CENG 334 – Intr. to Operating Systems

Programming HW #3 – File Systems

Due 28-5-2010 23:59

This homework is on basics of a file system implementation. Although they are mostly superseded by the more advanced NTFS, FAT file systems are still in use in many computer systems including digital cameras, flash disks and floppy disks because of their simplicity. In this homework you will be implementing a program that extracts some information from a FAT16 file system. Your program will operate on image files containing FAT16 partitions. You can find information about the specifics of FAT16 at the end of this document below.

The name of the image file will be given to your program as the first argument on the command line. Since your program needs to be able to do multiple tasks, the name of the tasks and their arguments will be given after the image file. An example usage that prints the boot sector info for the FAT16 partition contained in file.img is like this:

```
./hw3 file.img --printbootsector
```

After parsing the command line, your program will do the given task and then exit. Multiple tasks will not be given in the same command line. In the below sections, the tasks your program needs to do will be explained.

`--printbootsector`

In this task you should print information contained in the boot sector in the following format:

```
> ./hw3 file.img --printbootsector
bytes per sector: 512
sectors per cluster: 4
number of reserved sectors: 1
number of fat copies: 2
maximum root directory entries: 512
total sectors: 0
media descriptor type: f8
sectors per fat: 64
sectors per track: 32
number of heads: 64
number of hidden sectors: 0
number of sectors: 65536
signature: aa55
volume label: <VOLUMELABEL>
```

--printdiskusage

In this task, you should print information about the disk usage in the following format. Internal fragmentation is the sum of unused space at the last cluster of each file.

```
> ./hw3 file.img –printdiskusage
total clusters: 16343
used clusters: 1657
total size of files: 3391789
internal fragmentation: 1747
```

--listdirectoryentries root dir1 dir2

In this task you should list the names of files in a directory. This task takes a varying number of arguments and each argument represents a subdirectory in the path. In the above example, you should list the contents of root/dir1/dir2 directory. Note that in FAT, file names are padded with spaces, but you will not be printing them. You should also print the attributes of each file at the beginning of the line where;

1. The first char should be R or -. Depending on the file is read only or not.
2. The second char should be H or -. Depending on the file is hidden or not.
3. The third char should be S or -. Depending on this is a system file or not.
4. The fourth char should be D or -. Depending on this is a subdirectory or not.
5. The fifth char should be A or -. Depending on if the archive attribute is set or not.

```
> ./hw3 file.img –listdirectoryentries root dir1 dir2
---D- .
---D- ..
R---A readme.txt
----A webpage.htm
-H--- hidden.txt
--S-- command.com
RHS-A msdos.sys
---D- dir3
```

--printfilecontents root dir1 dir2 file

In this task, you should print the contents of the file to standard output. The directory parsing is similar to the above command. You should print solely the file contents, no additional spaces or newlines.

```
> ./hw3 file.img –printfilecontents root dir1 dir2 readme.txt
This is first line of readme.txt
This is line two.
Bye.
```

Specifications

1. You should submit a single tar file called **hw3.tar** through the COW system.
2. Your tar file should contain a makefile if non-default compiler flags are used.
3. Late submission policy is same as the other homeworks.
4. Using C++ or third party libraries is not allowed. Everything should be done by yourself in C.
5. The evaluation will be black-box and on inek machines.
6. Be very careful while printing strings, everything is case sensitive. Also avoid using extra white spaces and blank lines.
7. You may assume that the input file to your program is error-free. That means no bad-clusters or invalid filenames will be given.
8. Beware that size of some integer types might change between 32 bit and 64 bit machines. The compilers on inek machines are 64 bit.
9. Beware that compilers might adjust the alignment of data members.

Useful Links

1. http://en.wikipedia.org/wiki/File_Allocation_Table
2. http://home.teleport.com/~brainy/fat16.htm

Tips

You can create a toy FAT16 partition by issuing the following commands.

```
> dd if=/dev/zero of=test.img bs=1k count=32768
> mkdosfs -F 16 test.img
```

You can mount it and add files to it (if you have root access like) this:

```
> sudo mount -t msdos test.img /mnt/testimage
```