---|---|
| 128 | BREAK  Abort |
| 129 | IOCB Already Open (Input/Output Control Block) |
| 130 | Nonexistent  Device  Specified  -  You  typed  an undefined  device.  Legal  devices  are: D:,S:,C:,R:,P:,E:. |
| 131 | File/IOCB  Not  Open  For  Read |
| 132 | Invalid  IOCB  Command |
| 133 | Device  or  File/IOCB  Not  Open |
| 134 | Bad  IOCB  Number |
| 135 | File/IOCB  Not  Open  For  Write |
| 136 | End  of  File |
| 137 | Truncated  Record |
| 138 | Device  Timeout  (No  Drive  Found) |
| 139 | Device NAK (Not Acknowledged) This is a message you'll  get  when  trying  to  read  an  incompatible DOS  or  disk  not  in  place. |
| 144 | Device  Done  Error  (Bad  Sector/Disk  Write Protected) |
| 146 | Function  Not  Implemented  in  Handler |
| 150 | Directory  Not  Found |
| 151 | File  Exists.   May  not  replace  or  delete  file. Can happen when saving a file with a directory of  the  same  name  (dname  =  fname.ext). |
| 152 | Not  a  Binary  File |
| 160 | Drive  Number  Error |
| 162 | Disk  Full  (no  free  sectors) |
| 165 | File  Name  Error  -  Typed  illegal  characters  in filename. |
| 166 | Position  Range  Error |
| 167 | Cannot  Delete  Directory |
| 168 | Illegal  DOS  Command  /  Not  Implemented |
| 170 | File Not Found - you've mistyped a file or command name - tried a write operation with R O SpartaDOS. |

## XIO FUNCTIONS FROM BASIC

The following is a list of SpartaDOS XIO functions and how to implement them from BASIC. The DOS command, if applicable, follows the function name in parenthesis.

**RENAME FILE(s) (RENAME)**

**Format:**  XIO  32,#IOCB,0,0,"Dn:[path]fname.ext fname.ext"

**Notes:**  The IOCB must be currently closed for this operation to be used. Wild cards may be used in the file names.

**ERASE FILE(s) (ERASE)**

**Format:**  XIO  33,#IOCB,0,0,"Dn:fname.ext"

**Notes:**  The IOCB must be currently closed for this operation to be used. Wild cards may be used in the file names.

**SET FILE POSITION - POINT**

**Format:**  X = POSITION
Y = 0
POINT #IOCB,X,Y

**or:**  Y = INT(POSITION/65536)
POKE 846+IOCB*16,Y
POSITION = POSITION-Y*65536
Y = INT(POSITION/256)
POKE 845+IOCB*16,Y
POKE 844+IOCB*16,POSITION-Y*256
XIO 37,#IOCB,0,0,"Dn:"

C-1

**Notes:** In the first method, positions MUST be from 0 to 32767. The second method may take positions up to 8,388,607 (7FFFFF in HEX notation). You may position beyond the end of file if the file is opened in read/write mode. The space between the EOF and where you point is filled with zeros, but physically, no sectors are used to hold the zero data. Thus it is possible to have a file 32K in length but only 5 sectors long. If the data in the gap is accessed in any way, a sector will be created for the 128 or 256 byte area around the location accessed.

Note: POINT under SpartaDOS uses an absolute position relative to the beginning of the file. This is different from the sector number/position as in DOS 2.

## GET CURRENT FILE POSITION - NOTE

**Format:** NOTE #IOCB,X,Y
POSITION = X

**or:** XIO 38,#IOCB,0,0,"Dn:"
POSITION = PEEK(846+IOCB*16)*65536
POSITION = POSITION + PEEK(845+IOCB*16)*256
POSITION = POSITION + PEEK(844+IOCB*16)

**Notes:** In the first method, positions MUST be from 0 to 32767. The second method may return positions up to 8,388,607 (7FFFFF in HEX notation). Note that this is an absolute position relative to the beginning of the file. This is different from the sector number/position as in DOS 2.

## GET FILE END OF FILE POSITION/FILE LENGTH

**Format:**     XIO 39,#IOCB,0,0,'Dn:'
POSITION = PEEK(846+IOCB*16)*65536
POSITION = POSITION + PEEK(845+IOCB*16)*256
POSITION = POSITION + PEEK(844+IOCB*16)

**Notes:**      This returns the current file length (end-of-file pointer).

## LOAD BINARY FILE (LOAD)

**Format:**     XIO 40,#IOCB,4,X,'Dn:[path]fname.ext'

**Notes:**      This command will load a binary file. If X<128 then the INIT/RUN vectors will be used, otherwise they will be ignored. Note that the IOCB must currently not be open.

## SAVE BINARY FILE (SAVE and APPEND)

**Format:**     XIO 41,#IOCB,R,X,'Dn:[path]fname.ext addr1 addr2'

**Notes:**      This command will save a binary file between addr1 and addr2 where addr1 and addr2 are given in HEX. If R=8 then the file will be overwritten, else if R=9 then the file will be appended to (i.e. DOS's APPEND command). If X<128 then a binary file header of $FF,$FF will be written, otherwise it will not be written (preferable on APPENDing segments). Note that the IOCB must not currently be open.

## CREATE DIRECTORY (CREDIR)

**Format:**     XIO 42,#IOCB,0,0,"Dn:path"

**Notes:**      This command creates a new directory. The
                last name in the pathname is the directory
                to be created. The path leading up to the
                name must be a valid and existing path.
                Note that the IOCB must not currently be
                open.

## DELETE DIRECTORY (DELDIR)

**Format:**     XIO 43,#IOCB,0,0,"Dn:path"

**Notes:**      This command deletes a directory. The
                directory must contain no files in order
                for it to be deleted. The last name in
                the path name is the directory to be deleted.
                The path leading up to the name must be
                a valid and existing path. Note that the
                IOCB must not currently be open.

## CHANGE WORKING DIRECTORY (CWD)

**Format:**    XIO 44,#IOCB,0,0,"Dn:path"

**Notes:**    This changes the current default directory. The path name must be valid and all directory names in the path must exist. Note that the IOCB must not currently be open.

## DIRECTORY LISTING (DIR)

**Format:**    10 DIM A$(40):TRAP 40
20 OPEN #IOCB,6,X,"Dn:[path>]fname[.ext]"
30 INPUT #IOCB,A$:PRINT A$:GOTO 30
40 CLOSE #IOCB

**Notes:**    If X<128 then a standard DOS 2 listing is given. This is similar to the directory Atari DOS 2 displays, except the file size in sectors shows up as all zeros. If X>=128 then the expanded (SpartaDOS) listing is given, showing file size, date and time.

### SpartaDOS Data Table

The following is a list of functions that can be performed and the guidelines from the assembly language point of view. **DOSVEC ($0A,$0B)** points to a table of values called COMTAB. To get this value from BASIC use:

CMTAB = PEEK(10) + PEEK(11)*256

At certain offsets into this table are many important DOS values. The following is a list of some of these values:

**COMTAB+0**   A 6502 jump instruction to the command processor. BASIC enters here on a 'DOS' command

**COMTAB+3**   A 6502 jump instruction to the filename crunch routine. This is used by most external DOS commands to fetch the next file name on the command line. The command line is at COMTAB+63 and the crunched file name ends up at COMTAB+33. This routine supplies the default drive number if necessary. The zero flag on return is SET if no file name is on command line. Each call returns the next file name on the command line.

**COMTAB+6**   Address of the divert input/output routine. From assembly language program, do an indirect jump here with the file name at COMTAB+33 and the Y register equal to 0 if output (PRINT), or 1 if input (-fname).

**COMTAB+8**   Address of the stop divert input/output routine. From an assembly language program, do an indirect jump here with the Y register equal to 0 if to stop output (PRINT), or 1 if input (force EOF).

**COMTAB+10**   Current offset into the command line.

**COMTAB+11**   Address of the start of DOS. Normally $600 or $700.

**COMTAB+13**   Current date in DD/MM/YY format (3 bytes)

**COMTAB+16**  Current  time  in  HH/MM/SS  format  (3  bytes,  Not  BCD  format,  therefore  it  can  be  read  with  no  conversion  from  BASIC)

**COMTAB+19**  Alternate  date  in  DD/MM/YY  format  (3  bytes).

**COMTAB+22**  Alternate  time  in  HH/MM/SS  format  (3  bytes).

**COMTAB+25**  Time/date  override  flag.  0  if  to  use  current  time/date  on  new  files.  -1  (FF)  if  to  force  alternate  time/date.  This  is  used  on  commands  such  as  SPCOPY  from  DOS  2  to  SpartaDOS  copying.

**COMTAB+26**  Temporary  RUN  address  (see  COMTAB+61)

**COMTAB+28**  Always  128.

**COMTAB+29**  Density  (sector  size)  of  each  drive.  0  means  256  byte  sectors.  (1  byte  *  4  drives  =  4  bytes)

**COMTAB+33**  Buffer  for  crunch  routines  output.  (28  bytes)

**COMTAB+61**  Run  address  of  .COM  file.  If  no  address  is  specified  after  the  RUN  command,  this  is  the  address  to  be  run.

**COMTAB+63**  Input  buffer  (command  line  -  64  bytes).

**COMTAB-10**  Address  of  SIO  routine.  Many  commands  use  this  to  run  the  DOS's  SIO  routine  (High  speed  version).  In  low  speed  versions  this  will  be  $E459.  (2  bytes)

**Things to be VERY CAREFUL OF.**

Please, use unique volume names on all diskettes.

When formatting a diskette with another DOS in an UltraSpeed modified drive, first turn the disk drive off, then back on for a 'cold start'.

When using the COPY Command, always make sure you specify a destination filename when copying from any device other than a disk drive. If you don't you will lose the first file on your destination disk! ????????.??? will write over it.

## SOME VERY TECHNICAL DETAILS:

CIO open type flags e.g. OPEN #IOCB,T,X,"Dn:fname.ext" the meaning of the bits in **T** are:

Bit 0 (+01):  Open in append mode (in conjunction with write or update)

Bit 1 (+02):  Open formatted directory (in conjunction with read). If X>=128 then directory is in the expanded format.

Bit 2 (+04):  Open in read mode.

Bit 3 (+08):  Open in write mode. (If both read and write then it is considered update mode.)

Bit 4 (+16):  Open current directory file in unformatted (raw data) mode. NOTE: the position will automatically be set at 23, just past the directory header. (in conjunction with read or update)

Bit 5 (+32):  Open subdirectory entry (in conjunction with read only.. if write is used, the entry may possibly be permanent and useless)

## SECTOR MAP

| next | last | data | data | ... | data |

A sector map describes which sectors contain the data of the files.

next:    This is the sector number of the next sector map. It will be a zero if this is the last sector map.

last:    This is the sector number of the last sector map. It will be zero if this is the first sector map.

data:    These are the sector numbers of the data sectors of the file. If a data sector number is zero, then that portion of the file is not allocated. This can happen if a file is written at a low file position and then written to at a high file position without ever writing the in between data (see POINT).

## BIT MAP

A bit map is a sequence of bits that determine whether a sector is in use or not. Bit 7 represents the first sector in a group of 8 and Bit 0 represents the eighth sector in the group. The first byte of the bit map corresponds to sector numbers 0 through 7, the second corresponds to sector numbers 8 through 15, etc. (NOTE that sector '0' does not exist). If more than 1 bit map is required for the diskette, they will be sequential on the diskette. A sector is 'free' if the corresponding bit map bit is SET (1).

## THE DIRECTORY DATA STRUCTURE

The directory is a special file that gives information about each file and subdirectory it contains. Each directory is a file and can be accessed as such (refer to CIO open codes). Each directory entry is 23 bytes in length. The following describes each entry:

```
+00 :    Entry status
                00 if at end of directory file
                16 if file has been deleted
                40 if file is a subdirectory
                08 if file is a standard file
+01 :    First sector map of file
+03 :    Length of file (3 bytes)
+06 :    File name (8 bytes.. space padded)
+14 :    File name extension (3 bytes.. space padded)
+17 :    Date file was created (DD/MM/YY)
+20 :    Time file was created (HH/MM/SS)
+23 :    Length of each directory entry
```

The first directory entry has a special meaning. This entry actually describes the directory file as a whole and takes the place of a directory entry in the directory above the one in question. The sector has a special meaning; it is the first sector map or the directory that contained this directory as one of its entries. This is how relative paths are possible (i.e. '<dname' etc.). If this value is a zero, the directory in question is the MAIN directory, i.e. there is no directory above the MAIN directory.

## DISK DESCRIPTOR (SECTOR 1)

The diskette descriptor gives information above the disk-ette. The following is a list of information given by the descriptor:

+09 :   First sector map of the MAIN directory
+11 :   Total number of sectors on the diskette
+13 :   Number of free sectors on the diskette
+15 :   Number of bit map sectors used
+16 :   First bit map sector
+18 :   Sector number to begin file data sector allocation search
+20 :   Sector number to begin directory data sector allocation search
+22 :   Volume name (8 characters)
+30 :   Number of tracks
+31 :   Sector size (0 if 256 bytes, 128 if 128 byte sectors)
+32 :   DOS revision number
+33 :   Number of buffers to reserve (default if booted)
+34 :   Default drive to use (if booted)
+35 :   RESERVED
+36 :   RESERVED
+37 :   Number of sectors in DOS boot

## How to Access the Real-Time Clock

SpartaDOS keeps the internal time/date clock running and stores the values in memory. These can be used in your applications programs whenever access to time or date is desired. The values are stored in COMTAB+13 to COMTAB+18. (The pointer to 'COMTAB' is stored at DOSVEC 10,11)

COMTAB+13 = location of day
COMTAB+14 = location of month
COMTAB+15 = location of year
COMTAB+16 = location of hours (24 hour format)
COMTAB+17 = location of minutes
COMTAB+18 = location of seconds

The Basic Program below will display these values. It was written as a plain and simple example for those starting out in Basic or new to programming the Atari. To read the time/date values use PEEK and to change the values use POKE.

```
10 CMTAB=PEEK(10)+PEEK(11)*256
20 FOR T=13 TO 18
30 ? PEEK(CMTAB+T)
40 NEXT T
```

Note: A special SpartaDOS handler is used with the TD and TSET Commands to access our optional R-TIME Clock/Calendar Cartridge. This automatically updates the internal real time clock used by DOS. Since the cartridge is much more difficult to read directly, we recommend you read it through the DOS locations as shown in the above example.

## ALL ABOUT DISKETTES

CONSTRUCTION

The Atari disk drive uses 5 1/4 inch floppy diskettes. These are made of mylar with a magnetic coating and a dry surface lubricant much like recording tape. A semi protective lined jacket covers the diskette. Data is recorded in a digital format of 0s and 1s. Generally if a floppy disk loses just one of these bits your whole program is destroyed!

STORAGE AND HANDLING

The general handling rules for diskettes are:

1) Never touch the actual recording surface of the disk. This is the shiny part on each side through an elongated hole.

2) Do not bend the diskette.

3) Keep diskettes in their protective jackets when not in use.

4) Do not expose them to extreme temperatures. If this happens let the disks sit at room temperature for at least 1 hour before use.

5) Keep the diskettes away from magnets or magnetic fields. This includes TV sets, motors, transformers and power supplies, etc.

QUALITY (TPI and Density)

Diskettes are rated by the guaranteed quality of the re-cording surface. Sometimes this is given in TPI or tracks per inch. A disk rated at 96 TPI and will have fewer errors than a 48 TPI disk. A disk rated for single density will probably have more errors than one rated for double density. The higher quality disks will probably last longer and create less wear on your disk drive head since they use better lubricants. Single sided disks will work with double sided drives but they are not guaranteed like the double sided variety.

FORMAT STRUCTURE

Format on a single sided single density (SSSD) Atari drive consists of 40 tracks of 18 sectors each. Each sector holds 128 bytes but other DOS's use 3 of these bytes for mapping. Some of these sectors are also reserved for file management. SpartaDOS gives you 713 sectors of 128 bytes each for your use compared to 707 sectors of 125 bytes with Atari DOS 2.0. The raw SSSD format yields 92160 bytes per disk (90 KB).

1050 "Double Density" (1050DD) consists of 40 tracks of 28 sectors each. Each sector also has 128 bytes but is shorter in physical length. The raw 1050DD format yields 143360 bytes per disk (140 KB).

"True Double Density" (SSDD) uses 40 tracks of 18 sectors each but each sector stores 256 bytes. 184320 bytes (180 KB) is the total yield for a raw SSDD format.

WRITE PROTECT

There is a squared off notch on the upper right side of each diskette. This is the write protect notch. If this is covered with a protective tab, your drive will not be able to write to that diskette.

Note: The above holds true for 5 1/4 diskettes only. 8 inch disks are the opposite - cover the notch to write and uncover it to protect your data.

ON USING BOTH SIDES OF THE DISK

A common practice seen mostly with home computers, is to cut a notch on the side of the diskette opposite the write protect notch, which can be done easily with a 1/4 inch paper punch. This allows the user to then flip the disk over and use the back side for storage which effectively doubles the amount of data that a single sided diskette can hold. This practice works with most Atari compatible drives but can lead to eventual diskette damage.

The problem begins when the disk is flipped to Side B; the drive spins the disk the opposite direction it was turning while on Side A. Before flipping, the felt fibers inside the jacket were always wiping one way keeping the disk surface clean and lubricated. Dirt accumulated on the bottom of the jacket as the fibers continuously cleaned the disk surface. When the disk is flipped to side B, the fibers bend the other direction and dump much of the dirt onto the top side of the spinning disk. The process is repeated when the disk is flipped back to side A.

We hope this will help you to understand the problems associated with using both sides of the disk. We don't discourage the practice because it has its uses, such as media distribution or for disks that are not used very often. However, we strongly recommend you only use the front side of a disk if you are using it for business applications and when storing other irreplaceable data!

---

The  following  is  a  list  of  terms  and  definitions  which
may  appear  throughout  this  or  other  computer  manuals.

ADDRESS        a  location  in  memory  with  the  Atari  from
               0  to  $FFFF

APPEND         to  add  on  to,  to  append  two  files  is  to
               add  one  file  onto  the  end  of  the  other.

ASCII          the  American  Standard  Code  for  Information
               Exchange  defining  a  one  byte  code  from  0
               to  127  decimal.

BATCH          a  batch  file  is  a  file  containing  a  group
               of  commands  to  be  executed  consecutively.

BINARY         the  BASE  2  numbering  system;  only  containing
               2  numbers,  either  1  or  0.    For  ease  of  use
               Binary  numbers  are  usually  represented  in
               HEX  notation.    A  Binary  file  is  one  which
               is  directly  readable  by  the  computer  without
               going  through  an  interpreter.

BIT            a  binary  digit  -  either  a  0  or  1

BOOTUP         refers  to  system  initialization  which  sets
               up  the  computer  when  powering  up.  Also  called
               BOOT.

BUFFER         any  block  of  memory  specifically  set  aside
               for  use  as  temporary  storage.

BYTE           the  amount  of  information  a  computer  can
               process  in  one  cycle  -  the  Atari  byte  =
               8  bits.    The  byte  represents  a  number  from
               0  to  255  (0  to  $FF  HEX).

CIO            Central  Input/Output  -  one  part  of  the  oper-
               ating  system  that  handles  I/O.

COLD START    to start up the computer as if just powered up.

COMMAND    communication given from the human to the computer directing it to perform an action.

CURSOR    the pointer on the screen that marks where the next keystroke will appear or where the next action will take affect.

DATA    information generally used or operated on by a program.

DEBUG    to isolate and eliminate errors from a program.

DECIMAL    the BASE 10 numbering system; not very useful to computers unless translated to HEX or BINARY, but easy for humans to understand - uses the digits 0-9.

DEFAULT    the standard condition or value that exists upon running a program.

DENSITY    generally the number of bytes per sector is the disk density; single density being 128 bytes, double density being 256 bytes per sector. Actually density specifies the number of bytes per track on a disk.

DEVICE    the Atari defined devices are Dn:, E:, S:, R:, P:, C: ; referring to Disk drive (n=drive number), Editor portion of the screen display, the Screen, the RS232 device for communications, the Printer, and the Cassette storage device.

DIRECTORY    the list of all files stored in a given area of the diskette.

---

DOS          the Disk Operating System ; this is the program which manages the I/O to and from the computer.

DRIVER     same as HANDLER ; a program written to specifically handle one particular device or operation.

FILE          a collection of information usually stored as a named unit on a diskette.

FORMAT     the guidelines for the way in which the magnetic structure of the disk is layed down ; standard Atari format is 40 tracks (complete circles around the disk) with 18 sectors per track (18 blocks of 128 bytes spaced around each track like pie pieces).

HANDLER    a program written to handle a device.

HARDCOPY  printed on paper.

HEADER     the first few bytes of a program which tell it where it should be located, what type of program it is and how it should be used.

HEX           the BASE 16 numbering system ; there are sixteen unique single digits used for counting 0-F. A HEX number is usually proceeded by '$'.

I/O           Input/Output ; this is what ties a computer to the outside world.

ICD           Innovative Computer Design ; the company which authored SpartaDOS and other fine programs for the Atari Computer.

---

K in the computer world, one K or kilo is
equal to two to the tenth power or 1024.

KLUGE a 'Rube Goldberg' of software. A very compli-
cated and confusing way of doing something
relatively simple.

LANGUAGE a program which makes it easier or faster
in one way or another for humans to program
a computer. BASIC, LOGO, PASCAL, Assembler
are all languages for the Atari.

MACHINE CODE the lowest level programming language but
also the fastest running.

NESTED fitted within similar things.

PATH the trail or course taken from one place
to another ; when using subdirectories a
path specifies the trail from where you
are to where you want to go or retrieve
a file from.

PERIPHERAL an external device connected to your computer
like a disk drive, printer, etc.

PROGRAM a set of instructions to tell the computer
how to accomplish some certain task. These
instructions must conform to a particular
order and the conventions of the language
used.

PROMPT a signal to the user that some action may
be needed.

RAM Random Access Memory. The computer can
read and write to this but it is lost when
power goes down.

REALTIME    relating  to  'real  time'  as  on  the  standard
            clock  ;  a  realtime  program  uses  a  clock.

RELOCATABLE a  program  that  can  be  moved  to  different
            areas  in  memory  and  still  function  properly.

ROM         Read Only Memory.   Permanent memory that
            can  only  be  read.

SECTOR      the  standard  block  of  storage  used  on  floppy
            diskette  media  ;  can  be  128  or  256  bytes
            with  the  Atari  formats.

SPARTA      an powerful city in ancient Greece ; POWER!

SYNTAX      the  organization  or  arrangement  of  elements
            as  parts  of  a  command  line.

TRACK       a magnetic circle on the disk which contains
            the pattern of sectors.   There are 40 tracks
            on  a  standard  Atari  formatted  disk.

TRUNCATED   cut  short.

VARIABLE    something  that  changes or  has  no  fixed  value.

WARM START  a SYSTEM RESET without wiping out memory
            as  in  cold  start.

WILD  CARD  used when specifying filenames or pathnames
            to  ease  operator  entry  or  select  a  certain
            range  of  names.    *  and  ?  are  the  two  valid
            wild  cards.

| CODE # | ERROR CODE MESSAGE |
|--------|--------------------|
|        | **Atari Basic Messages** |
| 2      | Insufficient Memory |
| 3      | Value Error |
| 4      | Too Many Variables |
| 5      | String Length Error |
| 6      | Out of Data Error |
| 7      | Number > 32767 |
| 8      | Input Statement Error |
| 9      | Array or String DIM Error |
| 11     | Floating Point Overflow/Underflow Error |
| 12     | Line Not Found |
| 13     | No Matching FOR Statement |
| 15     | GOSUB or FOR Line Deleted |
| 16     | RETURN Error |
| 17     | Garbage Error |
| 18     | Invalid String Character |
| 19     | LOAD Program Too Long |
| 20     | Device Number >7 or =0 |
| 21     | LOAD File Error |

# CODE # ERROR CODE MESSAGE

## SpartaDOS Error Messages

| Code | Message |
|------|---------|
| 128 | BREAK Abort |
| 129 | IOCB Already Open (Input/Output Control Block) |
| 130 | Nonexistent Device Specified |
| 131 | File/IOCB Not Open For Read |
| 132 | Invalid IOCB Command |
| 133 | Device or File/IOCB Not Open |
| 134 | Bad IOCB Number |
| 135 | File/IOCB Not Open For Write |
| 136 | End of File |
| 137 | Truncated Record |
| 138 | Device Timeout (No Drive Found) |
| 139 | Device NAK (Not Acknowledged) |
| 144 | Device Done (Bad Sector/Disk Write Protected) |
| 146 | Function Not Implemented in Handler |
| 150 | Directory Not Found |
| 151 | File Exists |
| 152 | Not a Binary File |
| 160 | Drive Number Error |
| 162 | Disk Full (no free sectors) |
| 165 | File Name Error |
| 166 | Position Range Error |
| 167 | Cannot Delete Directory |
| 168 | Illegal DOS Command / Not Implemented |
| 170 | File Not Found |

# ICD