

COP4610

Introduction to Operating Systems

Project #3:

Implementing a FAT32

File System

Info

- TA/Grader: **Juan Pablo Conde**



- Office hours Wed 3:30-4:30pm, Thu 11:30am-12:30pm,
or by appointment
- Deadline: **December 1st, 11:59pm**
 - No late submissions
- Work in teams of 3 members
 - Can work alone with TA approval (not encouraged)
 - Submit your team information on Canvas

A word of advice

- Start early!
 - Takes time to understand (and to code)
- Ask questions
 - Recitation
 - TA's office hours
- Separate the workload evenly
 - Report problems to TA ASAP
 - Team-splitting only authorized by TA
 - Team members that work significantly less will get points deducted
 - If more than one working on the same computer and only one pushes changes, say it on the report

A word of advice

- Work together
 - No need to work on the same computer, but discuss things together
 - Split workload but keep in touch
 - Help teammates struggling with tasks
 - Be proactive

Don't even try to cheat

- Submissions compared
 - Other groups
 - Other submissions from previous classes
 - Public repositories
- Subject to penalties for violating academic honor policy
 - Fail the class
 - Other penalties
 - Not worth the risk
 - Fact: Two groups failed last spring

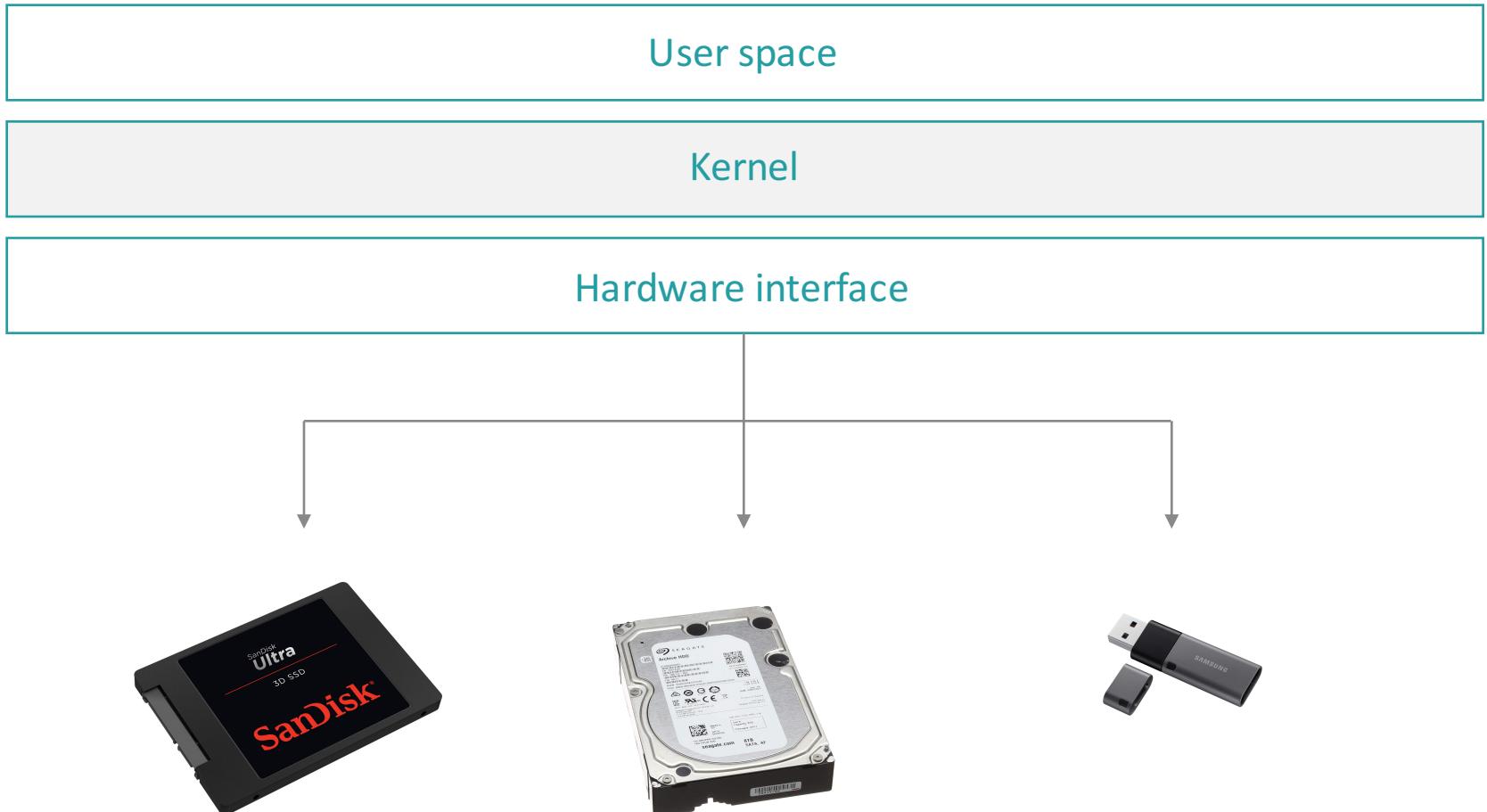
What if you didn't contribute?

- Too bad for you (bad grade)
 - Grades assigned individually
- Appeals on individual grading need to be justified
 - Teammates must support what you say (good luck)
 - Provide Gitlog
 - Other evidence might be requested by the TA
- **Keep this in mind, especially if you are expecting to graduate this semester ;)**

Outline

- FAT32 overview
- Implementation overview
- Hexadecimal and endianness refresher
- Exploring the FAT32 image file
- FAT32 specs
 - Layout
 - Reserved region
 - FAT
 - Data region

FAT32 overview



FAT32 overview

- A file system
- Defines how data is stored in the storage device
 - and also how it is accessed
- File allocation
 - How do I store a file?
- Navigation
 - How do I find a specific file and the blocks that belong to it?
 - How do I navigate through directories?
- Allocated and free space management

Implementation overview

- C language
- No kernel programming
 - Normal executable
- Work on image file
 - Raw bytes
 - Contents have a FAT32 structure
- Program features
 - Read input from user
 - Manipulate the image to access/write data

Hexadecimal refresher

- Base 16: digits range from 0 to 15
 - Represented by digits 0–9 and A–F
 - Prefix ‘0x’ to differentiate from decimal
- Binary to Hex matching
 - 0000 to 1111 → 0 to F (or 0 to 15 in decimal)
 - 4 bits needed to represent a hex digit (2^4)
- Two hex digits = 8 bits = 1 byte
 - E.g: 0xA1, 0xBE, 0x04

Endianness refresher

- Big endian
 - Most significant byte corresponds to least significant digits
 - E.g.: byte seq. 2A 3B 91 23 → 0x2A3B9123
- Little endian
 - Least significant byte corresponds to least significant digits
 - E.g.: byte seq. 2A 3B 91 23 → 0x23913B2A
- Intel architecture uses **little endian**

Exploring the FAT32 image file

- Two approaches
 - Mount the image
 - Use a hex editor
 - **Hexedit (Linux)**
 - Hex Fiend (Mac)
 - HxD (Windows)

Mounting the FAT32 image

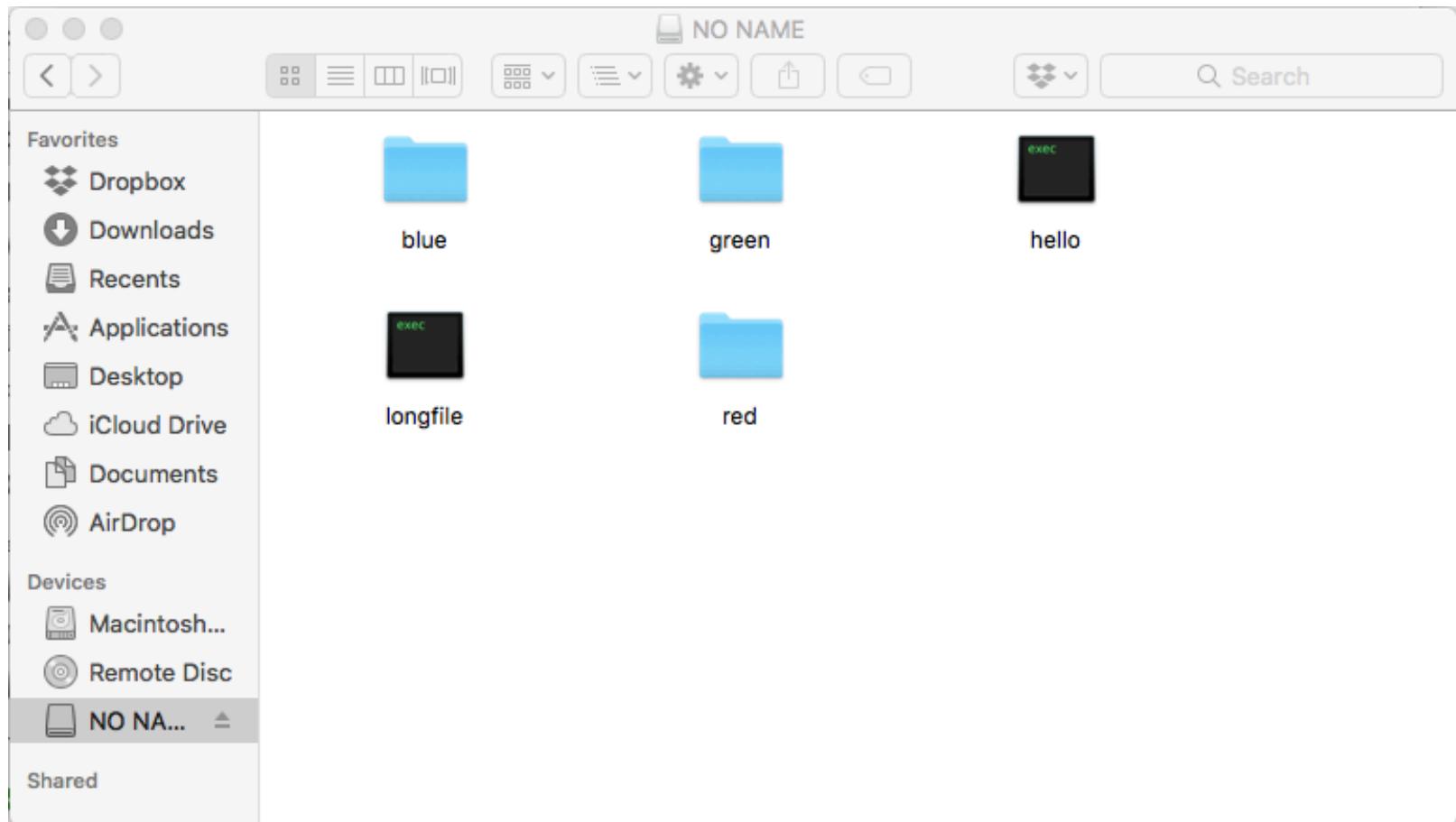
- In a Linux environment:

```
$ mkdir -p mnt_pnt  
$ sudo mount -o loop /path/to/image mnt_pnt  
$ cd mnt_pnt
```

- Explore the image as if it was a normal volume
 - CLI, Finder, Explorer, ...
- To unmount:

```
$ sudo umount mnt_pnt
```
- Note: Your program has to work with the image **unmounted**. Mount only to browse the image

Mounting the FAT32 image



Hexedit

- Often included in Linux distributions
- Free downloads for hex viewers available
 - View raw data content of an image file
- Test your project by exploring the image
 - Hexedit
 - Mounting
- Opening the image:
`$ hexedit fat32.img`

Hexedit

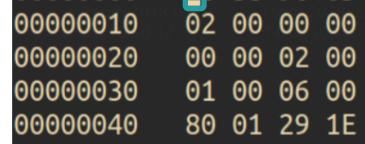
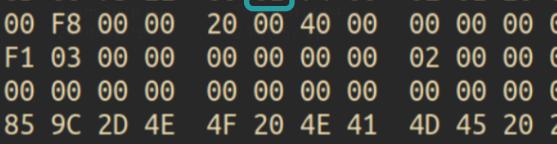
Starting byte offset for each line	Data bytes												ASCII representation			
00000000	E	B	5	8	9	0	6	D	6	B	6	6	7	3	2	E
00000010	0	2	0	0	0	0	0	0	F	8	0	0	0	0	0	0
00000020	0	0	0	0	2	0	0	F	1	0	3	0	0	0	0	0
00000030	0	1	0	0	6	0	0	0	0	0	0	0	0	0	0	0
00000040	8	0	0	1	2	9	1	E	8	5	9	C	2	D	4	E
00000050	2	0	2	0	4	6	4	1	5	4	3	3	2	2	0	0
00000060	2	2	C	0	7	4	0	B	5	6	B	4	0	E	BB	0
00000070	E	4	C	D	1	6	C	D	1	9	E	B	F	E	5	4
00000080	6	F	7	4	2	0	6	1	2	0	6	2	6	F	6	F
00000090	7	3	6	B	2	E	2	0	2	0	5	0	6	C	6	5
000000A0	7	2	7	4	2	0	6	1	2	0	6	2	6	F	6	F
000000B0	6	F	7	0	7	0	7	9	2	0	6	1	6	E	6	4
000000C0	6	1	6	E	7	9	2	0	6	B	6	5	7	9	2	0
000000D0	6	7	6	1	6	9	6	E	2	0	2	E	2	E	2	E
000000E0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
000000F0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
00000100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
00000110	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
00000120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
00000130	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
00000140	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
00000150	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
00000160	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
--- fat32.img -- 0x0/0x40000000---																

Hexedit

- Use **ctrl+g** or **F4** to jump to a new line

00000000	EB 58 90 6D 6B 66 73 2E 66 61 74 00 02 01 20 00 .X.mkfs.fat... .
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 00 02 00 00 00
00000030	01 00 06 00 00 00 00 00 00 00 00 00 00 00 00 00
00000040	80 01 29 1E 85 9C 2D 4E 4F 20 4E 41 4D 45 20 20 ..)....-NO NAME
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 6EThis 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
New position ? 0x █	
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 00
000000F0	00 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 00
00000110	00 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 00
00000130	00 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 00
00000150	00 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 00
--- fat32.img --0x0/0x40000000-----	

Hexedit

	One hex digit (4 bits)	Two hex digits (8 bits = 1 byte)
4 bytes (32 bits)		
One line (16 bytes)		
	00000000 E3 58 90 6D 6B 66 73 2E 66 61 74 00 02 01 20 00 .X.mkfs.fat... .	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 00 02 00 00 00	00000030 01 00 06 00 00 00 00 00 00 00 00 00 00 00 00 00
	00000040 80 01 29 1E 85 9C 2D 4E 4F 20 4E 41 4D 45 20 20 ..)...-NO NAME	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 6EThis 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 00	000000F0 00 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 00 ..	00000110 00 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 00 ..	00000120 00 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 00 ..	00000130 00 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 00 ..	00000140 00 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 00 ..	00000150 00 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 00 ..	00000160 00 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 00 ..	-- fat32.img -- 0x0/0x40000000--

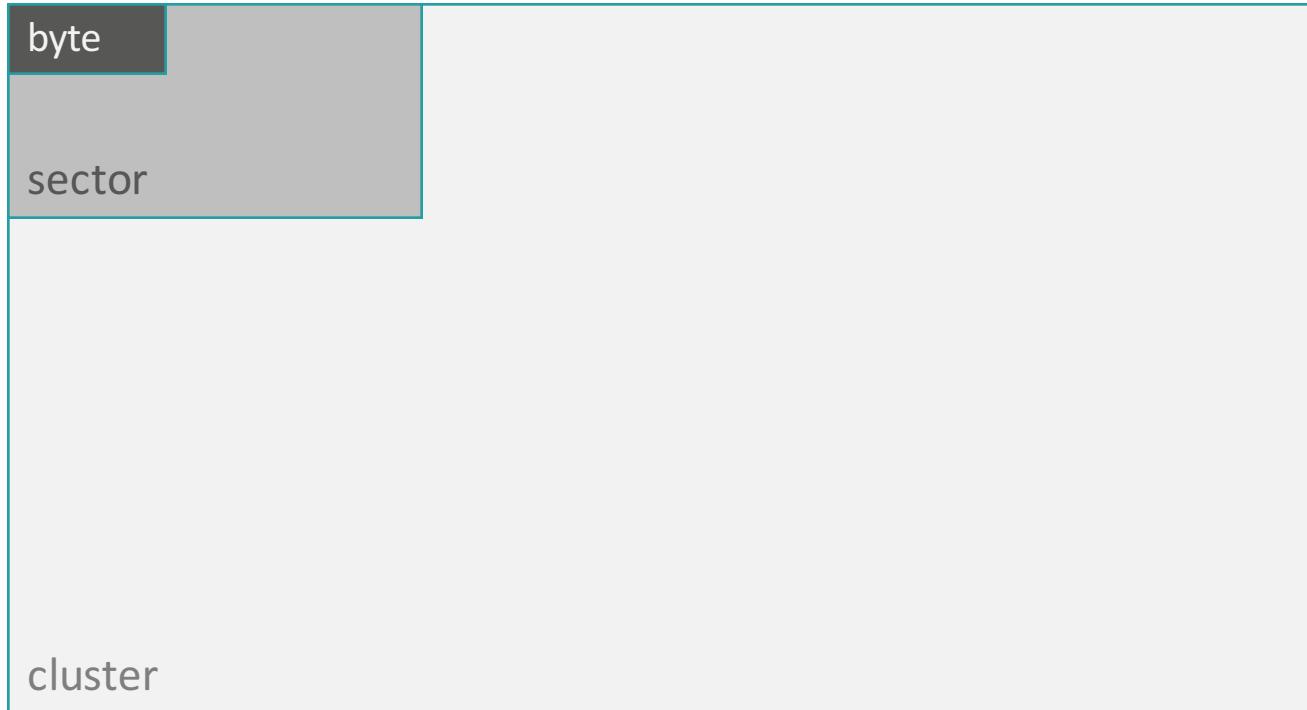
Hexedit

- To make things easier to read, set each line to 16 bytes
- For further help, press F1
 - list of commands and their descriptions

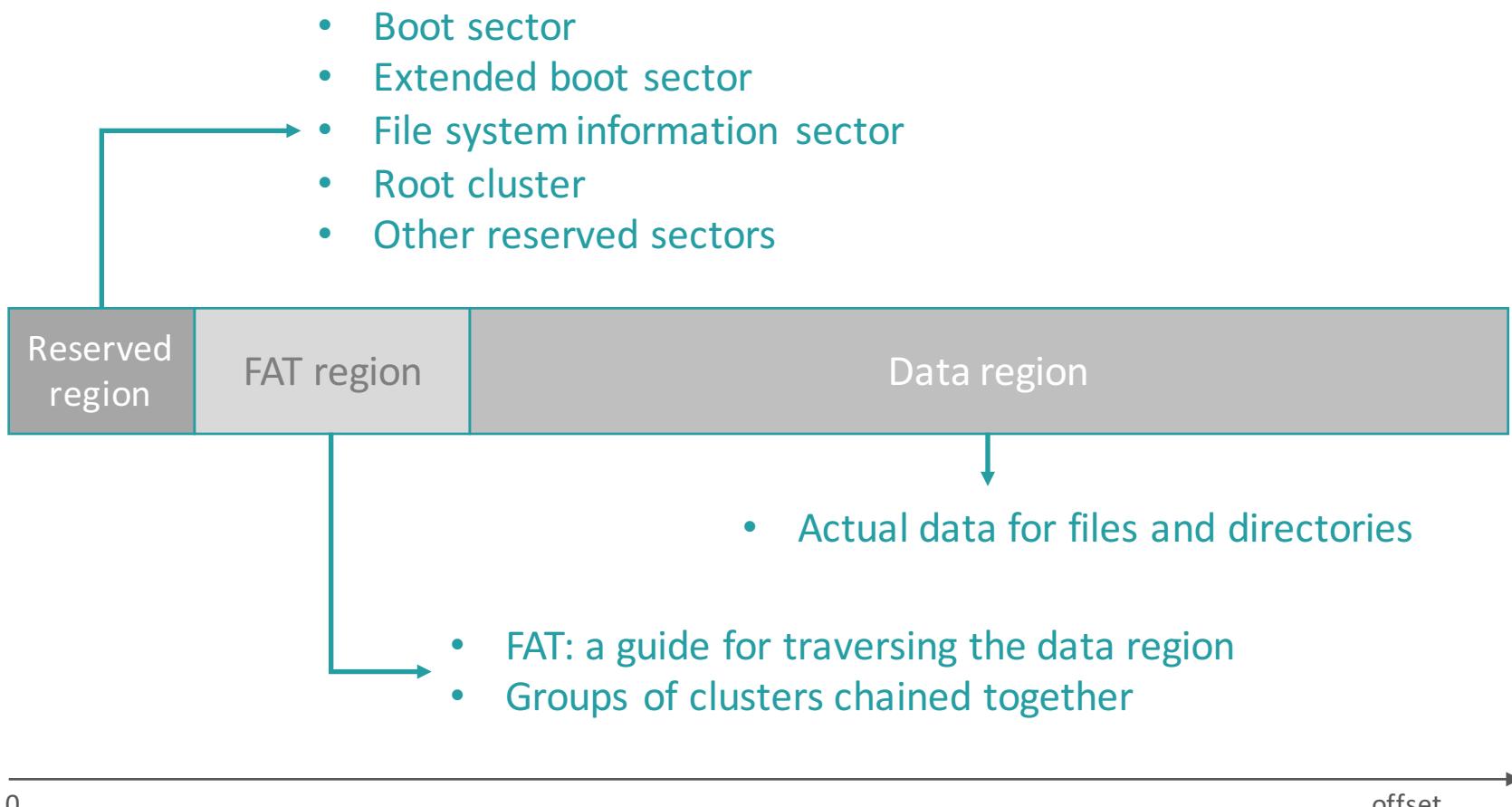
FAT32 terms

- Byte (8 bits)
 - The smallest addressable unit in modern processors
- Sector
 - Smallest addressable unit on a storage device
- Cluster
 - FAT32 specific term
 - A group of sectors representing a chunk of data
- FAT (File Allocation Table)
 - Maps files to data

Byte/Sector/Cluster



FAT32 layout



Reserved region

- Contains information about the file system itself
- When the file system is mounted, the OS reads the reserved region
 - Determines the type of file system
 - Characteristics
 - Size of clusters
 - Number of FATs
 - ...

FAT (File Allocation Table)

- Maps the clusters of data
 - Cluster is the smallest addressable unit for the FAT
- Keeps track of which clusters belong to which files
- Need first cluster to access a file
 - Then find subsequent clusters by using the FAT

FAT

Root cluster				
0000000	XXXXXXX	XXXXXXX	00000009	00000004
0000004	00000005	00000007	00000000	00000008
0000008	FFFFFFF	0000000A	0000000B	00000011
000000C	0000000D	0000000E	FFFFFFF	00000010
0000010	00000012	FFFFFFF	00000013	00000014
0000014	00000015	00000016	FFFFFFF	00000000
0000018	00000000	00000000	00000000	00000000
000001C	00000000	00000000	00000000	00000000
0000020	00000000	00000000	00000000	00000000
...

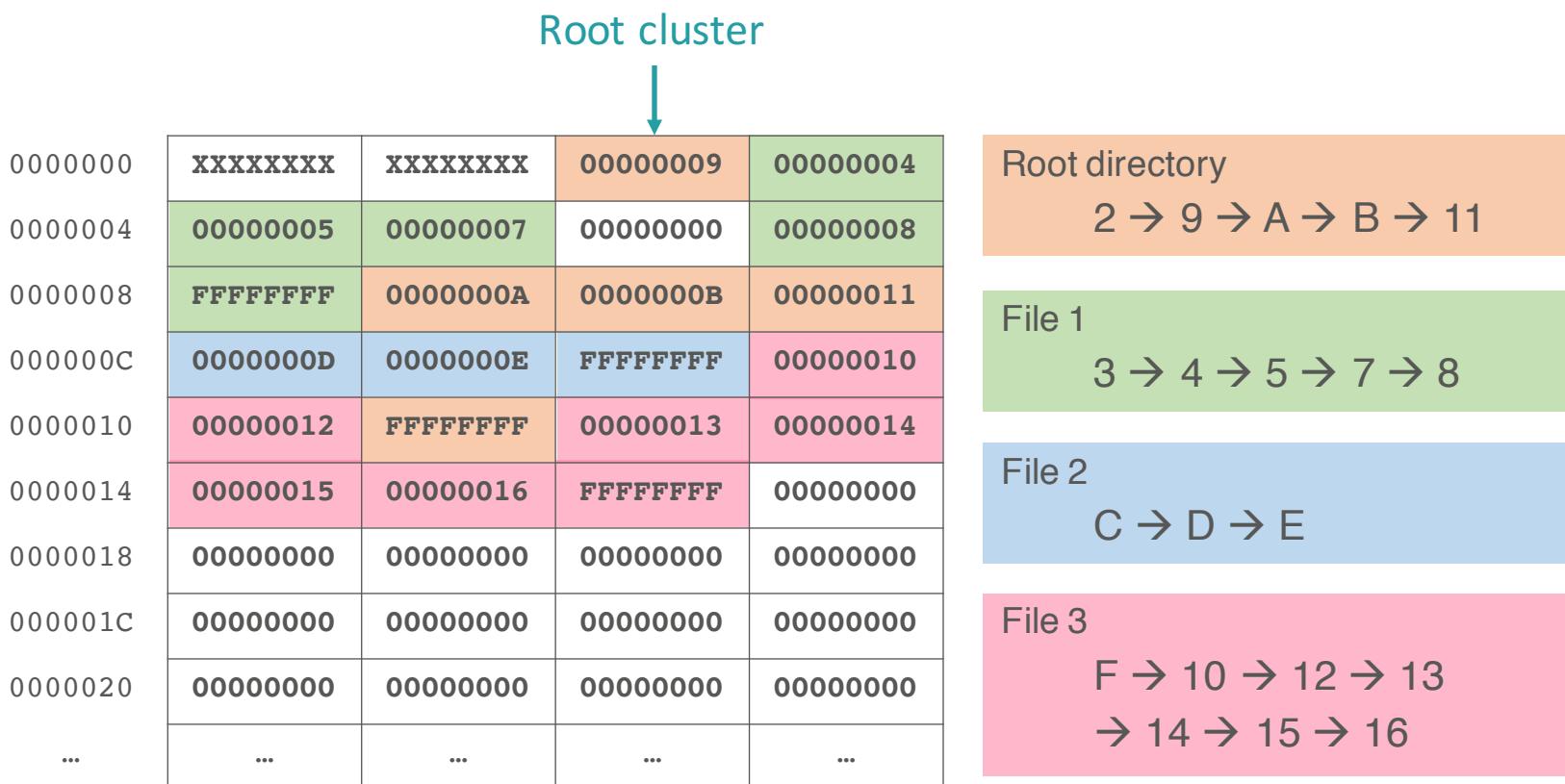
The root cluster is the first cluster for the root directory (noticed root cluster is 2?)

FAT entry indicates where the next cluster is located

- If 0x0: cluster not allocated
- If 0xFFFFFFFF*: last cluster
- Otherwise: indicates next cluster of the file/directory

*also check for values 0xFFFFFFF8 to 0xFFFFFFF in the image, which indicate last cluster too.

FAT



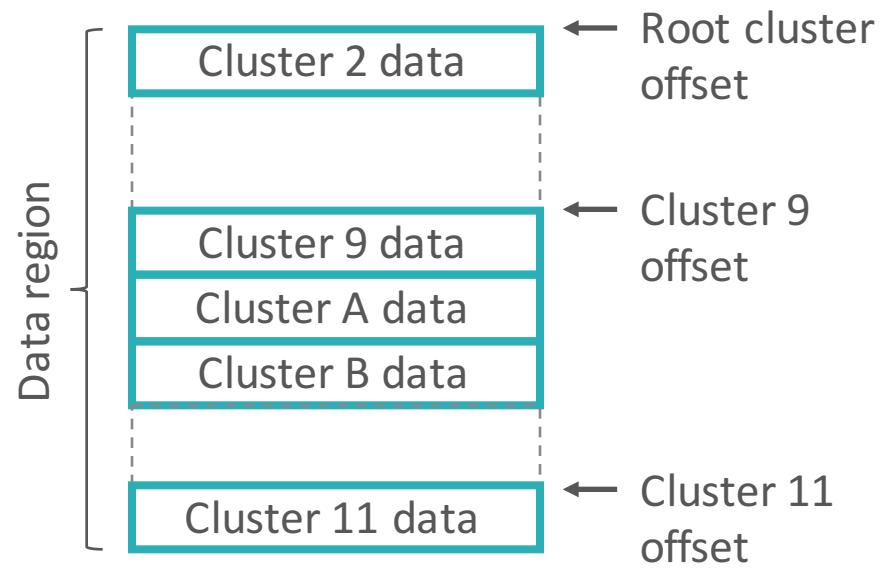
FAT

- So what do the lists of clusters mean?
 - Indicate the offsets within the data region
 - Data for file (or directory) starts at first cluster's offset, then continues on second's, and so on.

E.g: root directory

Clusters:

2 → 9 → A → B → 11



Data region

- Two types of data
 - Files
 - Directories (folders)
- Data stored in clusters
 - Accessing the data requires:
 - Knowing first cluster
 - FAT

Directories

- Composed of directory (DIR) entries
- Directory data (data region)
 - List of DIR entries
 - Each element represents an item (file, subdirectory)
- E.g: directory `my_dir`
 - Contains `a.txt`, `b.txt`, `my_subdir`
 - `my_dir` has a DIR entry for each in data region
 - DIR entries can be all in the same cluster or distributed among different clusters.
 - DIR entries belong to only one cluster (not fragmented)

Directories

- DIR entry
 - Fixed-size data structures
 - Contains information about the entry
 - Name
 - Size
 - Attributes
 - **First cluster number**
 - ...
 - Check FAT32 specs

Directories

- DIR entry
 - Example:

```
/mnt/mnt_pnt $ ls
blue green hello longfile red
```

 - In the data region (for the root directory):
 - Entries for directories blue, green and red
 - Entries for files hello and longfile
 - Every directory (except root directory) will have entries for
 - . (current directory)
 - .. (parent directory)

Directories

- DIR entries

001003F0	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00100400	41 6C 00 6F	00 6E 00 67	00 66 00 0F	00 97 69 00	Al.o.n.g.f....i.	
00100410	6C 00 65 00	00 00 FF FF	FF FF 00 00	FF FF FF FF	l.e.....	
00100420	4C 4F 4E 47	46 49 4C 45	20 20 20 20	00 64 04 8E	LONGFILE ..d..	
00100430	78 4E 49 4F	00 00 04 8E	78 4E 03 00	76 5B 03 00	xNIO....xN..v..	
00100440	41 68 00 65	00 6C 00 6C	00 6F 00 0F	00 14 00 00	Ah.e.l.l.o.....	
00100450	FF FF FF FF	FF FF FF FF	FF FF 00 00	FF FF FF FF	
00100460	48 45 4C 4C	4F 20 20 20	20 20 20 20	00 64 04 8E	HELLO ..d..	
00100470	78 4E 49 4F	00 00 04 8E	78 4E B1 01	0A 00 00 00	xNIO....xN.....	
00100480	41 62 00 6C	00 75 00 65	00 00 00 0F	00 55 FF FF	Ab.l.u.e....U..	
00100490	FF FF FF FF	FF FF FF FF	FF FF 00 00	FF FF FF FF	
001004A0	42 4C 55 45	20 20 20 20	20 20 20 10	00 64 04 8E	BLUE ..d..	
001004B0	78 4E 78 4E	00 00 04 8E	78 4E B2 01	00 00 00 00	xNxN....xN.....	
001004C0	41 67 00 72	00 65 00 65	00 6E 00 0F	00 42 00 00	Ag.r.e.e.n...B..	
001004D0	FF FF FF FF	FF FF FF FF	FF FF 00 00	FF FF FF FF	
001004E0	47 52 45 45	4E 20 20 20	20 20 20 10	00 64 04 8E	GREEN ..d..	
001004F0	78 4E 78 4E	00 00 04 8E	78 4E B3 01	00 00 00 00	xNxN....xN.....	
00100500	41 72 00 65	00 64 00 00	00 FF FF 0F	00 37 FF FF	Ar.e.d.....7..	
00100510	FF FF FF FF	FF FF FF FF	FF FF 00 00	FF FF FF FF	
00100520	52 45 44 20	20 20 20 20	20 20 20 10	00 00 05 8E	RED ..	
00100530	78 4E 78 4E	00 00 05 8E	78 4E B4 01	00 00 00 00	xNxN....xN.....	
00100540	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	
00100550	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	
--- fat32.img --0x100550/0x4000000---						

Long DIR
Name
Entries
(ignore)

DIR entry
for HELLO
file

DIR entry
for GREEN
directory

Files

- Just the content of the files in the data region
 - Plain data. No DIR entries or similar
- File's clusters make up the whole file

```
001005D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
001005E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
001005F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00100600 74 68 69 73 20 69 73 20 61 20 6C 6F 6F 6F 6E 67 this is a looong  
00100610 20 66 69 6C 65 0A 74 68 69 73 20 69 73 20 61 20 file.this is a  
00100620 6C 6F 6F 6E 67 20 66 69 6C 65 0A 74 68 69 73 looong file.this  
00100630 20 69 73 20 61 20 6C 6F 6F 6F 6E 67 20 66 69 6C is a looong fil  
00100640 65 0A 74 68 69 73 20 69 73 20 61 20 6C 6F 6F 6F e.this is a looo  
00100650 6E 67 20 66 69 6C 65 0A 74 68 69 73 20 69 73 20 ng file.this is  
00100660 61 20 6C 6F 6F 6E 67 20 66 69 6C 65 0A 74 68 a looong file.th  
00100670 69 73 20 69 73 20 61 20 6C 6F 6F 6F 6E 67 20 66 is is a looong f  
00100680 69 6C 65 0A 74 68 69 73 20 69 73 20 61 20 6C 6F ile.this is a lo  
00100690 6F 6F 6E 67 20 66 69 6C 65 0A 74 68 69 73 20 69 oong file.this i  
001006A0 73 20 61 20 6C 6F 6F 6E 67 20 66 69 6C 65 0A s a looong file.  
001006B0 74 68 69 73 20 69 73 20 61 20 6C 6F 6F 6E 67 this is a looong  
001006C0 20 66 69 6C 65 0A 74 68 69 73 20 69 73 20 61 20 file.this is a  
001006D0 6C 6F 6F 6E 67 20 66 69 6C 65 0A 74 68 69 73 looong file.this  
001006E0 20 69 73 20 61 20 6C 6F 6F 6F 6E 67 20 66 69 6C is a looong fil  
001006F0 65 0A 74 68 69 73 20 69 73 20 61 20 6C 6F 6F 6F e.this is a looo  
00100700 6E 67 20 66 69 6C 65 0A 74 68 69 73 20 69 73 20 ng file.this is  
00100710 61 20 6C 6F 6F 6E 67 20 66 69 6C 65 0A 74 68 a looong file.th  
00100720 69 73 20 69 73 20 61 20 6C 6F 6F 6F 6E 67 20 66 is is a looong f  
--- fat32.img -- 0x1005D0/0x40000000-----
```

Questions?