Tuan Nguyen

File System

A File system used to store and organizes data. Within the system, we can access individual file by called their name and the place they are stored. For example, home/dir1/file1. The file can contain information such as the size of the file, as well as its attributes, location and hierarchy in the directory metadata. Metadata can store and retrieve file such as, date created, data modified, data of access, user ID of file creator, and access permission.

**The Design for project are:**

- 8 blocks (4096 bytes) for File Allocation Table, each entry will store pointer

to next block (2 bytes)

- 2 blocks (1024 bytes) for Directory Entry Table which store filename, their first

block in FAT and meta data, each entry will be 32 bytes

- The rest are for Data, which will be 1048576 bytes in total

-Drive use: 2MB, 3MB, 5MB, 10MB

**Code Structure:**

-I used memory mapped file I/O in this project and divide the drive into 3 portions:

fatMap , dirTable and dataMap

-The project consists of 6 main IOCS(input output control system) functions and an User Interface to experiment with it

1) create:

-search for file in root directory, if already exist generate error

-else: create an entry, update Dir Table, update FAT, add the filename to the current

directory block

-if the file is a directory, update currentDir to indicate we are in the directory after

creation

2) delete:

-search for file in root directory, if not exist, generate error

-remove entry from dir Table

-Update FAT and the filename in current directory block

3) open:

-find the first block of the file

4) close: -go back to root directory

5) read: -from the 1st block, navigate all the other blocks and read it into a buffer

6) write: -from the 1st block, navigate to the last block, find new empty block to inser

data, then update FAT, meta data.

**Pseudocode**

**Main.c :** option for user to select( drive 2MB, 3MB, 5MB, 10MB), and 6 main IOCS(Input Output Control System) functions.

int main(){

char str1[20]; // allocate to store drive options

int input; // allocate to store user choice, like to open ,to close file

// drive option for user to select

printf("Enter the drive: (Drive2MB , Drive3MB ,Drive5MB , Drive10MB): \n");

scanf("%s" , str1); // get user input

setup(str1); // start up the filesystem

while (1){ // display options until user exits

Options(); //user option

printf("Enter user option: ");

scanf("%s", str); // scan input

Choice(str); // calls the appropriate filesystem method

// option to use file system

void Options()

printf("Here are your options:\n");

-Create

-Delete

-Read

-Write

-mkdir

-List

-Open

-Exit

-Close

// choice for user to select

void Choice(char str[]){

char fileName[11]; // hold user input to select from Option method

int firstBlock; // first block of file to open . start at 11 because

if (strcmp(str, "create") == 0){ // to create file

printf("Enter the name of to file to create: ");

scanf("%s", fileName);

called create() method

else if (strcmp(str, "delete") == 0) // to delete files

printf("Enter the name of to file to delete: ");

scanf("%12s", fileName);

called delete(fileName) method and pass in file to delete;

else if (strcmp(str, "read") == 0){ // read data blocks of a file

printf("Enter the name of file to read: ");

scanf("%12s", fileName);

called read(fileName) method and pass in file to read ;

else if (strcmp(str, "write") == 0){

char \*dataBlock = malloc(32768, sizeof(char)); // create an empty larger array

printf("Enter the name of file to write: ");

scanf("% 12s", fileName);

printf("--- Write --- \n");

if(read(STDIN\_FILENO, dataBlock, 32768) < 0){

write(fileName, dataBlock); // write to file

// free data in block

else if (strcmp(str, "mkdir") == 0){ // create a directory

printf("Enter the name of directory to make: ");

scanf("%s", fileName);

create(filename) // create directory

else if (strcmp(str, "list") == 0){

printf("List constent of current directory")

list();

else if(strcmp(str , "open") == 0){ // open file

printf("Enter the file name to open: ");

scanf("%12s", fileName);

else if(strcmp(str , "exit") == 0){ // exit program

exit();

exit(0);

else if (strcmp(str, "close") == 0){ // close file

close();

printf("Successfully close the file \n");

else {

printf("Invalid Command\n"); // invalid input

**Filesystem.c: 6 IOCS function and allocate space, FAT**

#define BLOCK\_SIZE 512 //block size in bytes

#define FILE\_MAX 64 // file max size

#define START\_OF\_FAT 0 //begin fat

#define START\_OF\_ROOT 8 // File Allocation Table

#define START\_OF\_DATA 10 //Start data at 10

#define FAT\_SIZE 2048 //total size of FAT

#define DIR\_TABLE\_SIZE 32 // total size of table

#define DATA\_SIZE 1048576 //total size of data in bytes

struct FAT{

uint16\_t next; //2 bytes

};

struct Entry{

char \*name; //11 bytes

uint16\_t create\_time; //2 bytes

uint16\_t last\_access; //2 bytes

uint16\_t modified\_time; //2 bytes

uint16\_t start\_blocks; //2 bytes

uint32\_t size; //4 bytes

}

//create a file

void create(char \*filename){

char \*f1 // store file name

if((filename= (f1){ file is exist

printf("file is already exist\n");

return;

}

else ( create a file entry )

//allocate space for file

//update that space in block

// if it a directory

//create a new directory entry with the following content:

-name

-creation time

-last access

-block size

// update currentDir to indicate we are in the directory after creation

//update directory table, FAT, and data

}

//delete a file

void delete(char \*filename){

char deleteName; //store name that matches user input

if((deleteName != (filename)) ){

printf("File cannot be found\n");

return;

}

else( remove entry from directory structure table

//replace entry in directory table with empty entry

// update space in directory table

//free block entry

}

//read a file

void read(char \*filename){

int blockStart = 0; // initial block

int blockEnnd // last block

char readFile

char buffer[512] // size of file

//index in FAT of the file

blockStart = (fatIndex – (start of data = 10)) x 512

//calculate the total size reading

blockEnd = blockStart + 512;

// file does exist

if((realdfile = filename))

// start read a file

//read block in file one by one

for(int i = blockStart ; i < blockEnd ; i++ , j++){

memcpy(&buffer[j] , sizeof(char));//read bytes to the buffer of size 512

printf("%c" , buffer[j]); // print character of each data reading

//update block is reading

else{ // file doesn’t exist

printf("File cannot be found\n");

return;

}

// write a file

void write(char \*filename , char \*writeData){

int dataSize // size data written

int blockStart = 0; // initial block

int blockEnd // last block

// length of written data

dataSize = strlen(writeData);

if(( finename is not found){ //search for exist file to write,

printf("File cannot be found\n");

return

else{// start written

blockStart = (freeBlock - START\_OF\_DATA) \* 512; // start point

blockEnd = blockStart + 512; // last point

for(int i = blockStart ; i < blockEnd ; i++ , size++){ //continue to write until the end of the string

memcpy( &writeData[size] , sizeof(char)); // copy each block of data into written file

}

dataSize -= size; // decrement size

}

//update allocate space

//update meta data

open a file

void open(char \*filename){

char f1; // store pointer to matches file name

if( filename = f1){

printf(“success open file)

else{

printf(“file can not be found”)

close a file

void fs\_close(){

set current directory = 0; closes a file descriptor

, so that it no longer refers to any file and may be

reused

}

exit the program

void exit(){

printf("File System is exit \n");

}

**Testing**

// Ask the drive user want to use

Enter the drive: (Drive2MB , Drive3MB ,Drive5MB , Drive10MB):

//for example : user select Drive2MB

//next Display a list of IOCS options

Here are your options:

-Create

-Delete

-Read

-Write

-mkdir

-List

-Open

-Exit

-Close

//For example : user select create

//ask to Enter the name of file to create: hello.c

//if create successfully: print success create file

//else print: can’t create file because of matching name or out of space

//For example : user select delete

//ask to Enter the name of file to delete: hello.c

//if delete successfully: print success delete file

//else printf (can’t delete hello.c)

//For example : user select read

//ask to Enter the name of file to read: hello.c

//If user hadn’t written in hello.c, then a black line appear on display

// else record every single character that user written in hello.c

//For example : user select write

//ask to Enter the name of file to write: hello.c

//if filename can’t found , print “can’t find hello.c to write”

//else user can start write to hello.c

//For example : user select mkdir

//ask to Enter the name of directory to make: dir1

//if success, printf “successfully create directory dir1”

else printf(can’t create directory )

//For example : user select list

//list the content of current directory

for example: hello.c

//For example : user select open

//ask to Enter the name of file to open: hello.c

//if success, printf “successfully open hello.c”

else printf(can’t open hello.c)

//For example : user select exit

//printf (file system is exit);

//exit the program

//For example : user select close

//ask to Enter the name of file to close: hello.c

//if success, printf “successfully close hello.c”)

else printf(can’t close hello.c)