

PART 3. UTILITIES IN F83 SYSTEM

CHAPTER 14. THE MS-DOS FILES

The source code managing files in the F83 system is scattered in screens 51 and 57 in KERNEL86.BLK and also in screens 7 to 12 in EXTEND86.BLK. Some of them were discussed in the chapter on the virtual memory.

14.1. CP/M-DOS FILE PRIMITIVE COMMANDS

The DOS file management system consists of a set of words in the DOS vocabulary that access the BDOS functions of the CP/M-DOS operating system, such as creating, opening, and deleting files. There is also a word that parses a string and creates a file control block (FCB). A very useful word SAVE is also provided to save the contents of memory as an executable DOS file. A number of words were also defined in the basic F83 system which are used to access the default file defined by the FCB1 control block.

VOCABULARY DOS	All the DOS words are put in this vocabulary. For CP/M systems, its name is CP/M, of course.
DOS DEFINITIONS	Make DOS the current vocabulary so that all subsequent words will be added to this vocabulary.
CREATE FCB1 B/FCB ALLOT	Allocate space for the first FCB block of the current file.
CREATE FCB2 B/FCB ALLOT	Allocate space for the second FCB block of the in-file.
: CLR-FCB (fcb ---)	Initialize the specified FCB.
DUP B/FCB ERASE	Clear the FCB to nulls.
1+ 11 BLANK	Initialize the file name and extension to blanks.
;	

The following words are simply BDOS functions with Forth names. Descriptive names make the Forth programs or definitions more readable.

: RESET (---)	Reset disk, flush file buffers, but does not close files.
0 13 BDOS DROP ;	
: CLOSE (fcb ---)	Close the given file and report errors.
16 BDOS	Call BDOS to close the file.
DOS-ERR?	If there is error,
ABORT" Close error"	report it.
;	
: SEARCH0 (fcb --- n)	17 BDOS ;

: SEARCH	(fcb --- n)	18 BDOS ;
: DELETE	(fcb --- n)	19 BDOS ;
: READ	(fcb ---)	Read the next record and report any error.
20 BDOS		Read next record.
DOS-ERR?		If read error,
ABORT" Read error"		abort with a message.
;		
: WRITE	(fcb ---)	Write the next record and report error if any.
21 BDOS		Write the record.
DOS-ERR?		Any error?
ABORT" Write error"		Report and abort.
;		
: MAKE-FILE	(fcb --)	Create a new DOS directory entry for a new file. Report error if any.
22 BDOS		Create directory entry.
DOS-ERR?		Error?
ABORT" Can't make file"		
;		

14.2. THE FILE CONTROL BLOCK

The file control block FCB is a table containing essential information so that the DOS system can manage the file in association with this block. The next two words build FCB blocks which is almost all that is needed to create files and gain access to them using the above commands.

: (!FCB) (addr len fcb ---)	Use the string at addr and the length, len, to set up a file control block. This is the primitive file name parsing word, which breaks the drive/filename/extension string into a drive specifier, the file name, and the extension, and inserts them into the proper fields in the FCB.
DUP B/FCB ERASE	Clear the entire FCB to zeros.
DUP 1+ 11 BLANK	Clear the name/extension fields to Ascii blanks.
>R	Save the FCB address for later use.
OVER 1+ C@	Get the second character in the string on stack.
ASCII : = IF	If it is a ':', then get the first character and use it as the drive specifier.
OVER C@	Get the first character.
[ASCII A] LITERAL	Store Ascii code of A here as a literal.
-	Subtract 65 (Ascii A) from the drive specifier. The result is the drive number.
R@ C!	Store it in the drive number field in FCB.
2 /STRING	Adjust the string address and length to point to the file name.
THEN	

R> 1+	Address of the name file in FCB.
-ROT	Get the string length to top of stack.
0 DO	Now fill the file name field.
DUP C@ ASCII . =	Is the character a period?
IF	Yes. End of file name and start of extension.
SWAP S	wap the FCB field pointer to top of stack.
8 I - +	Compute the address of the extension field in FCB.
ELSE	Not a period. Stuff the character in the name or extension field.
2DUP C@	Get the character from string.
SWAP C!	Store it in the FCB.
SWAP 1+	Increment the FCB pointer.
THEN	
SWAP 1+	Increment the string pointer also.
LOOP 2DROP	Clean the stack to exit.
;	
: !FCB (FCB-addr ---)	Use the following string as the file name string and create an FCB for it. If CAPS is false, allow lower case file names.
BL WORD	Parse out the next string and place it in the word buffer
COUNT	Get the string length from the word buffer address left by WORD.
CAPS @ IF	If CAPS is true,
2DUP UPPER	convert the string to upper case.
THEN	Otherwise, allow lower case string.
ROT	Get the FCB address to top of stack for (!FCB).
(!FCB)	Now, get (!FCB) to fill the FCB with the name string in the word buffer.
;	
: SELECT (drive ---)	Make the given drive the default drive.
14 BDOS DROP ;	

14.3. HIGH LEVEL FILE COMMANDS

The following words are defined in the basic F83 system as shown in KERNEL86.BLK file, screen 57. However, their functionalities make them a natural part of this chapter on the MS-DOS files. One of the problems in reading Forth source code is that the order in loading the Forth source codes does not necessarily bear any relationship with the logical order of words. In this book, I hope that grouping words together according to their functionalities will help you to perceive more clearly the logical structure in the F83 system.

: FILE-SIZE (fcb --- n)	Return the size of the current file in number of records.
35 BDOS DROP	BDOS function 35 returns the file size in the field of random record number.
RECORD# @	Get the file size.
;	

: DOS-ERR? (--- f) 255 = . ;	Return a true flag if the previous DOS operation is in error. BDOS returns 255 if an error occurred
: OPEN-FILE (---) IN-FILE @ 15 BDOS DOS-ERR? IF . " Open error" DISK-ABORT THEN DUP FILE-SIZE 1- SWAP MAXREC# ! ;	Open the current file and store the size of this file in MAXREC#. Open the in-file. Is there an error? If so, abort. Otherwise, size the file. Number of the last record in file. Save it.
92 CONSTANT DOS-FCB	The zero page address where DOS puts a parsed FCB.
: DEFAULT (---) FCB1 DUP IN-FILE ! DUP FILE ! CLR-FCB DOS-FCB 1+ C@ BL <> IF DOS-FCB FCB1 12 CMOVE OPEN-FILE THEN ;	Open the default DOS file. Move the parsed FCB block to FCB1 and open the file. If no file is in DOS-FCB, do nothing. Make the default file as specified by FCB1 both the in-file and the current file. Erase FCB1. Get the first character in the name field of the DOS file in DOS-FCB. If the first character of file name is not blank, there is a DOS file. Copy the drive number, file name. And extension into FCB1. Open the current file.
: CREATE-FILE (n ---) FCB2 DUP !FILES DUP !FCB MAKE-FILE MORE ;	Create a new file and allocate n blocks to this file. Set the file pointers in both the current file and in-file to point to FCB2. Build a FCB at FCB2 and make it the current file. The file name is taken from the input stream. Call BDOS to make the file. Allocate the require blocks.
: MORE (n ---) 1 ?ENOUGH CAPACITY SWAP SWAP DUP 8* FILE @ MAXREC# +! BOUNDS ?DO I BUFFER B/BUF BLANK UPDATE	Add n blocks to the current file. I need at least one stack item. Current maximum size in blocks. Record number to be added. Add to the maximum record field in the current FCB Now initialize the whole file to blanks. Get a disk buffer. Clear the disk buffer to blanks. Mark the buffer as modified. Next time BUFFER is called to

LOOP	use this buffer, the blanks will be written to the file.
SAVE-BUFFERS	Flush the remaining buffers out to disk.
FILE @ CLOSE	Close the file.
;	

14.4. SAVE CORE IMAGE TO A FILE

A very special usage of the file words is to save the entire core image in a file which can be called for execution from DOS. This will save lots of compiling time to load in many blocks of utilities. It is also a good way to build an application program without giving the user all the Forth source code, a good way to protect your software product.

DEFER HEADER	Create a vectored word.
' NOOP IS HEADER	HEADER is used in the DOS system.
 : SAVE (addr len ---)	 Use the name following as the file name and create an executable DOS file. Memory from addr to addr+len is saved into this file. The current file is not disturbed.
FCB2 DUP !FCB	Build a new FCB at FCB2, using the name following SAVE.
DUP DELETE DROP	If this file already exists, delete it.
DUP MAKE-FILE	Create a new file.
HEADER	Build an executable header.
-ROT BOUNDS DO	Scan the given range of memory.
I SET-DMA	Specify memory address for DMA transfer.
DUP WRITE	Write one record of 128 bytes.
128 +LOOP	Increment the index of length for next record.
CLOSE	Close the file. ;
 : SAVE-SYSTEM (---)	 The high level command to save the code image to a file.
256	You do not have to remember the dictionary addresses.
HERE	Starting memory address of the Forth dictionary.
SAVE	End of dictionary.
;	Make the executable file.

14.5. DIRECTORY ACCESSING

F83 can access the DOS directory on a disk directly without having to leave the Forth environment. They are conveniences that make you feel at home and eliminate the necessity of learning the DOS system and fighting against it.

: .NAME (n ---)	Print the name of the nth entry in the DOS directory.
#OUT @	Get the current output character count in #OUT.
C/L >	If it exceeds the line length,
IF CR THEN	send a CR to start a new line.

32 * PAD + 1+	The address of the nth entry, already copied to the PAD buffer.
8 2DUP TYPE SPACE	Print the file name.
+ 3 TYPE 3 SPACES	Print the extension.
;	
: DIR (---)	Print the DOS directory.
[DOS]	Switch context to CP/M vocabulary.
" ????????."?"	Put a file name template in PAD.
FCB2 (!FCB)	Create a new FCB with the ? marks in its name and extension fields.
CR PAD SET-DMA	Fetch the directory information to PAD.
SEARCH0	Search for the first directory entry that matches the ? mark name. Any valid file name would do. The stack item returned is the entry number of the file in PAD, just right for .NAME.
BEGIN	Scan the entire directory.
.NAME	Print the file name and extension.
SEARCH	Search the next matching file name, i.e., the next file name.
DUP DOS-ERR?	End of the directory?
UNTIL	If any error flag is returned, we have reached the end of the directory. Exit now. Otherwise, loop back to print the next file name.
DROP	Drop off the invalid entry number.
;	
: .FILE (addr ---)	Given the address of an FCB, print the name of this file.
COUNT ?DUP IF	If the drive number is not zero,
ASCII @ + EMIT ." :"	
THEN	then print the drive prompt.
8 2DUP	Name field width.
-TRAILING TYPE	Print the file name without the trailing blanks.
+	The address of the extension field.
." ."	Print a period sign between name and extension.
3 TYPE SPACE	Print the extension.
;	
: FILE? (---)	Print the name of the current file.
FILE @	Get the FCB of current file.
.FILE	Print its name.
;	

F83 allows you to have two files opened at the same time: a current file and an in-file. The in-file is used for input and the current file is used for output. The command SWITCH can be used to switch these two files so you can input from the previous current file and output to the previous in-file.

: SWITCH (---)	Exchange the current file and the in-file.
FILE @ IN-FILE @	Two fcb's.
FILE ! IN-FILE !	Exchange the fcb addresses.

```

;

: !FILES      ( fcb --- )    Set both the current file and the in-file to the given fcb.    DUP
FILE !       Set current file.
      IN-FILE !    Set in-file.
;

```

14.6. SYSTEM LEVEL FILE COMMANDS

The words defined above are mostly utility words which allow the F83 system to manage DOS files and the associated facility. As a user, you will probably have no need for them unless you have to dig down into the system level. To use the file management system, you need only a few words at the top Forth level to create files and to gain access to their contents. This section describes these words and their functions.

```

: FILE:      ( --- fcb )    Use the following string as the file name and create a new
                             file. The address of the FCB is returned on the stack.
      >IN @    Save the input character pointer because we will use the next
                             name more than once.
      CREATE   Create a Forth word using the following name. When this
                             word is referenced, the file of the same name in DOS will be
                             opened and made the current file.
      >IN !    Restore the input character pointer to the front of the file name.
      HERE DUP The parameter field address of the file definition.
      B/FCB ALLOT Put the FCB in the parameter field.
      !FCB      Now stuff the FCB with the new file name.
      DOES>    Now comes the execution part of the file definition.
      !FILES   Initialize both the current and the in-file.
;

: ?DEFINE    ( --- fcb )    Define the next word as a file if it is not already defined.
                             Leave its FCB address on stack.
      >IN @    Save the input character pointer.
      DEFINED  Search the dictionary for the next word, which is supposedly
                             a file name.
      IF NIP >BODY If the file definition is in the dictionary, discard the character
                             pointer because we will not need it. The cfa returned by
                             DEFINED is then changed to pfa which is the FCB of the
                             defined file.
      ELSE     No. The file was not defined.
      DROP    Throw away the word buffer address.
      >IN !    Restore the character pointer to the front of the file name.
      FILE:   Define a new file with a new file definition in the dictionary
      .      THEN ;

```

FORTH DEFINITIONS

All the file management words were put into the DOS vocabulary, which are not accessible from FORTH. The

two most used words concerning files are to be defined in the FORTH vocabulary so that they can be accessed conveniently.

: OPEN	(---)	Open the following file and make it the current file.
[DOS]		OPEN has to refer to words in the DOS vocabulary.
?DEFINE		Find the file in dictionary. If failed, create a new file.
!FILES		Make this file the current file.
OPEN-FILE		Open it.
;		
: DEFINE	(---)	Define the following word as a new file without opening it.
?DEFINE DROP ;		
: FROM	(---)	Make the next word in the input stream the FROM file. It will be created if not already being defined.
?DEFINE		Open a file.
IN-FILE !		Make it the in-file.
OPEN-FILE		And then open it.
;		
DEFER LOAD		Interpret a screen.

In the previous Forth systems, including F83 Version 1, LOAD always interprets a screen from the current file. To allow more natural and more convenient access to multiple files, F83 Version 2 modified the LOAD command so it will load a screen from the in-file, which is set up as the input file. Most of the other file commands access the current file as default. To make sure that other file commands can still access the current file, LOAD only loads one screen from the in-file and then restores the current file.

: (LOAD)	(n ---)	Interpret one screen from the in-file.
FILE @ >R		Save the current file fcb.
BLK @ >R		Save the currently processed screen number on the data stack.
>IN @ >R		Save the word parsing pointer also.
>IN OFF		Start at the beginning of the screen.
BLK !		Store n into BLK to process screen n.
IN-FILE @ FILE !		Make the in-file the current file for interpreting.
RUN		Interpret the screen.
R> >IN !		Restore the parsing pointer.
R> BLK !		Restore the previous screen number.
R> !FILES		After loading from the FROM file, restore the current file.
;		

' (LOAD) IS LOAD Vector LOAD to execute (LOAD).

1 CONSTANT INITIAL In all the F83 source files, screen 1 is always a load screen which loads in the code in the file. INITIAL is defined to load this screen.

: OK	(---)	Load applications in the current file.
INITIAL LOAD ;		
: A:	(---)	Select drive A as the default drive.
0 SELECT ;		
: B:	(---)	Select drive B as the default drive.
1 SELECT ;		