文件系统设计

1．目的和要求

本实验的目的是通过一个简单多用户文件系统的设计，加深理解文件系统的内部功能和内部实现。

2．实验内容

为 DOS 系统设计一个简单的二级文件系统。要求做到以下几点：

①可以实现下列几条命令

LOGIN　用户登陆

DIR　列文件目录

CREATE　创建文件

DELETE　删除文件

OPEN　打开文件

CLOSE　关闭文件

READ　读文件

WRITE　写文件

②列目录时要列出文件名，物理地址，保护码和文件长度。

③源文件可以进行读写保护。

3．实验环境

操作系统：WINDOWS XP

编译软件：WINTC

4．程序设计

（1）实验提示

①本文件系统采用两级目录，其中第一级对应于用户账号，第二级对应于用户账号下的文件。

②首先应确定文件系统的数据结构：主目录、子目录及活动文件等。主目录和子目录都以文件的形式存放于磁盘，这样便于查找和修改。

③用户创建的文件，可以编号存储于磁盘上。如：file0,file1,file2…并以编号作为物理地址，在目录中进行登记。

（2）主要数据结构

a）OSFILE 结点

```
typedef struct  /*the structure of OSFILE*/
{int  fpaddr;                /*file physical address*/
```

```
    int  flength;               /*file length*/
    int  fmode;   /*file mode:0-Read Only;1-Write Only;2-Read and
                  Write(default);*/
    char fname[MAXNAME];        /*file name*/
   } OSFILE;
```

b) OSUFD 结点

```
typedef struct     /*the structure of OSUFD*/
   { char ufdname[MAXNAME];   /*ufd name*/
     OSFILE ufdfile[MAXCHILD];   /*ufd own file*/
    }OSUFD;
```

c) 用户密码

```
 typedef struct  /*the structure of OSUFD'LOGIN*/
 {char ufdname[MAXNAME];       /*ufd name*/
  char ufdpword[8];            /*ufd password*/
  } OSUFD_LOGIN;
```

d) 文件打开模式

```
typedef struct     /*file open mode*/
{int ifopen;     /*ifopen:0-close,1-open*/
 int openmode;   /*0-read only,1-write only,2-read and write,3-initial*/
}OSUFD_OPENMODE;
```

e) i 结点

```
struct dinode
{
 unsigned short di_number;        /*关联文件数*/
 unsigned short di_mode;          /*存取权限*/
 unsigned short di_uid;
 unsigned short di_gid;
 unsigned long di_size;                /*文件大小*/
 unsigned int di_addr[NADDR];          /*物理块号*/
```

（3）主要函数

a) 登陆文件系统函数LoginF()
b) 目录操作函数DirF()
c) 改变当前目录函数CdF()
d) 创建文件函数CreateF()
e) 删除文件函数DeleteF()
f) 修改文件权限函数ModifyFM()
g) 打开文件函数OpenF()
h) 关闭文件函数CloseF()
i) 读文件函数ReadF()
j) 写文件函数WriteF()
k) 退出文件系统函数QuitF()
l) 帮助函数help();

5. 源代码

参考程序见下（本程序需要在 c:下建一个名为 osfile 的目录及一个名为 file 的子目

录)：

```
#include "stdio.h"
#include "string.h"
#include "conio.h"
#include "stdlib.h"
#define MAXNAME 25  /*the largest length of mfdname,ufdname,filename*/
#define MAXCHILD 50 /*the largest child*/
#define MAX (MAXCHILD*MAXCHILD) /*the size of fpaddrno*/

typedef struct  /*the structure of OSFILE*/
  {int  fpaddr;                /*file physical address*/
   int  flength;               /*file length*/
   int  fmode;   /*file mode:0-Read Only;1-Write Only;2-Read and
Write(default);*/
   char fname[MAXNAME];        /*file name*/
  } OSFILE;

typedef struct     /*the structure of OSUFD*/
  {char ufdname[MAXNAME];   /*ufd name*/
   OSFILE ufdfile[MAXCHILD];   /*ufd own file*/
  }OSUFD;

typedef struct  /*the structure of OSUFD'LOGIN*/
  {char ufdname[MAXNAME];       /*ufd name*/
   char ufdpword[8];            /*ufd password*/
  } OSUFD_LOGIN;

typedef struct     /*file open mode*/
  {int ifopen;     /*ifopen:0-close,1-open*/
   int openmode;   /*0-read only,1-write only,2-read and write,3-initial*/
  }OSUFD_OPENMODE;

OSUFD *ufd[MAXCHILD];   /*ufd and ufd own files*/
OSUFD_LOGIN ufd_lp;

int ucount=0;  /*the count of mfd's ufds*/
int fcount[MAXCHILD];  /*the count of ufd's files*/
int loginsuc=0; /*whether login successfully*/
char username[MAXNAME];  /*record login user's name22*/
char dirname[MAXNAME];/*record current directory*/
int fpaddrno[MAX];  /*record file physical address num*/
OSUFD_OPENMODE ifopen[MAXCHILD][MAXCHILD]; /*record file open/close*/
int wgetchar; /*whether getchar()*/
```

```
FILE *fp_mfd,*fp_ufd,*fp_file_p,*fp_file;

void main()
{int i,j,choice1;
 char choice[50];
 /*choice operation:dir,create,open,delete,modify,read,write*/
 int choiceend=1;  /*whether choice end*/
 char *rtrim(char *str);  /*remove the trailing blanks.*/
 char *ltrim(char *str);  /*remove the heading blanks.*/
 void LoginF();  /*LOGIN FileSystem*/
 void DirF();  /*Dir FileSystem*/
 void CdF();  /*Change Dir*/
 void CreateF();  /*Create File*/
 void DeleteF(); /*Delete File*/
 void ModifyFM(); /*Modify FileMode*/
 void OpenF();  /*Open File*/
 void CloseF();  /*Close File*/
 void ReadF(); /*Read File*/
 void WriteF(); /*Write File*/
 void QuitF(); /*Quit FileSystem*/
 void help();
 if((fp_mfd=fopen("c:\\osfile\\mfd","rb"))==NULL)
   {fp_mfd=fopen("c:\\osfile\\mfd","wb");
    fclose(fp_mfd);
   }

 for(i=0;i<MAX;i++) fpaddrno[i]=0;
 textattr(BLACK*16|WHITE);
 clrscr();   /*clear screen*/
 LoginF();   /*user login*/
 clrscr();

 if(loginsuc==1)  /*Login Successfully*/
 {while (1)
 {wgetchar=0;
  if (choiceend==1)
  {printf("\n\nC:\\%s>",strupr(dirname));}
  else printf("Bad command or file name.\nC:\\%s>",strupr(username));
   gets(choice);
   strcpy(choice,ltrim(rtrim(strlwr(choice))));
   if (strcmp(choice,"dir")==0) choice1=1;
   else if(strcmp(choice,"creat")==0) choice1=2;
   else if(strcmp(choice,"delete")==0) choice1=3;
   else if(strcmp(choice,"attrib")==0) choice1=4;
```

```
   else if(strcmp(choice,"open")==0) choice1=5;
   else if(strcmp(choice,"close")==0) choice1=6;
   else if(strcmp(choice,"read")==0) choice1=7;
   else if(strcmp(choice,"modify")==0) choice1=8;
   else if(strcmp(choice,"exit")==0) choice1=9;
   else if(strcmp(choice,"cls")==0) choice1=10;
   else if(strcmp(choice,"cd")==0) choice1=11;
   else if(strcmp(choice,"help")==0) choice1=20;
   else choice1=12;
     switch(choice1)
       {case 1:DirF();choiceend=1;break;
    case 2:CreateF();choiceend=1;if(!wgetchar) getchar();break;
    case 3:DeleteF();choiceend=1;if(!wgetchar)getchar();break;
    case 4:ModifyFM();choiceend=1;if(!wgetchar) getchar();break;
    case 5:choiceend=1;OpenF();if (!wgetchar) getchar();break;
    case 6:choiceend=1;CloseF();if (!wgetchar) getchar();break;
    case 7:choiceend=1;ReadF();if (!wgetchar) getchar();break;
    case 8:choiceend=1;WriteF();if (!wgetchar) getchar();break;
    case 9:printf("\nYou have exited this system.");
           QuitF();exit(0);break;
    case 10:choiceend=1;clrscr();break;
    case 11:CdF();choiceend=1;break;
    case 20:help();choiceend=1;break;
    default:choiceend=0;
       }
      }
}
else printf("\nAccess denied.");
}

void help(void)
{
printf("\nThe Command List\n");
printf("\nCd  Attrib  Creat  Modify  Read  Open  Cls  Delete  Exit  Close\n");
}
char *rtrim(char *str)  /*remove the trailing blanks.*/
{int n=strlen(str)-1;
 while(n>=0)
 {if(*(str+n)!=' ')
   {*(str+n+1)='\0';
    break;
   }
   else n--;
 }
```

```
 if (n<0) str[0]='\0';
 return str;
}

char *ltrim(char *str) /*remove the heading blanks.*/
{char *rtrim(char *str);
 strrev(str);
 rtrim(str);
 strrev(str);
 return str;
}

void LoginF()  /*LOGIN FileSystem*/
{char loginame[MAXNAME],loginpw[9],logincpw[9],str[50];
 int i,j,flag=1;
 char a[25];
 int findout; /*login user not exist*/
 char *rtrim(char *str);  /*remove the trailing blanks.*/
 char *ltrim(char *str);  /*remove the heading blanks.*/
 void InputPW(char *password);  /*input password,use '*' replace*/
 void SetPANo(int RorW);  /*Set physical address num*/
 while(1)
 {findout=0;
 printf("\n\nLogin Name:");
 gets(loginame);
 ltrim(rtrim(loginame));
 fp_mfd=fopen("c:\\osfile\\","rb");
 for(i=0;fread(&ufd_lp,sizeof(OSUFD_LOGIN),1,fp_mfd)!=0;i++)
  if (strcmp(strupr(ufd_lp.ufdname),strupr(loginame))==0)
    {findout=1;
     strcpy(logincpw,ufd_lp.ufdpword);
    }
 fclose(fp_mfd);
 if (findout==1)  /*user exist*/
    {printf("Login Password:");
     InputPW(loginpw); /*input password,use '*' replace*/
      if (strcmp(loginpw,logincpw)==0)
     {strcpy(username,strupr(loginame));
      strcpy(dirname,username);
      fp_mfd=fopen("c:\\osfile\\","rb");
      for(j=0;fread(&ufd_lp,sizeof(OSUFD_LOGIN),1,fp_mfd)!=0;j++)
        {strcpy(str,"c:\\osfile\\");
         strcat(str,ufd_lp.ufdname);
         ufd[j]=(OSUFD*)malloc(sizeof(OSUFD));
```

```
          strcpy(ufd[j]->ufdname,strupr(ufd_lp.ufdname));
          fp_ufd=fopen(str,"rb");
          fcount[j]=0;

for(i=0;fread(&ufd[j]->ufdfile[i],sizeof(OSFILE),1,fp_ufd)!=0;i++,fcount[j]++)
            {ifopen[j][i].ifopen=0;
             ifopen[j][i].openmode=4;}
          fclose(fp_ufd);}
       fclose(fp_mfd);
       ucount=j;
          SetPANo(0);
      printf("\n\nLogin successful! Welcome to this FileSystem\n\n");
      loginsuc=1;
      return;}
       else
      {printf("\n\n");
       flag=1;
       while(flag)
       {printf("Login Failed!  Password Error.  Try Again(Y/N):");
       gets(a);
       ltrim(rtrim(a));
       if (strcmp(strupr(a),"Y")==0) {loginsuc=0;flag=0;}
       else if(strcmp(strupr(a),"N")==0){loginsuc=0;flag=0;return;}
       }
      }
       }
 else
       {printf("New Password(<=8):");
        InputPW(loginpw); /*input new password,use '*' replace*/
        printf("\nConfirm Password(<=8):"); /*input new password,use '*' replace*/
        InputPW(logincpw);
       if (strcmp(loginpw,logincpw)==0)
      {strcpy(ufd_lp.ufdname,strupr(loginame));
       strcpy(ufd_lp.ufdpword,loginpw);
       fp_mfd=fopen("c:\\osfile\\","ab");
       fwrite(&ufd_lp,sizeof(OSUFD_LOGIN),1,fp_mfd);
       fclose(fp_mfd);
       strcpy(username,strupr(loginame));
       strcpy(dirname,loginame);
       strcpy(str,"c:\\osfile\\");
       strcat(str,username);
       if((fp_ufd=fopen(str,"rb"))==NULL)
         {fp_ufd=fopen(str,"wb");
          fclose(fp_ufd);
```

```
        }
       fp_mfd=fopen("c:\\osfile\\","rb");
       for(j=0;fread(&ufd_lp,sizeof(OSUFD_LOGIN),1,fp_mfd)!=0;j++)
         {strcpy(str,"c:\\osfile\\");
          strcat(str,ufd_lp.ufdname);
          ufd[j]=(OSUFD*)malloc(sizeof(OSUFD));
          strcpy(ufd[j]->ufdname,strupr(ufd_lp.ufdname));
          fp_ufd=fopen(str,"rb");

for(i=0;fread(&ufd[j]->ufdfile[i],sizeof(OSFILE),1,fp_ufd)!=0;i++,fcount[j]++)
             {ifopen[j][i].ifopen=0;
             ifopen[j][i].openmode=4;}
          fclose(fp_ufd);}
       fclose(fp_mfd);
       ucount=j;
       SetPANo(0);
       printf("\n\nLogin Successful! Welcome to this System\n\n");
       loginsuc=1;
       return;
       }
       else
       {printf("\n\n");
       flag=1;
       while(flag)
       {printf("Login Failed! Password Error. Try Again(Y/N):");
       gets(a);
       ltrim(rtrim(a));
       if (strcmp(strupr(a),"Y")==0) {loginsuc=0;flag=0;}
       else if(strcmp(strupr(a),"N")==0){loginsuc=0;flag=0;return;}
       }
       }
 }
 }
 }

void SetPANo(int RorW)  /*Set physical address num,0-read,1-write*/
{int i,j;
 if (RorW==0)
    {if((fp_file_p=fopen("c:\\osfile\\file\\file_p","rb"))==NULL)
        {fp_file_p=fopen("c:\\osfile\\file\\file_p","wb");
         fclose(fp_file_p);
        }
     fp_file_p=fopen("c:\\osfile\\file\\file_p","rb");
     for(i=0;fread(&j,sizeof(int),1,fp_file_p)!=0;i++)
```

```
        fpaddrno[j]=1;
      /*for(i=1;i<MAX;i++)
        if ((i%13)==0) fpaddrno[i]=1;*/
     }
 else
    {fp_file_p=fopen("c:\\osfile\\file\\file_p","wb");
     /*for(i=1;i<MAX;i++)
       if((i%13)==0) fpaddrno[i]=0;*/
     for(i=0;i<MAX;i++)
     if (fpaddrno[i]==1)
        fwrite(&i,sizeof(int),1,fp_file_p);
     }
 fclose(fp_file_p);
}

void InputPW(char *password)  /*input password,use '*' replace*/
{int j;
 for(j=0;j<=7;j++)
    {password[j]=getch();
     if ((int)(password[j])!=13)
       {if((int)(password[j])!=8)
          putchar('*');
        else
          {if (j>0)
        {j--;j--;
         putchar('\b');putchar(' ');putchar('\b');
        }
        else j--;
          }
        }
     else
      {password[j]='\0';
       break;
      }
    }
 password[j]='\0';
}

void DirF()  /*Dir FileSystem*/
{int i,j,count=0;
 char sfmode[25],sfpaddr[25],str[25];
 int ExistD(char *dirname);  /*Whether DirName Exist,Exist-i,Not Exist-0*/
 clrscr();
 if (strcmp(strupr(ltrim(rtrim(dirname))),"")!=0)
```

```
 {printf("\n\nC:\\%s>dir\n",dirname);

printf("\n%14s%16s%14s%10s%18s\n","FileName","FileAddress","FileLength","Type",
"FileMode");
  j=ExistD(dirname);
 for(i=0;i<fcount[j];i++)
   {if ((i%16==0)&&(i!=0))
     {printf("\nPress any key to continue..");
     getch();
     clrscr();

printf("\n%14s%16s%14s%10s%18s\n","FileName","FileAddress","FileLength","Type",
"FileMode");
     }
    itoa(ufd[j]->ufdfile[i].fpaddr,str,10);
    strcpy(sfpaddr,"file");
    strcat(sfpaddr,str);
    if (ufd[j]->ufdfile[i].fmode==0) strcpy(sfmode,"Read Only");
    else if(ufd[j]->ufdfile[i].fmode==1) strcpy(sfmode,"Write Only");
    else if(ufd[j]->ufdfile[i].fmode==2)strcpy(sfmode,"Read And Write");
    else strcpy(sfmode,"Protect");
    printf("%14s%16s%14d%10s%18s\n",ufd[j]->ufdfile[i].fname,sfpaddr,ufd[j]->uf
dfile[i].flength,"<FILE>",sfmode);
   }
 printf("\n %3d file(s)\n",fcount[j]);}
 else
 {printf("\n\nC:\\>dir\n");
  printf("\n%14s%18s%8s\n","DirName","OwnFileCount","Type");
  for(i=0;i<ucount;i++)
   {if ((i%16==0)&&(i!=0))
     {printf("\nPress any key to continue...");
     getch();
     clrscr();
       printf("\n%14s%18s%8s\n","DirName","OwnFileCount","Type");
    }
    printf("%14s%18d%8s\n",ufd[i]->ufdname,fcount[i],"<UFD>");
    count=count+fcount[i];
   }
  printf("\n %3d dir(s),%5d file(s)\n",ucount,count);
 }
}

int ExistD(char *dirname)  /*Whether DirName Exist,Exist-i,Not Exist-0*/
{int i;
```

```
 int exist=0;
 for(i=0;i<ucount;i++)
   if (strcmp(strupr(ufd[i]->ufdname),strupr(dirname))==0)
      {exist=1;
       break;
      }
 if (exist) return(i);
 else return(-1);
}

void CdF() /*Exchange Dir*/
{char dname[MAXNAME];
 char *rtrim(char *str);  /*remove the trailing blanks.*/
 char *ltrim(char *str);  /*remove the heading blanks.*/
 int ExistD(char *filename);  /*Whether FileName Exist,Exist-i,Not Exist-0*/
 printf("\nPlease input DirName (cd..-Previous dir; DirNAME-cd [DirNAME]):");
 gets(dname);
 ltrim(rtrim(dname));
 if (ExistD(dname)>=0)  strcpy(dirname,strupr(dname));
 else if(strcmp(strupr(dname),"CD..")==0)  strcpy(ltrim(rtrim(dirname)),"");
 else printf("\nError.\'%s\' does not exist.\n",dname);
}

void CreateF()  /*Create File*/
{int fpaddrno,flag=1,i;
 char fname[MAXNAME],str[50],str1[50],strtext[255],a[25];
 char fmode[25];

 char *rtrim(char *str);  /*remove the trailing blanks.*/
 char *ltrim(char *str);  /*remove the heading blanks.*/
 int FindPANo();  /*find out physical address num*/
 int WriteF1(); /*write file*/
 int ExistF(char *filename);  /*Whether FileName Exist,Exist-i,Not Exist-0*/
 int ExistD(char *dirname);
 if (strcmp(strupr(dirname),strupr(username))!=0)
 {printf("\nError. You must create file in your own dir.\n");wgetchar=1;}
 else
 {
 printf("\nPlease input FileName:");
 gets(fname);
 ltrim(rtrim(fname));
 if (ExistF(fname)>=0)
   {printf("\nError. Name \'%s\' has already existed.\n",fname);
    wgetchar=1;
```

```
   }
 else
   {printf("Please input FileMode(0-Read Only, 1-Write Only, 2-Read and Write,
3-Protect):");
    gets(fmode);
    ltrim(rtrim(fmode));
    if((strcmp(fmode,"0")==0)||(strcmp(fmode,"1")==0)||(strcmp(fmode,"2")==0)||
(strcmp(fmode,"3")==0))
      {fpaddrno=FindPANo();
       if (fpaddrno>=0)
     {i=ExistD(username);
      strcpy(ufd[i]->ufdfile[fcount[i]].fname,fname);
      ufd[i]->ufdfile[fcount[i]].fpaddr=fpaddrno;
      ufd[i]->ufdfile[fcount[i]].fmode=atoi(fmode);
      ifopen[i][fcount[i]].ifopen=0;
      ifopen[i][fcount[i]].openmode=4;
      strcpy(str,"c:\\osfile\\file\\file");
      itoa(fpaddrno,str1,10);
      strcat(str,str1);
      fp_file=fopen(str,"wb");
      fclose(fp_file);
      fcount[i]++;
      while(flag)
      {printf("Input text now(Y/N):");
       gets(a);
       ltrim(rtrim(a));
       ufd[i]->ufdfile[fcount[i]-1].flength=0;
       if(strcmp(strupr(a),"Y")==0)
        {fp_file=fopen(str,"wb+");
         ufd[i]->ufdfile[fcount[i]-1].flength=WriteF1();
         flag=0;
       }
       else if(strcmp(strupr(a),"N")==0){flag=0;wgetchar=1;}
      }
      printf("\n\'%s\' has been created successfully!\n",fname);
     }
       else
    {printf("\nFail!No Disk Space. Please format your disk.\n");wgetchar=1;}
      }
      else {printf("\nError. FileMode\'s Range is 0-3\n");wgetchar=1;}
   }}
}

int ExistF(char *filename)  /*Whether FileName Exist,Exist-i,Not Exist-0*/
```

```
{int i,j;
 int exist=0;
 int ExistD(char *dirname);
 j=ExistD(dirname);
 for(i=0;i<fcount[j];i++)
   if (strcmp(strupr(ufd[j]->ufdfile[i].fname),strupr(filename))==0)
      {exist=1;
       break;
      }
 if (exist) return(i);
 else return(-1);
}

int FindPANo()  /*find out physical address num*/
{int i;
 for(i=0;i<MAX;i++)
   if (fpaddrno[i]==0) {fpaddrno[i]=1;break;}
 if (i<MAX) return(i);
 else return(-1);
}

int WriteF1()  /*write file*/
{int length=0;
 char c;
 printf("Please input text(\'#\' stands for end):\n");
 while((c=getchar())!='#')
 {fprintf(fp_file,"%c",c);
  if (c!='\n') length++;
 }
 fprintf(fp_file,"\n");
 fclose(fp_file);
 return(length);
}

void DeleteF() /*Delete File*/
{char fname[MAXNAME];
 char str[50],str1[50];
 int i,j,k,flag=1;
 char a[25]; /*whether delete*/
 char *rtrim(char *str);  /*remove the trailing blanks.*/
 char *ltrim(char *str);  /*remove the heading blanks.*/
 int ExistF(char *filename);  /*Whether FileName Exist,Exist-i,Not Exist-0*/
 int ExistD(char *dirname);
 if (strcmp(strupr(dirname),strupr(username))!=0)
```

```
 {printf("\nError. You can only delete file in your own dir.\n");wgetchar=1;}
 else
 {printf("\nPlease input FileName:");
 gets(fname);
 ltrim(rtrim(fname));
 i=ExistF(fname);
 if (i>=0)
    {k=ExistD(username);
     if(ifopen[k][i].ifopen==1)
      {printf("\nError. \'%s\' is in open status. Close it before
delete.\n",fname);wgetchar=1;}
     else
     {
      while(flag)
       {printf("\'%s\' will be deleted. Are you sure(Y/N):",fname);
    gets(a);
    ltrim(rtrim(a));
    if(strcmp(strupr(a),"Y")==0)
    {fpaddrno[ufd[k]->ufdfile[i].fpaddr]=0;
    itoa(ufd[k]->ufdfile[i].fpaddr,str,10);
    for(j=i;j<fcount[k]-1;j++)
      {strcpy(ufd[k]->ufdfile[j].fname,ufd[k]->ufdfile[j+1].fname);
       ufd[k]->ufdfile[j].fpaddr=ufd[k]->ufdfile[j+1].fpaddr;
       ufd[k]->ufdfile[j].flength=ufd[k]->ufdfile[j+1].flength;
       ufd[k]->ufdfile[j].fmode=ufd[k]->ufdfile[j+1].fmode;
       ifopen[k][j]=ifopen[k][j+1];
      }
    fcount[k]--;
    strcpy(str1,"c:\\osfile\\file\\file");
    strcat(str1,str);
    remove(str1);
    flag=0;
    printf("\n\'%s\' has been deleted successfully.\n",fname);
    wgetchar=1;
       }
     else if(strcmp(strupr(a),"N")==0)
     {printf("\nError. \'%s\' hasn\'t been deleted.\n",fname);
      wgetchar=1;
      flag=0;}
      }}}
 else
   {printf("\nError. \'%s\' does not exist.\n",fname);wgetchar=1;}}
}
```

```
void ModifyFM() /*Modify FileMode*/
{char fname[MAXNAME],str[50];
 int i,j,k,flag;
 char fmode[25]; /*whether delete*/
 char *rtrim(char *str);  /*remove the trailing blanks.*/
 char *ltrim(char *str);  /*remove the heading blanks.*/
 void InputPW(char *password);  /*input password,use '*' replace*/
 void SetPANo(int RorW);  /*Set physical address num*/
 int ExistF(char *filename);  /*Whether FileName Exist,Exist-i,Not Exist-0*/
 int ExistD(char *dirname);
 if (strcmp(strupr(dirname),strupr(username))!=0) {printf("\nError.You can only
modify filemode in yourself dir.\n");wgetchar=1;}
 else
{ printf("\nPlease input FileName:");
 gets(fname);
 ltrim(rtrim(fname));
 i=ExistF(fname);
 if (i>=0)
    {k=ExistD(username);
     if(ifopen[k][i].ifopen==1)
     {printf("\nError.\'%s\' is in open status. Close it before
modify.\n",fname);wgetchar=1;}
     else
     {
     if(ufd[k]->ufdfile[i].fmode==0) strcpy(str,"read only");    /*FileMode*/
     else if(ufd[k]->ufdfile[i].fmode==1) strcpy(str,"write only");
     else if(ufd[k]->ufdfile[i].fmode==2) strcpy(str,"read and write");
     else strcpy(str,"Protect");

     printf("\'%s\' filemode is %s.\n",fname,strupr(str));
     printf("Modify to(0-read only,1-write only,2-read and write,3-Protect):");
     gets(fmode);
     ltrim(rtrim(fmode));
     if(strcmp(fmode,"0")==0)
       {ufd[k]->ufdfile[i].fmode=0;
    printf("\n\'%s\' has been modified to READ ONLY mode successfully.\n",fname);
    wgetchar=1;
       }
     else if(strcmp(fmode,"1")==0)
       {ufd[k]->ufdfile[i].fmode=1;
    printf("\n\'%s\' has been modified to WRITE ONLY mode successfully.\n",fname);
    wgetchar=1;
       }
     else if(strcmp(fmode,"2")==0)
```

```
       {ufd[k]->ufdfile[i].fmode=2;
    printf("\n\'%s\' has been modified to READ AND WRITE mode
successfully.\n",fname);
    wgetchar=1;
       }
     else if(strcmp(fmode,"3")==0)
       {ufd[k]->ufdfile[i].fmode=3;
    printf("\n\'%s\' has been modified to FORBID mode successfully.\n",fname);
    wgetchar=1;
       }
     else {printf("\nError.\'%s\' is not modified.\n",fname);wgetchar=1;}
    }
  }
 else
   {printf("\nError. \'%s\' dose not exist.\n",fname);wgetchar=1;}}
}


void OpenF() /*Open File*/
{char fname[MAXNAME];
 char str[25],str1[25],fmode[25];
 int i,k;
 char *rtrim(char *str);  /*remove the trailing blanks.*/
 char *ltrim(char *str);  /*remove the heading blanks.*/
 int ExistF(char *filename);  /*Whether FileName Exist,Exist-i,Not Exist-0*/
 int ExistD(char *dirname);
 if (strcmp(strupr(ltrim(rtrim(dirname))),"")==0)
 {printf("\nError. Please change to ufd dir before open.\n");wgetchar=1;return;}
 printf("\nPlease input FileName:");
 gets(fname);
 ltrim(rtrim(fname));
 i=ExistF(fname);
 if (i>=0)
  {k=ExistD(dirname);
   if(!ifopen[k][i].ifopen)
    {if (ufd[k]->ufdfile[i].fmode==3)
    {printf("\nError. The file\'s mode is FORBID. Can not open.\n");wgetchar=1;}
     else
     {printf("Please input FileOpenMode(0-Read Only,1-Write Only,2-Read and
Write):");
     gets(fmode);
     ltrim(rtrim(fmode));
     if((strcmp(fmode,"0")==0)||(strcmp(fmode,"1")==0)||(strcmp(fmode,"2")==0))
     {if(fmode[0]=='0')   /*open file with read only mode*/
```

```
    {strcpy(str,"read only");
     if((ufd[k]->ufdfile[i].fmode==0)||(ufd[k]->ufdfile[i].fmode==2))
ifopen[k][i].ifopen=1;
    }
      else if(fmode[0]=='1') /*open file with write only mode*/
    {strcpy(str,"write only");
     if((ufd[k]->ufdfile[i].fmode==1)||(ufd[k]->ufdfile[i].fmode==2))
ifopen[k][i].ifopen=1;
    }
      else if(fmode[0]=='2')  /*open file with read and write mode*/
    {strcpy(str,"read and write");
     if(ufd[k]->ufdfile[i].fmode==2) ifopen[k][i].ifopen=1;
    }
      if(ufd[k]->ufdfile[i].fmode==0) strcpy(str1,"read only");    /*FileMode*/
      else if(ufd[k]->ufdfile[i].fmode==1) strcpy(str1,"write only");
      else if(ufd[k]->ufdfile[i].fmode==2) strcpy(str1,"read and write");
      if(ifopen[k][i].ifopen==1)
    {ifopen[k][i].openmode=atoi(fmode);
     if (ifopen[k][i].openmode==0) strcpy(str,"read only");
     else if(ifopen[k][i].openmode==1) strcpy(str,"write only");
     else if(ifopen[k][i].openmode==2) strcpy(str,"read and write");
     printf("\n\'%s\' has been opened. OpenMode is %s,FileMode
is %s\n",fname,strupr(str),strupr(str1));
     wgetchar=1;
    }
       else
       {printf("\nError. \'%s\' hasn\'t been opened. OpenMode Error. OpenMode
is %s,but FileMode is %s\n",fname,strupr(str),strupr(str1));wgetchar=1;}
     }
     else {printf("\nError. FileOpenMode\'s Range is 0-2\n");wgetchar=1;}
     }}
    else {printf("\nError. \'%s\' is in open status.\n",fname);wgetchar=1;}
   }
 else
   {printf("\nError. \'%s\' does not exist.\n",fname);wgetchar=1;}
}

void CloseF() /*Close File*/
{int i,k,n=0;
 char fname[MAXNAME];
 char *rtrim(char *str);  /*remove the trailing blanks.*/
 char *ltrim(char *str);  /*remove the heading blanks.*/
 int ExistF(char *filename);  /*Whether FileName Exist,Exist-i,Not Exist-0*/
 int ExistD(char *dirname);
```

```
 if (strcmp(strupr(ltrim(rtrim(dirname))),"")==0)
 {printf("\nError. Please convert to ufd dir before close.\n");wgetchar=1;return;}
 k=ExistD(dirname);
 printf("\nOpen File(s) In This Ufd:\n");/*display openned file*/
 for(i=0;i<fcount[k];i++)
   {if (ifopen[k][i].ifopen==1) {printf("%15s",ufd[k]->ufdfile[i].fname);n++;}
    if((n%4==0)&&(n!=0)) printf("\n");
   }
 printf("\n%d files openned.\n",n);
 if (n==0) wgetchar=1;
 if(n!=0)
{printf("\nPlease input FileName:");
 gets(fname);
 ltrim(rtrim(fname));
 i=ExistF(fname);
 if(i>=0)
  {if(ifopen[k][i].ifopen==1)
   {ifopen[k][i].ifopen=0;
    ifopen[k][i].openmode=4;
    printf("\n\'%s\' has been closed successfully.\n",fname);
    wgetchar=1;
   }
   else {printf("\nError.\'%s\' is in closing status.\n",fname);wgetchar=1;}
  }
 else {printf("\nError. \'%s\' is not exist.\n",fname);wgetchar=1;}
}
}

void ReadF() /*Read File*/
{int i,k,n=0;
 char fname[MAXNAME];
 char str[255],str1[255],c;
 char *rtrim(char *str);  /*remove the trailing blanks.*/
 char *ltrim(char *str);  /*remove the heading blanks.*/
 int ExistF(char *filename);  /*Whether FileName Exist,Exist-i,Not Exist-0*/
 int ExistD(char *dirname);
 if (strcmp(strupr(ltrim(rtrim(dirname))),"")==0) {printf("\nError.Please
convert to ufd dir before read.\n");wgetchar=1;return;}
 printf("\nCaution:Open file first\n");
 printf("Opened File(s) List:\n");
 k=ExistD(dirname);
 for(i=0;i<fcount[k];i++)
   {if (ifopen[k][i].ifopen==1)
     if ((ifopen[k][i].openmode==0) ||(ifopen[k][i].openmode==2))
```

```
{printf("%15s",ufd[k]->ufdfile[i].fname);n++;}
    if((n%4==0)&&(n!=0)) printf("\n");
   }
 printf("\n%d files openned.\n",n);
 if (n==0) wgetchar=1;
 if(n!=0)
{printf("\nPlease input FileName:");
 gets(fname);
 ltrim(rtrim(fname));
 i=ExistF(fname);
 if(i>=0)
  {if(ifopen[k][i].ifopen==1)
   {if((ifopen[k][i].openmode==0) ||(ifopen[k][i].openmode==2))
     {itoa(ufd[k]->ufdfile[i].fpaddr,str,10);
      strcpy(str1,"file");
      strcat(str1,str);
      strcpy(str,"c:\\osfile\\file\\");
      strcat(str,str1);
      fp_file=fopen(str,"rb");
      fseek(fp_file,0,0);
      printf("\nThe text is:\n\n");
      printf("   ");
      while(fscanf(fp_file,"%c",&c)!=EOF)
       if (c=='\n') printf("\n   ");
       else printf("%c",c);
      printf("\n\n%d Length.\n",ufd[k]->ufdfile[i].flength);
      fclose(fp_file);
      wgetchar=1;
     }
    else
    {printf("\nError.\'%s\' has been opened with WRITE ONLY mode. It isn\'t
read.\n",fname);wgetchar=1;}
   }
   else {printf("\nError.\'%s\' is in closing status. Please open it before
read\n",fname);wgetchar=1;}
  }
 else {printf("\nError. \'%s\' does not exist.\n",fname);wgetchar=1;}
}
}

void WriteF() /*Write File*/
{int i,k,n=0;
 char fname[MAXNAME];
 char str[50],str1[50],a[50];
```

```
 char *rtrim(char *str);  /*remove the trailing blanks.*/
 char *ltrim(char *str);  /*remove the heading blanks.*/
 int ExistF(char *filename);  /*Whether FileName Exist,Exist-i,Not Exist-0*/
 int ExistD(char *dirname);
 int WriteF1(); /*write file*/
 if (strcmp(strupr(ltrim(rtrim(dirname))),"")==0) {printf("\nError. Please
convert to ufd dir before write.\n");wgetchar=1;return;}
 k=ExistD(dirname);
 printf("\nOpen File(s) with write only mode or read and write mode:\n");/*display
openned files with writable mode*/
 for(i=0;i<fcount[k];i++)
   {if (ifopen[k][i].ifopen==1)
     if ((ifopen[k][i].openmode==1) ||(ifopen[k][i].openmode==2))
{printf("%15s",ufd[k]->ufdfile[i].fname);n++;}
    if((n%4==0)&&(n!=0)) printf("\n");
   }
 printf("\n%d files open.\n",n);
 if (n==0) wgetchar=1;
 if(n!=0)
{printf("\nPlease input FileName:");
 gets(fname);
 ltrim(rtrim(fname));
 i=ExistF(fname);
 if(i>=0)
  {if(ifopen[k][i].ifopen==1)
   {if((ifopen[k][i].openmode==1) ||(ifopen[k][i].openmode==2))
     {itoa(ufd[k]->ufdfile[i].fpaddr,str,10);
      strcpy(str1,"file");
      strcat(str1,str);
      strcpy(str,"c:\\osfile\\file\\");
      strcat(str,str1);
      if (ufd[k]->ufdfile[i].flength!=0)
    {printf("\n\'%s\' has text. Overwrite or Append(O-overwrite,A-Append,else-not
write):",fname);
     gets(a);
     ltrim(rtrim(a));
     if (fp_file!=NULL) fclose(fp_file);
     if (strcmp(strupr(a),"O")==0)
       {printf("\nOverwrite\n");
        fp_file=fopen(str,"wb");
        ufd[k]->ufdfile[i].flength=0;
        ufd[k]->ufdfile[i].flength=WriteF1();
       }
     else if(strcmp(strupr(a),"A")==0)
```

```
        {printf("\nAppend\n");
         fp_file=fopen(str,"ab");
         ufd[k]->ufdfile[i].flength=ufd[k]->ufdfile[i].flength+WriteF1();
        }
      else
        {printf("\nError.\'%s\' has not been written.\n",fname);
         fclose(fp_file);
         wgetchar=1;
        }
     }
        else
      {fp_file=fopen(str,"wb");
       ufd[k]->ufdfile[i].flength=WriteF1();
      }
      }
     else
     {printf("\nError. \'%s\' has been opened with read only mode.It isn\'t
writed.\n",fname);wgetchar=1;}
    }
    else
     {printf("\nError. \'%s\' is in closing status. Please open it before
write\n",fname);wgetchar=1;}
   }
 else
   {printf("\nError. \'%s\' does not exist.\n",fname);wgetchar=1;}
}
}

void QuitF() /*Quit FileSystem*/
{int i,j;
 char str[50];
 void SetPANo(int RorW);  /*Set physical address num,0-read,1-write*/
 SetPANo(1);
 if (fp_mfd!=NULL) fclose(fp_mfd);
 if (fp_ufd!=NULL) fclose(fp_ufd);
 if (fp_file!=NULL) fclose(fp_file);
 for(j=0;j<ucount;j++)
{strcpy(str,"c:\\osfile\\");
 strcat(str,ufd[j]->ufdname);
 ltrim(rtrim(str));
 fp_ufd=fopen(str,"wb");
 fclose(fp_ufd);
 fp_ufd=fopen(str,"ab");
 for(i=0;i<fcount[j];i++)
```

```
    fwrite(&ufd[j]->ufdfile[i],sizeof(OSFILE),1,fp_ufd);
  fclose(fp_ufd);}
}
```

6. 实验运行结果

（1）登陆界面，见图 1

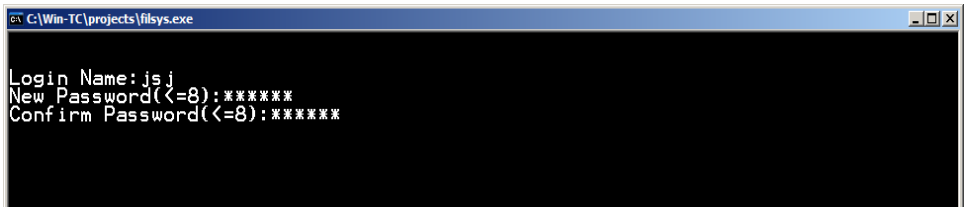

图 1 登陆界面

（2）创建文件界面，见图 2

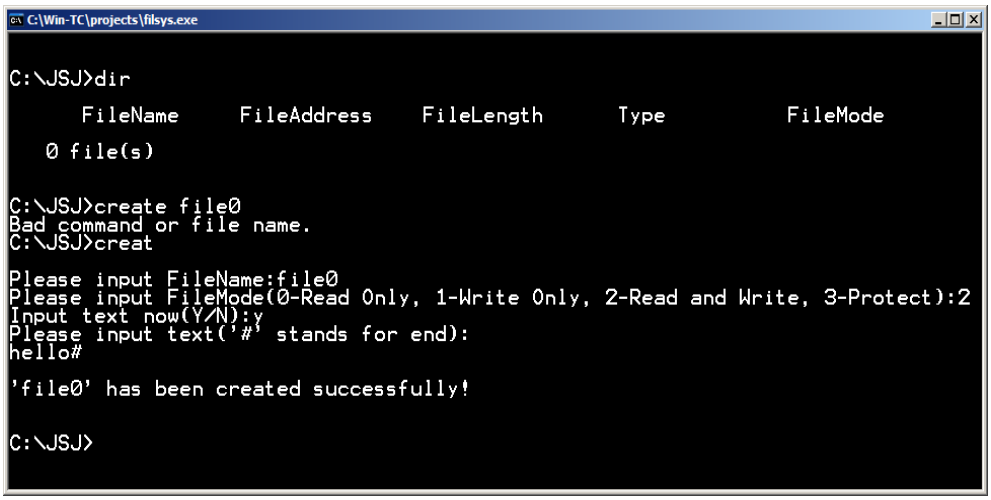

图 2 创建文件界面

（3）删除文件界面，见图 3

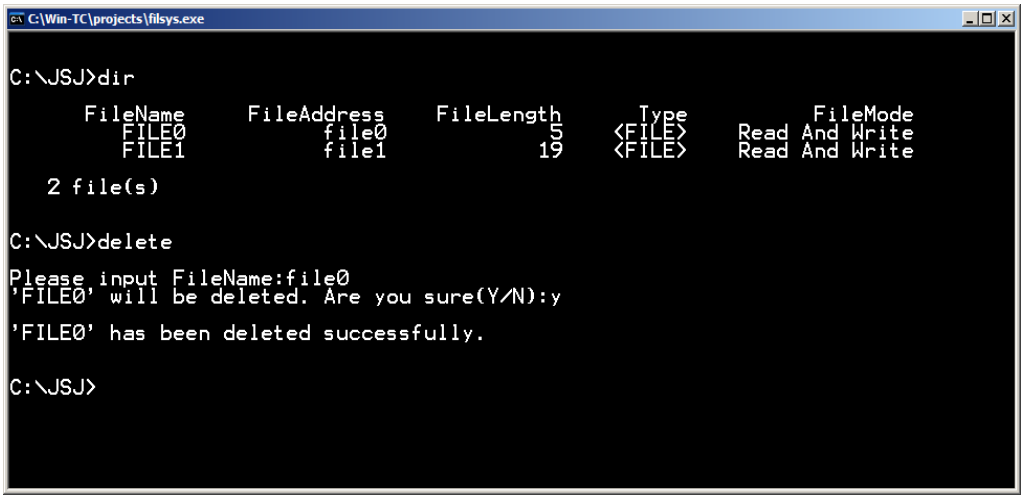

图 3 删除文件界面

（4）打开文件界面，见图 4

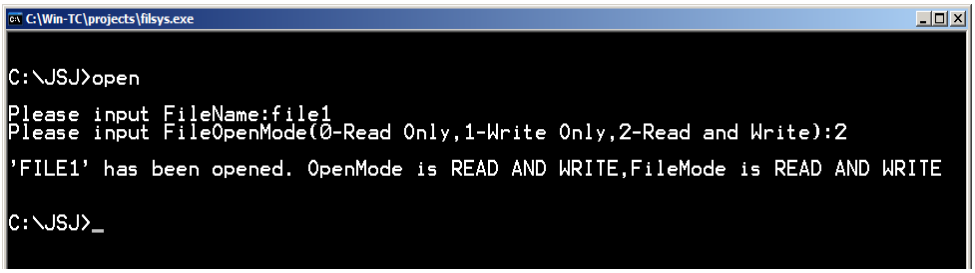

图 4 打开文件界面

（5）修改文件界面，见图 5

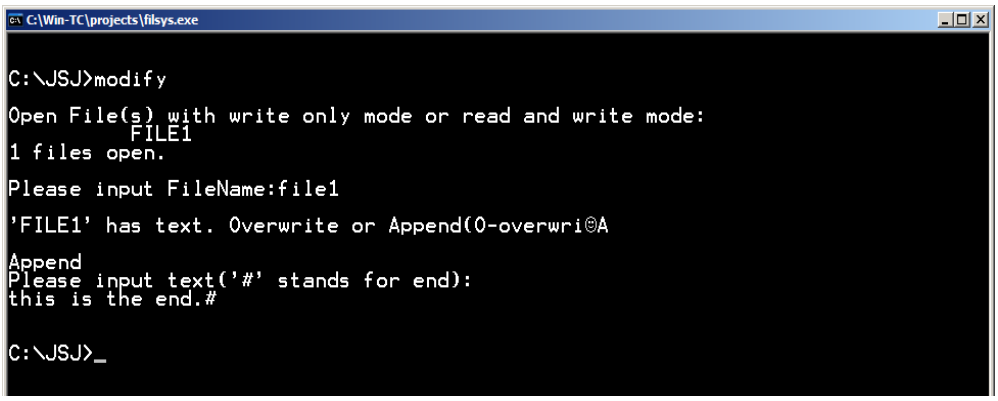

图 5 修改文件界面

（6）读文件界面，见图 6

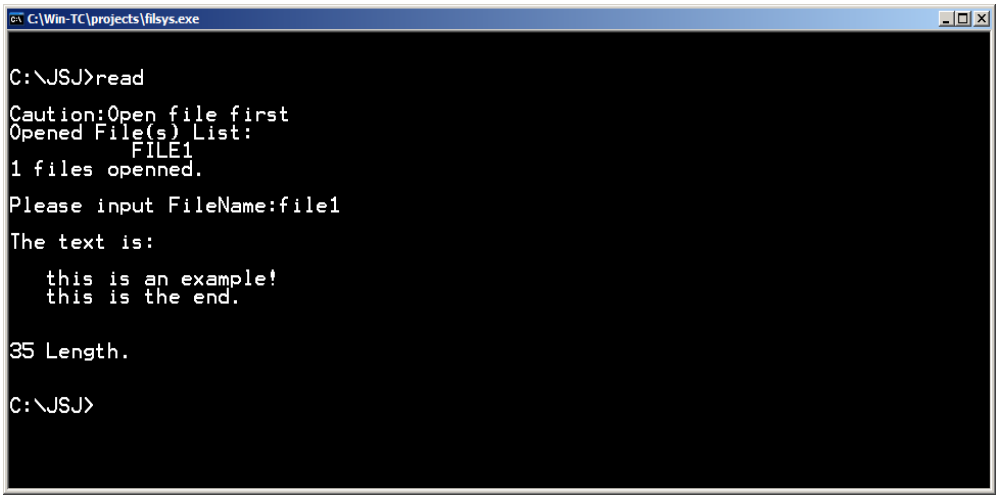

图 6 读文件界面

7. 心得体会