CS474 Operating Systems Project #2

Michael Smith

*Computer Science Dept, NMSU*

p173939@nmsu.edu

Abstract-This document describes the process needed to ensure mutually excluded access to a shared buffer and safe access to such buffer.

Introduction

A program written in the C Language “buffer.c” was written to accomplish the goal of demonstrating how a shared buffer, when protected adequately with mutual exclusion using semaphores, can be access safely with different threads.

Program

“buffer.c” first initializes three semaphores, “full”, “empty” and “mutex”. Then it creates 2 threads, the producer and the consumer. These threads are then joined so that the parent process that create them will only continue when both of the created threads have finished. A shared memory buffer “buffer” with 15 slots is also created and shared between the two threads. The consumer thread immediately goes into waiting but the producer thread starts reading from file “mydata.dat” character by character and inserts them into the shared buffer until the buffer is full the it signals the “full” and “mutex” semaphores and goes into waiting. The consumer then takes over and prints each character in the buffer until it is empty then signals “empty” and “mutex” and goes into waiting. The process continues until the producer reads EOF and writes it to the buffer then signals “full” and “mutex” and the thread ends. The consumer then prints out the last remaining characters and then ends itself. Control is then granted back to the parent process and it releases the semaphores and the shared buffer.

Results

The program will print any string of characters, 15 at a time, one for each line until the end of file is reached. Each character printed will only print in increments of 1 second due to a sleep(1) present between each print in the consumer thread. An example output for a file containing the ABCs is listed below:

writing A to buffer

writing B to buffer

writing C to buffer

writing D to buffer

writing E to buffer

writing F to buffer

writing G to buffer

writing H to buffer

writing I to buffer

writing J to buffer

writing K to buffer

writing L to buffer

writing M to buffer

writing N to buffer

writing O to buffer

Buffer full transferring to consumer

Reading A from buffer

Reading B from buffer

Reading C from buffer

Reading D from buffer

Reading E from buffer

Reading F from buffer

Reading G from buffer

Reading H from buffer

Reading I from buffer

Reading J from buffer

Reading K from buffer

Reading L from buffer

Reading M from buffer

Reading N from buffer

Reading O from buffer

Buffer read, transferring control to producer

writing P to buffer

writing Q to buffer

writing R to buffer

writing S to buffer

writing T to buffer

writing U to buffer

writing V to buffer

writing W to buffer

writing X to buffer

writing Y to buffer

writing Z to buffer

File completely read, wrote EOF and transferred control to consumer

Reading P from buffer

Reading Q from buffer

Reading R from buffer

Reading S from buffer

Reading T from buffer

Reading U from buffer

Reading V from buffer

Reading W from buffer

Reading X from buffer

Reading Y from buffer

Reading Z from buffer

read EOF, exiting

It is important to note that the producer does not 1 second between each of its writes to the buffer and to the user it will be instantaneous. There is a limit of 150 characters according to the assignment an example with 150 characters is given below: