Operating Systems - Homework 3 - Report

Assumptions

First of all given development virtual machines did not work because of various reasons. So, I decided to use my own development system. I use 64-bit Ubuntu 20.04.4 LTS as operating system.

Also I use CMake instead of Makefile, I wanted to use some new stuff and there is no given restriction to use Makefile.

Introduction

I used C++ for coding. I belive the object oriented approach is more appropriate for this homework. FileSystem class represent the file system and related operations. Also I used CMake. It creates two executable with names “makeFileSystem” and “fileSystemOper”. Here is CMakeLists.txt:

```
cmake_minimum_required(VERSION 3.22)
project(file_system)

set(CMAKE_CXX_STANDARD 14)

add_executable(makeFileSystem MakeFileSystem.cpp Helper.cpp Helper.h FileSystem.cpp FileSystem.h)
add_executable(fileSystemOper FileSystemOperation.cpp Helper.cpp Helper.h FileSystem.cpp FileSystem.h)
```

Super Block

My super block contains following data.

```
struct SuperBlock {
    uint32_t diskSize;
    uint32_t blockSize;
    uint32_t blockCount;
    uint32_t freeBlockCount;
    uint32_t firstBlock;
    uint32_t inodeSize;
    uint32_t inodeCount;
    uint32_t freeInodeCount;
    uint32_t firstInode;
    uint32_t inodeBitmapBlockCount;
    uint32_t blockBitmapBlockCount;
    uint32_t inodeBlockCount;
    uint32_t dataBlockCount;
    uint32_t inodeBitmapStartBlock;
    uint32_t blockBitmapStartBlock;
    uint32_t inodeStartBlock;
    uint32_t dataStartBlock;
    uint32_t rootInode;
    uint32_t directoryCount;
```

```
    uint32_t fileCount;
};
```

When makeFileSystem program run, it creates a file that represents a file system. It gets block size as parameter. My default file system size is 16MB. According to these parameters, I calculate these required data in “create” method of “FileSystem” class. The super block is written to first block of the system. When a file system loaded, its super block is loaded to memory, then operations use necessary information from this struct.

Free Space Management

I manage free inodes and blocks with bitmaps. They are stored in file system blocks, right after the super block. When a file system loaded, these blocks are loaded to memory. They are converted to bool array. If related index is false, it means the inode/block is empty, else used. When there is need to empty inode or block, a loop traverse over these arrays and select first empty index.

Directory Structure and Directory Entries

```
struct Inode {
    uint8_t mode;
    uint8_t blockCount;
    uint16_t blocks[7];
    uint32_t size;
    uint32_t creationTime;
    uint32_t lastAccessTime;
    uint32_t lastModificationTime;
};

struct DirectoryEntry {
    uint16_t inodeNumber;
    unsigned char name[14];
};
```

File System Operations

dir

```
void listDirectory(char *path);
```

mkdir

```
void makeDirectory(char *path);
```

rmdir

```
void removeDirectory(char *path);
```

dumpe2fs

```
  void dumpe2fs() const;
```

write

```
 void writeFile(char *path, char *file);
```

read

```
void readFile(char *path, char *file);
```

del

```
void removeFile(char *path);
```