Lab Two Report

Paul Clark

Kevin Conyers

June 11, 2019

Part One

The objective of this lab is to explore the Data Shared problem between two processes. To do this we were provided with code that instantiates a parent process and a child process. These two processes share a variable called countptr. Theses two processes are supposed to increment the variable by different values, the parent process is supposed to increment by twenty each time it runs, while the child process is supposed to increment by two each time it runs. So if the parent process is called twice, and the child is called 3 times, the final value is should be forty-six.

Question One

To test if this is the case, we compile the code in lab2-1.c and run it. If the code is correct we should see that countptr ends up being equal to the number of times parent process is called multiplied by twenty plus the number of times child process runs multiplied by two. The following is the output from a run of the given code:

```
Child process –>>counter= 2
Child process –>>counter= 4
Child process –>>counter= 6
Child process –>>counter= 8
Child process –>>counter= 10
Child process –>>counter= 12
Child process –>>counter= 14
Child process –>>counter= 16
Child process –>>counter= 18
Parent process –>>counter = 20
Child process –>>counter= 20
Child process –>>counter= 22
Child process –>>counter= 24
Child process –>>counter= 26
Child process –>>counter= 28
Child process –>>counter= 30
Child process –>>counter= 32
Child process –>>counter= 34
Child process –>>counter= 36
Child process –>>counter= 38
Child process –>>counter= 40
Parent process –>>counter = 40
Child process –>>counter= 42
Child process –>>counter= 44
Child process –>>counter= 46
Child process –>>counter= 48
Child process –>>counter= 50
Parent process –>>counter = 60
```

So the child process is called twenty-five times and the parent process is called three times. However, counter's final value is sixty, instead of 110 as would be expected. So clearly this code is not working as intended.

Question Two

The variable countptr is most certainly not a shared variable. The changes made by Child Process are not visible by the parent, or vice versa.

Part Two

In part two we were tasked with modifying lab2-2.c so that Peterson's solution was properly implemented.

Question One

The counter is visble to both the child an parent process.

Question Two

Clearly this is not functioning as intended, as the counter is not increasing by on two's and twenty's Instead the counter is increasing unpredictably because access to it is not controlled.

Code Report

The code that I wrote for this lab works as intended. Each process alternative increments to pointer from it's previous by the amount that it should. The current limit is 50, so as soon as the counter value is 50 or more, the processes exit. Currently, that means that the final value reached is 64 because if the process reaches the critical section, it first checks the current value of counter. When parent process enters it's critical zone when counter equals 44, it increments by 20. Then child process starts again and sees that counter is 64 and exits.

Possible Issues

It is possible that synchronization is not actually enforced. This is due to the way C handles mmap files.