Kernel File-related Data Structures and Interfaces

- We have reviewed how files and directories are stored on disk

- We know the UNIX file system-call interface

  fd = open("file",…),

  close(fd),

  read(fd,…), write(fd,…), lseek(fd,…),…..

- What is in between?

What do we need to keep track of?

- ## File descriptors
  - Each open file has a file descriptor
  - Read/Write/lseek/…. use them to specify which file to operate on.

- ## File pointer
  - Determines where in the file the next read or write is performed

- ## Mode
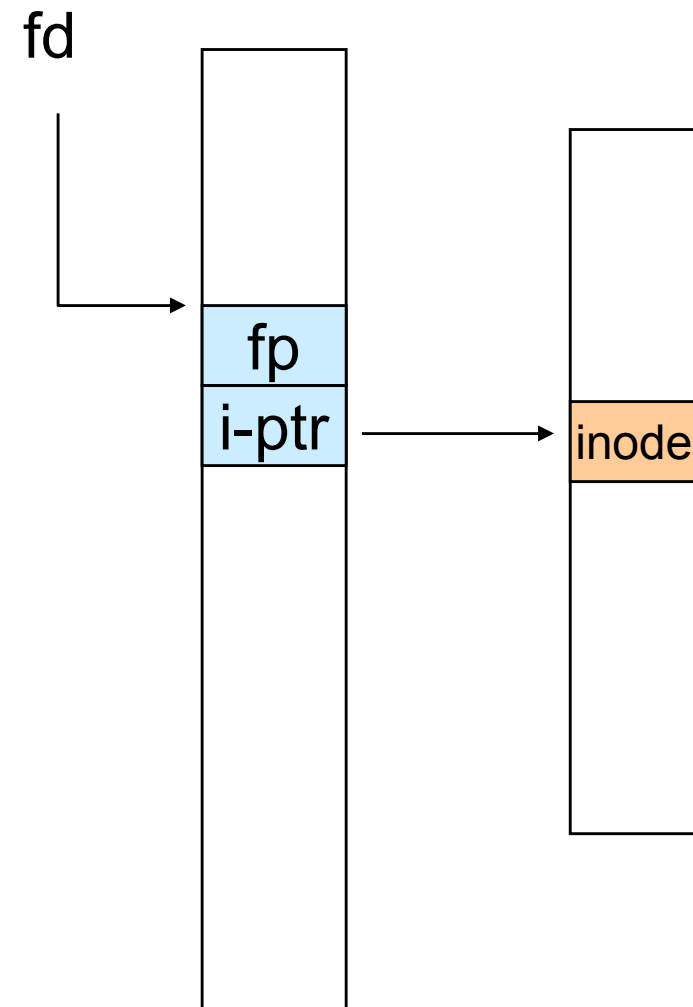 - Was the file opened read-only, etc….

An Option?

- Use inode numbers as file descriptors and add a file pointer to the inode

- Problems
  - What happens when we concurrently open the same file twice?
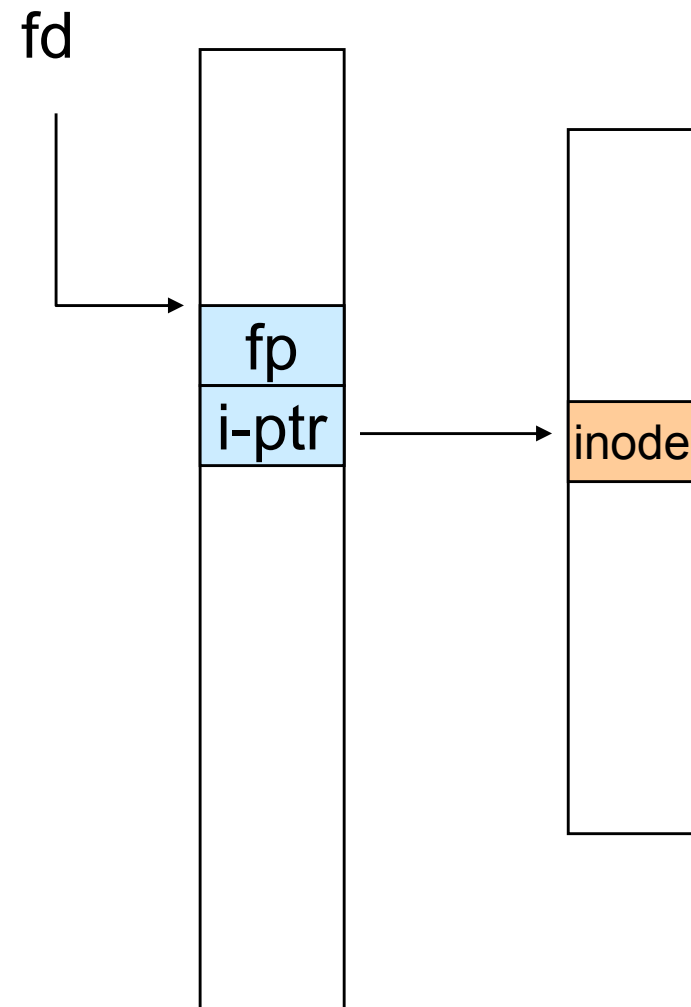   - We should get two separate file descriptors and file pointers….

An Option?

- **Single global open file array**
  - *fd* is an index into the array
  - Entries contain file pointer and pointer to an inode
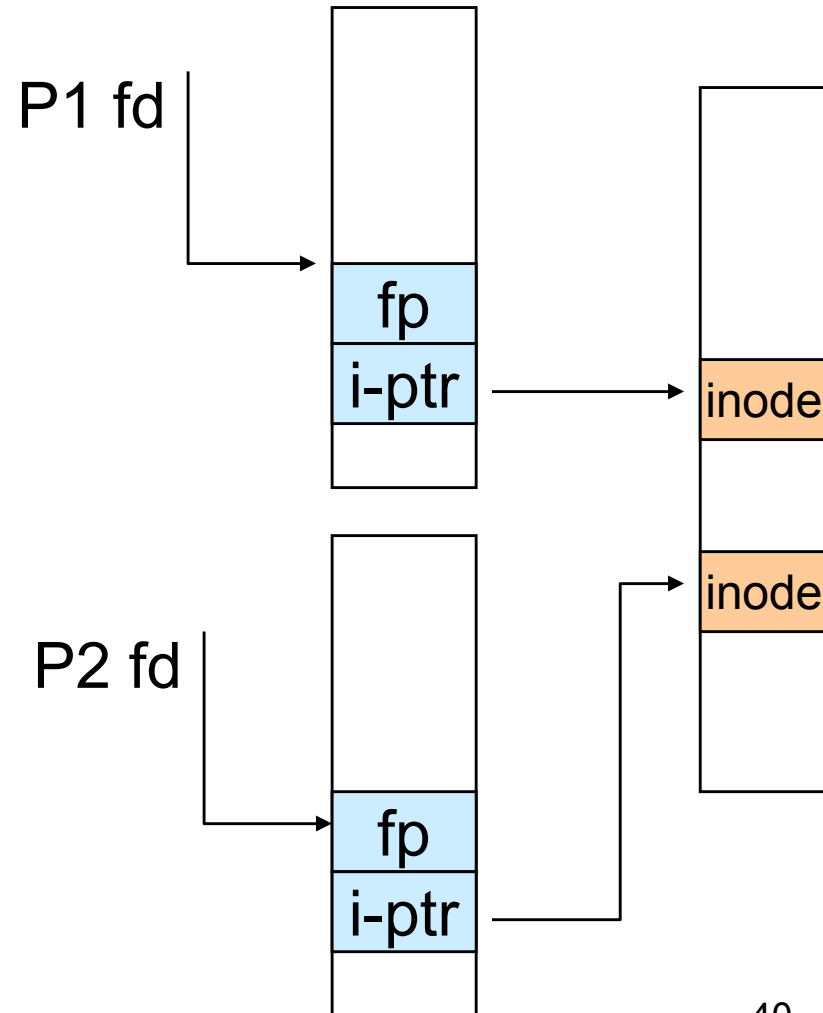
fd

fp

i-ptr

inode

Issues

- **File descriptor 1 is stdout**
  - Stdout is
    - console for some processes
    - A file for others

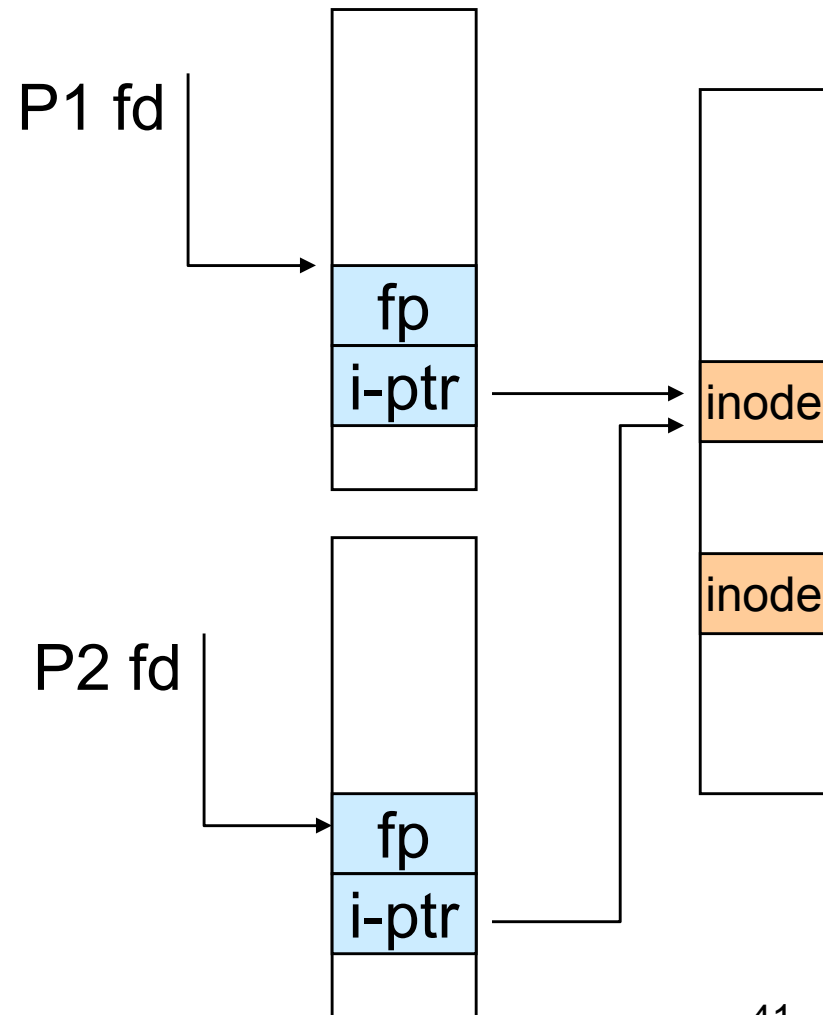- **Entry 1 needs to be different per process!**

fd

fp

i-ptr

inode

Per-process File Descriptor Array

- **Each process has its own open file array**
  - Contains fp, i-ptr etc.
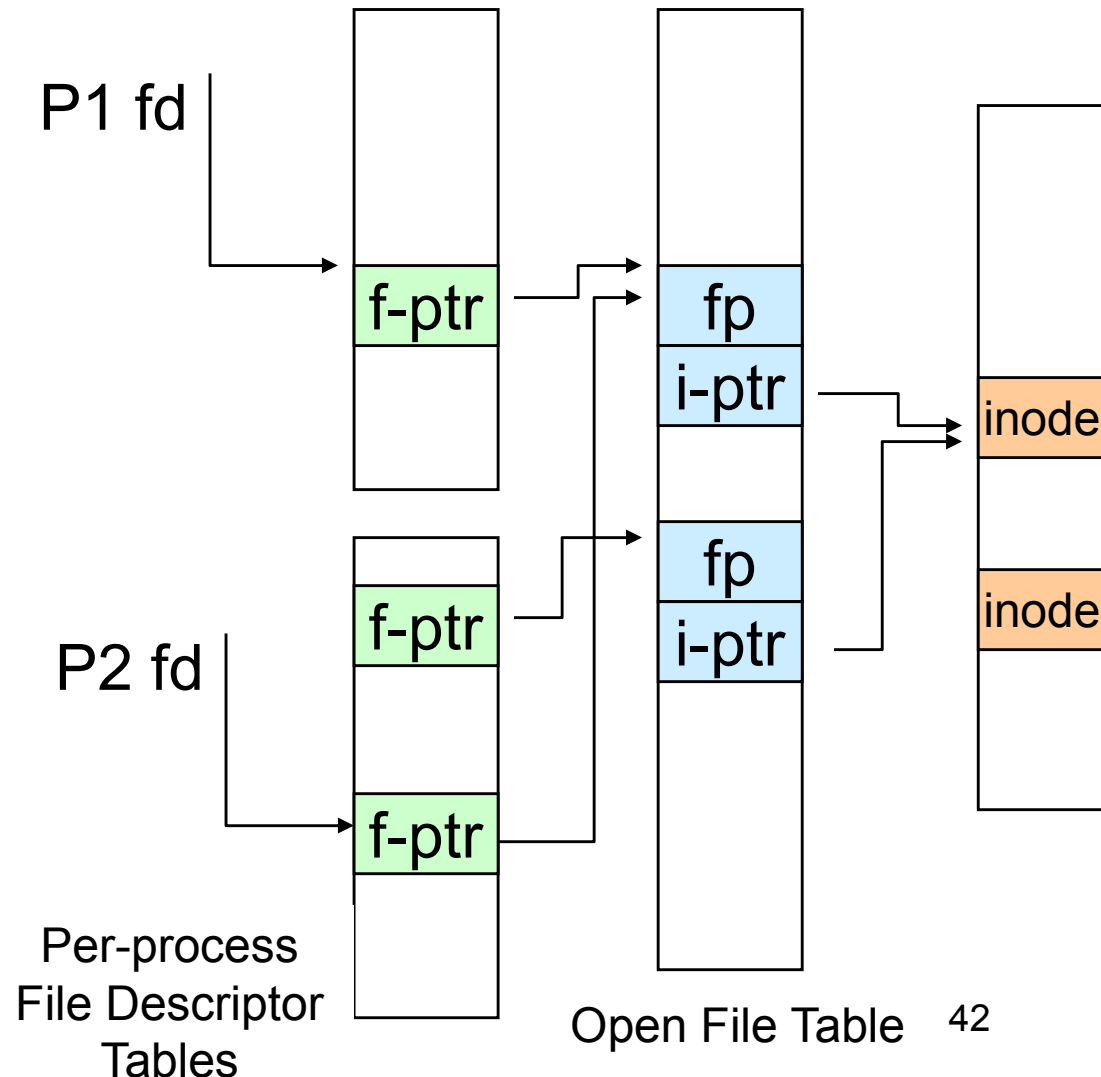 - *Fd* 1 can be any inode for each process (console, log file).

P1 fd

fp
i-ptr → inode

P2 fd

inode

fp
i-ptr

Issue

- **Fork**
  - Fork defines that the child shares the file pointer with the parent

- **Dup2**
  - Also defines the file descriptors share the file pointer

- **With per-process table, we can only have independent file pointers**
  - Even when accessing the same file

P1 fd

fp
i-ptr

inode
inode

P2 fd

fp
i-ptr

Per-Process *fd* table with global open file table

- Per-process file descriptor array
  - Contains pointers to *open file table entry*
- Open file table array
  - Contain entries with a fp and pointer to an inode.
- Provides
  - Shared file pointers if required
  - Independent file pointers if required
- Example:
  - All three *fds* refer to the same file, two share a file pointer, one has an independent file pointer

P1 fd

f-ptr

fp

i-ptr

inode

f-ptr

P2 fd

f-ptr

fp

i-ptr

inode

Per-process File Descriptor Tables

Open File Table

42

Per-Process *fd* table with global open file table

- Used by Linux and most other Unix operating systems

P1 fd

f-ptr

fp

i-ptr

inode

P2 fd

f-ptr

fp

i-ptr

inode

f-ptr

Per-process File Descriptor Tables

Open File Table

43