21 File Management In DOS

The DOS file management functions are among the most basic available to the programmer. However, programmers using high level languages seldom access these DOS functions directly because the languages often have their own methods of file management. This chapter describes how the file functions are organized and how you can use them from higher level languages.

Two Sides Of DOS

The term *file management functions* refers to the functions used to manage files, such as creating, deleting, opening, closing, reading from, and writing to files. Operating systems such as DOS provide the programmer with functions for file management. For example, DOS provides functions that return special file information or rename a file.

One peculiarity of DOS is these functions exist in two forms because of the combined CP/M & UNIX compatibility. For every UNIX compatible file function, there is also a CP/M compatible file function. Versions 2.0 and up of DOS borrowed ideas from this UNIX compatibility.

FCB (File Control Block) functions

The CP/M compatible functions are designated as FCB functions because they are based on a data structure called the FCB (File Control Block). DOS uses this data structure for storing information during file manipulation. The user must reserve space for the FCB within this program. The FCB permits access to the FCB functions which open, close, read from, and write to files. Since the FCB functions were developed for compatibility with CP/M's functions, and since CP/M doesn't have a hierarchical file system, FCB functions don't support paths. As a result, FCB functions can only access files that are in the current directory.

DOS Versions 2.0 and up support handle functions, which were first used in the UNIX environment. However, the UNIX compatible handle functions don't have the problems resulting from FCB functions. As the name suggests, a handle is used to identify the file to be accessed. DOS stores information about each open file in an area that is separate from the program.

Remember the differences between these function groups are related to how these files are created, instead of their actual structures. Files created and edited using FCB functions can cause problems if subsequently accessed by handle functions and vice versa.

The most important fact to remember is to keep the two groups of functions separate when developing high level language programs. In the following sections, we'll take a closer look at each group of functions.

Handle Functions

It's easier for the programmer to access a file using the handle functions than using the FCB functions. With handle functions a programmer doesn't have to use a data structure for file access like the FCB functions do. Similar to the functions of the UNIX operating system, file access is performed using a filename. The filename is passed as an ASCII string when the file is opened or first created. This must be performed before the first write or read operation to the file. In addition to the filename, it may contain a device designator, a pathname, and a file extension. The ASCII string ends with the end character (ASCII code 0). After the file is opened, a numeric value called the handle is returned. Any further operations to this file are performed using this 16-bit handle. For a subsequent read or write operation, the handle, instead of the filename, is passed to the appropriate function.

For each open file, DOS saves certain information pertaining to that file. If the FCB functions are used, DOS saves the information in the FCB table within the program's memory block. When the handle functions are used, the information is stored in an area outside of the program's memory block in a table that is maintained by DOS. The number of open files is therefore limited by the amount of available table space. The amount of table space set aside by DOS is specified by the FILES parameter of the CONFIG.SYS file:

```
FILES = X
```

In DOS Version 3.0, this maximum is 255. If you change the maximum number of files in the CONFIG.SYS file, the change will not become effective until the next time DOS is booted.

```
FILES
```

While the FILES parameter (CONFIG.SYS) specifies the maximum number of open files for the entire operating system, DOS limits the number of open files to 20 per program. Since five handles are assigned to standard devices, such as the keyboard, monitor, and line printer, only 15 handles are available for the program. For example, if a program opens three files, DOS assigns three available File handles and reduces the number of additional available handles by three. If this program calls another program, the three files opened by the original program remain open. If the new program opens additional files, the remaining number of handles available is reduced even further.

Variable access length

Another difference from FCB functions lies in read and write functions. While FCB functions work with records of constant lengths, handle functions specify how many bytes should be read or written. This makes dynamic access to consecutive data records possible.

Besides the standard read and write functions, there is also a file positioning function. This lets you specify an exact location within the file for the next data access. Knowing both a record number and the length of each data record allows you to specify the position to access a particular data record:

```
position = record_number * record_length
```

This function isn't used during sequential file access because DOS sets the file pointer during the opening or creation of a file to the first byte within the file. Each subsequent read or write operation moves the file pointer, by the number of bytes read, towards the end of the file so the next file access starts where the previous one ended.

The following table summarizes the handle functions. For a more detailed description of these functions, see Appendix D, which documents the DOS API functions.

You'll find additional information on the companion CD-ROM

Appendix D on the companion CD-ROM will give you a more detailed description of these handles. Appendix D also documents the DOS API functions.

Function	Operation	Function	Operation
3CH	Create file	57H/01H	Read/Write modifications & date/time of file
3DH	Open file	5AH	Create temporary file
3EH	Close file	5BH	Create new file
42H	Move file pointer/determine file size	5CH/00H	Protect file range against access
43H	Read/Write file attribute	5CH/01H	Release protected file range
56H	Rename file	6CH	Extended OPEN function
57H/00H	Read/Write modifications & date/time of file		

Here are a few general rules to follow when using these functions:

1. Functions that expect a filename or the address of a filename as an argument (e.g., Create File and Open File) expect the segment address of the name in the DS register and the offset address in the DX register. If the function successfully returns a handle, it's returned in the AX register.
2. Functions that expect a handle as an argument expect to find it in the DX register. After the call, the carry flag indicates whether an error occurred during execution. If an error occurs, the carry flag is set and the error code is returned in the AX register.
3. Function 59H of DOS interrupt 21H returns very detailed information about errors that occur during disk operations. This function is available only in DOS Versions 3.0 and higher.

FCB (File Control Block) Functions

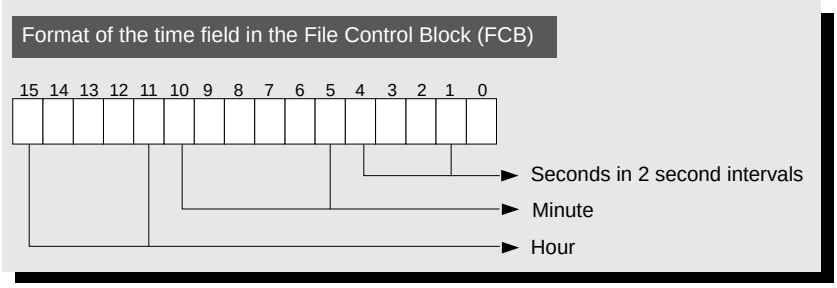
As we discussed, DOS uses an FCB data structure for managing a file. The programmer can use this data structure to obtain information about a file or change information about a file. So, we'll examine the structure of an FCB before discussing the individual FCB functions. The FCB is a 37-byte data structure that can be subdivided into different data fields. The following figure illustrates these fields:

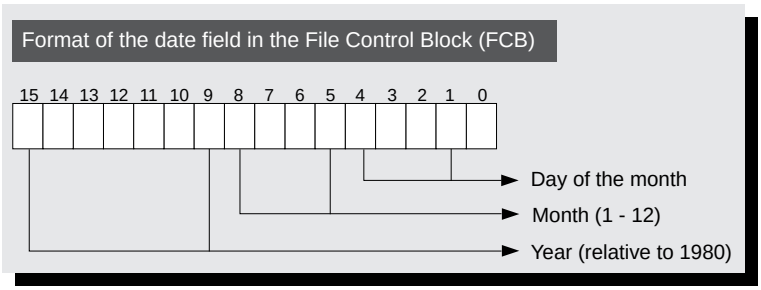
FCB 37-byte data structure					
Address	Contents	Type	Address	Contents	Type
+00H	Device name	1 byte	+14H	Modification date	1 word
+01H	Filename	8 bytes	+16H	Modification time	1 word
+09H	File mode	3 bytes	+18H	Reserved	1 word
+0CH	Current block number	1 word	+20H	Current data record number	1 byte
+0EH	Data record size	1 word	+21H	Data record number for random access	2 words
+10H	File size	2 words			

Notice the name of the file is found beginning at offsets 01H through 0BH of the FCB. The byte at offset 0 is the device indicator, 0 is the current drive, 1 drive A, 2 drive B, etc.

The filename that begins at offset 1 is an ASCII string. It may not contain a pathname since it's limited to 8 characters. For this reason, the FCB functions can access only files in the current directory. Filenames shorter than eight characters are padded with spaces (ASCII code 32). The file extension, if any, occupies the next three bytes of the FCB. At offset 0CH of the FCB is the current number of the block for sequential file access. The two bytes at offset 0EH are the record size. The four bytes at offset 10H are the length of the file.

The date and time of the last modifications to the file are stored beginning at offset 14H of the FCB in encoded form.





An eight-byte data area follows and is reserved for DOS (no user modifications allowed). The use of this area varies from one version of DOS to another.

Following this reserved data area is the current record number which is used with the current block number to simulate CP/M operations.

Random files

The last data field of the FCB is used for a type of access in which the data within the file may be retrieved or written in a non-sequential order. This field is four bytes long. If a record is equal to or larger than 64 bytes, only the first three bytes are used for indicating the current record number. All four bytes of this field are used for records smaller than 64 bytes.

Extended FCB

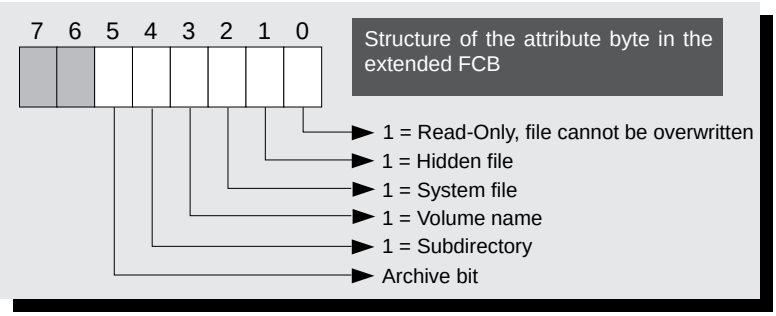
Besides a standard FCB, DOS also supports the extended FCB. Unlike normal FCBs, extended FCBs access files with special attributes, such as hidden files or system files. They also permit access to volume names and subdirectories (this doesn't mean that you can access files in other directories besides the current directory).

An extended FCB is similar to a standard FCB, but it's seven bytes larger. These seven bytes are located at the beginning of the data structure. So, all subsequent fields are displaced by seven bytes.

Extended FCB data structure

Address	Contents	Type	Address	Contents	Type
+00H	FF	1 byte	+15H	File record size	1 word
+01H	Reserved(0)	5 bytes	+17H	File size	2 words
+06H	File attribute	1 byte	+1BH	Modification date	1 word
+07H	Device name	1 byte	+1DH	Modification time	1 word
+08H	Filename	8 bytes	+1FH	Reserved	8 bytes
+10H	File extension	3 bytes	+27H	Current data record number	1 byte
+13H	Current block number	1 word	+28H	Data record number	2 words

The first byte of an extended FCB always contains the value 255 and identifies this as an extended FCB. Since this address contains the device number in a normal FCB and therefore cannot contain the value 255, DOS can tell the difference between a normal and an extended FCB. The next five bytes are reserved exclusively for use by DOS. They shouldn't be changed. The seventh byte is a file attribute byte. Refer to the "Floppy Diskette And Hard Drive Structure" section of Chapter 14 for the details of the file attribute byte.



Now that you're familiar with the FCB structures, the next section focuses on using FCBs for accessing files.

FCB and file access

Before accessing a file, an FCB must be built in the program's memory area. The area can be reserved within the data segment of the program or by allocating additional memory using another DOS function (see Appendix D).

Although it's possible to write the data directly into the FCB, it is better to use one of the appropriate DOS functions to do this. For example, to set the filename in the FCB you can use DOS function 29H. The function number is passed in the AH register. The address of the FCB is passed in the ES:DI register pair. The address of the filename is passed in the DS:SI register pair. The filename is an ASCII string terminated by the end character (ASCII code 0). The AL register contains flags for converting the filename and are discussed in more detail in Appendix C.

Open FCB

After the FCB is properly formatted, the file can be opened or created using a DOS function. When this happens, DOS stores information about that file, such as the file size, date and time of file creation, etc., in the FCB. At this point the FCB is considered opened.

By default, the record length is set to 128 bytes when the FCB is opened. To override this record length, store the desired record length at offset 0EH of the FCB after it's opened. Otherwise, the default length will be used.

DTA

For record lengths greater than 128 bytes, the record buffer, also known as the DTA (Disk Transfer Area), must be moved to accommodate the longer record size. Usually DOS builds the DTA in the PSP (Program Segment Prefix). Accessing the file using the default DTA for a record length greater than 128 bytes would overwrite some of the other fields in the PSP.

The most convenient way to select a new DTA is to reserve the space in the program's data segment. To change the address of the DTA, use DOS function 1AH. The address of the new DTA is passed in the DS:DX register pair. Since DOS assumes that you've set aside an area large enough to accommodate your largest record length, you don't have to specify the new length.

File access

For sequential file access, processing begins at the first record in the file. DOS maintains a record pointer in the FCB to keep track of the current record within the file. Each time the file is accessed, DOS advances the pointer so the second, third, fourth, etc. record is processed sequentially.

For random file access, the records can be processed in any order. The position of each record relative to the beginning of the file determines its record number. This record number is then passed to DOS to access a specific record. The last field of the FCB is used to specify the record number to DOS.

It's also possible to change from sequential access mode to random access mode and vice versa, since processing depends on a specific DOS function to access the file. There are actually two sets of independent functions, one for sequential access and one for random functions.

The following table lists all the FCB functions of DOS interrupt 21H. A more detailed description of the functions is found in Appendix D which you'll find on the companion CD-ROM.

FCB functions of DOS interrupt 21H			
Function	Task	Function	Task
0FH	Open file	21H	Random Read (of record)
10H	Close file	22H	Random Write (of record)
13H	Delete file	23H	Determine file size
14H	Sequential read	24H	Set record number for random access
15H	Sequential write	27H	Random read (one or more records)
16H	Create file	28H	Random write (one or more records)
17H	Rename file	29H	Enter filename into FCB
1AH	Set DTA address		

The following are some basic rules about these functions:

Using the FCB functions, you can access several files, each with their own unique FCB. To tell DOS which file should be accessed, pass the address of the file's FCB in the DS:DX register pair.

Most of the functions return an error code in the AL register or the value zero if the function was successfully completed. For functions that open, close, create, or delete a file, a code of 255 is returned if an error occurs. The other functions return specific error codes. More detailed information about these errors can be determined by calling DOS function 59H but this is available only in DOS Versions 3.0 or later.

You'll find additional information on the companion CD-ROM

Appendix D on the companion CD-ROM provides a more detailed description of these functions.

Handles versus FCBs

Now we'll briefly discuss the advantages and disadvantages of the individual functions. If you want to convert a program from the CP/M or UNIX operating systems into DOS, the choice will be easy. However, if you want to develop a new program under DOS, the following explanations should help you determine which set of functions to use.

Handles

There are two main advantages to using handle functions. The first is the ability to access a file in any subdirectory of the disk. The second is the handle functions aren't limited to the number of FCBs that can be stored in a program's memory space.

There are also several additional considerations. You can access the name of a disk drive only by using an FCB. When the FCB is opened, you can easily determine its file size and the date of the last modification. The handle functions automatically provide an area large enough to accommodate the records in the file.

As you can see, there are arguments for and against using either the FCB functions or the handle functions. For future versions of DOS, the handle functions will become more important and the importance of the FCB functions will diminish. This is reason enough to use the handle functions for your new program development.