Newly Released Requirements

File system is the core storage management sub-system of OS. All files (executable, not-executable …) should be grouped and organized by File System (FS).

Even the whole kernel source code file is loaded supervised by FS. In this newly released requirements sheet, you should create a file system (of course, you can do teamwork or solo work).

Here are some key points you may need to guide you to design a simple FS.

Overview

Design and implement a simplified inode-based file system inspired by XV6, specifically tailored for the QEMU RISC-V platform. The system should provide basic file operations through a clean, portable interface.

DISK LAYOUT DESIGN

[Boot Block]

[Superblock]

[Inode Blocks]

[Bitmap Block]

[Data Blocks]

ESSENTIAL DATA STRUCTURES

FILE STRUCT DEF:

```
struct file {
  enum { FD_NONE, FD_PIPE, FD_INODE } type;
  int ref; // reference count
  char readable;
  char writable;
  struct pipe *pipe;
  struct inode *ip;
  uint off;
};
```

INODE STRUCT DEF:

```
struct inode {
  uint dev;           // Device number
  uint inum;          // Inode number
  int ref;            // Reference count
  struct sleeplock lock; // protects everything below here
  int valid;          // inode has been read from disk?

  short type;         // copy of disk inode
```

```
    short major;
    short minor;
    short nlink;
    uint size;
    uint addrs[NDIRECT+1];
};
```

REFERENCE:

https://github.com/mit-pdos/xv6-public