SYSTEM PROGRAMMING

Hw5 Report

Elifnur Kabalcı
1801042617

Main.c

Here, I have stored the data received from the user as an argument in variables. Before starting the actual code, I made the time definitions. To calculate the elapsed time, I used the gettimeofday method as stated in the homework pdf. I called the directory function between the start and end of the period. Since I will go to other transactions from here, all transactions remained within the time value range.

I got some of the codes from the textbook. I made other designs based on it.

Directory.c

Here I check if directory is a private directory (is_Regular). I am checking if the string given as Directory represents a directory (is_directory). I am examining the directory given inside the Directory method and examining the nested directory states. Then I copy. I use the copy_files method when copying. This is also in the copy.c file.

Copy.c

Here I create the number of threads specified in the arguments to copy the files. After creating all of them with the for loop, I join the threads with the for loop again. I used the thread copy method while creating threads. In this method, the file location is copied first with the number of buffers used to copy the files in the source and destination locations. Then the file contents are copied with the while loop.

Makefile

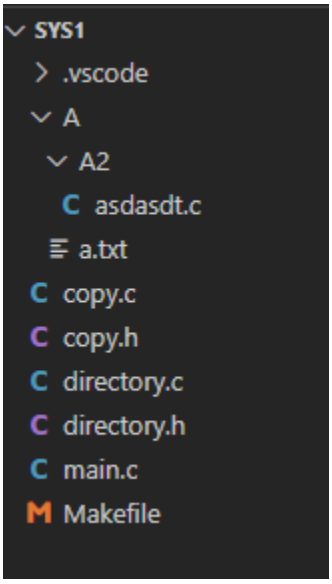
There are a few important points in the Makefile section. One of these points is to add the lpthread flag while compiling the program. Thus, the program can connect to the thread library of POSIX and run correctly. The second point is while main.o , copy.o and directory.o are sufficient for normal c code. It didn't work here at first. I also had to compile pcp.c which was not created by me. Then the problem is solved. The pCp just below the All part is the most important part. It is where the compiled files come together. Compiled files are recompiled and a single producer-consumer problem file is obtained.

```
all: pCp

pCp: main.o copy.o directory.o -lpthread
    gcc main.o copy.o directory.o -lpthread -o pCp

pCp.o: pCp.o
    gcc -c pCp.c

main.o: main.c directory.h copy.h -lpthread
    gcc -c main.c -lpthread

copy.o: copy.c copy.h -lpthread
    gcc -c copy.c -lpthread

directory.o: directory.c directory.h copy.h -lpthread
    gcc -c directory.c -lpthread


clean:
    rm -f *.o pCp
```

Running Screen:

Step by step

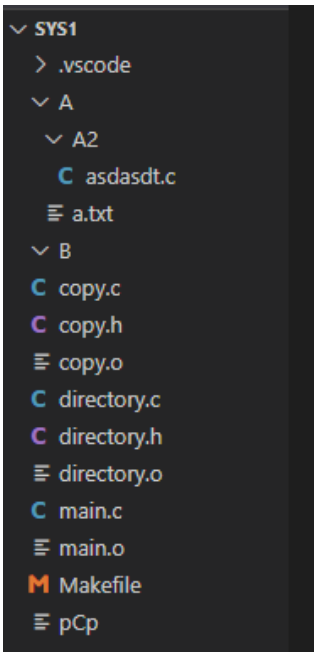
```
latulipenoirez@Elifnur-PC:/mnt/c/Users/e.kabalci2018/Desktop/sys1$ make clean
rm -f *.o pCp
```



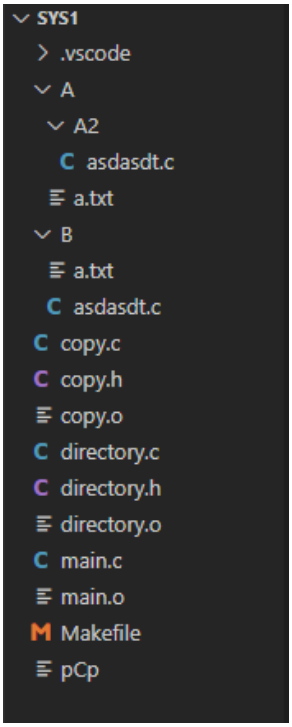
->Initial condition

```
latulipenoirez@Elifnur-PC:/mnt/c/Users/e.kabalci2018/Desktop/sys1$ make
gcc -c main.c -lpthread
gcc -c copy.c -lpthread
gcc -c directory.c -lpthread
gcc main.o copy.o directory.o -lpthread -o pCp
```

After Compile:



After Run:



Output:

The program browses and collects all nodes in the first of the given directories and copies them to the second shown location. Files and their contents are also copied.

While copying according to the number of threads and buffer size given as parameters, I printed how much data the thread copied. In the meantime, because the data to be copied was not finished, it reused the threads from 1 to the number of threads.

The operating time in microseconds is printed on the screen. Apart from that, I wrote the screen press only for error situations.