操作系统研讨课

蒋德钧

Fall Term 2017-2018

email: jiangdejun@ict.ac.cn
office phone: 62601007

中国科学院大学
University of Chinese Academy of Sciences

Lecture 6 File System

2017.12.20

中国科学院大学
University of Chinese Academy of Sciences

Schedule

- Project 5 due
- Project 6 assignment
  - final project, no final exam for this course

Project 5 Due

- P5 review
  - We test
    - VM paging without page fault
    - VM on-demand paging
  - Please compile your code, running the code on your board, and show the results to TA
  - Answer any questions we may ask

Project 5 Due

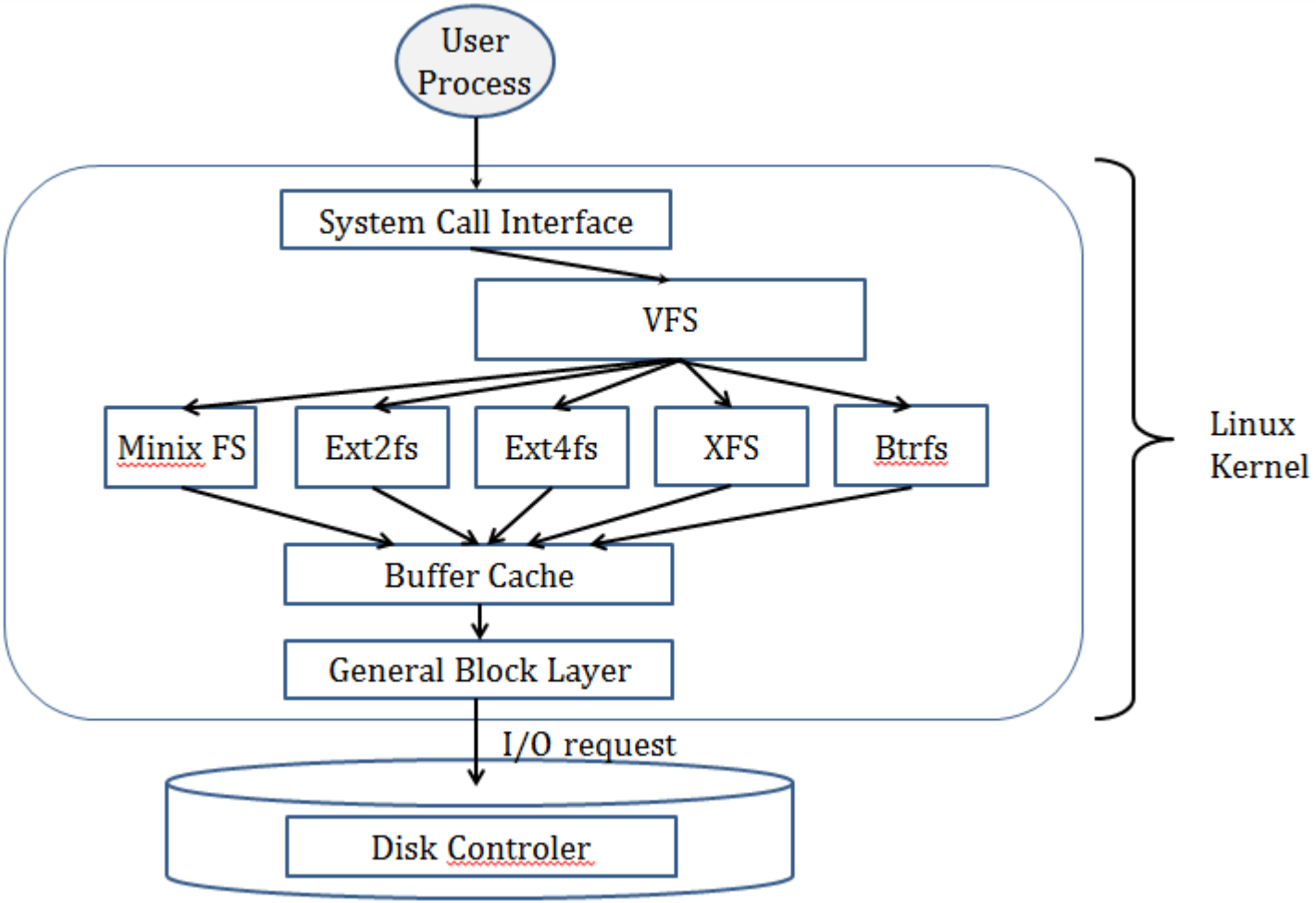
- Requirement for developing (60 points)
  - Implement virtual memory for user-space process with TLB miss but without page fault(25)
  - Implement virtual memory for user-space process with TLB miss, page fault, and page replacement (35)
  - Bonus
    - Efficient page replacement
    - Two-level page tables

Project 6 File System

- Requirement
  - Implement a FUSE based file system
    - Disk and File system metadata management
    - Hierarchical directory structure
    - Common file system operations
      - browse/create/delete directories
      - open/release/read/write/link/unlink files etc.
      - mkfs, mount, stat
    - Concurrency control should be considered for FS in-memory data structure

Project 6 File System

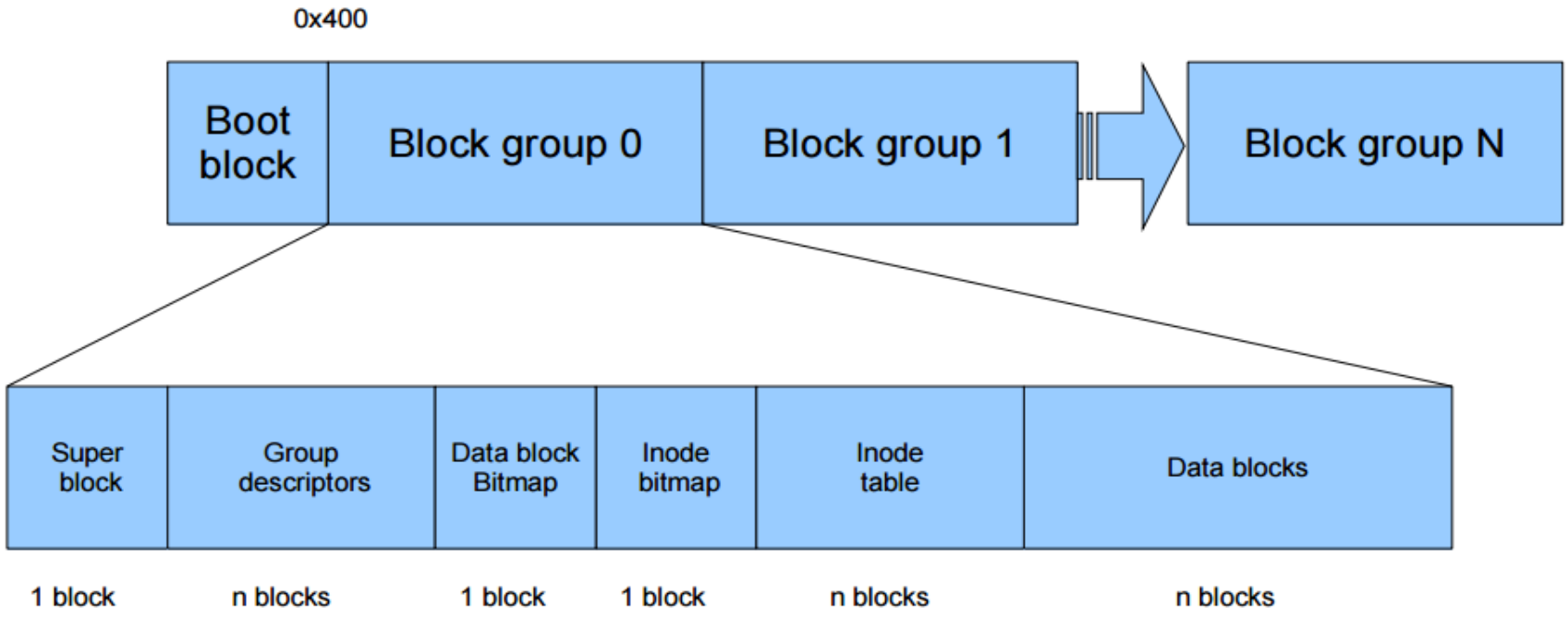
- File system



Project 6 File System

- Disk layout
  - How to manage the disk?

An example: ext2 FS disk layout



Project 6 File System

- Disk layout
  - Design your own disk layout
  - Note that, it is unnecessary to leave the space for boot block in this project

Project 6 File System

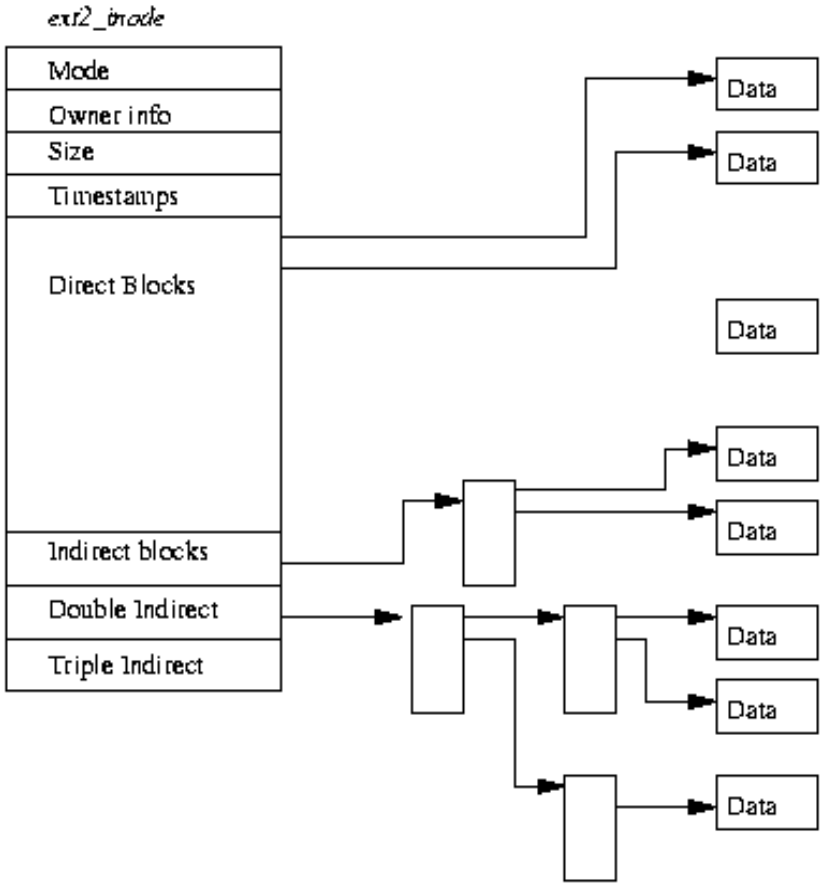
- Superblock

– Metadata to describe the structure of the file system, e.g.
  - Size
  - Num. of inodes
  - Num. of data blocks
  - Start address of inodes
  - Start address of data blocks
  - Magic number: used to judge an existing FS or not
  - In-memory lock?

- Note that, we require 2 superblocks in the disk layout.

Project 6 File System

- Inodes
  - Metadata to describe file/directory
    - Mode
    - Size
    - Other info
    - Addresses of data blocks
    - in-memory lock?



Project 6 File System

- File descriptor table
  - Keeping information of opening files, e.g.
    - file descriptor number (fd)
    - inode number
    - Availability
    - Current seek position (not required in this project)

Project 6 File System

- FS operations – init
  - Initialize data structure and resources used by FS
  - Note that when init is called, the disk is not necessarily formatted
    - What to do if disk is already formatted?
    - What to do if disk is not formatted?
    - Call mkfs or mount?
  - Note that, in FUSE, when running ./examplefs /mountted-dir, init is called

Project 6 File System

- FS operations – mkfs
  - Write FS metadata, e.g. superblock, block info, inode table etc.
  - Create root directory
  - Initialize file descriptor table
- FS operations – mount
  - Read on-disk structures of FS metadata into memory, and initialize the in-memory structure
  - Initialize root directory and file descriptor table

Project 6 File System

- FS operations – statfs
  - Return metadata info of a file system
  - Where to get the metadata info?

Project 6 File System

- File operations – open
  - Open an existing/ a non-existing file
  - Flags indicating the operation mode
    - read_only, write_only, rd_wr
  - Return a file descriptor by a successful call
- File operations – release
  - Free file descriptor
  - How to deal with the space occupied by the file?

Project 6 File System

- File operations – link
  - Create a hard link to a file
- File operations – unlink
  - Remove a link to a file
  - Delete the file if the link count is 0
- File operations – symlink
  - Create a symbolic link to a file/directory

Project 6 File System

- File operations
  - mknod: create a file
  - read: read bytes from an open file
  - write: write bytes into an open file
  - truncate: truncate a file to specified length
  - getattr: return info about a file
    - Where to get the info about a file?
  - rename

Project 6 File System

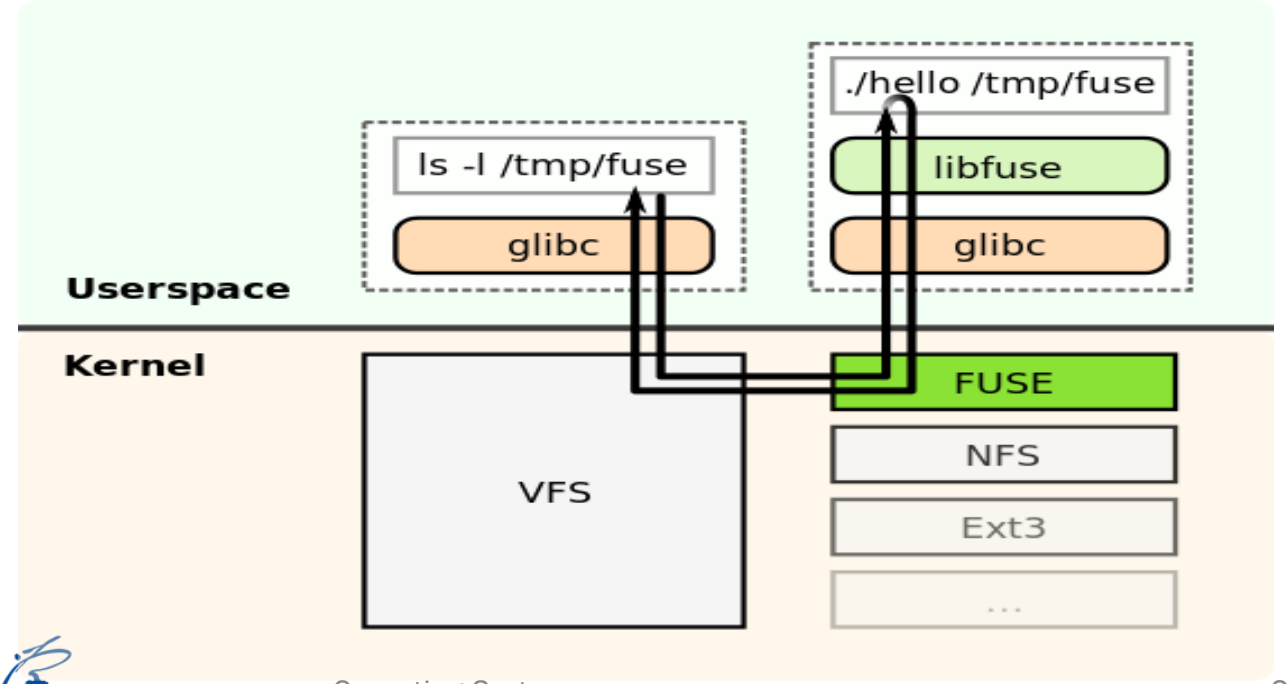
- Directories
  - A special file containing list of files and directories
  - Including file name and inode number
  - Always has two entries
    - Current directory "."
    - Parent directory ".."
  - Consider carefully about indexing the contents within dentry

Project 6 File System

- Directories operation – mkdir
  - Create a directory
    - Create an entry in parent directory
    - Create two directories "." and ".."
- Directories operation – rmdir
  - Remove a directory
- Directories operation – readdir
  - List all contents within an directory

Project 6 File System

- Tips on implementing file system
  - We implement the file system on the virtual machine, not on the board.
  - We use FUSE to implement a user-space file system



Project 6 File System

- Tips on implementing file system
  - Use the SD card as the disk for your file system
  - Pay attention to inode/superblock alignment when writing to disk
  - Pay attention to dealing with manipulating existing files/directories
  - Please refer the provided example and task documents carefully

Project 6 File System

- Step by step
  - Step 1: design and implement various FS metadata, e.g. superblock, inode, file descriptor, block allocation. Make sure your file system can be correctly mounted at this step.
  - Step2: design and implement various FS operations

Project 6 File System

- Requirements for design review (40 points)
  - What is the disk layout in your design?
  - How do you design the structures of FS metadata, e.g. superblock, inode, dentry
  - How do you track all blocks and do block allocation?
  - What to do when initializing a file system?
  - How do you handle path lookup?

Project 6 File System

- Requirements of developing (60 points)
  - Implement init, mkfs, mount, statfs (20)
  - Implement open/release/mknod/read/write/link/unlink/symlink/truncate/getattr/rename (25)
  - Implement mkdir/rmdir/readdir (15)

Project 6 File System

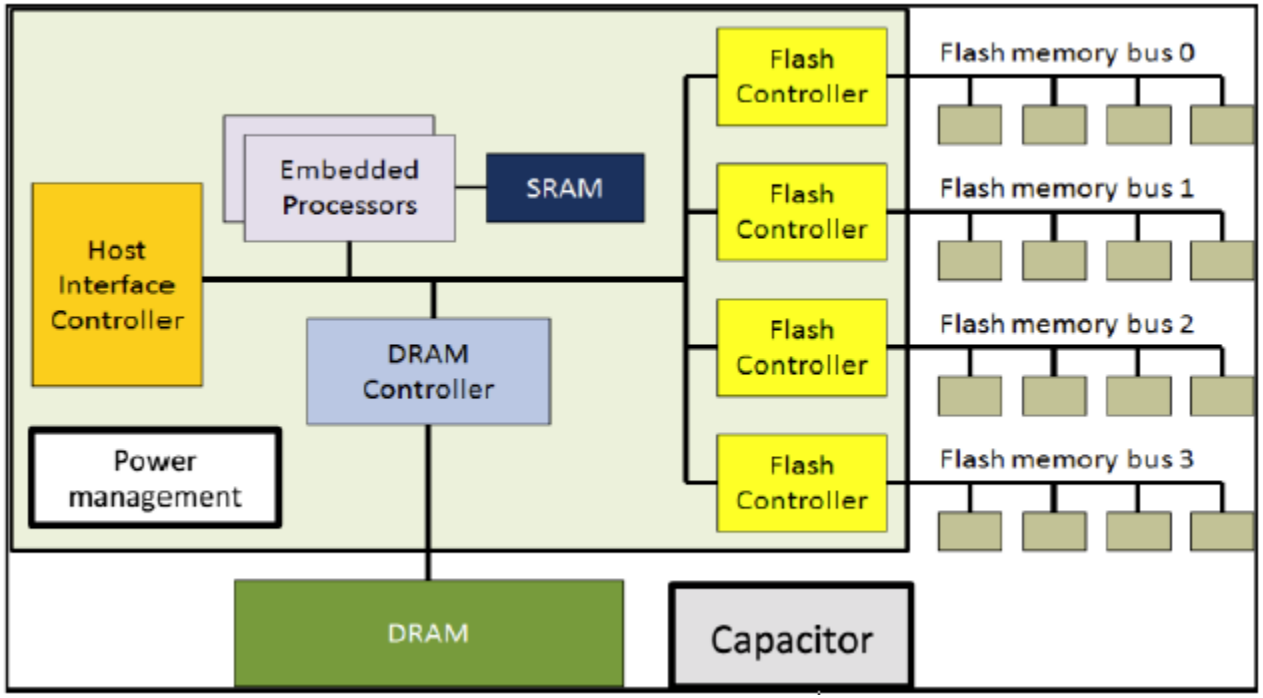
- Bonus 1 (1 points)
  - Implement a few more FS operations, including
    - utime
    - chmod
    - destroy
    - readlink

Project 6 File System

- Bonus 2 (3 points)
  - Implement SSD friendly file system
    - Read the provided paper carefully and try to understand the implication of log-structure on SSD
    - You are encouraged to search further reading materials to help you understand SSD specific design for file system
    - Assuming direct blocks

Project 6 File System

- Bonus 2 (3 points)
  - Implement SSD friendly file system
    - SSD internal architecture



Project 6 File System

- Bonus 2 (3 points)
  - Implement SSD friendly file system
    - SSD specific features
      - Write after erase
    - File system design
      - In-place update vs. out-of-place update
      - Garbage collection

Project 6 File System

- P6 schedule
  - P6 design review: 27th Dec.
  - P6 due: 10th Jan. 2018
  - 3rd Jan. 2018: No class