CSCI 360 - Lab 3 Report

Sami Al-Qusus
Oct 2, 2018

Assignment Goals:

- Learning about interprocess communications with shared memory.

Assignment Instructions (Dr. LoPinto):

- Create two processes - one for a sender and one for a receiver.
- Create a shared memory segment of 101 bytes. Use the first byte as a flag for synchronization and the remaining 100 bytes to carry data from the sender to the receiver.
    - The sender should busy-wait on the flag. When the flag is in the proper state, the sender writes text into the shared buffer.
    - The receiver should also busy-wait on the flag. When the flag is in its proper the receiver should print get the message from shared memory and print it.
    - Both the sender and receiver should run in a loop until terminated by the user. You can refer to this example sender to see how ftok, shmget, and shmat system calls are done. You should describe these calls in your PDF report and explain what permissions you used for shmget in the sender and the receiver. Also you should explain what ipcs and ipcrm do.

What I did:

- I first searched up all the unfamiliar terms like how ftok, shmget, and shmat. I then looked at a simple example using them.
- I then attempted to run the example code provided to us and went through it line by line to understand what every line does so I can build on top of it for lab3.
- I had couldn't get it to work for me because of this flag; IPC_EXCL
    - Its used along with IPC_CREATE to create a new segment and fails if the segment already exists. So because I had created the segment before I couldn't proceed because it returns -1. Once I took it out it worked. Though I then found

out that I can just destroy the segment before calling shmget or use ipcrm!

- Once I had that figured out, I put the code in two files, a receiver and a sender.
    - Once both get the shared memory location they go into a loop until the user terminates them.
    - At the begging of the loop they both check who has the right of go with an if statement. Whoever has right of way will do get access to shared memory and the other will busy-wait. That is keep checking until the flag changed by the other.
    - When the sender gets the access, it takes a message from user, changes the flag and puts message on the buffers.
    - Then sender goes into busy-wait and receiver reads what is on the buffers and changes flag to give the sender right of way.

What I learnt:

- ftok - generates an IPC key, because if we use a random number as a key, we might get a clash with some other unrelated program.
- IPC stands for inter-related process communication.
- shmget()-Creates the shared memory segment or use an already created shared memory segment.
    - IPC_CREAT: flag for creating new segment
    - IPC_EXCL (Used with IPC_CREAT to create new segment and the call fails, if the segment already exists
    - Call would return a valid shared memory identifier (used for further calls of shared memory) on success and -1 in case of failure.
    - S_IRUSR | S_IWUSR
        - These are the user read and write permissions.
    -
    -
- shmat()-Attaches the process to the already created shared memory segment.
    - We can use shmdt() to Detach the process from the already attached shared memory segment.
- shmctl()- destroys the shared memory.
- ipcrm - removes the shared memory segment associated by an id. Can also remove queues or semaphores set.
- ipcs- command to find the identifiers and keys.

Credits:

MODERN OPERATING SYSTEMS

- Dr. LoPinto lab3 example.
- https://www.geeksforgeeks.org/ipc-shared-memory/
- https://www.tutorialspoint.com/inter_process_communication/inter_process_communication_shared_memory.htm
- https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.bpxa500/ipcrm.htm