操作系统 lab5 文件系统

邱梓豪

141130077

很惭愧，临近期末，事情比较多，我在 os 的网站上看到了助教给的框架代码，于是我研究了一下这份代码，按照我的想法进行了一些修改。

1.文件系统布局

磁盘的划分方式为最开始的一个扇区为 bootloader，之后 200 个扇区为 Kernel，之后所有的磁盘空间划分为 Filesystem。

```
//#define SECTOR_SIZE 512 // i.e., 512 B, defined in i
#define SECTORS_PER_BLOCK 2
#define POINTER_NUM 12
#define NAME_LENGTH 64

/* Dependent Variables & Constants */
#define BLOCK_SIZE (SECTOR_SIZE * SECTORS_PER_BLOCK)
```

代码中将两个 Sector 视为一个 Block，每个 Block 的大小即为 1024 字节。

整个 Filesystem 被划分为若干个 BlockGroup。每个 Block Group 在头部为 MetaBlock，记录文件系统的元数据，依次包含 SuperBlock、GroupDescriptorTable、InodeBitmap、BlockBitmap；MetaBlock 之后则为 InodeTable 以及真正存储通常文件以及目录文件的 DataBlock。

SuperBlock 中记录了整个文件系统的信息，例如总扇区数，

Inode 总数，Block 总数，可用 Inode 数，可用 Block 数等。Group Descriptor Table 中则记录了各个 Block Group 的信息，例如可用 Inode 数，可用 Block 数。InodeBitmap 以及 Block Bitmap 则分别用于对该 BlockGroup 中 InodeTable 以及 DataBlock 的使用情况进行记录

2.文件系统格式化

```
int main(int argc, char *argv[]) {
    char driver[NAME_LENGTH];
    char srcFilePath[NAME_LENGTH];
    char destFilePath[NAME_LENGTH];

    stringCpy("fs.bin", driver, NAME_LENGTH - 1);
    format(driver, SECTOR_NUM, SECTORS_PER_BLOCK);

    stringCpy("/boot", destFilePath, NAME_LENGTH - 1);
    mkdir(driver, destFilePath);
    stringCpy(argv[1], srcFilePath, NAME_LENGTH - 1);
    stringCpy("/boot/initrd", destFilePath, NAME_LENGTH - 1);
//  touch(driver, destFilePath);
    cp(driver, srcFilePath, destFilePath);
//  touch(driver, destFilePath);

    stringCpy("/dev", destFilePath, NAME_LENGTH - 1);
    mkdir(driver, destFilePath);
    stringCpy("/dev/stdin", destFilePath, NAME_LENGTH - 1);
    touch(driver, destFilePath);
    stringCpy("/dev/stdout", destFilePath, NAME_LENGTH - 1);
    touch(driver, destFilePath);

    stringCpy("/usr", destFilePath, NAME_LENGTH - 1);
    mkdir(driver, destFilePath);

    return 0;
}
```

genFS 的主要任务就是对文件系统进行格式化以及创建初始的文件和目录。其中格式化的任务由 format 函数实现。

```
int format (const char *driver, int sectorNum, int sectorsPerBlock) {
    int i = 0;
    int ret = 0;
    FILE *file = NULL;
    uint8_t byte[SECTOR_SIZE];
    SuperBlock superBlock;
    GroupDesc groupDesc[MAX_GROUP_NUM];
    if (driver == NULL) {
        printf("driver == NULL.\n");
        return -1;
    }
    file = fopen(driver, "w+"); // create / truncating / writing / reading
    if (file == NULL) {
        printf("Failed to open driver.\n");
        return -1;
    }
    for (i = 0; i < SECTOR_SIZE; i++) {
        byte[i] = 0;
    }
    for (i = 0; i < sectorNum; i++) {
        fwrite((void*)byte, sizeof(uint8_t), SECTOR_SIZE, file);
    }
    ret = initGroupHeader(file, sectorNum, sectorsPerBlock, &superBlock, groupDesc);
    if (ret == -1) {
        printf("Failed to format: No enough sectors.\n");
        fclose(file);
        return;
    }
    ret = initRootDir(file, &superBlock, groupDesc);
    if (ret == -1) {
```

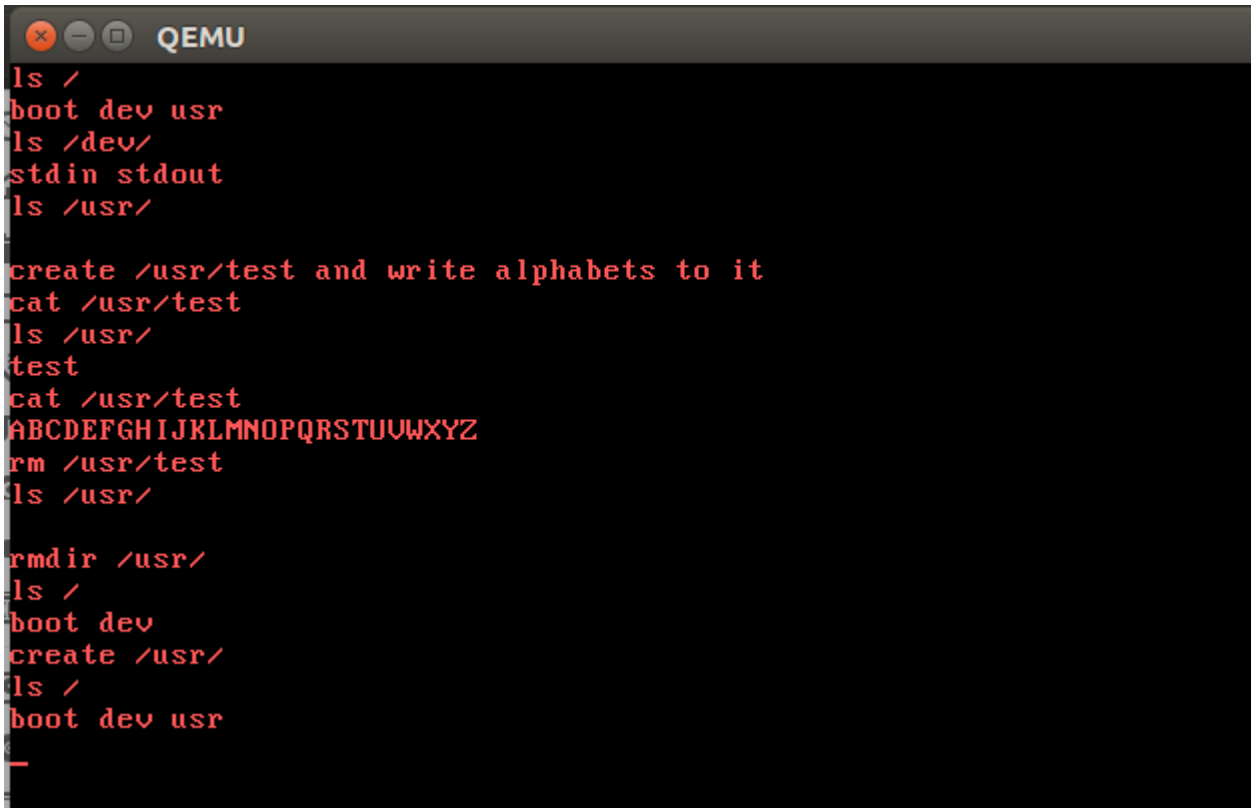
在该程序中，磁盘文件的句柄为“file”，通过向这个文件写入格式化信息来构建磁盘文件。主要写入的信息是每个 BlockGroup 头部的有关信息，及初始化根目录。

3.文件系统相关接口

对文件系统的操作接口有如下几个：open，write，read，lseek，close，remove。这些功能的实现，都是利用了 fs.h 中提供的若干函数。这些函数封装了对 filesystem 中 inode 和 block 的一些操作，比如分配，释放，读写。利用这些文件系统给出的接口，就可以实现上层的系统调用了。

```
#ifndef __FS_H__
#define __FS_H__

#include "fs/ext.h"

int readGroupHeader (SuperBlock *superBlock, GroupDesc *groupDesc);

int allocInode (SuperBlock *superBlock, GroupDesc *groupDesc,
        Inode *fatherInode, int fatherInodeOffset,
        Inode *destInode, int *destInodeOffset, const char *destFilename, int destFilety
int freeInode (SuperBlock *superBlock, GroupDesc *groupDesc,
        Inode *fatherInode, int fatherInodeOffset,
        Inode *destInode, int *destInodeOffset, const char *destFilename, int destFilety

int readInode (SuperBlock *superBlock, GroupDesc *groupDesc,
        Inode *destInode, int *inodeOffset, const char *destFilePath);

int allocBlock (SuperBlock *superBlock, GroupDesc *groupDesc, Inode *inode, int inodeOf

int readBlock (SuperBlock *superBlock, Inode *inode, int blockIndex, uint8_t *buffer);

int writeBlock (SuperBlock *superBlock, Inode *inode, int blockIndex, uint8_t *buffer);

int getDirEntry (SuperBlock *superBlock, Inode *inode, int dirIndex, DirEntry *destDirEn
```

最后的运行效果如下：



注：我 lab1 到 lab4 都已按时完成，如果助教发现有问题，麻烦联系我，我的 qq 是 2957606241