Z R D O S

Version 1.0

Echelon Z-System Disk Operating System

PROGRAMMER'S GUIDE

by

Dennis L. Wright

Z R D O S

Version 1.0

Echelon Z-System Disk Operating System

PROGRAMMER'S GUIDE

by

Dennis L. Wright

1 January 1985

ZRDOS is a Z80 coded CP/M 2.2 compatible Disk Operating System. Use of Z80 code allows addition of many new features. This document explains these features and differences between ZRDOS Version 1.0 and CP/M 2.2.

ZRDOS PLUS

ADDENDUM

TO ZRDOS VERSION 1.0 PROGRAMMER'S GUIDE

ZRDOS Plus is the re-entrant version of ZRDOS Version 1.0. All features and function calls are identical to those outlined in the ZRDOS Version 1.0 Programmer's Guide with the following addition.

Programs that intercept BIOS calls from ZRDOS Plus can be written to make calls to ZRDOS without destroying the original DOS callers pointers and parameters.

Re-entrance can be accomplished by first saving the current ZRDOS Plus buffers. This is done by copying the ZRDOS Plus buffers to a user assigned save buffer area of 147 bytes. Once the DOS data has been saved the user program is free to make any DOS calls necessary. Before returning to the original DOS caller, the ZRDOS Plus buffers must be restored. The beginning of the ZRDOS Plus parameter-Buffer area is located at ZRDOS Plus base + 5 (ZRDOS Plus base is the address specified in system page zero, hex location 06) and is 147 bytes in length.

The main purpose of making ZRDOS re-entrant is to allow the efficient programming of ZCPR3 I/O Packages, packages (modules of 1.5k-bytes length) that redirect Device Record (Console, List, Reader and Punch) input and output to and from disk files. Echelon, Inc. offers several IOPs that make use of this feature.

EXAMPLE METHOD OF SAVING AND RESTORING ZRDOS PLUS BUFFERS

The following routines demonstrate a method that can be used to save and restore the ZRDOS Plus buffers to allow re-entrant calls to ZRDOS Plus.

```
;
DOS     EQU     5
BUFOFF  EQU     5         ; Offset from beginning of ZRDOS Plus to
                          ; internal dos buffers.
;
; This routine gets the address of the ZRDOS Plus parameter buffer.
;
GETBUF: LHLD    06        ; Get dos address.
        LXI     D,BUFOFF  ; Add offset to ZRDOS Plus internal buffer.
        DAD     D
        SHLD    DOSBUF    ; Save as dosbuf pointer.
        RET
;
; This routine saves ZRDOS Plus parameters to allow re-entry.
;
SAVDOS: LHLD    DOSBUF    ; Save ZRDOS Plus parameter buffer.
        LXI     D,DOSSAV
        CALL    MOVIT
        MVI     C,47      ; Function 47, get current dma address.
        CALL    DOS
        SHLD    CURDMA    ; Save it.
        RET
;
; This routine restores original ZRDOS Plus parameters.
;
RSTDOS: LDED    CURDMA    ; Restore dma address
        MVI     C,26      ; Function 26, set dma address.
        CALL    DOS
        LXI     H,DOSSAV  ; Restore ZRDOS Plus parm buffer.
        LDED    DOSBUF
MOVIT:  LXI     B,147     ; Move 147 bytes.
        LDIR
        RET
;
CURDMA: DW      0         ; Save area for current DMA address.
DOSBUF: DW      0         ; Save area for pointer to ZRDOS Plus parms.
DOSSAV: DS      147       ; Save area for ZRDOS Plus parm buffer.
;
```

T A B L E O F C O N T E N T S

L I S T O F F I G U R E S

L I S T O F T A B L E S

Z R D O S

Version 1.0

Echelon Z-System Disk Operating System

PROGRAMMER'S GUIDE

by

Dennis L. Wright

1. DIFFERENCES FROM THE STANDARD CP/M CCP

ZRDOS is a Z80 coded, CP/M 2.2 compatible disk operating system designed to be used with Echelon Z80/HD64180 Command Processor ZCPR3 written by Richard Conn and auto-install Z3-Dot-Com by Joseph Wright.

1.1. ZCPR3 Utilities and Features

ZRDOS is compatible with all of the ZCPR3 utilities. Named Directories, Redirectable I/O, and all other ZCPR3 features can be used with ZRDOS. For installation of these additional features please refer to original ZCPR3 documentation package.

2. DIFFERENCES FROM CP/M 2.2 BDOS

2.1. Disk Change

ZRDOS allows the changing of disks without the need of a warm boot. CHANGED DISKS WILL BE AUTOMATICALLY LOGGED IN.

NOTE: The above mentioned auto login will not occur if a file was open when the disk was changed and a write operation is attempted to that file on the new diskette instead the following error message will be printed:

Disk Changed Error On Drive B:

It should also be noted that the automatic logon may or may not be able to handle changes in density or number of sides. This depends on how your bios handles deblocking and double sided disks. However if the disk is of the same density and number of sides as the disk it is being swapped with there will be no problems.

2.2. Read Only Disk Status

Under ZRDOS a disk can only be set to R/O status by executing function call 28 (Protect Drive).

ZRDOS function call 37 (Reset Drive) is different in that it will only reset the Read Only bits for the drive(s) specified in the user passed drive map in the register pair (DE). With CP/M function 37 will also reset the bits for any drive not currently active.

Function call 13 (Reset Disk System) is different in that it will not reset drives that are set to Read Only but will instead reset the disk changed vector.

2.3. Read Console Buffer

The Read console buffer routine (Function 10) for ZRDOS is different in the following ways:

- o Rubout (DEL) is treated the same as a backspace.
- o The Control-R edit function is not implemented.

NOTE: In CP/M these are teletype oriented edit commands and were felt not to be desired in ZRDOS.

2.4. File Archiving

ZRDOS supports the use of the file archive attribute. The support of this feature is compatible with both CP/M 3.0 and MP/M. This bit when set indicates an archived file. That is a file which has not been altered. The bit can be set by using a Function 30 (Set File Attributes) function call. Any update to this file once the bit has been set will cause the bit to be reset. This can then be used by a copy utility to indicate the need to backup the file. The utility that backs up the file should then set the archive bit to indicate the file has been backed up.

2.5. Wheel Protection

ZRDOS uses the ZCPR3 wheel byte and a new file attribute bit to protect files from non-wheel users. This bit when set will write protect the file as long as the wheel byte is off. If the wheel byte is set the file is treated as a normal file.

If a non-wheel user attempts to change a wheel protected file the following error message will be displayed:

```
A0>File W/P Error on A:
```

2.6. Error Messages

Error Number	ZRDOS	CP/M
1	Read Error On A:	Bdos Err On A:Bad Sector
2	Drive Select Error On A:	Bdos Err On A:Select
3	Disk R/O Error On A:	Bdos Err On A:R/O
4	File R/O Error On A:	Bdos Err On A:File R/O
5	Disk Changed Error On A:	n/a
6	File W/P Error On A:	n/a

With ZRDOS all non-retryable errors jump directly to warm boot after the error message has been printed. Read Errors allow the user the option of retrying the operation by pressing any key but control-c or aborting by pressing control-c. Error numbers shown above are returned in the (A) register. The selected drive number is returned in register (E).

NOTE: If the warm boot trap (see FUNCTION 50) is set ZRDOS jumps directly to the warm boot vector and no error messages are displayed. (User program stack pointer not returned.)

CP/M handles the errors in the same manner except a key must be pressed before CP/M will return from any type of error and CP/M doesn't return an error or drive number. Nor does CP/M have a warm boot trap function.

3. ZRDOS EXTENDED FUNCTION CALLS

3.1. Function 47: Get Current DMA Address

This function will return the currently assigned DMA address in the register pair (HL).

3.2. Function 48: Return ZRDOS Version Number

This function works the same as CP/M function call 12 except the ZRDOS Version number is returned instead of the CP/M version number. To maintain CP/M compatibility ZRDOS will return version number 2.2 on a function 12 call. As with function 12 function 48 uses the register pair (HL) to return the version number. If user programs that use the extended ZRDOS functions are written this function should first be used to determine if the program is being run under ZRDOS.

3.3. Function 50: Set Warm Boot Trap

A new function call is provided that allows the user to set a trap on warm boot to a user specified address. The trap is set by executing a function 50 call with the trap address in the register pair (DE). The Warm boot jump address at location 001H is replaced with the user supplied trap address. Warm boots executed after the trap is set will cause a jump to the trap address. ZRDOS error messages are suppressed allowing the user to print his own error messages. As noted in the ZRDOS error section above errors detected by ZRDOS return an error number and the active drive number which the user can then use to determine how best to handle the error.

WARNING: Caution should be exercised when using this function as the results will be unpredictable if a program that has set the trap terminates without first resetting the trap. See FUNCTION 52 below.

3.4. Function 52: Reset Warm Boot Trap

This new function call will reset the warm boot trap set by function call 50. The trap is reset by executing a function call 52. The Real bios warm boot address is restored to location 0001H. If function call 50 is used in a user program a function call 52 should be executed before control is returned to the operating system.

4. ZRDOS VERSION 1.0 FUNCTION CALLS

FUNCTION NUMBER	DESCRIPTION OF ZRDOS OPERATION PERFORMED
0	System Reset
1	Console Input
2	Console Output
3	Reader Input
4	Punch Output
5	List Output
6	Direct Console I/O
7	Get I/O Byte
8	Set I/O Byte
9	Print String
10	Read console Buffer
11	Get Console Status
12	Return Version Number (CP/M)
13	Reset Disk System
14	Select Disk
15	Open File
16	Close File
17	Search For First
18	Search For Next
19	Delete File
20	Read Sequential
21	Write Sequential
22	Make File
23	Rename File
24	Return Login Vector
25	Return Current Disk
26	Set DMA Address
27	Get Allocation Vector Address
28	Write Protect Disk
29	Get Read/Only Vector
30	Set File Attributes
31	Get Disk Parameter Block Address
32	Set/Get User Code
33	Read Random
34	Write Random
35	Compute File Size
36	Set Random Record
37	Reset Drive
40	Write Random With Zero Fill
47	Return Current DMA Address
48	Return Version number (ZRDOS1)
50	Set Warm Boot trap
52	Reset Warm Boot trap

Table 4-1: ZRDOS Version 1.0 Function Calls

4.1. FUNCTION 0: SYSTEM RESET

Entry Parameters	Returned Value
Register (C): 00H	None

Function to terminate program and reset the system. Same results as performing a jump to location 0000H. The disk system is reset; that is disks marked as changed are cleared and the directory check information is discarded.

4.2. FUNCTION 1: CONSOLE INPUT

Entry Parameters	Returned Value
Register (C): 01H	Register (A): ASCII Character

Function to get character from console device. A byte from the device currently assigned to CON: is returned in register (A). The byte is echoed to the terminal. If no byte is ready at the time the call is made, the calling program is suspended until a byte becomes available.

4.3. FUNCTION 2: CONSOLE OUTPUT

Entry Parameters	Returned Value
Register (C): 01H Register (E): ASCII Character	None

Function to output character in register (E) to the device currently assigned to CON: and expand tabs if necessary. A Control-S (pause) and Control-P (echo to printer) console input test is also performed.

4.4. FUNCTION 3: READER INPUT

Entry Parameters	Returned Value
Register (C): 03H	Register (A): ASCII Character

Function to get character from the reader device. The next byte from the device currently assigned to RDR: is returned in register (A). All 8 bits are returned. The calling program is suspended until a byte is ready.

4.5. FUNCTION 4: PUNCH OUTPUT

Entry Parameters	Returned Value
Register (C): 04H Register (E): ASCII Character	None

Function to output a character to the punch device. The byte in register (E) is sent to the device currently assigned to PUN:. The program is suspended until the device is ready to accept the byte.

4.6. FUNCTION 5: LIST OUTPUT

Entry Parameters	Returned Value
Register (C): 05H Register (E): ASCII Character	None

Function to output a character to the list device. The byte in register (E) is sent to the device currently assigned to LST:. The program is suspended until the device is ready to accept the byte.

4.7. FUNCTION 6: DIRECT CONSOLE I/O

Entry Parameters		Returned Value
Register (C): 06H		Register (A): ASCII Char
Register (E): 0FFH	(input)	or
0FEH	(status)	status
ASCII Char	(output)	

Function to perform direct console i/o. If register (E) contains (FF) then this is an input request. If register (E) contains (FE) then this is a status request. Otherwise the character in register (E) will be sent to the device currently assigned to CON:. This request bypasses all control character checks.

4.8. FUNCTION 7: GET I/O BYTE

Entry Parameters	Returned Value
Register (C): 07H	Register (A): I/O Byte Value

Function to return the i/o byte. The current value of the IOBYTE (memory location 0003H) is returned in register (A).

4.9. FUNCTION 8: SET I/O BYTE

Entry Parameters	Returned Value
Register (C): 08H Register (E): I/O Byte Value	None

Function to set the i/o byte. The value in (E) is set as the current IOBYTE (memory location 0003H). It changes control of the output direction immediately.

The I/O byte located at memory location 0003H is made up of the four fields shown in the following table:

IOBYTE:	7 6	5 4	3 2	1 0
bit value	list (LST:)	punch (PUN:)	reader (RDR:)	console (CON:)
00	TTY:	TTY:	TTY:	TTY:
01	CRT:	PTP:	PTR:	CRT:
10	LPT:	UP1:	UR1:	BAT:
11	UL1:	UP2:	UR2:	UC1:

Table 4-2: IOBYTE Format

4.10. FUNCTION 9: PRINT STRING

Entry Parameters	Returned Value
Register (C): 09H Register (DE): String address	None

Function to send the character string pointed to by (DE) to the device currently assigned to CON:. The printing of the string to the console device will continue until a '$' is encountered in the string. Console input control character checks are made and tabs are expanded.

4.11. FUNCTION 10: READ CONSOLE BUFFER

Entry Parameters	Returned Value
Register (C): 0AH Register (DE): Buffer address	Console Characters in Buffer

Function to execute a buffered read. ZRDOS notes the current cursor position as it knows it, then reads characters from the console device until a CR or LF is received, or until the maximum number of characters have been received.

ZRDOS unlike standard bdos treats the rubout (del) key the same as a backspace. Also the control-R function has been eliminated.

The form of the read buffer is as follows:

BASE = Address in (DE)

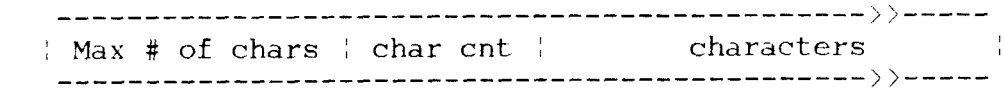

Fig. 4-1: Console Buffer Format

4.12. FUNCTION 11: GET CONSOLE STATUS

Entry Parameters	Returned Value
Register (C): 0BH	Register (A): Console Status

Function to interrogate the console device. The device currently assigned to CON: is polled. If a byte is ready for input, a nonzero value is returned in register (A), otherwise 00H is returned.

4.13. FUNCTION 12: RETURN VERSION NUMBER

Entry Parameters	Returned Value
Register (C): 0CH	Register (HL): Version Number

Function to return the current version number. Version number 2.2 is returned to maintain CP/M compatibility. Function 48 should be used to get the ZRDOS version number.

4.14. FUNCTION 13: RESET DISK SYSTEM

Entry Parameters	Returned Value
Register (C): 0DH	Register (A): 0FFH if the current default drive contains a file name beginning with a $ 00H if not.

Function to reset the disk system. All active drives are reset to an unknown condition. Drive A is relogged in and the DMA address is reset to 80H. Unlike CP/M ZRDOS does not reset the read only vector of drives that have been set to read only status but instead resets the ZRDOS disk changed vector for any drives that are marked as changed.

4.15. FUNCTION 14: SELECT DISK

Entry Parameters	Returned Value
Register (C): 0EH Register (E): Selected Disk	None

Function to set the active disk number. Register (E) contains a number in the range 0 - 15, signifying disk A - P respectively. If the selected drive is not the current default drive, it is made the default drive. If it has not been selected since the last warm start or disk reset, its directory is scanned and new allocation and check vectors are built.

4.16. FUNCTION 15: OPEN FILE

Entry Parameters	Returned Value
Register (C): 0FH Register (DE): FCB Address	Register (A): Directory Code

Function to open a specified file. The drive code, if not zero, is used to select a drive. The directory is scanned for the first match to the filename and extent number in the fcb pointed to by the register pair (DE). The filename may contain wildcards. A matching directory entry is then copied into the specified fcb and register (A) is return with the directory sector location code 0 - 3 (See section on Directory Codes). If no match is found register (A) will contain 0FFH.

4.17. FUNCTION 16: CLOSE FILE

Entry Parameters	Returned Value
Register (C): 10H Register (DE): FCB Address	Register (A): Directory Code

Function to close a specified file. The filename and extent number in the fcb pointed to by the register pair (DE) are located in the directory. The file name may contain wildcards. If they are found, the record count and data map from the specified fcb are copied into the directory entry and the directory location code 0 - 3 is returned in register (A). If the filename can not be found 0FFH is returned in register (A).

4.18. FUNCTION 17: SEARCH FOR FIRST

Entry Parameters	Returned Value
Register (C): 11H Register (DE): FCB Address	Register (A): Directory Code

Function to return the first occurrence of a specified file name. The directory of the default drive is scanned for an entry that matches the filename and extent number in the fcb pointed to by the register pair (DE). The filename may contain wildcards. If a match is found, the directory location code 0 - 3 is returned in register (A). If no match is found 0FFH is returned in register (A). If the extent number contains 00H, only the first extent for a file can be matched. If the extent number contains a question mark, the first entry found is returned. If the drive number of the specified fcb contains a question mark, all directory entries of any user code, and entries of any type including those not in use, are compared.

4.19. FUNCTION 18: SEARCH FOR NEXT

Entry Parameters	Returned Value
Register (C): 12H Register (DE): FCB Address	Register (A): Directory Code

Function to return the next occurrence of a file name. This function performs the same as function 17 except that the search begins with the entry following the one returned by the last search (function 17 or 18). For this function to work correctly it must be proceeded by a search function.

4.20. FUNCTION 19: DELETE FILE

Entry Parameters	Returned Value
Register (C): 13H Register (DE): FCB Address	Register (A): Directory Code

Function to delete a file by name. The drive code, if not zero, is used to select a drive. The directory is scanned for all entries that match the given filename (which may contain wildcards). Only files in the active user area are considered.

4.21. FUNCTION 20: READ SEQUENTIAL

Entry Parameters	Returned Value
Register (C): 14H Register (DE): FCB Address	Register (A): Directory Code

Function to execute a sequential read of the specified record number. The drive code, if not zero, is used to select a drive. The 128-byte record referenced to by the (cr) byte is read and placed into the current file buffer. The (cr) byte is incremented. If it then equals the (rc) byte, the entire extent has bee read; the directory entry describing the next extent of the file is copied into the FCB and (cr(is zeroed. If there are no further extents the extent map in the referenced fcb is set to zero. If the record is successfully read, register (A) is returned containing 00H. If an end of file occurs, register (A) is returned containing 0FFH. The format of the referenced fcb is shown below:

0	1 - 8	9 - B	C	D	E	F
dr	filename	filetype	ex	s1	s2	rc
extent map						
cr						

Fig. 4-2: FCB format

4.22. FUNCTION 21: WRITE SEQUENTIAL

Entry Parameters	Returned Value
Register (C): 15H Register (DE): FCB Address	Register (A): Directory Code

Function to write the next sequential record. The drive code if not zero, is used to select a drive. If no block has been allocated to the record referenced by (cr) of this extent, one is allocated and entered in the bit map. The record in the current file buffer is written into the position referenced by (cr). The FCB bytes (cr) and (rc) are then incremented. If the extent is then full, the FCB is copied into the matching directory entry and a new entry is made for the next extent, the (cr) and (rc) bytes are reset to zero as is the data map area of the FCB. If the write was successful, 00H is returned in register (A) otherwise a none zero value is returned.

4.23. FUNCTION 22: MAKE FILE

Entry Parameters	Returned Value
Register (C): 16H Register (DE): FCB Address	Register (A): Directory Code

Function to create a file. A directory entry is created for the filename specified by the fcb pointed to by register pair (DE). The newly created entry will contain a pointer to the first extent but with no space allocated to it. Upon return register (A) will contain the Directory Code for the new fcb if the operation was successful or 0FFH if no more directory space is available. A successfully created file can be treated as open.

4.24. FUNCTION 23: RENAME FILE

Entry Parameters	Returned Value
Register (C): 17H Register (DE): FCB Address	Register (A): Directory Code

Function to rename a file. The drive code if not zero, is used to select a drive. The directory is scanned and all entries for the explicit filename in bytes 01H - 0BH of the fcb are changed to that in bytes 11H - 1BH. If no such directory entry is found, 0FFH is returned in register (A) else the Directory Code is returned in register (A).

4.25. FUNCTION 24: RETURN LOGIN VECTOR

Entry Parameters	Returned Value
Register (C): 18H	Register (HL): Login Vector

Function to return the login vector. A bit map of the drives that are currently active is returned in the (HL). The bits of the map stand for drives as follows:

```
                    Register (H)        Register (L)
                         |                   |
                  ---------------     ---------------
Bit numbers:      7 6 5 4 3 2 1 0     7 6 5 4 3 2 1 0
                  | | | | | | | |     | | | | | | | |
Drive ID:         P O N M L K J I     H G F E D C B A
```

Fig. 4-3: Login Vector Bit Map

4.26. FUNCTION 25: RETURN CURRENT DISK

Entry Parameters	Returned Value
Register (C): 19H	Register (A): Current Disk

Function to return the current disk assignment. The disk number of the currently logged in drive is returned in register (A). The number returned ranges from 0 to 15 and corresponds to drives A through P respectively.

4.27. FUNCTION 26: SET DMA ADDRESS

Entry Parameters	Returned Value
Register (C): 1AH	Register (DE): DMA Address

Function to set the dma address to the address supplied in the register pair (DE). The Direct Memory Address used to address a 128 byte file buffer for disk read/write transfers is set to the address specified in the register pair (DE). The default DMA address used by ZRDOS is 0080H.

4.28. FUNCTION 27: GET ADDR(ALLOC)

Entry Parameters	Returned Value
Register (C): 1BH	Register (HL): ALLOC Address

Function to return the allocation vector. The address of the allocation vector for the currently logged in drive is returned in the register pair (HL).

4.29. FUNCTION 28: WRITE PROTECT DISK

Entry Parameters	Returned Value
Register (C): 1CH	None

Function to write protect the current disk. The default drive is set to read-only status. Under ZRDOS1 unlike CP/M the protected drive will retain this status until it is reset with a function call 37 or cold boot.

4.30. FUNCTION 29: GET READ ONLY VECTOR

Entry Parameters	Returned Value
Register (C): 1DH	Register (HL): R/O Vector Value

Function to return the read-only status vector. A bit map of the drives that are currently marked read-only is returned in the (HL). The bits of the map stand for drives as follows:

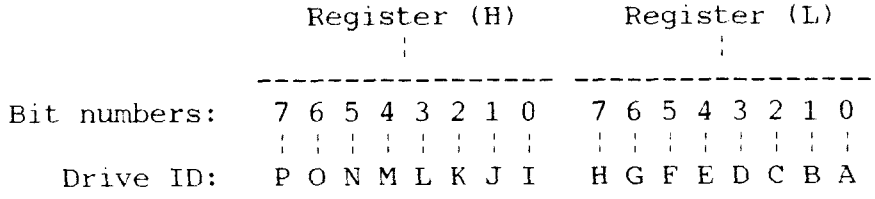

Fig. 4-4: Read Only Vector Bit Map

4.31. FUNCTION 30: SET FILE ATTRIBUTES

Entry Parameters	Returned Value
Register (C): 1EH Register (DE): FCB Address	Register (A): Directory Code

Function to set the file attributes. The attributes f1-f4 can be used by the user for any purpose. The next three are reserved for future use. Attribute t1 is the File Read Only attribute and is used to prevent a file from being written to. The t2 attribute is the system attribute it alerts the ZCPR3 DIR command that this file is not to be displayed. The t3 attribute is the file archive attribute and is used to indicate whether a file has been updated. Attributes are set by turning on the high order bit of the specified byte and reset by turning it off. The f8 attribute is the Wheel Protect attribute. If this bit is set and the ZCPR3 Wheel byte is off the file can not be written to nor can the file's attributes be changed. If the Wheel byte is set the file is treated as any other file.

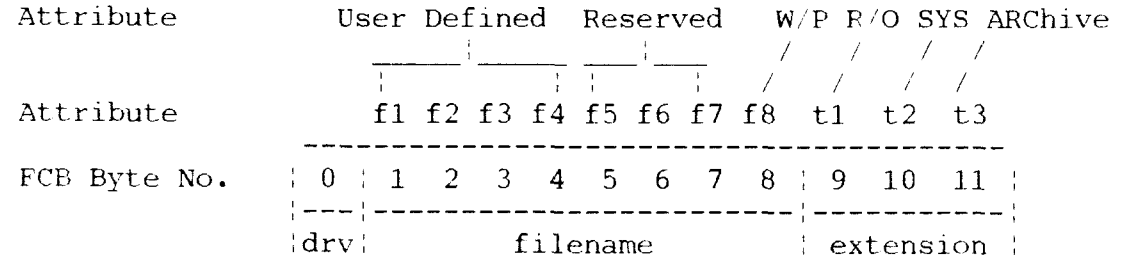

Fig. 4-5: File Attribute Format.

4.32. FUNCTION 31: GET ADDR(DISK PARMS)

Entry Parameters	Returned Value
Register (C): 1FH	Register (HL): DPB Address

Function to return the address of the disk parameter block for the current drive. The address of the Disk Parameter Block is returned in the register pair (HL).

4.33. FUNCTION 32: SET/GET USER CODE

Entry Parameters	Returned Value
Register (C): 20H Register (DE): 0FFH (get) or User Code (set)	Register (A): Current Code or no value

Function to get or set the user number. If (E) was 0FFH then this is a request to return the current user number. Else set the user number from (E).

4.34. FUNCTION 33: READ RANDOM

Entry Parameters	Returned Value
Register (C): 21H Register (DE): FCB Address	Register (A): Return Code

Function to read a random record from a file. The 'r0', 'r1' and 'r2' bytes used to construct the fcb pointer to the specified record number (see fig.4-6 on next page). Unlike a sequential read operation, the record number is not advanced. Thus, if the calling program does not increment the record number subsequent random read operations will continue to read the same record.

As each random read operation automatically sets the extent and record values into the specified fcb the file can then be sequentially read or written, starting from the currently accessed position.

Upon return from a random read operation register (A) contains 00H if the operation was a success or one of the following error codes:

01 - Reading unwritten data
03 - Cannot close current extent
04 - Seek to unwritten extent
06 - Seek past physical end of disk

4.35. FUNCTION 34: WRITE RANDOM

Entry Parameters	Returned Value
Register (C): 21H Register (DE): FCB Address	Register (A): Return Code

Function to write a random record to a file. This operation is similar to the Read Random operation, except that data is written to the specified record from the currently defined DMA address. If the addressed extent or record has not yet been allocated, an automatic allocation will be performed before the data is written.

Upon return register (A) contains 00H if the operation was successful or an error code if not. The error codes are the same as those returned for a Random Read operation with the addition of the following code:

05 - Directory Overflow

For random R/W, the fcb for the desired record number is set per the 'r0,r1,r2' bytes. These bytes in the fcb are used as follows:

	fcb+35	fcb+34		fcb+33
Byte	r2	r1		r0
Bit #	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0		7 6 5 4 3 2 1 0
	overflow	extra extent and 's2'	extent number	record number

Fig. 4-6: Use of FCB bytes 'r0','r1' and 'r2'.

4.36. FUNCTION 35: COMPUTE FILE SIZE

Entry Parameters	Returned Value
Register (C): 23H Register (DE): FCB Address	Random Record Field Set

Function to compute the size of a random file. The directory is scanned to find the highest numbered extent of the filename in the fcb specified by register pair (DE). The direct address of the specified file's last record, plus one, is set in the record address field of the specified fcb.

4.37. FUNCTION 36: SET RANDOM RECORD

Entry Parameters	Returned Value
Register (C): 24H Register (DE): FCB Address	Random Record Field Set

Function to return the random record position of a given file which has been read in sequential mode up to now. The extent number and current record number of the fcb specified by register pair (DE) are used to calculate the direct address of the record returned by the last sequential read operation.

4.38. FUNCTION 37: RESET DRIVE

Entry Parameters	Returned Value
Register (C): 25H Register (DE): Drive Vector	Register (A): 00H

Function to allow a program to log off any drives. On entry, set (DE) to contain a word with bits set for those drives that are to be logged off. The log-in vector and the write protect vector will be updated. Drives to be reset are specified in the register pair (DE) as follows:

	Register (D)								Register (E)							
Bit numbers:	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Drive ID:	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A

Fig. 4-7: Active Drive Vector Bit Map.

NOTE: This function differs from CP/M in that only those drives specified in the (DE) will be unprotected and not drives which are not currently active as is done in CP/M.

4.39. FUNCTION 40: WRITE RANDOM WITH ZERO FILL

Entry Parameters	Returned Value
Register (C): 28H Register (DE): FCB Address	Register (A): Return Code

Function to write random records with zero fill. When Direct Access Write (Function 34) is used to build a file, unwritten records within an allocation block contain unpredictable garbage. This request fills the unwritten records of each new block with binary zeros.

4.40. FUNCTION 47: RETURN CURRENT DMA ADDRESS

Entry Parameters	Returned Value
Register (C): 2FH	Register (HL): DMA Address

Function to return the current DMA Address.

4.41. FUNCTION 48: RETURN ZRDOS VERSION NUMBER

Entry Parameters	Returned Value
Register (C): 30H	Register (HL): Version number

Function to return the current ZRDOS version number.

4.42. FUNCTION 50: SET WARM BOOT TRAP

Entry Parameters	Returned Value
Register (C): 32H Register (DE): Trap address	None

This function replaces the warm boot jump address at 0001H with a trap address. Warm boots will then be diverted to the trap address.

4.43. FUNCTION 52: RESET WARM BOOT TRAP

Entry Parameters	Returned Value
Register (C): 34H	None

Function to reset the warm boot trap. The real warm boot address is stored at 0001H. Warm boots will now be directed to the real warm boot. This function will take affect only if the warm boot trap was previously set by function 50.

5. DIRECTORY CODES

Many of the ZRDOS functions return a directory code as a return parameter. The Directory Code is actually a multiplier to be used in determining the directory entry location in the default file buffer. The default buffer (location 80H) contains 128 bytes (one sector) of the directory entries read off of the specified disk. There are four 32 byte directory entries to a sector of directory information. The returned Directory Code points to one of these entries. The specified entry can be found by multiplying the Directory Code times 32 and adding this offset to the beginning address of the default buffer (80H). Below is shown a hex ASCII image of a typical directory sector loaded into the default buffer:

DEFAULT BUFFER ADDRESS

	HEX IMAGE			ASCII IMAGE
0080	00444454 20202020	20434F4D 00000026	.DDT COM...&
0090	07000000 00000000	00000000 00000000	.....
00A0	00454449 54202020	20434F4D 0000004C	.EDIT COM...L
00B0	0A0B0000 00000000	00000000 00000000	.....
00C0	00474F54 4F202020	20434F4D 00000006	.GOTO COM....
00D0	91000000 00000000	00000000 00000000	.....
00E0	0048454C 4C4F2020	20202020 00000009	.HELLO ....
00F0	3D000000 00000000	00000000 00000000	=.....

Fig. 5-1: Example Directory Sector.

The Directory code for the directory entry EDIT.COM is 1. So multiplying the directory code by 32 gives us 32 decimal. 32 decimal is 20 hex. Adding 20 hex to 80 hex gives us A0 hex.

A. INDEX

D

E

F

G

I

L

N

O

P

R

S

T

U

W

Z