操作系统课程设计报告

学院：计算机学院

班级：15052316

姓名：鞠畅

学号:15051407

# 实验5 简单文件系统的实现

## 1 实验要求

(1) 在内存中开辟一个虚拟磁盘空间作为文件存储分区，在其上实现一个简单的基于多级目录的单用户单任务系统中的文件系统。在退出该文件系统的使用时，应将该虚拟文件系统以一个文件的方式保存到磁盘上，以便下次可以再将它恢复到内存的虚拟磁盘空间中。

(2) 文件存储空间的分配可采用显式链接分配或其他的办法。

(3) 空闲磁盘空间的管理可选择位示图或其他的办法。如果采用位示图来管理文件存储空间，并采用显式链接分配方式，那么可以将位示图合并到 FAT 中。

(4) 文件目录结构采用多级目录结构。为了简单起见，可以不使用索引结点，其中的每个目录项应包含文件名、物理地址、长度等信息，还可以通过目录项实现对文件的读和写的保护。

(5) 要求提供以下操作命令：

my\_format：对文件存储器进行格式化，即按照文件系统的结构对虚拟磁盘空

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。

my\_mkdir：用于创建子目录。

my\_rmdir：用于删除子目录。

my\_ls：用于显示目录中的内容。

my\_cd：用于更改当前目录。

my\_create：用于创建文件。

my\_open：用于打开文件。

my\_close：用于关闭文件。

my\_write：用于写文件。

my\_read：用于读文件。

my\_rm：用于删除文件。

my\_exitsys：用于退出文件系统。

## 2 实验步骤

1.声明常量和结构体FCB、FAT、USEROPEN、BLOCK0

2.编写主函数。首先调用startsys()函数，初始化文件系统。然后循环请求输入命令，直至输入exit。对输入的字符串进行切分并匹配命令，调用相关函数。

3.按实验指导书的步骤完成文件系统函数的编写

## 3 核心思路及代码

下面详细介绍下几个比较重要的函数的思路

（1）findFree()

**功能：**在FAT表中找空的表项，找到则返回表内偏移量，找不到则返回END（65535，FAT16的最大值）

**实现步骤：**

1.定位到FAT表的起始位置

2.从第6个位置开始找（前五个FAT表项分别是引导块（1块）、FAT1（2块）、FAT2（2块）），如果所指向位置的id属性是FREE，表示还未分配，返回此时的偏移量，否则继续找

3.所有的位置都遍历之后返回END

unsigned short findFree()

{

unsigned short i;

fat \*fat1,\*fatptr;

fat1=(fat \*)(myvhard+BLOCKSIZE);

for(i=6; i<END; i++)

{

fatptr=fat1+i;

if(fatptr->id == FREE)

{

return i;

}

}

printf("Error,Can't find free block!\n");

return END;

}

（2）int findFreeO()

**功能**：寻找空闲文件表项

**实现思路**：遍历所有文件打开表项，找到free属性为0的表项并返回偏移量，否则报错并返回-1。

int findFreeO()

{

int i;

for(i=0;i<MAX\_OPEN\_FILE;i++)

{

if(openfilelist[i].free==0)

{

return i;

}

}

printf("Error,open too many files!\n");

return -1;

}

（3）my\_mkdir(char \*dirname)

**实现步骤**：

1.调用do\_read()将当前目录的FCB读入到内存

2.检查是否存在同名的目录，有则报错并退出，没有则执行下一步

3. 在文件打开表中寻找空闲的表项，在FAT表中寻找空闲的盘块

4. 在当前目录FCB的末尾追加一个新的目录项，修改当前目录的length属性，将USEROPEN中的fcbstate置1

5.创建新建目录的FCB，写入当前目录中的目录项

6.为新建目录文件创建“.”和“..”两个目录项并写入3中分配的磁盘块中

（4）my\_rmdir(char\*dirname)

**实现步骤：**

1. 调用do\_read()将当前目录的FCB读入到内存

2. 判断目录是否为根目录，是则报错返回

3. 判断要删除的目录是否存在，不存在则报错返回

3. 检查待删除的文件除了“.”和“..”之外有没有其他子目录或文件，有则报错返回

4. 从FAT表中该目录的起始盘块开始，指针移向盘块id属性所指的位置，并释放当前盘块，直至盘块的id是END

5.更新父目录的FCB

**代码**：

void my\_rmdir(char \*dirname)

{

int rbn,fd;

char text[MAX\_TXT\_SIZE];

fcb \*fcbptr,\*fcbtmp,\*fcbtmp2;

unsigned short bknum;

int i,j;

fat \*fat1,\*fatptr;

/\*判断是否是根目录\*/

if(strcmp(dirname,".")==0 || strcmp(dirname,"..")==0)

{

printf("Error,can't remove this directory.\n");

return;

}

fat1=(fat \*)(myvhard+BLOCKSIZE);

openfilelist[curfd].count=0;

rbn=do\_read(curfd,openfilelist[curfd].length,text);

fcbptr=(fcb \*)text;

/\*判断当前文件是否存在\*/

for(i=0;i<rbn/sizeof(fcb);i++)

{

if(strcmp(dirname,fcbptr->filename)==0)

{

break;

}

fcbptr++;

}

if(i >= rbn/sizeof(fcb))

{

printf("Error,the directory is not exist.\n");

return -1;

}

/\*判断目录是否为空\*/

bknum=fcbptr->first;

fcbtmp2=fcbtmp=(fcb \*)(myvhard+bknum\*BLOCKSIZE);

for(j=0;j<fcbtmp->length/sizeof(fcb);j++)

{

if(strcmp(fcbtmp2->filename,".") && strcmp(fcbtmp2->filename,"..") && fcbtmp2->filename[0]!='\0')

{

printf("Error,the directory is not empty.\n");

return -1;

}

fcbtmp2++;

}

/\*将FAT表中的块号置为FREE\*/

while(bknum!=END)

{

fatptr=fat1+bknum;

bknum=fatptr->id;

fatptr->id=FREE;

}

/\*更新父目录中的文件\*/

strcpy(fcbptr->filename,"");

strcpy(fcbptr->exname,"");

fcbptr->first=END;

openfilelist[curfd].count=0;

do\_write(curfd,text,openfilelist[curfd].length,2);

}

## 4 心得体会

虽然我们的文件系统非常的简单，但是光是这样就已经非常复杂了，尤其是里面每个函数之间的相互调用，虚拟硬盘上磁盘块的读取和查找以及每一步操作后需要修改的各种属性信息。由于一开始对文件系统的结构不太熟悉，所以先在网上找了前辈的代码，仔细阅梳理清楚，之后针对代码中存在的一些小问题以及指导书中提到却没有实现的功能进行了编写。这次实验将课上的理论转换到了函数层面。比如找到块内偏移量再写出磁盘块号再转换成内存中的位置，课上学会了流程和方法，在实验过程中通过编程的方式让自己理解的更加透彻深刻。