Project #3

(A shared Protected Circular Queue and Communication between threads)

kyuminlee@ou.edu

Kyumin Lee
University of Oklahoma
Norman, OK

***Abstract*—The project implements a producer-consumer problem using a circular buffer, pthreads, and semaphores in C. Its objective is to synchronize two threads – a producer that reads characters from a file and a consumer that prints these characters to the screen – while managing access to a shared circular buffer.**

Keywords—Semaphores, circular buffer, thread, producer, consumer

I. To solve the problem

I created a circular buffer with 15 positions, each storing one character. The circular functionality is managed by 'in' and 'out' pointers. Two threads were synchronized. Producer thread reads from 'mytest.dat' and writes to the buffer. Consumer thread reads from the buffer and prints characters, with a one second delay between reads. Three semaphores controlled buffer access and synchronization. 'empty' was the count of empty buffer slots. 'full' was the count of full buffer slots. And 'mutex' ensured mutual exclusion for buffer access. The producer signaled the end of the file to the consumer using a special character. The program needs to be compiled using 'gcc' with the '-lpthread' and '-lrt' flags for linking pthread and real-time libraries.

II. Results

I've created my own mytest.dat file to just check if it would output correctly.
Content of my own mytest.dat:

```
Hello, World!
My name is Kyumin Lee.
This is a test file.
```

Output of the program:

```
lee0436@gpel8:~/project3$ ./project3
Hello, World!
My name is Kyumin Lee.
This is a test file.
```

III. Conclusion

The project successfully demonstrated the use of semaphores and threads to manage a shared resource in a concurrent environment. The implementation of the circular buffer and the synchronization between the producer and consumer threads worked as expected. The one-second delay in the consumer thread effectively illustrated the control of execution speed and thread synchronization.