COP 4600-002: Operating Systems
Fall 2017

Project #1: Shared Memory

Alan Rodriguez, U86831061

I. INTRODUCTION

FOR this project 4 processes were created. These processes modify a shared variable "total". The processes increment until their respective bounds, 100,000, 200,000, 300,000 and 500,000.
Child processes are created and modify the shared variable as well. The parent process does not terminate until all child processes have exited. This is accomplished by using the waitpid function.
Once all the parent processes have terminated upon completion, the shared memory must be detached. This is done by passing our shared variable to the function call, shmdt(total).
Now that shared memory is detached, it can be removed with shmctl(shmid, IPC_RMID, NULL); This ensures that shared memory is removed and there is no memory leakage.
After running the program several times, my results were nearly identical.

II. RESULTS

From Process 1: counter = 100000
From Process 2: counter = 300000
From Process 3: counter = 600000
Child 96 pid has just exited.
Child 97 pid has just exited.
Child 98 pid has just exited.
From Process 4: counter = 1100000
Child 99 pid has just exited.
End of Program.

A second trial of the program,
From Process 1: counter = 100000
From Process 2: counter = 300000
From Process 3: counter = 600000
Child 107 pid has just exited.
Child 108 pid has just exited.
From Process 4: counter = 1100000
Child 109 pid has just exited.
Child 110 pid has just exited.
End of Program.

I decided to increase the number being incremented for each process by a factor of 10 and then noticed a change in the results.

From Process 1: counter = 1003267
Child 142 pid has just exited.
From Process 2: counter = 2930444
Child 143 pid has just exited.
From Process 3: counter = 4249931
Child 144 pid has just exited.
From Process 4: counter = 7185701
Child 145 pid has just exited.
End of Program.

Another simulation,
From Process 1: counter = 1006044
Child 147 pid has just exited.
From Process 2: counter = 2693580
Child 148 pid has just exited.
From Process 3: counter = 4785531
Child 149 pid has just exited.
From Process 4: counter = 6847721
Child 150 pid has just exited.
End of Program.

III. CONCLUSION:

From this data, I was able to conclude that the program would never produce similar results. The results of the counter always vary which implies that the processes are not modifying the shared memory properly. The value of total is accessed by the child processes and increments the variable, but at the same time other processes access that variable and the value is not properly incremented at that time, thus the counter is never accurate and does not work as it was intended to.