[http://www.disc.ua.es/~gil/FAT12Description.pdf]

CSSE 332 – Operating Systems
Spring 2004-2005

Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Prof. Archana Chidanandan

**FAT file name and extension representation**

File names in DOS traditionally have a limit of 8 characters for the name, and 3 characters for the extension. There are a few things to be aware of:

- File/directory names and extensions are *not* null-terminated within the directory entry
- File/directory names always occupy 8 bytes--if the file/directory name is shorter than 8 bytes (characters) pad the remaining bytes with spaces (ASCII 32, or Hex 0x20). This also applies to 3-character extensions.
- File/directory names and extensions are *always* uppercase. Always convert given file/directory names to uppercase.
- Directory names can have extensions too.
- "FILE1" and "FILE1.TXT" are unique (the extension *does* matter).
- Files and directories *cannot* have the same name (even though the attributes are different). Here are examples of how some file names would translate into the 11 bytes allocated for the file/directory name and extension in the directory entry (white space between quotes should be considered as spaces).

- *filename provided* [01234567012]
- "foo.bar" -> "FOO BAR"
- "FOO.BAR" -> "FOO BAR"
- "Foo.Bar" -> "FOO BAR"
- "foo" -> "FOO "
- "foo." -> "FOO "
- "PICKLE.A" -> "PICKLE A "
- "prettybg.big" -> "PRETTYBGBIG"
- ".big" -> *illegal*! file/directory names cannot begin with a "."

## The Root Directory

The root directory contains an entry for each file whose name appears at the *root* (the top level) of the file system. Other directories can appear within the root directory; they are called *subdirectories*. The main difference between the two is that space for the root directory is allocated statically, when the disk is formatted; there is thus a finite upper limit on the number of files that can appear in the root directory.

Subdirectories are just files with special data in them, so they can be as large or small as desired.

The format of all directories is the same. Each entry is 32 bytes (0x20) in size, so a single block can contain 16 of them. The following table shows a summary of a single directory entry; note that the offset is merely from the start of that particular entry, not from the start of the block.

| Offset | Length | Description |
|---|---|---|
| 0x00 | 8 bytes | Filename |
| 0x08 | 3 bytes | Filename extension |
| 0x0b | 1 byte | File attributes |
| 0x0c | 10 bytes | Reserved |
| 0x16 | 2 bytes | Time created or last updated |
| 0x18 | 2 bytes | Date created or last updated |
| 0x1a | 2 bytes | Starting cluster number for file |
| 0x1c | 4 bytes | File size in bytes |

### The Filename

The eight bytes from offset 0x00 to 0x07 represent the filename. The first byte of the filename indicates its status. Usually, it contains a normal filename character (e.g. 'A'), but there are some special values:

0x00
: Filename never used.

0xe5
: The filename has been used, but the file has been deleted.

0x05
: The first character of the filename is actually 0xe5.

0x2e
: The entry is for a directory, not a normal file. If the second byte is also 0x2e, the cluster field contains the cluster number of this directory's parent directory. If the parent directory is the root directory (which is statically allocated and doesn't have a cluster number), cluster number 0x0000 is specified here.

Any other character
: This is the first character of a real filename.

If a filename is fewer than eight characters in length, it is padded with space characters.

### The Filename Extension

The three bytes from offset 0x08 to 0x0a indicate the filename extension. There are no special characters. Note that the dot used to separate the filename and the filename extension is implied, and is not actually stored anywhere; it is just used when referring to the file. If the filename extension is fewer than three characters in length, it is padded with space characters.

### The File Attributes

The single byte at offset 0x0b contains flags that provide information about the file and its permissions, etc. The flags are single bits, and have meanings as follows. Each bit is given as its numerical value, and these are combined to give the actual attribute value:

0x01
: Indicates that the file is read only.

0x02
: Indicates a hidden file. Such files can be displayed if it is really required.

0x04
: Indicates a system file. These are hidden as well.

0x08
: Indicates a special entry containing the disk's volume label, instead of describing a file. This kind of entry appears only in the root directory.

0x10
: The entry describes a subdirectory.

0x20
: This is the archive flag. This can be set and cleared by the programmer or user, but is always set when the file is modified. It is used by backup programs.

0x40
: Not used; must be set to 0.

0x80
: Not used; must be set to 0.

## The File Time

The two bytes at offsets 0x16 and 0x17 are treated as a 16 bit value; remember that the least significant byte is at offset 0x16. They contain the time when the file was created or last updated. The time is mapped in the bits as follows; the first line indicates the byte's offset, the second line indicates (in decimal) individual bit numbers in the 16 bit value, and the third line indicates what is stored in each bit.

```
<------- 0x17 --------> <------- 0x16 -------->
15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00
h  h  h  h  h  m  m  m  m  m  m  x  x  x  x  x
```

where:

hhhhh
: indicates the binary number of hours (0-23)

mmmmmm
: indicates the binary number of minutes (0-59)

xxxxx
: indicates the binary number of two-second periods (0-29), representing seconds 0 to 58.

## The File Date

The two bytes at offsets 0x18 and 0x19 are treated as a 16 bit value; remember that the least significant byte is at offset 0x18. They contain the date when the file was created or last updated. The date is mapped in the bits as follows; the first line indicates the byte's offset, the second line indicates (in decimal) individual bit numbers in the 16 bit value, and the third line indicates what is stored in each bit.

```
<------- 0x19 --------> <------- 0x18 -------->
15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00
y  y  y  y  y  y  y  m  m  m  m  d  d  d  d  d
```

where:

yyyyyyy
: indicates the binary year offset from 1980 (0-119), representing the years 1980 to 2099

mmmm

indicates the binary month number (1-12)

`ddddd`

indicates the binary day number (1-31)

## The Starting Cluster Number

The two bytes at offsets 0x1a and 0x1b are treated as a 16 bit value; remember that the least significant byte is at offset 0x1a. The first cluster for data space on the disk is always numbered as 0x0002. This strange arrangement is because the first two entries in the FAT are reserved for other purposes.

## The File Size

The four bytes at offsets 0x1c to 0x1f are treated as a 32 bit value; remember that the least significant byte is at offset 0x1c. They hold the actual file size, in bytes.