

---

**Week 11**

**FAT32 Boot Sector,  
Locating Files and Dirs**

---

Classes COP4610 / CGS5765  
Florida State University

---

# Outline

- Recap of last week's lecture
  - Introduction to project 3
  - Introduction to FAT32
- Starting Project 3
  - How to parse the boot sector
  - More on serialization
  - Finding the root directory and files
  - Understanding open, close, read, write

---

# Recap – Intro to Project 3 and FAT32

---

---

# Project 3

- You will create a user-space utility to manipulate a FAT32 file system image
  - No more kernel programming!

---

# FAT32 Manipulation Utility

Utility only recognizes the following built-in commands:

- open
- close
- create
- rm
- size
- cd
- ls
- mkdir
- rmdir
- read
- write

---

# File System Image

- Manipulation utility will work on a pre-configured FAT32 ***file system image***
  - Actually a file
- File system image will have raw FAT32 data structures inside
  - Just like looking at the raw bytes inside of a disk partition

---

# File System Image

- Your FAT32 manipulation utility will have to
  - Open the FAT32 file system image
  - Read parts of the FAT32 file system image and interpret the raw bytes inside to service your utility's file system commands...

...just like a file system!

---

# Terminology

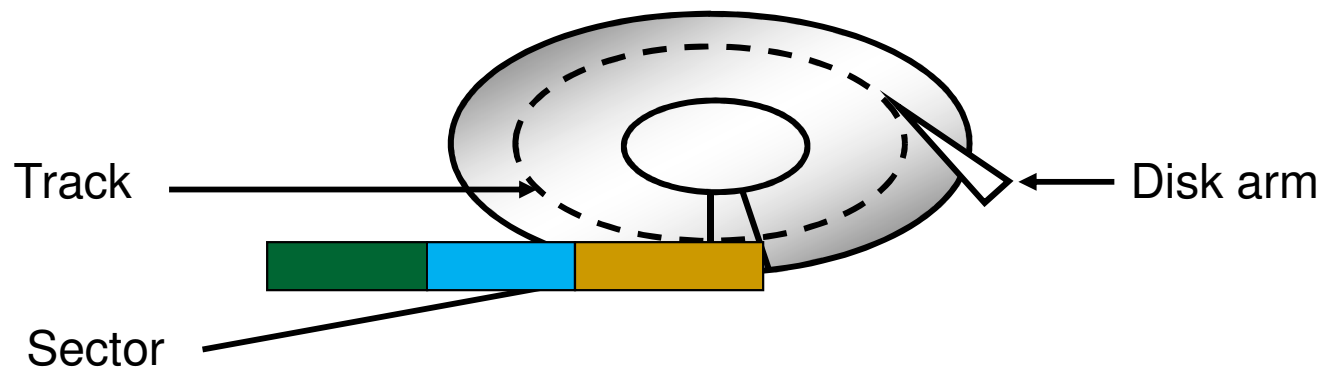
- **Byte** – 8 bits of data, the smallest addressable unit in modern processors
- **Sector** – Smallest addressable unit on a storage device. Usually this is 512 bytes
- **Cluster** – FAT32-specific term. A group of sectors representing a chunk of data
- **FAT** – Stands for *file allocation table* and is a map of files to data



---

# FAT32 Disk Layout

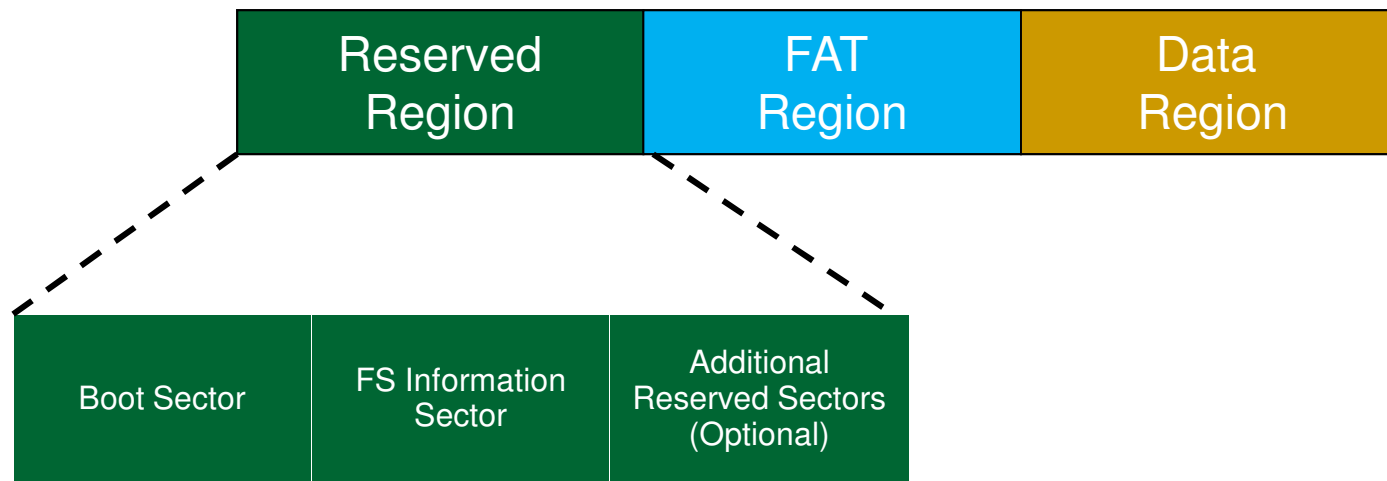
- 3 main regions...



---

# Reserved Region

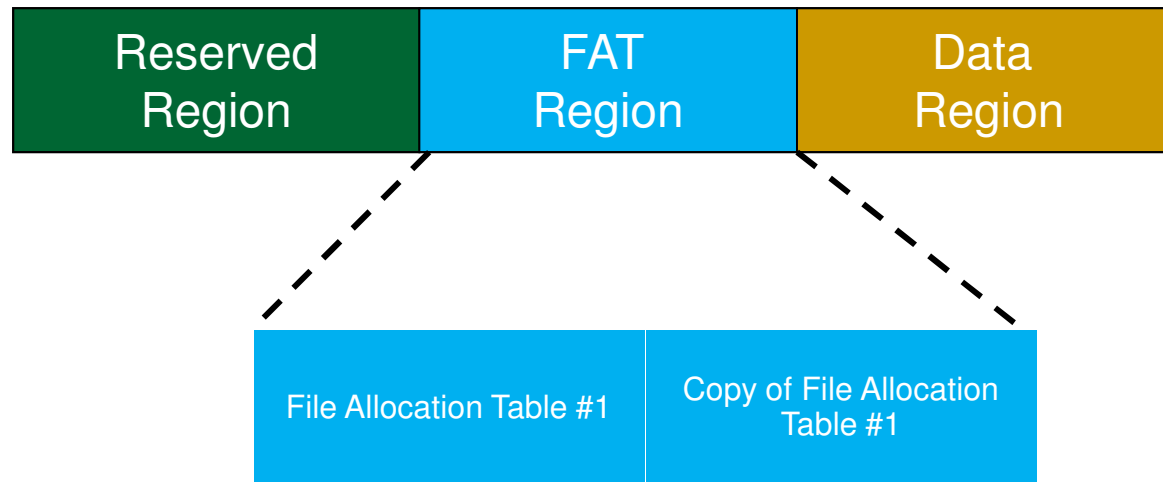
- **Reserved Region** – Includes the boot sector, the extended boot sector, the file system information sector, and a few other reserved sectors



---

# FAT Region

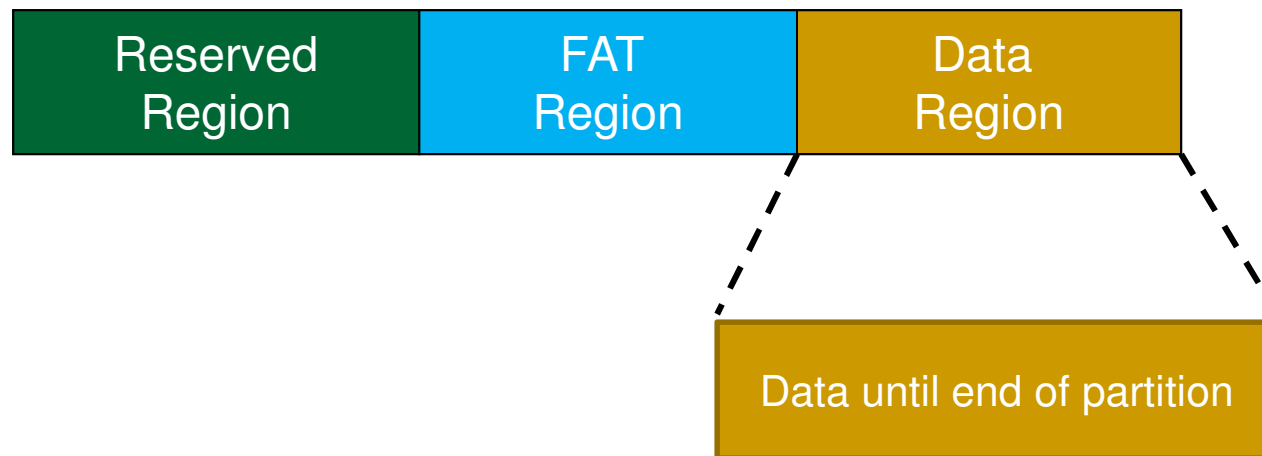
- **FAT Region** – A map used to traverse the data region. Contains mappings from cluster locations to cluster locations



---

# Data Region

- **Data Region** – Using the addresses from the FAT region, contains actual file/directory data



---

# Where to begin?

1. Mount the file system image with the OS FAT32 driver and take a look around
2. Find the FAT32 spec from Microsoft inside the image, read it

# File System Image Mount Example

```
$> sudo mount -o loop fat32.img /mnt  
$> cd /mnt
```

- `fat32.img` is your image file
- `/mnt` is your mounting directory
- Once the file is mounted, you can go into the `/mnt` directory and issue all your normal file system commands like:
  - `ls`, `cat`, `cd`, ...

---

## Hint

- As you work, it might make sense to first take a look at the raw file system image
- Hexedit to the rescue!

---

# Hexedit

```
$> hexedit [filename]
```

- View files in hexadecimal or ASCII
- Why wouldn't you want to view the file system image file in your regular editor?



# Hexedit

```
user@cop4610: ~  
00000000 EB 58 90 6D 6B 64 6F 73 66 73 00 00 02 01 20 00 .X.mkdosfs....  
00000010 02 00 00 00 00 F8 00 00 20 00 40 00 00 00 00 00 .....@.....  
00000020 00 00 02 00 F1 03 00 00 00 00 00 00 02 00 00 00 .....  
00000030 01 00 06 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000040 00 00 29 6E FA 2E 43 20 20 20 20 20 20 20 20 ..)n..C  
00000050 20 20 46 41 54 33 32 20 20 20 0E 1F BE 77 7C AC FAT32 ...w|.  
00000060 22 C0 74 0B 56 B4 0E BB 07 00 CD 10 5E EB F0 32 ".t.v.....^..2  
00000070 E4 CD 16 CD 19 EB FE 54 68 69 73 20 69 73 20 6E .....This is n  
00000080 6F 74 20 61 20 62 6F 6F 74 61 62 6C 65 20 64 69 ot a bootable di  
00000090 73 6B 2E 20 20 50 6C 65 61 73 65 20 69 6E 73 65 sk. Please inse  
000000A0 72 74 20 61 20 62 6F 6F 74 61 62 6C 65 20 66 6C rt a bootable fl  
000000B0 6F 70 70 79 20 61 6E 64 0D 0A 70 72 65 73 73 20 oppy and..press  
000000C0 61 6E 79 20 6B 65 79 20 74 6F 20 74 72 79 20 61 any key to try a  
000000D0 67 61 69 6E 20 2E 2E 2E 20 0D 0A 00 00 00 00 00 gain ... ..  
000000E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
000000F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000160 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000170 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

# Hexedit

Line numbers in hex

```
user@cop4610: ~  
00000000 EB 58 90 6D 6B 64 6F 73 66 73 00 00 02 01 20 00 .X.mkdosfs....  
00000010 02 00 00 00 00 F8 00 00 20 00 40 00 00 00 00 00 .....@.....  
00000020 00 00 02 00 F1 03 00 00 00 00 00 00 02 00 00 00 .....  
00000030 01 00 06 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000040 00 00 29 6E FA 2E 43 20 20 20 20 20 20 20 20 ..)n..C  
00000050 20 20 46 41 54 33 32 20 20 20 0E 1F BE 77 7C AC FAT32 ...w|.  
00000060 22 C0 74 0B 56 B4 0E BB 07 00 CD 10 5E EB F0 32 ".t.v.....^..2  
00000070 E4 CD 16 CD 19 EB FE 54 68 69 73 20 69 73 20 6E .....This is n  
00000080 6F 74 20 61 20 62 6F 6F 74 61 62 6C 65 20 64 69 ot a bootable di  
00000090 73 6B 2E 20 20 50 6C 65 61 73 65 20 69 6E 73 65 sk. Please inse  
000000A0 72 74 20 61 20 62 6F 6F 74 61 62 6C 65 20 66 6C rt a bootable fl  
000000B0 6F 70 70 79 20 61 6E 64 0D 0A 70 72 65 73 73 20 oppy and..press  
000000C0 61 6E 79 20 6B 65 79 20 74 6F 20 74 72 79 20 61 any key to try a  
000000D0 67 61 69 6E 20 2E 2E 2E 20 0D 0A 00 00 00 00 00 gain ... ..  
000000E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
000000F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000160 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000170 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

# Hexedit

Content in  
hex

```
user@cop4610: ~
00000000 EB 58 90 6D 6B 64 6F 73 66 73 00 00 02 01 20 00 .X.mkdosfs....
00000010 02 00 00 00 00 F8 00 00 20 00 40 00 00 00 00 00 .....@.....
00000020 00 00 02 00 F1 03 00 00 00 00 00 00 02 00 00 00 .....
00000030 01 00 06 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000040 00 00 29 6E FA 2E 43 20 20 20 20 20 20 20 20 ..)n..C
00000050 20 20 46 41 54 33 32 20 20 20 0E 1F BE 77 7C AC FAT32 ...w|.
00000060 22 C0 74 0B 56 B4 0E BB 07 00 CD 10 5E EB F0 32 ".t.v.....^..2
00000070 E4 CD 16 CD 19 EB FE 54 68 69 73 20 69 73 20 6E .....This is n
00000080 6F 74 20 61 20 62 6F 6F 74 61 62 6C 65 20 64 69 ot a bootable di
00000090 73 6B 2E 20 20 50 6C 65 61 73 65 20 69 6E 73 65 sk. Please inse
000000A0 72 74 20 61 20 62 6F 6F 74 61 62 6C 65 20 66 6C rt a bootable fl
000000B0 6F 70 70 79 20 61 6E 64 0D 0A 70 72 65 73 73 20 oppy and..press
000000C0 61 6E 79 20 6B 65 79 20 74 6F 20 74 72 79 20 61 any key to try a
000000D0 67 61 69 6E 20 2E 2E 2E 20 0D 0A 00 00 00 00 00 gain ...
000000E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000000F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000160 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000170 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

# Hexedit

Content in  
printable  
ASCII

```
user@cop4610: ~
00000000 EB 58 90 6D 6B 64 6F 73 66 73 00 00 02 01 20 00 .X.mkdosfs....
00000010 02 00 00 00 00 F8 00 00 20 00 40 00 00 00 00 00 .....@.....
00000020 00 00 02 00 F1 03 00 00 00 00 00 00 02 00 00 00 .....
00000030 01 00 06 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000040 00 00 29 6E FA 2E 43 20 20 20 20 20 20 20 20 ..)n..C
00000050 20 20 46 41 54 33 32 20 20 20 0E 1F BE 77 7C AC FAT32 ...w|.
00000060 22 C0 74 0B 56 B4 0E BB 07 00 CD 10 5E EB F0 32 ".t.v.....^..2
00000070 E4 CD 16 CD 19 EB FE 54 68 69 73 20 69 73 20 6E .....This is n
00000080 6F 74 20 61 20 62 6F 6F 74 61 62 6C 65 20 64 69 ot a bootable di
00000090 73 6B 2E 20 20 50 6C 65 61 73 65 20 69 6E 73 65 sk. Please inse
000000A0 72 74 20 61 20 62 6F 6F 74 61 62 6C 65 20 66 6C rt a bootable fl
000000B0 6F 70 70 79 20 61 6E 64 0D 0A 70 72 65 73 73 20 oppy and..press
000000C0 61 6E 79 20 6B 65 79 20 74 6F 20 74 72 79 20 61 any key to try a
000000D0 67 61 69 6E 20 2E 2E 2E 20 0D 0A 00 00 00 00 00 gain ...
000000E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
000000F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000160 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000170 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

---

# Hexadecimal Hints

- Hex is base 16 – one hexadecimal can represent 0-15
- It takes 4 binary bits to represent values 0-15
  - $0000 = 0$
  - $1111 = 15$

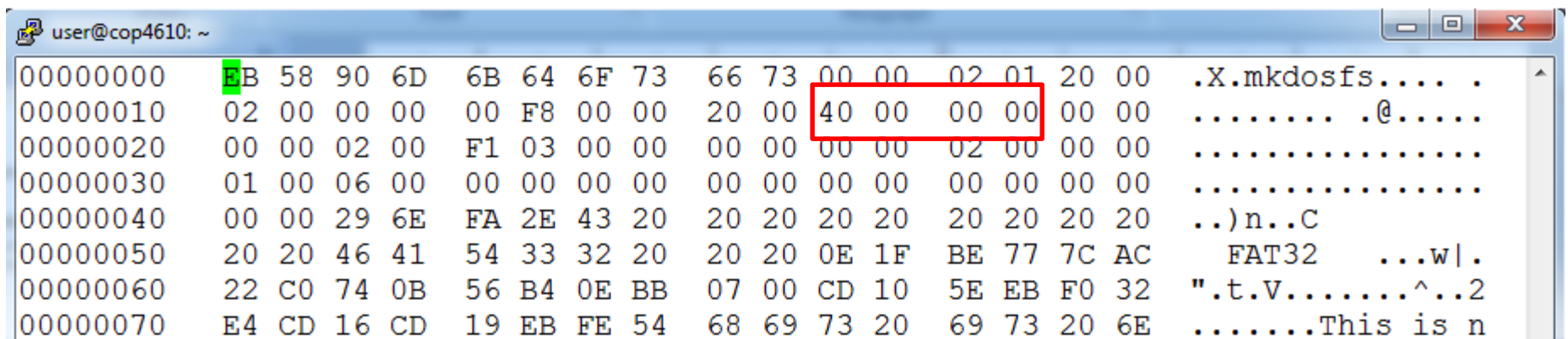
---

# Hexadecimal Hints

- If it takes **4 bits** to represent one hexadecimal number, it takes **8 bits** to represent two hexadecimal numbers
  - 8 bits = 1 byte
- Two hex numbers together symbolize one byte
  - That's why hex numbers are in groups of two

# Endianness

- FAT32 is represented in little endian byte order
  - Reading left to right, you encounter least-significant byte first
  - What 32-bit number is this? 0x0000040 or 0x40000000?



```
user@cop4610: ~  
00000000 EB 58 90 6D 6B 64 6F 73 66 73 00 00 02 01 20 00 .X.mkdosfs....  
00000010 02 00 00 00 00 F8 00 00 20 00 40 00 00 00 00 00 .....@.....  
00000020 00 00 02 00 F1 03 00 00 00 00 00 00 02 00 00 00 .....  
00000030 01 00 06 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000040 00 00 29 6E FA 2E 43 20 20 20 20 20 20 20 20 ..)n..C  
00000050 20 20 46 41 54 33 32 20 20 20 0E 1F BE 77 7C AC FAT32 ...w|.  
00000060 22 C0 74 0B 56 B4 0E BB 07 00 CD 10 5E EB F0 32 ".t.V.....^..2  
00000070 E4 CD 16 CD 19 EB FE 54 68 69 73 20 69 73 20 6E .....This is n
```

# Endianness

- Why are characters in order (readable) if some numbers are not?

```
user@cop4610: ~  
00000000 EB 58 90 6D 6B 64 6F 73 66 73 00 00 02 01 20 00 .X.mkdosfs....  
00000010 02 00 00 00 00 F8 00 00 20 00 40 00 00 00 00 00 .....@.....  
00000020 00 00 02 00 F1 03 00 00 00 00 00 00 02 00 00 00 .....  
00000030 01 00 06 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000040 00 00 29 6E FA 2E 43 20 20 20 20 20 20 20 20 ..)n..C  
00000050 20 20 46 41 54 33 32 20 20 20 0E 1F BE 77 7C AC FAT32 ...w|.  
00000060 22 C0 74 0B 56 B4 0E BB 07 00 CD 10 5E EB F0 32 ".t.v.....^..2  
00000070 E4 CD 16 CD 19 EB FE 54 68 69 73 20 69 73 20 6E .....This is n  
00000080 6F 74 20 61 20 62 6F 6F 74 61 62 6C 65 20 64 69 ot a bootable di  
00000090 73 6B 2E 20 20 50 6C 65 61 73 65 20 69 6E 73 65 sk. Please inse  
000000A0 72 74 20 61 20 62 6F 6F 74 61 62 6C 65 20 66 6C rt a bootable fl  
000000B0 6F 70 70 79 20 61 6E 64 0D 0A 70 72 65 73 73 20 oppy and..press
```



---

# Endianness

- You **must** account for little endianness across bytes when reading in numbers of size larger than one byte
  - Characters are only one byte, no re-ordering necessary

---

# Starting Project 3

---

---

# Parse the Boot Sector

- We need to gather important information about the file system in the boot sector

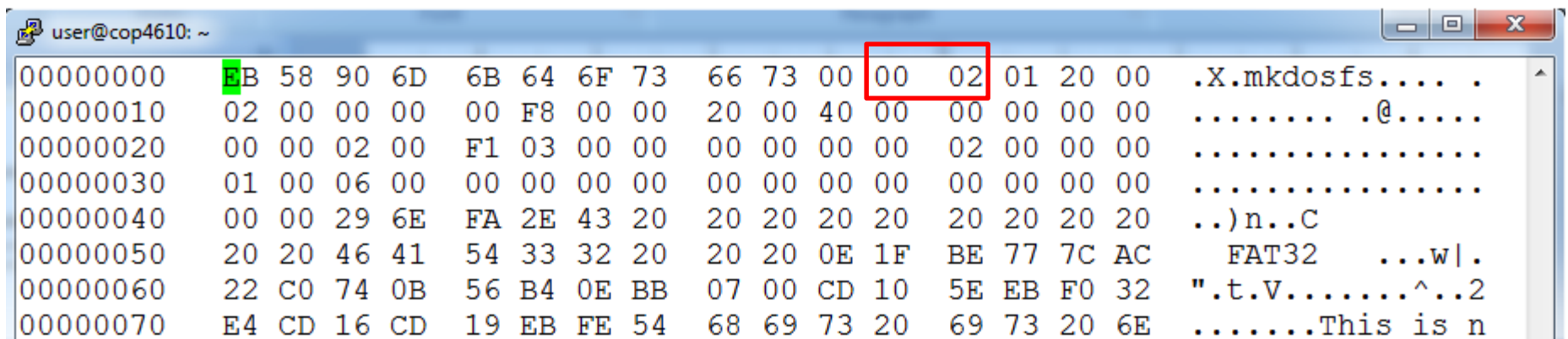
---

# Important Boot Sector Information

- Size of each region
  - BPB\_BytesPerSec
  - BPB\_SecPerClus
  - BPB\_RsvdSecCnt
  - BPB\_NumFATS
  - BPB\_FATSz32
- Root directory (first directory in tree)
  - BPB\_RootClus
- Warning: this list is not exhaustive!

# Important Boot Sector Information

- BPB\_BytesPerSector
  - Offset 11, size 2 bytes
  - 0x0200 = 512



```
user@cop4610: ~  
00000000 EB 58 90 6D 6B 64 6F 73 66 73 00 00 02 01 20 00 .X.mkdosfs.... .  
00000010 02 00 00 00 00 F8 00 00 20 00 40 00 00 00 00 00 ..... @.....  
00000020 00 00 02 00 F1 03 00 00 00 00 00 00 02 00 00 00 .....  
00000030 01 00 06 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000040 00 00 29 6E FA 2E 43 20 20 20 20 20 20 20 20 ..)n..C  
00000050 20 20 46 41 54 33 32 20 20 20 0E 1F BE 77 7C AC FAT32 ...w|.  
00000060 22 C0 74 0B 56 B4 0E BB 07 00 CD 10 5E EB F0 32 ".t.V.....^..2  
00000070 E4 CD 16 CD 19 EB FE 54 68 69 73 20 69 73 20 6E .....This is n
```

---

# Next Steps

- After you have parsed the boot sector and saved key values, you may want to find the root directory
  - Everything starts here...

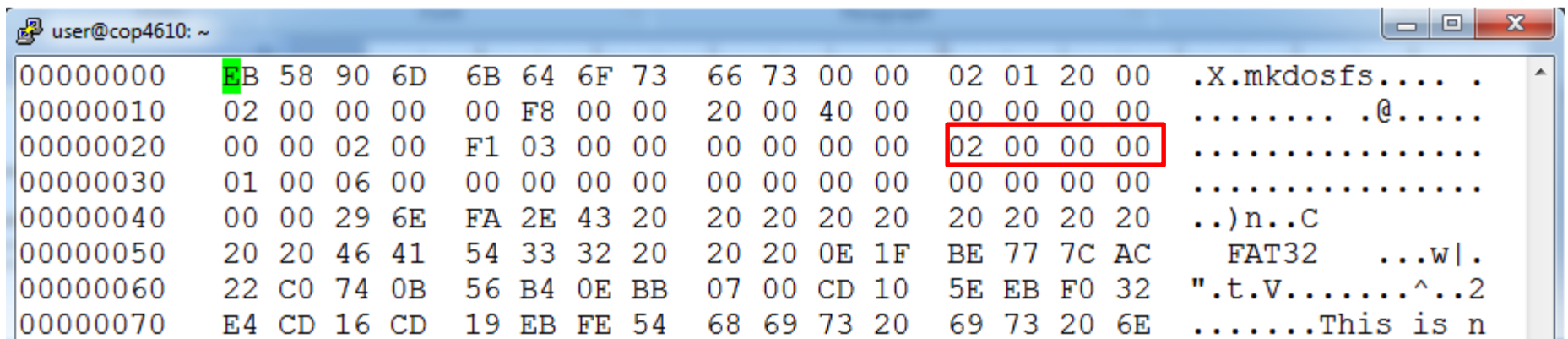
---

# Finding the Root Directory

1. Figure out the ***root directory cluster number*** from the boot sector

# Finding the Root Directory

- BPB\_RootClus
  - Offset 44, size 4 bytes
  - 0x00000002 = 2



The screenshot shows a hex editor window titled "user@cop4610: ~". The main content is a hex dump of a file system boot sector. The first column shows offsets from 00000000 to 00000070. The second column shows the hex values. The third column shows the ASCII representation of the hex values. The value 02 00 00 00 at offset 00000020 is highlighted with a red box. The ASCII representation for this value is ".....".

```
00000000  EB 58 90 6D 6B 64 6F 73 66 73 00 00 02 01 20 00 .X.mkdosfs.... .
00000010  02 00 00 00 00 F8 00 00 20 00 40 00 00 00 00 00 ..... @.....
00000020  00 00 02 00 F1 03 00 00 00 00 00 00 02 00 00 00 .....
00000030  01 00 06 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00000040  00 00 29 6E FA 2E 43 20 20 20 20 20 20 20 20 ..)n..C
00000050  20 20 46 41 54 33 32 20 20 20 0E 1F BE 77 7C AC FAT32 ...w|.
00000060  22 C0 74 0B 56 B4 0E BB 07 00 CD 10 5E EB F0 32 ".t.V.....^..2
00000070  E4 CD 16 CD 19 EB FE 54 68 69 73 20 69 73 20 6E .....This is n
```



---

# Finding the Root Directory

2. Figure out where the root directory starts in the data region, where  $N = \text{cluster number}$ 
  - (We just found  $N=2$ )

```
FirstSectorofCluster = ((N - 2) * BPB_SecPerClus) +  
                        FirstDataSector;
```

# Finding the Root Directory

3. Read in the root directory structure located at the first sector of the root directory cluster

```
user@cop4610: ~
001003F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00100400  41 63 00 6F 00 64 00 65 00 00 00 0F 00 FE FF FF Ac.o.d.e.....
00100410  FF FF FF FF FF FF FF FF FF FF 00 00 FF FF FF .....
00100420  43 4F 44 45 20 20 20 20 20 20 20 10 00 64 B2 6C CODE ..d.l
00100430  5C 3D 5C 3D 00 00 B2 6C 5C 3D 03 00 00 00 00 00 \=\=...l\=\=...
00100440  41 64 00 69 00 72 00 73 00 00 00 0F 00 5D FF FF Ad.i.r.s.....]..
00100450  FF FF FF FF FF FF FF FF FF FF 00 00 FF FF FF .....
00100460  44 49 52 53 20 20 20 20 20 20 20 10 00 64 B2 6C DIRS ..d.l
00100470  5C 3D 5C 3D 00 00 B2 6C 5C 3D 07 00 00 00 00 00 \=\=...l\=\=...
00100480  41 66 00 61 00 74 00 67 00 65 00 0F 00 16 6E 00 Af.a.t.g.e....n.
00100490  31 00 30 00 33 00 2E 00 70 00 00 00 64 00 66 00 1.0.3...p...d.f.
001004A0  46 41 54 47 45 4E 7E 31 50 44 46 20 00 64 B2 6C FATGEN~1PDF .d.l
001004B0  5C 3D 5C 3D 00 00 B2 6C 5C 3D 11 00 FD 89 07 00 \=\=...l\=\=...
001004C0  41 46 00 41 00 54 00 69 00 6E 00 0F 00 D2 66 00 AF.A.T.i.n....f.
001004D0  6F 00 2E 00 74 00 78 00 74 00 00 00 00 00 FF FF o...t.x.t.....
001004E0  46 41 54 49 4E 46 4F 20 54 58 54 20 00 64 B2 6C FATINFO TXT .d.l
001004F0  5C 3D 5C 3D 00 00 B2 6C 5C 3D D6 03 35 01 00 00 \=\=...l\=\=...5...
00100500  41 66 00 69 00 6C 00 65 00 73 00 0F 00 79 00 00 Af.i.l.e.s...y..0.
--- fat32.img --0x100400/0x4000000-----0.
```

---

# Finding the Root Directory

4. Does the root directory span more than one cluster? Look up the *next cluster number* in the FAT.
  - ❑ Find `ThisFATSecNum` and `ThisFATEntOffset` for the current cluster number
  - ❑ Go to `ThisFATSecNum` and read the 32-bit unsigned value starting at offset `ThisFATEntOffset`
  - ❑ FAT will either give you the next cluster number in the directory or the **End of Cluster Chain** value

---

# What exactly is the FAT?

- Files and directories may span multiple clusters
- FAT is a database or array of pointers to the ***next*** cluster number of the file or directory

# Finding the Root Directory

- Next cluster number of root directory in FAT
  - EoC=0x0FFFFFFF8 – directory does not go on

```
user@cop4610: ~
00004000  F8 FF FF 0F  FF FF FF 0F  F8 FF FF 0F  FF FF FF 0F  .....
00004010  FF FF FF 0F  FF FF FF 0F  FF FF FF 0F  FF FF FF 0F  .....
00004020  FF FF FF 0F  FF FF FF 0F  FF FF FF 0F  FF FF FF 0F  .....
00004030  FF FF FF 0F  FF FF FF 0F  FF FF FF 0F  FF FF FF 0F  .....
00004040  FF FF FF 0F  12 00 00 00  13 00 00 00  14 00 00 00  .....
00004050  15 00 00 00  16 00 00 00  17 00 00 00  18 00 00 00  .....
00004060  19 00 00 00  1A 00 00 00  1B 00 00 00  1C 00 00 00  .....
00004070  1D 00 00 00  1E 00 00 00  1F 00 00 00  20 00 00 00  .....
00004080  21 00 00 00  22 00 00 00  23 00 00 00  24 00 00 00  !..."....#...$...
00004090  25 00 00 00  26 00 00 00  27 00 00 00  28 00 00 00  %...&...'...( ...
000040A0  29 00 00 00  2A 00 00 00  2B 00 00 00  2C 00 00 00  )...*...+...,...
000040B0  2D 00 00 00  2E 00 00 00  2F 00 00 00  30 00 00 00  -...../...0...
000040C0  31 00 00 00  32 00 00 00  33 00 00 00  34 00 00 00  1...2...3...4...
000040D0  35 00 00 00  36 00 00 00  37 00 00 00  38 00 00 00  5...6...7...8...
000040E0  39 00 00 00  3A 00 00 00  3B 00 00 00  3C 00 00 00  9...:...;...<...
000040F0  3D 00 00 00  3E 00 00 00  3F 00 00 00  40 00 00 00  =...>...?...@...
00004100  41 00 00 00  42 00 00 00  43 00 00 00  44 00 00 00  A...B...C...D...
00004110  45 00 00 00  46 00 00 00  47 00 00 00  48 00 00 00  E...F...G...H...0.
--- fat32.img --0x4008/0x4000000-----0.
```

# Finding the Root Directory

- Next cluster number of root directory in FAT
  - EoC=0x0FFFFFFF8 – directory does not go on

```
user@cop4610: ~
00004000  F8 FF FF 0F  FF FF FF 0F  F8 FF FF 0F  FF FF FF 0F  .....
00004010  FF FF FF 0F  FF FF FF 0F  FF FF FF 0F  FF FF FF 0F  .....
00004020  FF FF FF 0F  FF FF FF 0F  FF FF FF 0F  FF FF FF 0F  .....
00004030  FF FF FF 0F  FF FF FF 0F  FF FF 0F  FF FF FF 0F  .....
00004040  FF FF FF 0F  12 00 00 00  00 00 00  14 00 00 00  .....
00004050  15 00 00 00  16 00 00 00  17 00 00 00  18 00 00 00  .....
00004060  19 00 00 00  1A 00 00 00  1B 00 00 00  1C 00 00 00  .....
00004070  1D 00 00 00  1E 00 00 00  1F 00 00 00  20 00 00 00  .....
00004080  21 00 00 00  22 00 00 00  23 00 00 00  24 00 00 00  !..."....#...$...
00004090  25 00 00 00  26 00 00 00  27 00 00 00  28 00 00 00  %...&...'...( ...
000040A0  29 00 00 00  2A 00 00 00  2B 00 00 00  2C 00 00 00  )...*...+...,...
000040B0  2D 00 00 00  2E 00 00 00  2F 00 00 00  30 00 00 00  -...../...0...
000040C0  31 00 00 00  32 00 00 00  33 00 00 00  34 00 00 00  1...2...3...4...
000040D0  35 00 00 00  36 00 00 00  37 00 00 00  38 00 00 00  5...6...7...8...
000040E0  39 00 00 00  3A 00 00 00  3B 00 00 00  3C 00 00 00  9...:...;...<...
000040F0  3D 00 00 00  3E 00 00 00  3F 00 00 00  40 00 00 00  =...>...?...@...
00004100  41 00 00 00  42 00 00 00  43 00 00 00  44 00 00 00  A...B...C...D...
00004110  45 00 00 00  46 00 00 00  47 00 00 00  48 00 00 00  E...F...G...H...0.
--- fat32.img --0x4008/0x4000000-----0.
```

---

# Directory Structure

- Each directory is made up of one or more ***directory entries*** that contain
  - File name (or sub-directory name)
  - Attributes
  - First cluster number
    - Cluster number where file or directory in question starts
  - More...

---

# Finding Files and Directories

- Files and sub-directory entries can be found by going to their ***first cluster number***
  - Found in the directory entry



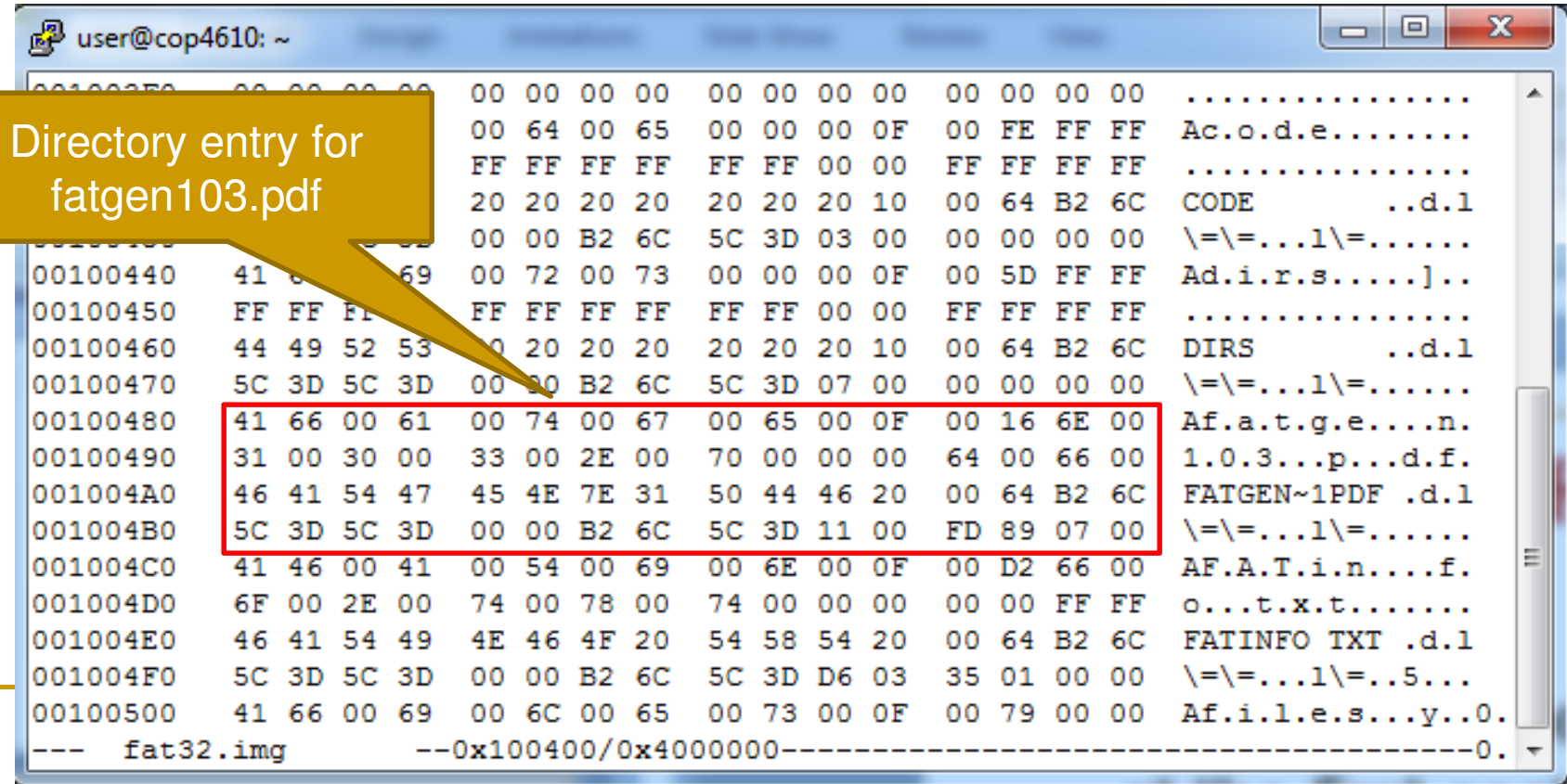
# Finding fatgen103.pdf

- Suppose we have read in the root directory and want to find the file 'fatgen103.pdf'

```
user@cop4610: ~
001003F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00100400  41 63 00 6F 00 64 00 65 00 00 00 0F 00 FE FF FF Ac.o.d.e.....
00100410  FF FF FF FF FF FF FF FF FF FF 00 00 FF FF FF FF .....
00100420  43 4F 44 45 20 20 20 20 20 20 20 10 00 64 B2 6C CODE ..d.l
00100430  5C 3D 5C 3D 00 00 B2 6C 5C 3D 03 00 00 00 00 00 \=\=...l\=.....
00100440  41 64 00 69 00 72 00 73 00 00 00 0F 00 5D FF FF Ad.i.r.s.....]..
00100450  FF FF FF FF FF FF FF FF FF FF 00 00 FF FF FF FF .....
00100460  44 49 52 53 20 20 20 20 20 20 20 10 00 64 B2 6C DIRS ..d.l
00100470  5C 3D 5C 3D 00 00 B2 6C 5C 3D 07 00 00 00 00 00 \=\=...l\=.....
00100480  41 66 00 61 00 74 00 67 00 65 00 0F 00 16 6E 00 Af.a.t.g.e....n.
00100490  31 00 30 00 33 00 2E 00 70 00 00 00 64 00 66 00 1.0.3...p...d.f.
001004A0  46 41 54 47 45 4E 7E 31 50 44 46 20 00 64 B2 6C FATGEN~1PDF .d.l
001004B0  5C 3D 5C 3D 00 00 B2 6C 5C 3D 11 00 FD 89 07 00 \=\=...l\=.....
001004C0  41 46 00 41 00 54 00 69 00 6E 00 0F 00 D2 66 00 AF.A.T.i.n....f.
001004D0  6F 00 2E 00 74 00 78 00 74 00 00 00 00 00 FF FF o...t.x.t.....
001004E0  46 41 54 49 4E 46 4F 20 54 58 54 20 00 64 B2 6C FATINFO TXT .d.l
001004F0  5C 3D 5C 3D 00 00 B2 6C 5C 3D D6 03 35 01 00 00 \=\=...l\=..5...
00100500  41 66 00 69 00 6C 00 65 00 73 00 0F 00 79 00 00 Af.i.l.e.s...y..0.
--- fat32.img --0x100400/0x4000000-----0.
```

# Finding fatgen103.pdf

- Suppose we have read in the root directory and want to find the file 'fatgen103.pdf'



```
user@cop4610: ~
001003E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
001003F0 00 64 00 65 00 00 00 0F 00 FE FF FF Ac.o.d.e.....
00100400 FF FF FF FF FF FF 00 00 FF FF FF FF .....
00100410 20 20 20 20 20 20 20 10 00 64 B2 6C CODE ..d.l
00100420 00 00 B2 6C 5C 3D 03 00 00 00 00 00 \=\=...l\=.....
00100430 00 00 00 00 00 00 00 00 00 00 00 00 .....
00100440 41 00 00 69 00 72 00 73 00 00 00 0F 00 5D FF FF Ad.i.r.s.....]..
00100450 FF FF FF FF FF FF 00 00 FF FF FF FF .....
00100460 44 49 52 53 00 20 20 20 20 20 20 10 00 64 B2 6C DIRS ..d.l
00100470 5C 3D 5C 3D 00 00 B2 6C 5C 3D 07 00 \=\=...l\=.....
00100480 41 66 00 61 00 74 00 67 00 65 00 0F 00 16 6E 00 Af.a.t.g.e....n.
00100490 31 00 30 00 33 00 2E 00 70 00 00 00 64 00 66 00 1.0.3...p...d.f.
001004A0 46 41 54 47 45 4E 7E 31 50 44 46 20 00 64 B2 6C FATGEN~1PDF .d.l
001004B0 5C 3D 5C 3D 00 00 B2 6C 5C 3D 11 00 \=\=...l\=.....
001004C0 41 46 00 41 00 54 00 69 00 6E 00 0F 00 D2 66 00 AF.A.T.i.n....f.
001004D0 6F 00 2E 00 74 00 78 00 74 00 00 00 00 00 FF FF o...t.x.t.....
001004E0 46 41 54 49 4E 46 4F 20 54 58 54 20 00 64 B2 6C FATINFO TXT .d.l
001004F0 5C 3D 5C 3D 00 00 B2 6C 5C 3D D6 03 \=\=...l\=..5...
00100500 41 66 00 69 00 6C 00 65 00 73 00 0F 00 79 00 00 Af.i.l.e.s...y..0.
--- fat32.img --0x100400/0x4000000-----0.
```

Directory entry for fatgen103.pdf

# Finding fatgen103.pdf

- Entry's first cluster number
  - $0x000011 = 17$

```
user@cop4610: ~
001003F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00100400  41 63 00 6F 00 64 00 65 00 00 00 0F 00 FE FF FF Ac.o.d.e.....
00100410  FF FF FF FF FF FF FF FF FF FF 00 00 FF FF FF FF .....
00100420  43 4F 44 45 20 20 20 20 20 20 20 10 00 64 B2 6C CODE ..d.l
00100430  5C 3D 5C 3D 00 00 B2 6C 5C 3D 03 00 00 00 00 00 \=\=...l\=.....
00100440  00 00 00 00 00 00 00 00 00 00 00 00 00 5D FF FF Ad.i.r.s.....]..
00100450  FF FF FF FF FF FF FF FF FF FF FF FF FF FF .....
00100460  00 20 00 00 00 00 00 00 00 64 B2 6C DIRS ..d.l
00100470  5C 3D 5C 3D 00 00 B2 6C 5C 3D 00 00 00 00 00 00 \=\=...l\=.....
00100480  41 66 00 61 00 74 00 67 00 65 00 0F 00 16 6E 00 Af.a.t.g.e....n.
00100490  31 00 30 00 00 00 2E 00 70 00 00 00 64 00 66 00 1.0.3...p...d.f.
001004A0  46 41 54 47 43 4E 7E 31 50 44 46 20 00 64 B2 6C FATGEN~1PDF .d.l
001004B0  5C 3D 5C 3D 00 00 B2 6C 5C 3D 11 00 FD 89 07 00 \=\=...l\=.....
001004C0  41 46 00 41 00 54 00 69 00 6E 00 0F 00 D2 66 00 AF.A.T.i.n....f.
001004D0  6F 00 2E 00 74 00 78 00 74 00 00 00 00 00 FF FF o...t.x.t.....
001004E0  46 41 54 49 4E 46 4F 20 54 58 54 20 00 64 B2 6C FATINFO TXT .d.l
001004F0  5C 3D 5C 3D 00 00 B2 6C 5C 3D D6 03 35 01 00 00 \=\=...l\=..5...
00100500  41 66 00 69 00 6C 00 65 00 73 00 0F 00 79 00 00 Af.i.l.e.s...y..0.
--- fat32.img --0x100400/0x400000-----0.
```

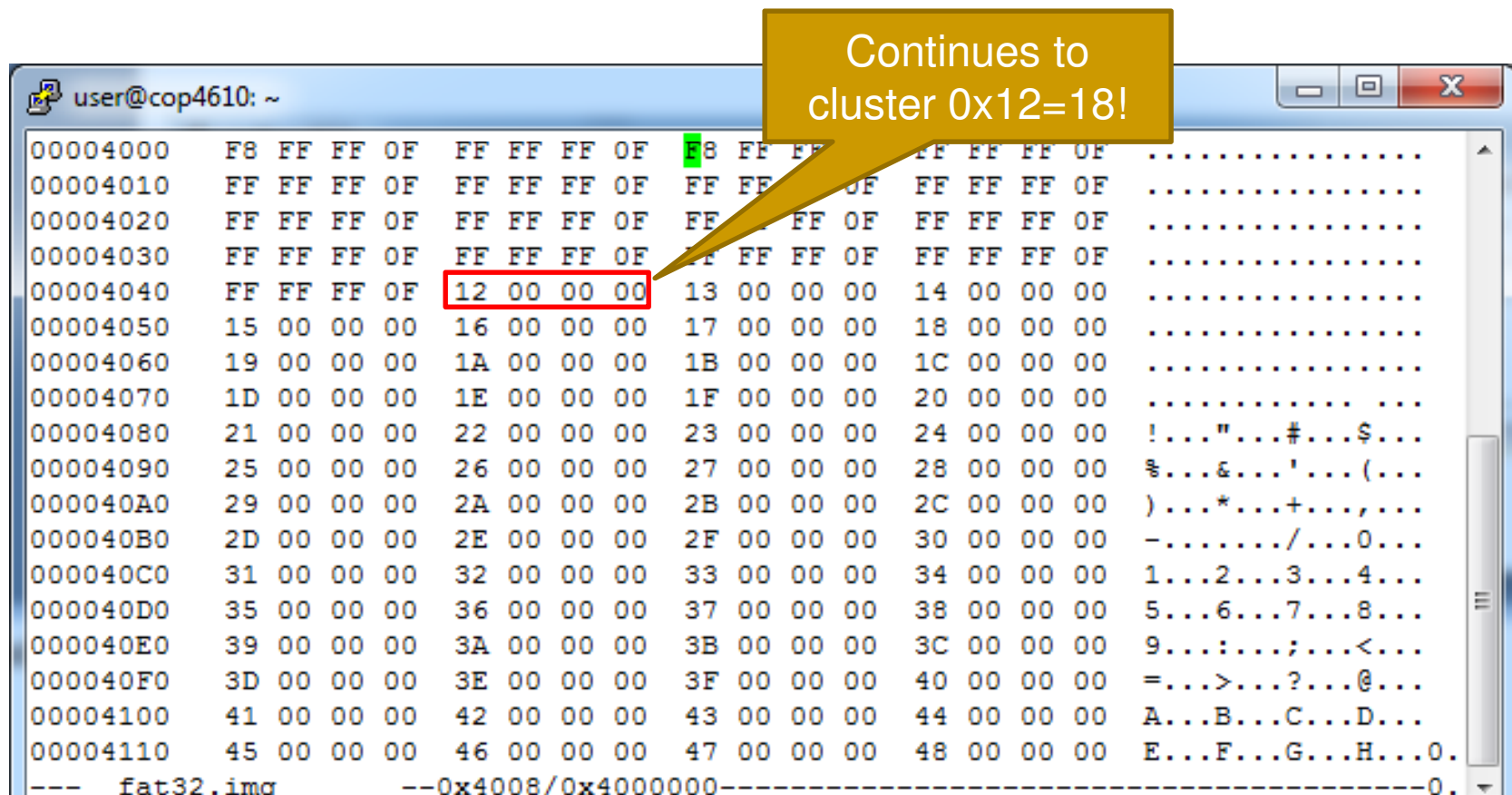
# Finding fatgen103.pdf

- Plug  $N=17$  into `FirstSectorofCluster` equation, go to that sector...

```
user@cop4610: ~
001021F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
00102200  25 50 44 46 2D 31 2E 34 0A 25 C3 A4 C3 BC C3 B6 %PDF-1.4%.....
00102210  C3 9F 0A 32 20 30 20 6F 62 6A 0A 3C 3C 2F 4C 65 ...2 0 obj.<</Le
00102220  6E 67 74 68 20 33 20 30 20 52 2F 46 69 6C 74 65 ngth 3 0 R/Filte
00102230  72 2F 46 6C 61 74 65 44 65 63 6F 64 65 3E 3E 0A r/FlateDecode>>.
00102240  73 74 72 65 61 6D 0A 78 9C ED 3D C9 8E 1C B9 72 stream.x..=....r
00102250  F7 FA 8A 3A 0F 50 6D EE 99 04 84 06 7A 35 E0 DB ...:Pm.....z5..
00102260  D8 02 7C 18 BC 93 FD C6 82 31 32 30 73 79 BF 6F ..|.....120sy.o
00102270  C6 46 46 56 25 99 A5 D6 3C 65 6B 54 10 C4 CE 28 .FFV%...<ekT...(
00102280  92 C1 D8 18 11 24 73 31 77 F6 F8 8F C3 EF 47 73 .....$s1w.....Gs
00102290  3C 99 72 99 AC 2B E5 94 A1 FC E3 EF C7 FF FC E9 <.r..+.....
001022A0  F8 7F 87 9F 8F BF 1F 33 56 96 D2 1F 83 4F 77 F9 .....3V....Ow.
001022B0  68 A7 BB 54 5B 98 23 FC FB E3 7F 0E 73 86 DE 66 h..T[.##.....s.f
001022C0  BE 8B C7 CF 07 1B 7C 69 43 D0 6F 47 9B E7 3B 57 .....|iC.oG...;W
001022D0  21 17 DD DD 54 21 6F A6 82 98 A1 83 4F F6 6E AE !...T!o.....O.n.
001022E0  75 C1 A6 BB D0 A0 C9 94 C1 05 8A 2E AA 11 08 4A u.....J
001022F0  D9 96 F2 B7 03 B5 64 88 B1 08 44 23 54 08 47 17 .....d...D#T.G.
--- fat32.img          --0x102200/0x400000-----0.
```

# Finding fatgen103.pdf

- Does the file continue after this cluster?
  - Look up current cluster number 17 in FAT...



Continues to cluster 0x12=18!

```
user@cop4610: ~
00004000  F8 FF FF 0F  FF FF FF 0F  F8 FF FF  FF FF FF 0F  .....
00004010  FF FF FF 0F  FF FF FF 0F  FF FF  FF FF FF 0F  .....
00004020  FF FF FF 0F  FF FF FF 0F  FF  FF 0F  FF FF FF 0F  .....
00004030  FF FF FF 0F  FF FF FF 0F  FF  FF 0F  FF FF FF 0F  .....
00004040  FF FF FF 0F  12 00 00 00  13 00 00 00  14 00 00 00  .....
00004050  15 00 00 00  16 00 00 00  17 00 00 00  18 00 00 00  .....
00004060  19 00 00 00  1A 00 00 00  1B 00 00 00  1C 00 00 00  .....
00004070  1D 00 00 00  1E 00 00 00  1F 00 00 00  20 00 00 00  .....
00004080  21 00 00 00  22 00 00 00  23 00 00 00  24 00 00 00  !..."....#...$...
00004090  25 00 00 00  26 00 00 00  27 00 00 00  28 00 00 00  %...&... '... (...
000040A0  29 00 00 00  2A 00 00 00  2B 00 00 00  2C 00 00 00  )...*...+...,...
000040B0  2D 00 00 00  2E 00 00 00  2F 00 00 00  30 00 00 00  -...../...0...
000040C0  31 00 00 00  32 00 00 00  33 00 00 00  34 00 00 00  1...2...3...4...
000040D0  35 00 00 00  36 00 00 00  37 00 00 00  38 00 00 00  5...6...7...8...
000040E0  39 00 00 00  3A 00 00 00  3B 00 00 00  3C 00 00 00  9...:...;...<...
000040F0  3D 00 00 00  3E 00 00 00  3F 00 00 00  40 00 00 00  =...>...?...@...
00004100  41 00 00 00  42 00 00 00  43 00 00 00  44 00 00 00  A...B...C...D...
00004110  45 00 00 00  46 00 00 00  47 00 00 00  48 00 00 00  E...F...G...H...0.
--- fat32.img          --0x4008/0x4000000-----0.
```

---

# Summary of Finding Files/Dirs

1. Find ***first cluster number*** in directory entry of interesting file or directory
2. Figure out the sector to read using cluster number and `FirstSectorofCluster` equation
3. Read that cluster
4. Figure out if file or directory continues past cluster by looking up `FAT[current cluster number]`
  - If EoC mark stop
  - Else go to 3 with `cluster=FAT[current cluster number]`

---

# File System Utility Operations

---

Open, Close, Read, Write

---

# Starting our Utility

```
$> ./fat32_reader fat32.img  
/]
```

- Fat32\_reader is name of our utility
- Should return a different prompt (like “/]”) to symbolize that user is inside utility



---

# Handling Open Files

- An open file is a file we allow I/O operations to be performed on
    - read
    - write
  - To handle open files, we must maintain a table of files that are open
-

---

# Opening Files

```
/] open "fatinfo.txt" rw
```

- If “fatinfo.txt” is found in the current directory, open “fatinfo.txt”
    - In order to find fatinfo.txt, you must be able to interpret the current directory and determine whether fatinfo.txt is part of that directory
  - Once you open fatinfo.txt, store its name (at least) in the open file table
-

---

# I/O on Open Files

```
/] read "fatinfo.txt" 0 100
```

- Only allow the read if fatinfo.txt is open
  - In order to read fatinfo.txt, you must look into the open file table, look up the address of fatinfo.txt (or store it with its name), and read enough of the data clusters to satisfy the read request
-

---

# I/O on Open Files

```
/] write "fatinfo.txt" 0 "Hello"
```

- If write stays within the cluster
  - Just write data
- If write goes beyond cluster
  1. Find a free cluster, remember as `next_cluster_number`
  2. Change `FAT[current_cluster]` from EoC to `next_cluster_number`
  3. Change `FAT[next_cluster_number]` to EoC
  4. Write the data in the cluster `next_cluster_number`

---

# Closing Files

```
/] close "fatinfo.txt"
```

- Remove this file's entry from the open file table

---

# To Do

- Write code to parse the boot directory
  - Cannot go anywhere without this code
- Write code to open and close files
- Write code to read directories and files
- Try writing to files and directories

---

# Next Time

- Discussion of other file operations