24

ZAP80™

A Disk Access Utility

for

CP/M, SB/80, DMADOS

and

Compatible Operating Systems

TABLE OF CONTENTS

INTRODUCTION to ZAP80™

ZAP80 is a disk/diskette access utility for CP/M, DMADOS and other compatible 8080/Z80 operating systems. ZAP80 was initially written to allow direct access to the disk surface by specifying a track and sector number for any sector. In its development, it has been expanded to include extensive file utility servicing to access and patch file sectors. The selection of file commands is provided in a separate File Menu.

In addition, several other facilities have been added. These include comparing files, saving a sector in a temporary buffer for subsequent copying to a different location, displaying any location in memory, initializing directory and/or data tracks to e5h, and chaining from ZAP80 directly to a COM file on disk.

*** OPERATIONAL WARNING ***

Care must be taken when using ZAP80 to patch such areas as the directory. ZAP80 allows access to any disk sector, including those on the operating system and directory tracks. In particular, it is possible to patch a directory sector in such a way as to make it incompatible with the actual data in the data area tracks.

ZAP80™ OPERATION

ZAP80 is invoked by keying in ZAP plus an optional drive and optional filename.typ on the console command line.
Type:

```
ZAP [<drive:><filename.typ>]<cr>
```

ZAP80 will then signon with a brief message and a display of the current operating system it identified, followed by:

```
                                     ZAP80 version n.n

                              Menu's
                               PM          =  primary menu
                               FM          =  file menu
                               MM          =  miscellaneous menu
                               SM          =  special functions menu

                               CS          =  configure system for ZAP80

                          LM,<?>,<help>    =  redisplay this menu

   file parameters==>     track 44    sector 21    file = B:ZAP.COM     LSN = 000
   logon parameters==>    track  2    sector  1    drive B       dens = 52(o/s)
   COMMAND== >
```

ZAP80 maintains two sets of parameters for accessing disk sectors: file parameters and logon parameters. The file parameters allow access to any sector of a specified file, while the logon parameters allow direct access to the disk surface by specifying a track and sector on the current logged-on drive. The file does not have to be on the logged-on drive, and can be left open while a logon sector is accessed.

For file operations, ZAP80 maintains an internal file control block, and all file accesses are performed through standard bdos function calls. Random reads are made by specifying either an address or a logical sector number (LSN), which ZAP80 internally translates to the proper file extent and relative sector number within the extent, before making the function call.

For file reads, ZAP80 expects the bios to perform a set track and set sector during the read operation. If a bios does not do so, the resulting file parameter display of the track and sector values will be arbitrary. Note, however, that the track and sector display is a convenience (allowing, for instance, subsequent access to a file sector by using the logon parameters) and does not materially affect actual file i/o operations.

ZAP80 specifically recognizes CP/M 2.0+, DMADOS 8.0+ and CDOS operating systems; all others default to CP/M 1.4. For CP/M 1.4, ZAP80 uses its own internal parameters, which are user modifiable; otherwise, the parameters provided in the bios are used. For CDOS, standard CP/M 1.4 parameters are used; however, the cdosfg must be set to a non-zero value in order to bypass the CP/M version test function call (this MUST be done using DEBUG before ZAP is run). For systems with non-standard bios parameters, a system override flag is provided to force the use of ZAP80 parameters. See the section on ZAP80 System Configuration.

If a file was specified on the console command line, the file parameters are set to that file's first sector (LSN=0); otherwise, a 'no file currently open' message is displayed. The logon parameters are set to the first sector of the directory track, using the o/s track offset parameter of CP/M 2.0+ and DMADOS 8.0+ (default=2 for CP/M 1.4, and can be set in the system configuration table).

All disk i/o except for the copy and initialize commands is performed to/from a ZAP80 buffer called SBUF (identified in the menus as screenbuf). Thus, for instance, if a sector has been saved in the temporary buffer, it must be retrieved back into screenbuf before it can be written to disk.

Displaying memory does not alter screenbuf; however, the display of screenbuf may be removed from the screen, necessitating an appropriate command to identify its contents before writing back to disk (DS to display screenbuf, or reread the sector).

All patching is performed in screenbuf and does not alter the disk surface until it is explicitly written back using an FW (file write) or WL (write logon) command. A sector could therefore be patched, saved in the temporary buffer, reread from disk, and then compared with the patched version by retrieving from the temporary buffer before writing back.

ZAP80 maintains two flags for physical writes to ensure that the buffer is flushed for double density. One is set on whenever a logon sector write is performed, the other whenever a file write is performed. When a file is opened, or ZAP is exited, the file write flag is checked. If it is on, a file close function call is made before performing the requested operation, and both flags are turned off. If ZAP is exited and the file write flag is off, the logon write flag is checked. If it is on, ZAP80 (arbitrarily) reads track 1, sector 1 to ensure that the buffer is flushed.

Input from the console in response to command prompts adhere to the following conventions: prompts for address inputs expect hex; drive specifications, alpha (A,B,etc.); track and sector, decimal.

Most of the commands are two-character commands, although there are a few control character and single-character commands. Command input can be either upper or lower case, and a delimiter (i.e., a <cr>) is not required after the command is keyed in; ZAP processes it immediately. No command alters the disk surface without providing a prompt first, except for FW, file write.

For a more detailed discussion of ZAP80 operation, see the system configuration section, following, as well as the command descriptions themselves.

ZAP80™ SYSTEM CONFIGURATION

The minimum system configuration required for the initial proper operation of ZAP80 is a user provided cursor up string, which can be inserted using the CS (configure system) command. (CDOS requires the cdosfg to be set to a non-zero value — this MUST be done using DEBUG before ZAP is run). The patch commands call a clear screen routine, which can also be inserted with the CS command. In addition, several of the cursor movement functions test for cursor addressability, and work more efficiently if it is provided. This is also included in the CS command as a convenience.

CS ——— Configure System

In order to use this command, you must be logged on to the drive containing the file named ZAP.COM. If you want to rename the file, configure the system first. The command provides for inserting a cursor up string, a clear screen string and cursor addressing. The cursor up is required, the clear string and cursor addressing optional (for further discussion of cursor addressing, see the pgm listing, further below).

After the cursor up string is inserted, ZAP can be used to patch any of the parameters which need modification. Four of the parameters bear discussion at this point:

numdvs ——— # of drives on the system

ZAP80 is distributed with this parameter set to 8 drives, and is used by the SD (select drive) command as an overflow test. If it is set to 2, but there are 3 drives on the system, ZAP80 will never allow drive C to be selected. On the other hand, if it is left at 8 and there are only 3 drives on the system, trying to select drive D, E, etc. will cause a normal operating system error message.

tperdk ——— # of tracks per disk

ZAP80 is distributed with this parameter set to 77 tracks per disk, and functions as an overflow test in a manner similar to the number of drives above. No provision has been made in this release of ZAP80 for managing overflow for systems consisting of different sized disks. If this is the case, set this value to the number of tracks for the largest disk on your system; overflow on smaller disks will cause an o/s message.

deffbs ——— default file base address

ZAP80 is distributed with this parameter set to 100h, the usual beginning of the transient program area (tpa). This value MUST be the address that the operating system expects for loading COM files, as this is the value used by the LD (load) command.

Besides the LD command, the normal function of this value is to make file address displays consistent with source listing addresses for ease in patching. The FB (file base) command is provided for dynamically altering this value for patching programs org'd at some value other than 100h.

For data files, address values relative to the beginning of the file are usually unnecessary; however, an accurate value can be obtained by using the FB command to temporarily set the file base to 0.

maxlsn ——— maximum logical sector number

ZAP80 is distributed with this value set to logical section number 0f1ch, which is the largest value for a standard 8" double density diskette. Again, the value is used as an overflow test in a manner similar to other commands above; however, it has one additional efficiency function:

ZAP80 performs its own file control block maintenance for random file reads. That is, it calculates the proper extent for any given logical sector number, and then tries to open that extent. If the open is unsuccessful, it decrements the extent number, and repeats the process. Keeping the maxlsn parameter as small as feasible helps prevent overextended open attempts.

The remainder of this section refers to the rest of the system configuration parameters, and applies only to the operation of the logon parameter commands. Since all file operations are performed through standard function calls, they are not affected by the system configuration parameter values.

ZAP80 has been designed to accommodate many different BIOS implementations, both standard and non-standard (where standard is determined by two criteria: 1) the first sector number of each track is 1; 2) the operating system is either cp/m 2.0+ or dmados 8.0+ with well-formed sector translation tables, or single-density cp/m 1.4).

For standard systems, the following discussion will be informative in nature; the actual parameter values will probably not need to be altered. If there are any doubts, or if ZAP does not function as expected, refer to the following discussion, as well as to the discussion accompanying the appropriate command, and make any necessary adjustment(s).

Several ZAP80 commands use the first sector number and sectors per track parameters to check for valid input entries and overflow conditions. In addition, the copy, initialize, and skew sector read commands use the sector translation tables in their operation.

Non-standard BIOS implementations would not function properly with ZAP80 without special considerations. As an example, there are implementations which use the same sector translation table for both single and double density (the BIOS performs the sector translation internally, and the SECTRAN function of cp/m 2.0 has been disabled). Such cases can be handled by ZAP80, as described below.

In addition, ZAP80 can handle double density implementations of cp/m 1.4, which in all ways conform to standard procedures except for providing the necessary parameters.

In order to handle these non-standard cases, ZAP80 maintains, in its internal working storage area, the following values for the current logged on drive:

- first sector number (assumed to be consistent for the system, the default value is 1, and can be set to 0 if necessary)
- current number of sectors per track
- address of the current sector translation table

and ZAP80 goes through the following procedure whenever a select drive command is processed:

1. retrieve the o/s type identified at signon
2. set the values above as follows:

   a. if the o/s is cp/m 2.0+ or dmados 8.0+, check to see if the 'override o/s translation' flag is set;

   - if the flag is set on, use the o/s value for the # of sectors per track, but ignore the o/s sector translation table address;

     - if the # of sectors per track equals ZAP's single density value, use the single density ZAP table, and mark the density display as a '(zap)' table

     - otherwise, use the double density ZAP table, and mark the density display as a '(zap)' table

   - if the flag is not set on, use the appropriate o/s function calls to set the values, and mark the density display as an '(o/s)' table

   b. otherwise, set the values to those for 'single' density in the ZAP80 configuration tables, and mark the density display as a '(zap)' table. (Note: we have used the arbitrary labels 'single' and 'double' density; the values can be anything necessary to conform to the user's system.)

Now, if the D (toggle density) command is used, ZAP80 will toggle back and forth between the two sets of values in the system configuration tables (i.e., the o/s values will be overriden if the operating system is cp/m 2.0+ or dmados 8.0+), and the density display will be marked as a '(zap)' table. To reset the o/s values (for cp/m 2.0+ and dmados 8.0+), reselect the drive using the SD command.

The first sector number, which is normally 1, can be set to 0 in the system configuration table if necessary; the SF (set first) command is provided to dynamically change the first sector number, for systems in which the first sector number is different for single density than it is for double.

cdosfg ——— cromemco CDOS o/s flag

Set this flag to non-zero for Cromemco's CDOS, or any operating system which does not recognize cp/m's get version number function call (decimal 12, hex 0c). This will cause ZAP80 to bypass the function call and automatically use the ZAP80 internal parameter tables. (The flag MUST be set using DEBUG before ZAP is run.)

° ° °

ZAP80 maintains 100h-23fh as a system configuration area; the program itself starts at 240h. The patch addresses are provided below, and can be examined using the DM (display memory) command after invoking ZAP.

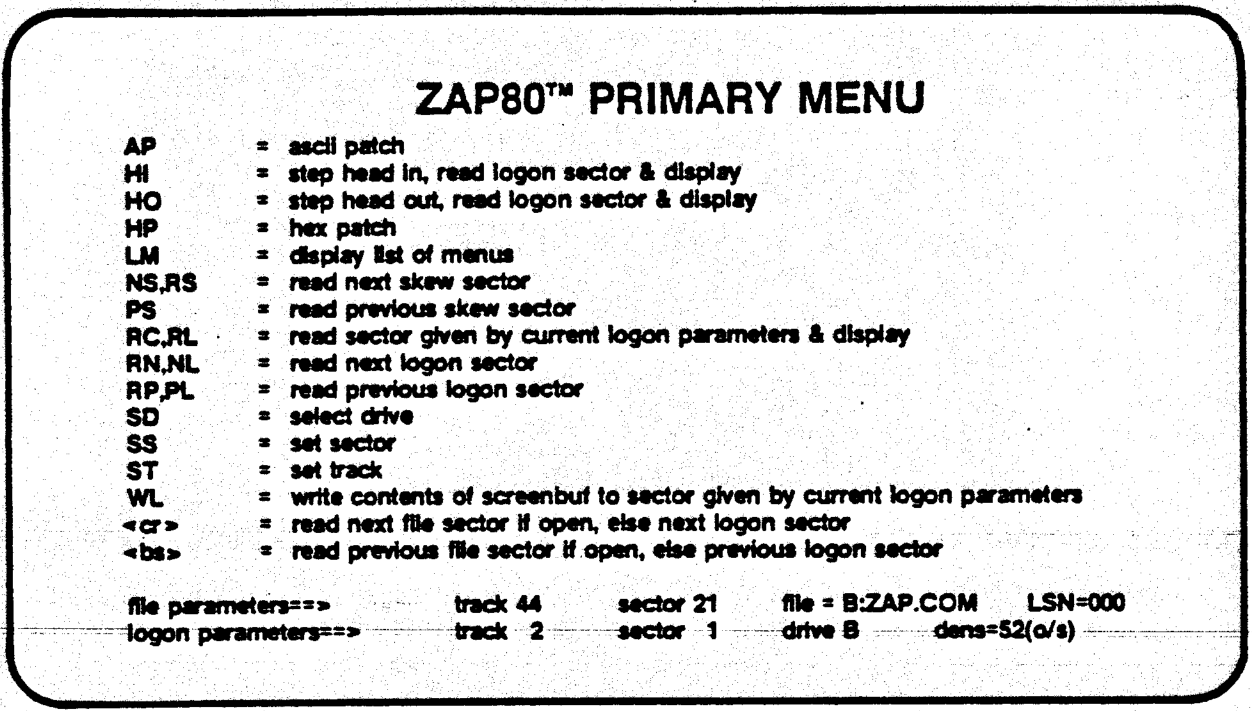
```
0100     C34002                   jmp    zapstrt
                       ;
0103-015F              ; reserved (version, serial #, copyright)
                       ;
0160     01            frstsn:    db     1                 ;1st sctr # on track, patch to
                                                           ;0 if necessary
```

```
0161  1A          sdsptk:   db   26         ;'sngl dens' # sctrs/trk
0162  34          ddsptk:   db   52         ;'dbl dens' # sctrs/trk
0163  08          numdvs:   db   8          ;# drives on system
0164  4C          tperdk:   db   77         ;# of trks/dsk
0165  02          nmostk:   db   2          ;# of o/s trks (o/s trk offset
                                            ;parm for cp/m 2.0+ & dmados)
0166  0001        deffbs:   dw   100h       ;default filebase address
0168  1C0F        maxlsn:   dw   0f1ch      ;maximum logical sctr # (this
                                            ;is the value for 8" dbl dens)
016A  00          ovrost:   db   0          ;override o/s sctr translation flag
016B  00000000              db   0,0,0,0    ;reserved for expansion
016F  00          cdosfg:   db   0          ;cromemco o/s flag (non-0 causes
                                            ;jump around get vers # fn call)
                  :
                  :
                  :
                  ucrsup:                   ;user provided cursor up string
                                            ;the following is an example:
0170  02                    db   2          ; # of chars in string
0171  1B                    db   1bh        ; 1st char =  esc
0172  41                    db   'A'        ; 2nd char = 'A'
0173  0000000000            db   0,0,0,0,0  ; room for 3rd-7th chars
                  :
                  uclrsc:                   ;user provided clear screen
                                            ;the following is an example:
0178  01                    db   1          ; # of char in string
0179  0C                    db   0ch        ; 1st char = ascii formfeed
017A  0000000000            db   0,0,0,0,0  ; room for 2nd-6th chars
                  :
017F  00                    db   0          ;reserved
                  :
                  :
                  ;user provided address cursor strings
                  ;
                  ; normal operation is:  lead-in string , followed by
                  ;  col + col offset , followed by  row + row offset ;
                  ;
                   accommodation is also made for an intermediate string
                  ; (between the col and row outputs), a trailing string,
                   and provision for row being sent before col (patch
                  ; 'cb4rfg' to non-zero).

0180  =           uadcrs:   equ  $          ;user provided addr cursor strings
                                            ;the following is an example:
                                            ; sends  esc,'Y',c+20h,r+20h
                  crsldi:                   ;cursor lead-in string
0180  00                    db   2          ; # of chars in string
0181  1B                    db   1bh        ; 1st char =  esc
0182  59                    db   'Y'        ; 2nd char = 'Y'
0183  0000000000            db   0,0,0,0,0  ; room for 3rd-7th chars
                  crsint:                   ;intermed string betw row & col
0188  00                    db   0          ; null string in this example
0189  0000000000            db   0,0,0,0,0  ; room for 7 chars
018E  0000                  db   0,0
                  crstrl:                   ;cursor trail string
```

```
0190   00                    db   0           ; null string in this example
0191   0000000000            db   0,0,0,0,0   ; room for 7 chars
0196   0000                  db   0,0

0198   00          cb4rfg:   db   0           ;non-zero to send col before row
0199   20          rowoff:   db   20h         ;add this # to row (0=top)
019A   20          coloff:   db   20h         ;add this # to col (0=left)
019B   0000000000            db   0,0,0,0,0   ;room for expansion
                   :
                   :
                   :
                   sdttbl:                    ;sngl dns translation table
01A0   01070D13              db   1,7,13,19
01A4   19050B11              db   25,5,11,17
01A8   1703090F              db   23,3,9,15
01AC   1502080E              db   21,2,8,14
01B0   141A060C              db   20,26,6,12
01B4   1218040A              db   18,2,4,10
01B8   10160000              db   16,22,0,0
01BC   00000000              db   0,0,0,0
                   :
                   :
                   ddttbl:                    ;dbl dns translation table
01C0   01020708              db   1,2,7,8
01C4   0D0E1314              db   13,14,19,20
01C8   191A1F20              db   25,26,31,32
01CC   25262B2C              db   37,38,43,44
01D0   31320304              db   49,50,3,4
01D4   090A0F10              db   9,10,15,16
01D8   15161B1C              db   21,22,27,28
01DC   21222728              db   33,34,39,40
01E0   2D2E3334              db   45,46,51,52
01E4   05060B0C              db   5,6,11,12
01E8   11121718              db   17,18,23,24
01EC   1D1E2324              db   29,30,35,36
01F0   292A2F30              db   41,42,47,48
01F4-  01FF                                   ;room for more sectors
                   :
                   vctrtb:                    ;BIOS jmp vector table
0200-  0232                  db   0,0,...
0233-  023F                                   ;room for expansion
                   :
                   zapstrt:
0240   31xxxx                lxi  sp,stack    ;set up local stack
   .
   .
   .
```



```
                    ZAP80™ PRIMARY MENU
AP          =  ascii patch
HI          =  step head in, read logon sector & display
HO          =  step head out, read logon sector & display
HP          =  hex patch
LM          =  display list of menus
NS,RS       =  read next skew sector
PS          =  read previous skew sector
RC,RL       =  read sector given by current logon parameters & display
RN,NL       =  read next logon sector
RP,PL       =  read previous logon sector
SD          =  select drive
SS          =  set sector
ST          =  set track
WL          =  write contents of screenbuf to sector given by current logon parameters
<cr>        =  read next file sector if open, else next logon sector
<bs>        =  read previous file sector if open, else previous logon sector

file parameters==>      track 44     sector 21     file = B:ZAP.COM     LSN=000
logon parameters==>     track  2     sector  1     drive B         dens=52(o/s)
```

AP ——— Ascii Patch

Patch the current sector in screenbuf in ascii. This command prompts for a byte number if a logon sector is in screenbuf, or an address if a file sector is in screenbuf. Typing a <cr> or a <sp> in response to the prompt sets the patch to the beginning of the sector. The command will then perform a clear screen, redisplay the entire sector, and then display the sector again with the bytes preceding the specified byte blanked out and the cursor positioned at the byte specified.

Full cursor control is provided within the patch sector (see the Special Functions Menu). To end the patch, use the <esc> key. If a patch is attempted past the end of the sector, the patch will be automatically ended: however, cursoring past the end will position the cursor at the beginning of the sector. Furthermore, cursoring over any blanked out byte will redisplay the byte.

HI ——— Head In

Using the logon parameters, step the head in one track (i.e., track=track+1), read the sector, and display.

HO ——— Head Out

Same as HI, but reverse.

HP ——— Hex Patch

Same as AP, but in hex. See discussion for AP.

LM ——— List Menus

Display list of menus. Same display as signon.

NS,RS ——— Read Next Skew Sector

Using the logon parameters and the sector translation table, read the next skew sector and display. CP/M 1.4 defaults to the ZAP80 internal tables. For further discussion, see the section on translation tables within the system configuration section, as well as the toggle density (^D) function in the Miscellaneous Menu.

PS ——— Read Previous Skew Sector

Same as NS, but reverse.

RC,RL ——— Read Current Logon Sector

Using the logon parameters, read the current sector and display.

RN,NL ——— Read Next Logon Sector

Using the logon parameters, read the next sector and display. This command uses the first sector number and number of sectors per track parameters to determine if the current sector is last sector on the track. If so, it sets the current sector to the first sector number, and then tests for next track overflow using the tracks per disk parameter. If an overflow is found, the sector remains set to the first sector number, and the track to the last track.

Care must therefore be taken on double density diskettes where the first and/or second tracks are single density and the remainder of the disk is double density. Let's assume, for instance, the operating system is CP/M 2.2, the disk is set up with track 0 as single density, with the rest as double density, and the BIOS is set up for standard double density parameters (52 sectors per track). If the current logon parameters indicate sector 26, track 0 and you attempt an RN command, you will get an operating system read error. In this case, you must use the SS and ST commands to read the next sector.

RP,PL ——— Read Previous Logon Sector

Same as RN, but reverse. See the discussion for RN.

SD ——— Select Drive

This command uses the number of drives parameter in the system configuration table to check for overflow (ZAP80 is distributed with the default set to 8 drives). However, this is an internal check only. Since the actual select drive is made with a standard bdos call, the usual operating system constraints apply.

For DMADOS 8.0+, since there is no direct system call to obtain the parameter table address, an additional call is made using the BIOS jump table directly to get that address.

For CP/M 2.0+ and DMADOS 8.0+, if the toggle density command (^D) has been used to override the operating system parameters, they can be reset by reselecting the drive.

SS ——— Set Sector

Set the logon parameters to the specified sector. This command uses the first sector number and the sectors per track parameters to test for a valid sector number.

ST ——— Set Track

Set the logon parameters to the specified track. This command uses the tracks per disk parameter to test for a valid track number.

WL ——— Write (Current) Logon Sector

Using the logon parameters, write the contents of screenbuf to the current sector. This command issues a prompt to ensure that it is the logon sector that is desired for the write.

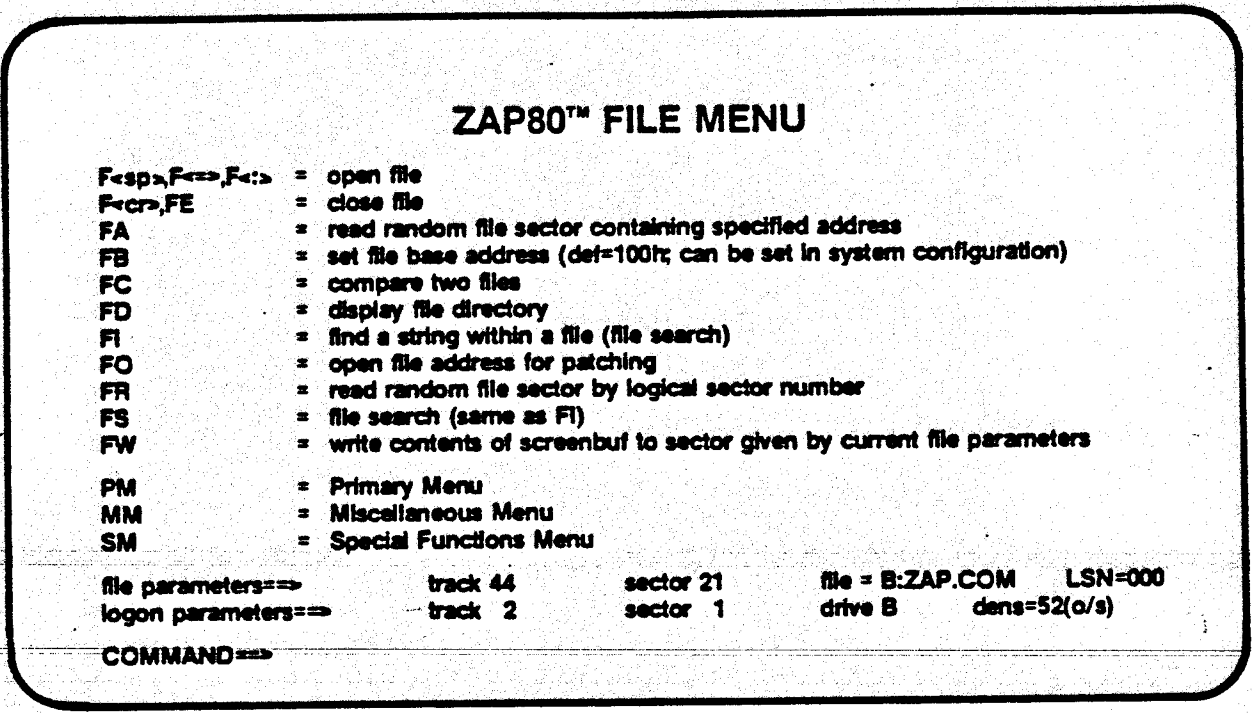
<cr> ——— (read next file or logon sector)

If a file is open, this command reads and displays the next file sector; otherwise the next logon sector. If an attempt is made to read past the end of a file, a warning prompt is issued.

<bs> ——— (read previous file or logon sector)

If a file is open; this command reads and displays the previous file sector; otherwise the previous logon sector. An attempt to read through the beginning of a file simply rereads the first sector.

— This page intentionally left blank —



ZAP80™ FILE MENU

```
F<sp>,F<=>,F<:>   =  open file
F<cr>,FE          =  close file
FA                =  read random file sector containing specified address
FB                =  set file base address (def=100h; can be set in system configuration)
FC                =  compare two files
FD                =  display file directory
FI                =  find a string within a file (file search)
FO                =  open file address for patching
FR                =  read random file sector by logical sector number
FS                =  file search (same as FI)
FW                =  write contents of screenbuf to sector given by current file parameters

PM                =  Primary Menu
MM                =  Miscellaneous Menu
SM                =  Special Functions Menu

file parameters==>       track 44      sector 21      file = B:ZAP.COM    LSN=000
logon parameters==>      track  2      sector  1      drive B        dens=52(o/s)

COMMAND==>
```

F<sp>,F<=>,F<:> ——— open file

Open a file for processing. Type F, followed by any of the three delimiters above, followed by an optional [drive:], followed by a specific <filename.typ>. The file does not have to be on the logged on drive to open it.

F<cr>,FE ——— close file

ZAP80 maintains 2 flags for physical writes to the disk surface (see the discussion on ZAP80 Operation). This command performs an explicit function call to close the file if the file has been written to (the flag is on), otherwise, it simply clears the internal ZAP80 file control block to prevent further access to the file.

FA ——— File Address

If a file has not been opened, this command displays a 'no file open' message. Otherwise, it prompts for an address, and then reads and displays the file sector containing the specified address, returning the file parameters for the sector.

FB ——— File Base

This command allows the user to dynamically set a file base address for ZAP80 without altering the default value, which is normally 100h (the usual beginning address of the transient program area, or tpa). This address is then used as the beginning file address for displaying the file sectors.

Although the default value can be altered in the system configuration table, care should be taken when doing so, as the LD (load) command uses the default address to load a program for chaining, and this location MUST be the one expected by the operating system.

FC ——— File Compare

Compare two files. This command prompts for three display conditions:

1. display the file parameters?
2. display the sectors?
3. stop for differences found in a sector?

The default for all three is 'yes' if the response is anything but 'N' or 'n'. The first displays the file parameters for each sector if set to 'yes'. The second alternately displays 16 bytes (in both hex and ascii) from the first file and then 16 bytes from the second file. All of the bytes from the second file which are identical to the corresponding byte from the first file are blanked out. The display will not stop (pause) unless the third is set to yes. Furthermore, if the stop is set to 'yes' and the sector display is set to 'no', and differences are found (causing a 'stop'), that sector will be displayed for examination.

The compare is resumed by pressing anything but 'E', '?', 'S', or 'C':

E ——— causes an immediate end to the compare, with a display of the number of differences found to that point.

? ——— displays the break options.

S ——— allows any of the three conditions above to be 'set' to 'yes':

- SF ——— sets the file parameter display
- SD ——— sets the sector display
- SS ——— sets the stop for differences found

C ———allows any of the three conditions above to be 'cancelled' ('set' to 'no'):

- CF ——— cancels the file parameter display
- CD ——— cancels the sector display
- CS ——— cancels the stop for differences found

Further control over the compare operation is given by providing a 'break' function which is initiated by pressing either <esc> or ^S. That is, even if all three of the display conditions above are set to 'no', pressing <esc> or ^S will cause the compare to temporarily halt. At that point, pressing 'E', '?', 'S', or 'C' is recognized and processed as described above. Pressing any other key will resume the compare with the conditions unchanged.

If the compare command is allowed to terminate normally, it will display which of the two files, if either, is longer, and how many differences are found (up to 255); otherwise, only the number of differences found up to the termination point are displayed. In either case, if there were no differences, both files are closed; otherwise, the second file is left open, pointing to the last sector read so that it can be patched, if desired.

FD ——— File Directory

This command displays the directory of any specified drive. Type:

FD<cr> ——— for the current drive
FD<sp><drive> ——— for any other drive

The command displays the standard information from the directory file control block: filename, filetype, extent number, and number of 128-byte pages (converted to decimal).

FI ——— Find

This command finds either a hex or ascii string within a file, and leaves the file open to the first byte of the string for patching. The maximum length of the string is 20 bytes, and if the string splits two sectors, ZAP80 will still properly identify and locate its position at the first byte.

The first prompt is for the code type (hex or ascii), the second for the string itself. ZAP80 maintains the most recent code type and string buffer inputs, so that inputting a <cr> or <sp> after the first prompt will repeat the most recent find, if there has been one. The 'R' (repeat) option is provided to accomplish the same thing immediately after a successful find.

FO ——— File Open (Address)

This command 'opens' a file address for patching, and is most useful for patching a program source file for which a listing containing the file addresses is available (see the discussion accompanying the FB (file base) command).

If a file has not been previously opened, the command first prompts for a filename and opens the file before prompting for the file address.

FR ——— File Read

This command reads the random file sector specified after the prompt for a logical sector number. Inputting a <cr> or <sp> reads the sector corresponding to the LSN display on the file parameter line.

After reading the input, ZAP80 calculates the proper extent containing the LSN, and attempts to open that extent. If it is beyond the end of the file, the extent is decremented, and the command reads forward until the highest numbered extent is opened. For this reason, it is prudent to have some idea of the size of the file, in terms of the number of sectors. See the discussion of the maximum logical sector number parameter in the system configuration section.

FS ——— File Search

Same as the FI (find) command.

FW ——— File Write

This command writes the contents of screenbuf to the sector given by the current file parameters. There is no prompt given before execution, as there is for the WL (write logon) command.

DM ——— Display Memory

This command allows the display of any location in memory from 0h to 0fffh. The display is in 128 byte blocks, beginning with the address input immediately after 'DM' is input. ZAP80 maintains the last byte accessed by a DM command, so that inputting a <cr> or <sp> after the DM will display the next block.

DP ——— Display Parameters

This command displays the current file and logon parameters. Although these parameters are displayed after nearly all operations are completed, it is particularly useful after the MM (Miscellaneous Menu) and SM (Special Functions Menu) commands, which do not display them.

DS ——— Display Screenbuf

This command displays the current contents of screenbuf, in case it is needed.

^F ——— Fill (temporary buffer)

This command allows the temporary buffer to be filled with a hex character, so that it can subsequently be written to a sector on disk.

GB ——— Get Buffer

This command retrieves the contents of the temporary buffer and places it into screenbuf, again allowing for subsequent writing to a sector on disk.

GI ——— Get Initbuf

This command gets an internal buffer of e5h's and places it in screenbuf for subsequent writing to a sector on disk. It accomplishes the same thing as using the ^F (fill) command followed by a subsequent GB (get buffer) command.

ID ——— Initialize Drive (with e5h's)

This command is most useful for initializing the directory tracks on disks which have been acquired that contain 'junk' bytes in the directory. However, it also allows all of the data tracks to be initialized for users who like to ensure 'clean' data tracks.

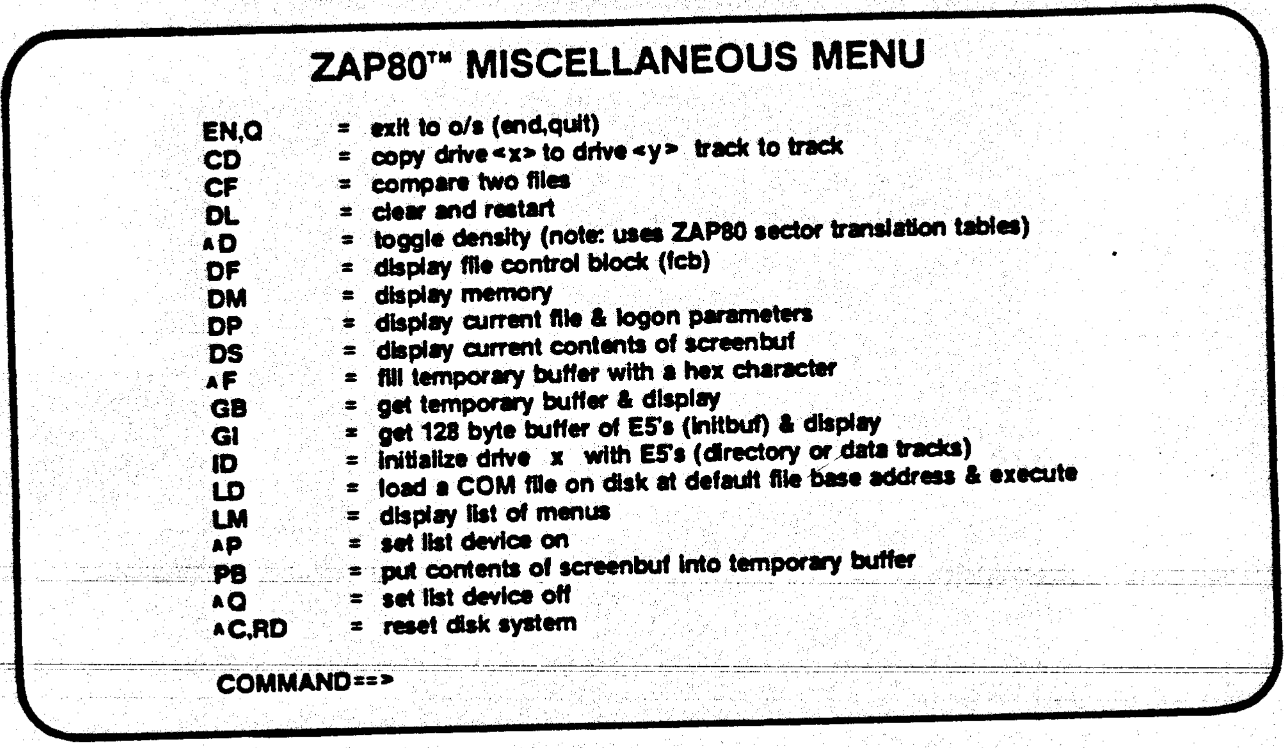
LD ——— Load

This command allows chaining directly from ZAP to a COM file on disk. It loads a COM file specified after the prompt at the default file base address, pushes a jump to 0h onto the stack (for programs which exit via a 'ret' instruction), and jumps to the default file base address. See the discussion accompanying the file base address on the File Menu and in the system configuration section.

After keying in 'LD', the command prompts with a space, after which the filename is entered. Type:

```
LD <filename>
```

The command will not prevent you from entering the type extension along with the filename; however, it automatically overwrites the type field with 'COM', and prompts with a 'COM file not found' message if appropriate.



```
ZAP80™ MISCELLANEOUS MENU

EN,Q     =  exit to o/s (end,quit)
CD       =  copy drive<x> to drive<y>  track to track
CF       =  compare two files
DL       =  clear and restart
^D       =  toggle density (note: uses ZAP80 sector translation tables)
DF       =  display file control block (fcb)
DM       =  display memory
DP       =  display current file & logon parameters
DS       =  display current contents of screenbuf
^F       =  fill temporary buffer with a hex character
GB       =  get temporary buffer & display
GI       =  get 128 byte buffer of E5's (initbuf) & display
ID       =  initialize drive  x  with E5's (directory or data tracks)
LD       =  load a COM file on disk at default file base address & execute
LM       =  display list of menus
^P       =  set list device on
PB       =  put contents of screenbuf into temporary buffer
^Q       =  set list device off
^C,RD    =  reset disk system

COMMAND==>
```

EN,Q ——— exit to o/s (end,quit)

This command logs on the original drive upon entry to ZAP, and performs a jump to the warm boot routine (0h).

CD ——— Copy Drive

This command performs a track to track copy from one drive to another. There is a clear warning that any existing data will be destroyed, and a chance to abort after the prompts and before actual execution.

Since it is a track to track copy, both drives must have the same characteristics (single or double density, number of sectors per track), although no internal check is made. If you blow it, ZAP blows it.

CF ——— Compare Files

Same as FC (file compare) command on the File Menu.

CL ——— Clear & Restart

This command reinitializes ZAP and restarts, with the exception that any file input from the console command line will not be reopened on restart. It must be reopened after the CL, if desired.

^D ——— Toggle Density

This command overrides any o/s parameters if the operating system is cp/m 2.0+ or dmados 8.0+, and uses the ZAP80 internal system configuration parameters. See the extensive discussion in the system configuration section.

DF ——— Display FCB (file control block)

This command displays the current contents of the internal file control block maintained by ZAP80 for the 'file parameter line' commands.

DM ——— Display Memory

This command allows the display of any location in memory from 0h to 0ffffh. The display is in 128 byte blocks, beginning with the address input immediately after 'DM' is input. ZAP80 maintains the last byte accessed by a DM command, so that inputting a <cr> or <sp> after the DM will display the next block.

DP ——— Display Parameters

This command displays the current file and logon parameters. Although these parameters are displayed after nearly all operations are completed, it is particularly useful after the MM (Miscellaneous Menu) and SM (Special Functions Menu) commands, which do not display them.

DS ——— Display Screenbuf

This command displays the current contents of screenbuf, in case it is needed.

^F ——— Fill (temporary buffer)

This command allows the temporary buffer to be filled with a hex character, so that it can subsequently be written to a sector on disk.

GB ——— Get Buffer

This command retrieves the contents of the temporary buffer and places it into screenbuf, again allowing for subsequent writing to a sector on disk.

GI ——— Get Initbuf

This command gets an internal buffer of e5h's and places it in screenbuf for subsequent writing to a sector on disk. It accomplishes the same thing as using the ^F (fill) command followed by a subsequent GB (get buffer) command.

ID ——— Initialize Drive (with e5h's)

This command is most useful for initializing the directory tracks on disks which have been acquired that contain 'junk' bytes in the directory. However, it also allows all of the data tracks to be initialized for users who like to ensure 'clean' data tracks.

LD ——— Load

This command allows chaining directly from ZAP to a COM file on disk. It loads a COM file specified after the prompt at the default file base address, pushes a jump to 0h onto the stack (for programs which exit via a 'ret' instruction), and jumps to the default file base address. See the discussion accompanying the file base address on the File Menu and in the system configuration section.

After keying in 'LD', the command prompts with a space, after which the filename is entered. Type:

```
LD <filename>
```

The command will not prevent you from entering the type extension along with the filename; however, it automatically overwrites the type field with 'COM', and prompts with a 'COM file not found' message if appropriate.

LM ——— List Menus

This command redisplays the list of menus screen which immediately followed the initial signon to ZAP.

▲P ——— Printer On

This command sets the list device on. All subsequent bytes sent to the console by ZAP are sent to the list device as well, until a ▲Q (list device off) command is received.

PB ——— Put Buffer

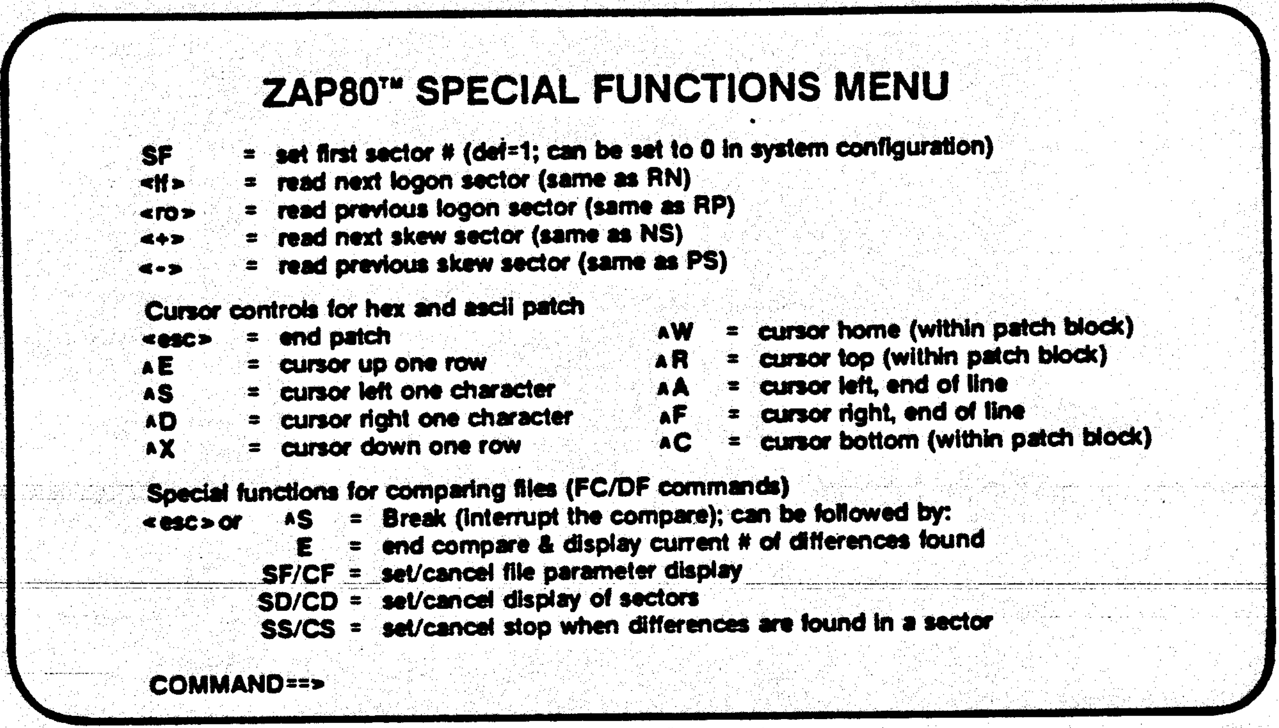
This command puts the current contents of screenbuf into the temporary buffer for subsequent retrieval.

▲Q ——— Printer Off

This command sets the list device off.

▲C,RD ——— Reset Disk System

This command performs a standard reset function call.



```
                ZAP80™ SPECIAL FUNCTIONS MENU

SF       =  set first sector # (def=1; can be set to 0 in system configuration)
<lf>     =  read next logon sector (same as RN)
<ro>     =  read previous logon sector (same as RP)
<+>      =  read next skew sector (same as NS)
<->      =  read previous skew sector (same as PS)

Cursor controls for hex and ascii patch
<esc>    =  end patch                        ^W  =  cursor home (within patch block)
^E       =  cursor up one row                ^R  =  cursor top (within patch block)
^S       =  cursor left one character        ^A  =  cursor left, end of line
^D       =  cursor right one character       ^F  =  cursor right, end of line
^X       =  cursor down one row              ^C  =  cursor bottom (within patch block)

Special functions for comparing files (FC/DF commands)
<esc>or     ^S  =  Break (interrupt the compare); can be followed by:
             E  =  end compare & display current # of differences found
         SF/CF  =  set/cancel file parameter display
         SD/CD  =  set/cancel display of sectors
         SS/CS  =  set/cancel stop when differences are found in a sector

COMMAND==>
```

SF ——— Set First (sector number)

This command allows the first sector number parameter to be dynamically altered without changing the default value. See the discussion in the system configuration section.

For a discussion of the rest of the commands on this menu, see the accompanying commands on the appropriate menu:

- the read command is found on the Primary Menu
- the patch commands are also found on the Primary Menu, with the following additional cursor controls:
  - <sp> functions the same as ^D for hex only (left one char)
  - <bs> functions the same as ^S (right one char)
- The file compare command is found on the File Menu