Project 4 Report

Name: Hannah Reinbolt
Date: 4-17-2021
Assignment: Project 4
Class: CS7500

1.1 Project and Design of cpmFS

The cpmFS system project design is function-oriented, including a wide range of functions that help run the command line functions. These functions are broken up into three main files; cpmfsys, which contains the main command line management functionality; diskSimulator, which contains the main disk functionality; and fsysdriver, which contains the main program that calls both the cpmfsys and diskSimulator functions. The program synchronizes with global variables located in the main cpmfsys and diskSimulator files.

The diskSimulator focuses on reading, writing, and printing blocks stored in the system as the main disk as well as writing and reading images input by the user.
The cpmfsys focuses on implementing and manipulating each block for the disk. Each block consists of a 1 byte extent, 8 byte file name, 3 byte file name extension, 1 byte extent number low, 1 byte number of bytes past last sector, 1 byte extent number high, 1 byte number of sectors used in final block, and 16 bytes that hold file data. After creating each block, functionality exists to create, rename, copy, delete, read, write, open, and close files. Several internal functions exist to help sort, check legality of names, and print information for files and more. Below is the design of each function in cpmfsys.

1.11 mkDirStruct
This function is the initialization of a block. Here every part of a block is updated initially according to block0's index.

1.12 writeDirStruct
This function is the copy constructor of a block. A current block is passed to this function and every part of the pointer to block0 is updated with the current block. Once done, the pointer to block0 is written to disk and freed.

1.13 makeFreeList
This function frees up a block on the global "freeblocks" list. Fully clears freeblocks list and reads current block0 pointer. From here it re-creates freeblocks from what is actually free in block0.

1.14 printFreeList
This function prints what is in the global "freeblocks" list. Prints in 16x16 format with "*" meaning full and "." meaning empty.

1.15 findExtentWithName

This function finds the extent number for current name. It makes a copy of the name and extension passed in name, checks both for legality, extracts the extension, updates the name if it is legal and if the extension and name both are then it returns the number of extent.

1.16 checkLegalName

This function checks if name is legal. First it checks if the name exists and first letter is not a number, then checks if it is a space, control character or punctuation. Everything else is valid.

1.17 cpmDir

This function prints the file directory. It updates a new block with block0 and checks for empty blocks. Then it calculates the block length/size and prints the name, extension and size.

1.18 cpmRename

This function modifies an existing file name with a new filename. First it creates a copy of the real filename, then checks the name for validity, calculates the extent number, creates a new directory block and updates that block with the new name.

1.19 cpmDelete

This function deletes a file or block. First it reads block0 and calculates the extent number. Then it sets the status of that block to 0xe5 with means “Unused”. This will signal the program that this block is free. The block is written to the next available block and freed.

1.20 checkName

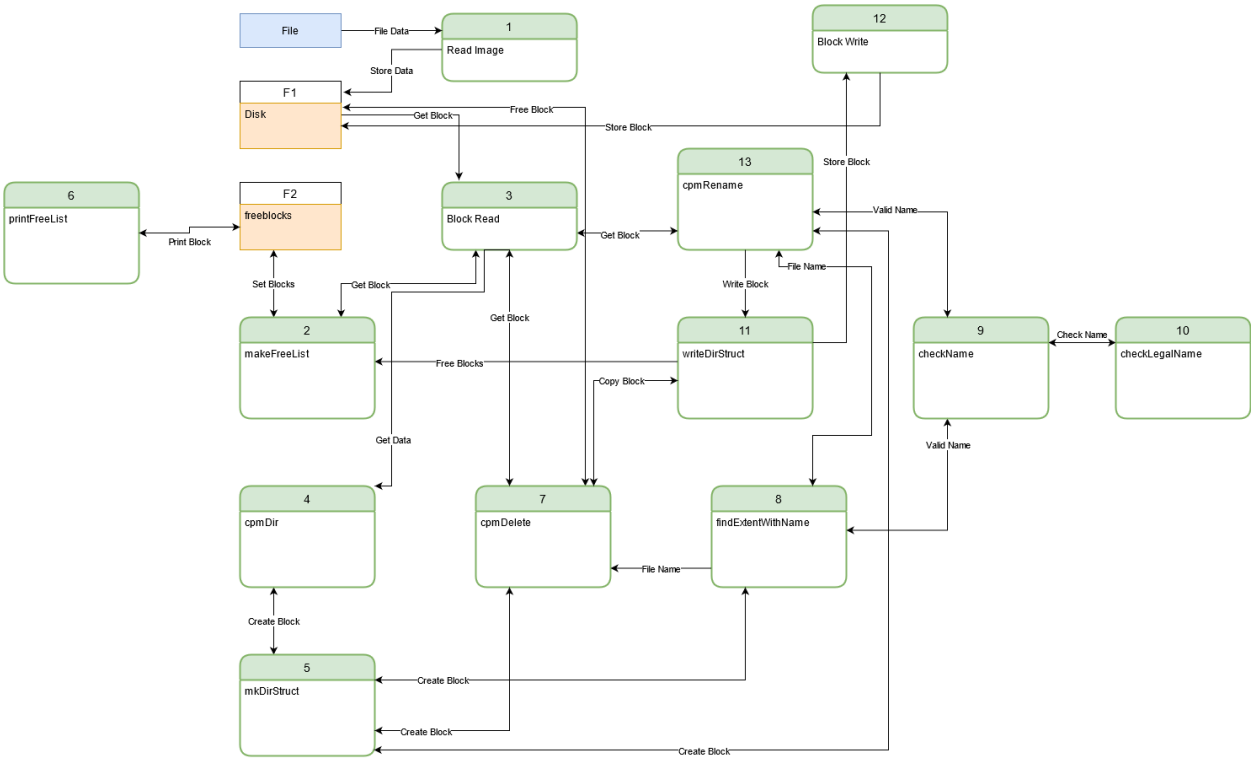
This function is an additional function to help check the validity of a name. First it makes a copy of the new name and separates the extension. Then it empties the rest of the name space with empties as well as the extension. The name is then checked for legality.

The fsysdriver calls the block and disk functionality. Here it will help print out necessary information and end the program. The disk is updated as these functions are called.

Potential design issues arose when first starting to build the system functions. Specifically when to call “BlockRead” and “BlockWrite”. Also how to implement the free block list global variable. After some thought and tinkering both became more clear on how to implement them. Also in order to cut down on code repetition, I built a helper function to help check if names and extensions were valid.

Finally, below is the cpmFS DFD Diagram for this program.

Diagram: DFD Flow Diagram



1.2 Lessons Learned

During the development of this project there were several learning moments. Below are some notes on those lessons.

1. Print statements are your friend. There were a few times I had a segfault in the program and print statements really helped me narrow down what was the issue.

2. Don’t forget to increment counters. A few times I forgot to do this and it messed up a lot. Haha.

3. Read the notes, watch the lectures, do some google searching on different topics, read through all the given code SLOWELY. This will help a lot to start to understand what to do.

4. Don’t procrastinate. This project took me a few days. It could have been worse but it’s still a good thing to remember to dedicate enough time.

These are the big lessons I learned while designing this cpmFS system.