文件管理系统设计

1. **项目需求**

在内存中开辟一个空间作为文件存储器，在其上实现一个简单的文件系统。退出这个文件系统时，需要该文件系统的内容保存到磁盘上，以便下次可以将其恢复到内存中来。并实现操作：对文件存储器进行格式化（即按照文件系统的结构对内存中的虚拟磁盘空间进行布局，并在其上创建根目录以及用于管理文件存储空间等的数据结构）；创建子目录；删除子目录；显示目录；更改当前目录；创建文件；打开文件；关闭文件；写文件；读文件；删除文件。

1. **设计思路**

通过创建一个二进制文件模拟创建磁盘。该文件系统以盘块为基本分配单位，每个盘块（BLOCK\_SIZE）大小为512字节。最开始的第0块为引导块。文件存储空间管理采取两张FAT表，FAT2对FAT1的备份，每个FAT表项长度为8位，即最大的磁盘容量为256\*BLOCK\_SIZE字节。每张FAT表共有256个表项，因此两张FAT表刚好是512字节，也即一个BLOCK\_SIZE。文件目录采用多级目录结构，接下来的第2块就是根目录，在格式化的时候，根目录中已经被增入两个目录项：”.”和”..”，分别表示当前目录和上层目录。空闲空间管理采用位图。

1. **详细设计**

**1.**数据结构

struct FCB

{

char filename[8];//文件名

char exname[3];//文件扩展名

unsigned char attribute;//值为0时表示为目录文件，值为1时表示为数据文件

struct tm date\_time;//处理创建日期和时间

unsigned short first;//文件起始盘块号

unsigned long length;//文件长度(字节数)

char free;//若值为0，表示目录项为空，值为1，表示已分配

};

struct FAT

{

unsigned short id;

};

typedef struct USEROPEN //用户打开文件表 只用于文件

{

char filename[8];//文件名

char exname[3];//文件扩展名

unsigned char attribute;//值为0时表示为文件目录，值为1时表示数据文件

unsigned short time;//文件创建时间

unsigned short data;//文件创建日期

unsigned short first;//文件起始盘块号

unsigned long length;//文件长度

char free;//若值为0，表示目录项为空，值为1，表示已分配

int dirno;// 相应打开文件的目录项在父目录文件中的盘块号

int diroff;//相应打开文件的目录项在父目录文件的dirno盘块中的目录项序号

char dir[80];//相应打开文件所在的目录名，这样方便快速检查出指定文件是否已经打int count;//读写指针在文件中的位置

char fcbstate;//如果修改了文件的FCB的内容置为1，否则为0

char topenfile;//若值为0，表示该用户打开表项为空，否则表示已被某打开文件占据

}useropen;

struct BLOCK0

{

char information[200];//存储磁盘块大小，磁盘块数量，最多打开文件数等信息

unsigned short root;//根目录文件的起始盘块号

char \*startblock;//虚拟磁盘上数据区开始位置

};

struct FAT \*fat; //FAT表的起始位置

struct FCB \*root; //根目录

struct FCB \*cur\_dir; //当前目录

struct FCB \*father\_dir; //父目录地址

char \*myvhard; //指向虚拟磁盘的起始地址

useropen openfilelist[MAXOPENFILE];//用户打开文件表数组

useropen \*ptrcurdir; //指向用户打开文件表中的当前目录所在打开文件表项的位置

char currentdir[80]; //记录当前目录的目录名

char \*startp; //记录虚拟磁盘上数据区开始位置

2.函数

void startsys();

进入并初始化建立文件系统，以供用户使用。

void format();

对虚拟磁盘进行格式化：将虚拟磁盘第一块作为引导路块，在之后写入文件系统的描述信息；在引导块后建立两张完全一样的FAT表，用于记录文件所占据的磁盘块及管理虚拟磁盘块的分配；在第二张FAT后创建根目录文件root，将数据区的第一块分配给根目录，在该磁盘上创建两个特殊的目录项：”.”和”..”。

void cd(char \*dirname);

改变当前目录到指定的名为dirname的目录。

void mkdir(char \*dirname);

在当前目录下创建名为dirname的子目录。为新建的子目录文件分配一个空闲打开文件表项和一个盘块，在当前目录中为新建目录文件寻找一个空的目录项或为其追加一个新的目录项。

void rmdir(char \*dirname);

在当前目录下删除名为dirname的子目录。回收该目录文件所占的磁盘块，并修改FAT，当前目录文件的用户打开表项中的长度信息。

void ls(void);

显示当前目录的内容。

int create(char \*filename);

创建名为filename的新文件，为新文件分配一个盘块，并在父目录为新文件寻找一个空闲的目录项或为其追加一个新的目录项，将FCB的内容写到新的目录项中。

void rm(char \*filename);

删除名为filename的文件，回收该文件所占据的额磁盘块，修改FAT，并修改该父目录文件的用户打开文件表项中的长度信息。

int open(char \*filename);

打开当前目录下名为filename的文件。

void close(int fd);

关闭之前由open（）打开的文件，以覆盖写的方式将欲关闭文件的FCB写入父目录文件的相应盘块中，回收该文件占据的用户打开文件表表项。

int write(int fd);

将用户通过键盘输入的内容写到fd所指定的文件中，在写操作时，先将数据写在缓冲区中，缓冲区的大小与磁盘块的大小相同，然后将缓冲区中的数据一次性写到磁盘块中。

int read(int fd,int len);

读出指定文件中从读写指针开始的长度为len的内容到用户空间，读出时先将一个磁盘块中的内容都出到缓冲区，然后再传送到用户区。