Shared Memory Project 1

Karshan Arjun
COP4600: Operating Systems
University of South Florida, USA

Abstract

The purpose of the this project is to give a detail look into the use of processes that are creating by the CPU and then is managed by the kernel. The use of shared memory is also explained in the report.

Design

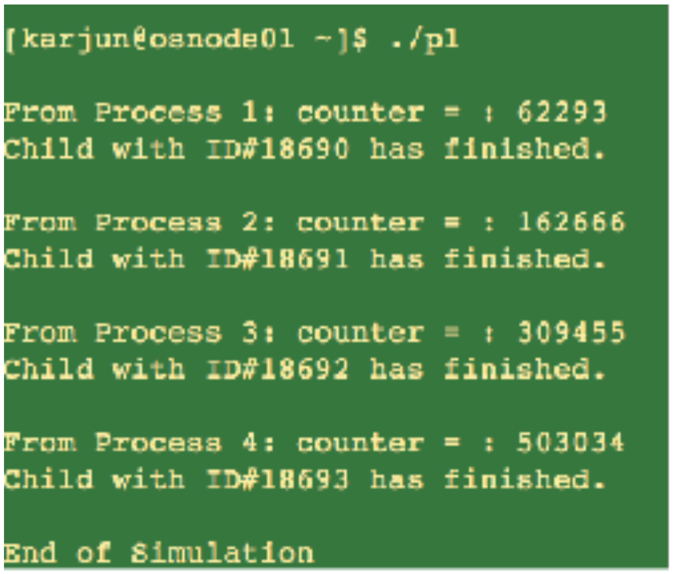
The design of the Shared memory program revolved around using processes to do the work of containing increasing the value of a struct which is shared among the the four processes. Four process functions were made. The processes 1-4 had a values of 100,000, 200,000, 300,000 and 500,000 respectively.

In the main function we create a tunnel to get the a shared memory segment by calling shmget function. And allocating the struct with shmat function. After the setup for gathering the shared memory is over, each of the process id will be initialized with fork function and then process itself will be ran. Once the process id is now a value and not 0, then the next process will be ran.

Once the all the processes are ran, we wait for the waitpid function to not return a -1, this lets us know that the process is over. We then release the the shared memory with function with the shmclt function and end the simulation of shared memory.

Results

Running the program will first go through each process in a upon call order. Then will end the child process from the parent.



```
[karjun@osnode01 ~]$ ./p1

From Process 1: counter = : 62293
Child with ID#18690 has finished.

From Process 2: counter = : 162666
Child with ID#18691 has finished.

From Process 3: counter = : 309455
Child with ID#18692 has finished.

From Process 4: counter = : 503034
Child with ID#18693 has finished.

End of Simulation
```

Performance

The performance of the program was done by using the time unix command to get the time taken to run the program on the system and user, which relates to the kernel mode and the user mode. The user mode gave a 0m0.012s to run and the kernel mode called sys took 0m0.002s to be ran.