CS347m/CS219 Lecture 22

persistence

the file abstraction

fd	inode	block
process	FS	disk

inode — metadata object for files

- filetype = dir/file/...
- name, size, permissions
- datablocks (loc^{n}s of disk for file content)

(linear order specified via block nos.)

23, 47, 58

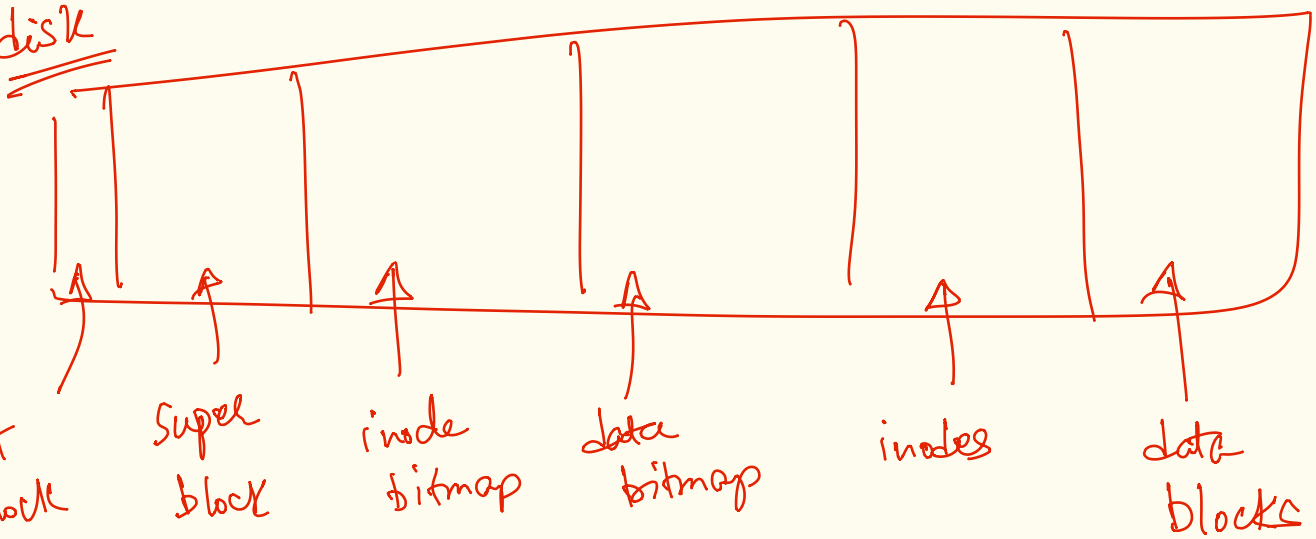
Lab quiz

1. 5th Nov. 830 am SL2
2. SL2 TA hours
3. Fri. 31st Oct. in-class hands-on session

CC105

- xv6
- syscall interface usage

disk



(#) what happens on a process file access.

e.g: fd = open("foo.txt", R|W);

6

read(fd, buffer, size);

FS

fd → inode number

inode number → inode → checks: size, permissions

offset → datablock index/number.

(perform/access block from disk)

(Q) how to find the inode / inode number?

fd = open("/home/foo.txt", R|W);

pathname ~ 3 components

2 + 1 = 3 files

directories file

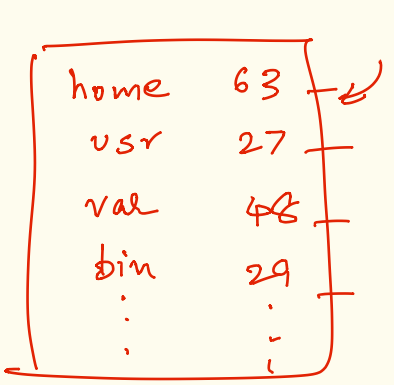
(1) inode number of "/" is well known

(2) read inode of "/" & check permissions. & ensure the inode flag is set to directory

③ use datablocks to read contents / listing of directory?

- FS-specific seq. of directory listing in the datablocks

e.g:

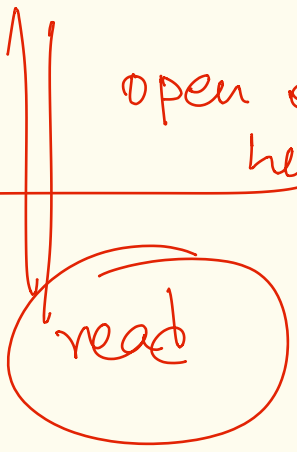


- read datablocks of "/"
  - look for home
    - note its inode number

④ lookup inode of home whose parent is /
- read datablocks
  - find foo.txt — check ~~perm~~ permissions
  - ~~look~~ note inode number of foo.txt.

⑤ look inode of foo.txt
- check permissions
- map offset to datablock
- read the block
- copy to user buffer
- return from syscall.

open ends here



⊛ for FS operations open/read/write — single syscall
⇒ multiple reads/writes by FS to disk!

~ FS optimizations. (in-memory objects)

① dentry cache

directory entry cache

parent directory info.	PATH component	→ inode number
	/	2
ptr. to parent	home	63
	usr	23

build this mapping from disk & store in-memory

② inode cache

inode number → inode contents

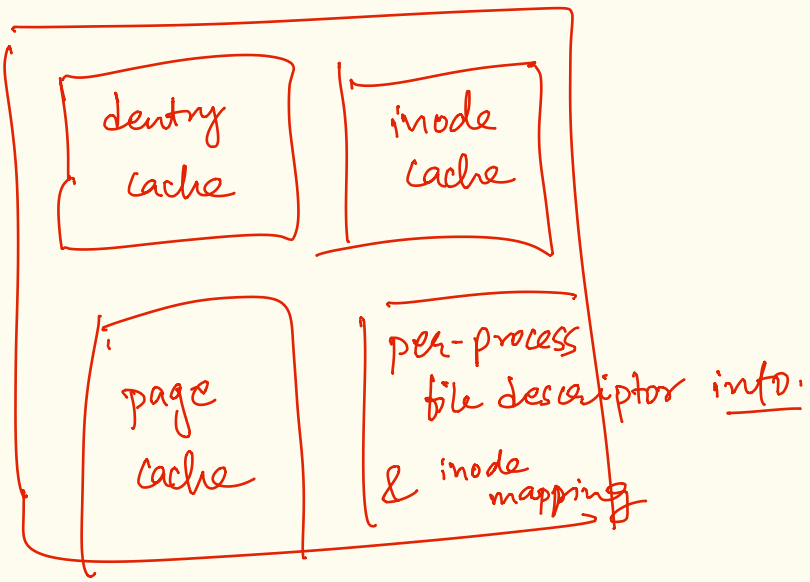
2 → inode object / all disk inode contents + some more

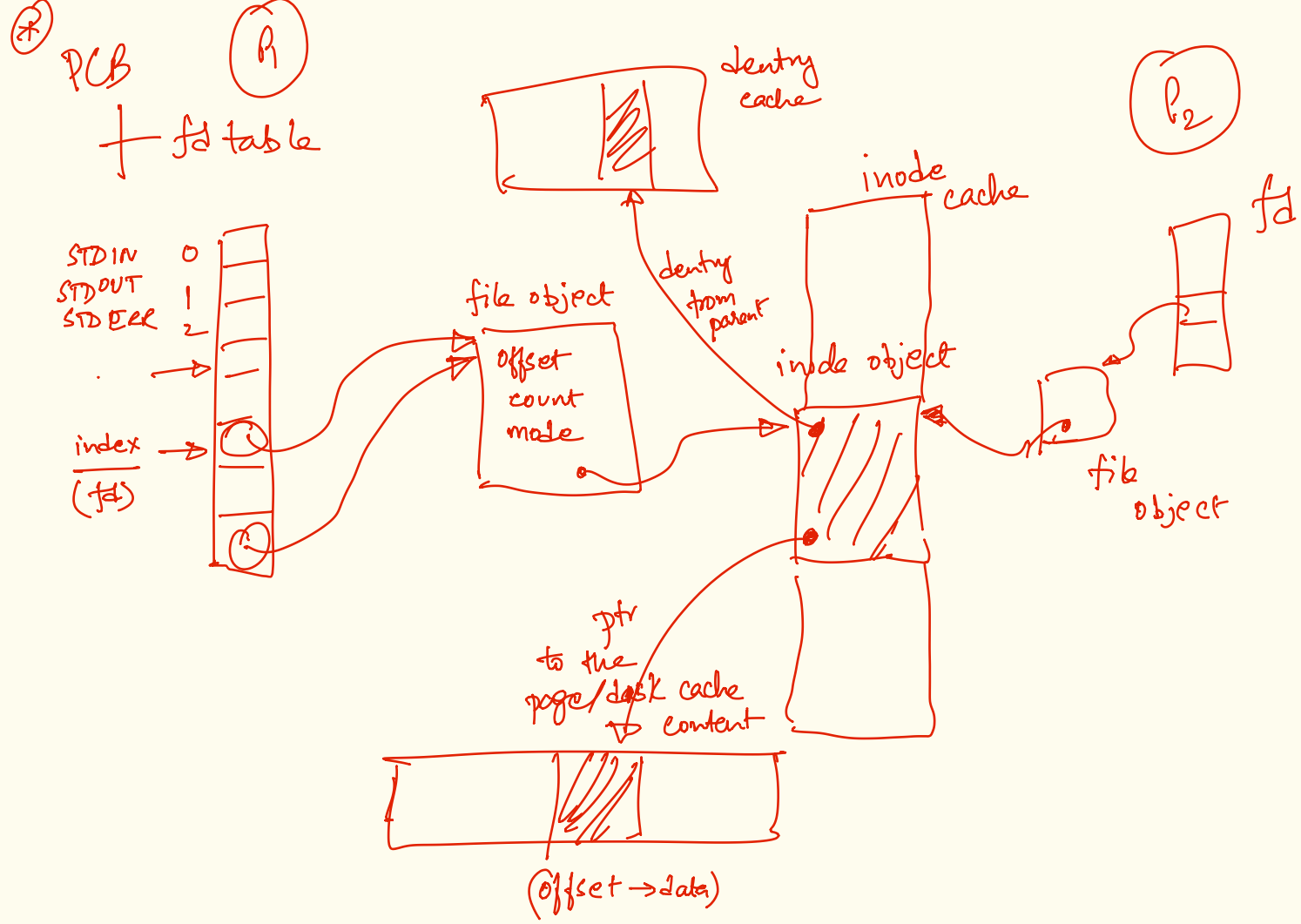
③ page cache

disk

inode, offset → data/content

FS objects (in-memory)





---

inode

- name, size, filetype
- permissions
- ref. count
- datablocks (set!)

(i) DIRECT MAPPING.

ordered list of blocks

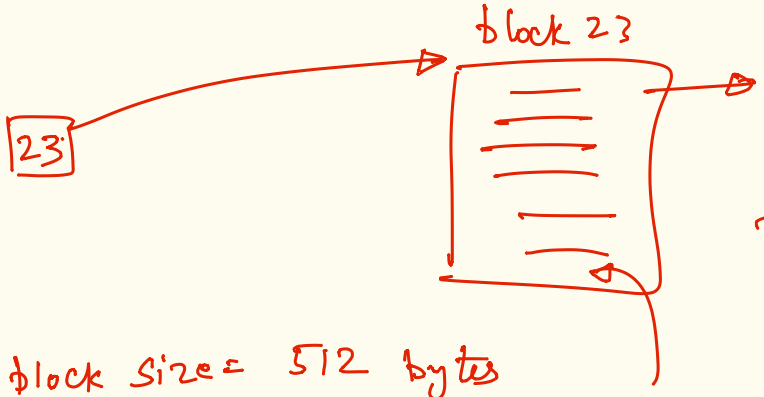
first list maps to data block 0.

datablock [ ]

(ii) INDIRECT- MAPPING

storing a pointer to a datablock whose contents are a list of block numbers.

inode

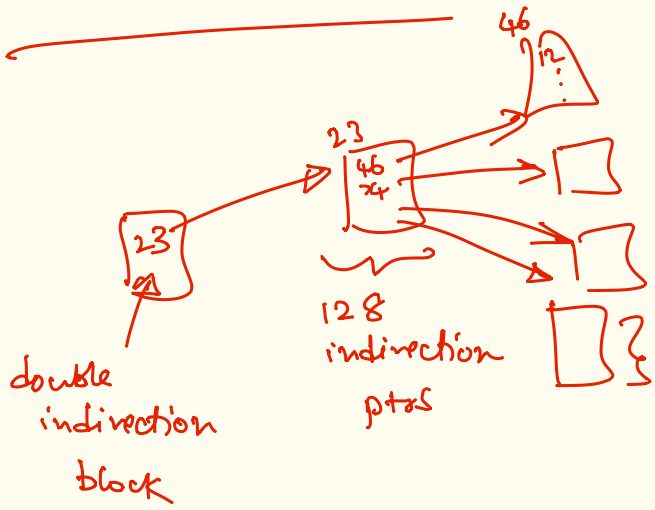


block size = 512 bytes

every entry = 4 bytes

$\frac{512}{4} = 128$ entries

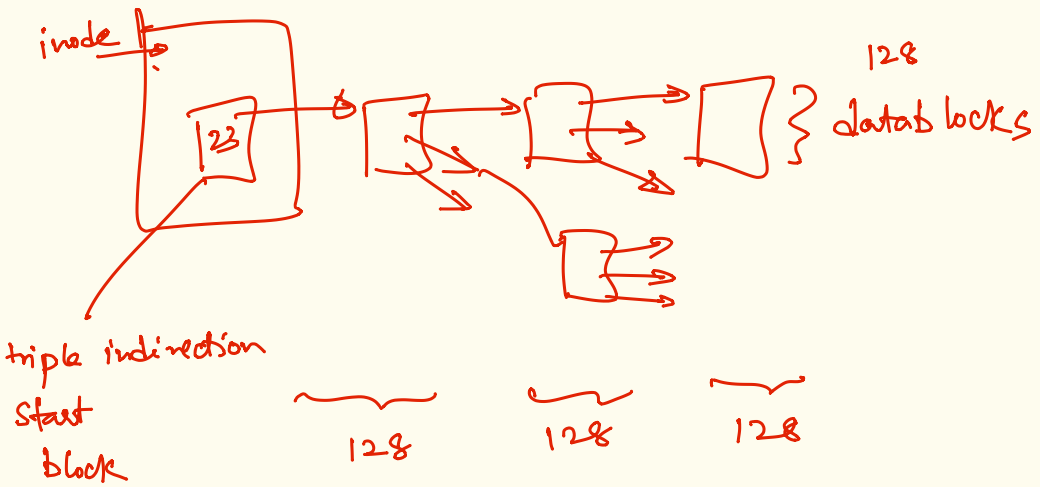
(iii) double indirect.



max file size

$= 128 \times 128$ blocks

(iv) triple indirect



max file size

$= 128 \times 128 \times 128$ blocks

XV6

12 DIRECT MAPPING BLOCKS

1 INDIRECT

+12
128
} 140 blocks

max file size