Gavin Le - U54163659
Aung Khaing - U43209795
February 3, 2020

Operating System - COP4600 - Spring 2020

Project-1 Report: System Calls and Context Switch Measurements

Introduction

Most of every one now owns a laptop or computer. However, not many people know or wonder how fast or effective their computer can be.

In this project, we run a simple test to measure the cost of system calls and the cost of context switch of the processor. In order to achieve the most correct and accurate result, we run the test with a large number of iterations. Moreover, we measure the runtime with function clock_gettime(), which is also a system call that allows us to get time in nanosecond.

For system calls, we repeatedly call a system call function gettimeofday() with a for loop. Firstly, we run clock_getime() function to capture the time at that moment then store into timespec type variable as the starting point. Then, we run the loop for a large number of iterations. After the loop is done, we run clock_gettime() function again to capture the time and store as the ending point. Then, we convert the time to nanosecond and subtract the ending time by the starting time to get the runtime of the loop. Finally, we divided runtime by the number of iterations to get the average cost of gettimeofday() system call.

For context switch, this test is trickier than system call test. First, we create two ends parent and child with read and write ability. Then, we connect two ends together by running a system called pipe(parent) and pipe(child), and we run fork() system call and check if pipe is successful or not, if it's not the program will exit. After that, we start the context switch by sending and receiving messages between parent and child processors by reading and writing. If the PID is not 0, we start the parent process. To write the message from parent to child, we close the parent processor reading end, write the message using write() system-call, then close the writing end of parent processor, on the other hand, to read the message from child to parent, we close parent

reading end, then read the message using read() system-call function, after that, we close reading end of parent. Same for a child processor when PID is 0. We run these processes for a large number of iterations, and measure the runtime the same way as we did for the system call test. Finally, we divide the runtime by the number of iterations to find the cost of context switch.

Approach

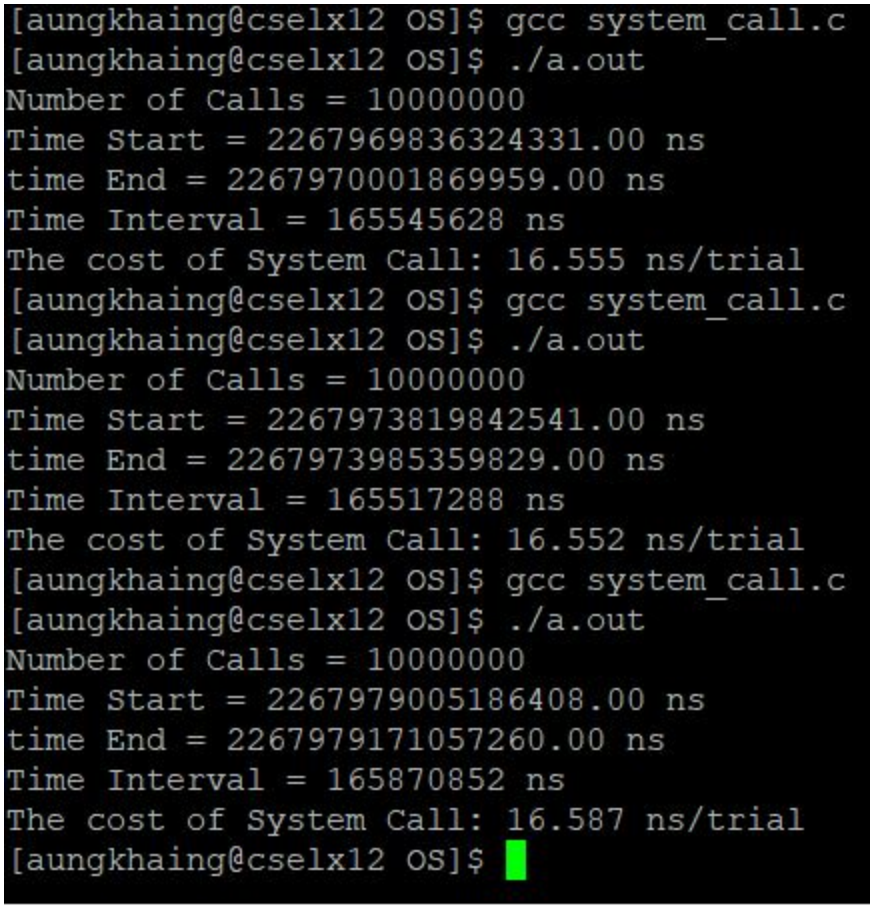
The approach we took is to make the program simple and clear. To measure the cost of system call we run a loop of 10000000 trials for a system call (gettimeofday(&temp, NULL)). Then we use clock_gettime(CLOCK_MONOTONIC &start) to record the start time and end time of the loop, and calculate the elapsed time by subtracting the recorded end time with the start time. Then divide the time interval by the number of trials to get the cost of system call.

Approach for context switch is the same as system call, simple and clear. To measure context switch we took the fork, read, write, pipe approach. Create two string messages, one for the parent process and one for the child, which will be read and written in parent and child array made for piping. We then pipe the two processes, and assign pid_t to fork() to create a new process.

Gavin Le - U54163659
Aung Khaing - U43209795
February 3, 2020

Output

System call: We display the number of iterations/calls of system call, the start time and end time recorded with clock_gettime() in nanoseconds. Then the subtracted elapsed time followed by the cost of system call. We tested the program multiple times and the results are very accurate, about +/-0.03 nanoseconds margin of error.



Gavin Le - U54163659
Aung Khaing - U43209795
February 3, 2020

Context Switch: Below figure shows the result of context switch measurement with two different measurements of the same program running multiple times.

```
[aungkhaing@cselx12 OS]$ ./a.out
Time Interval = 34656705 ns
The cost of Context Switch is 3465.671 ns/trial
Time Interval = 34598750 ns
The cost of Context Switch is 3459.875 ns/trial
[aungkhaing@cselx12 OS]$ ./a.out
Time Interval = 34848265 ns
The cost of Context Switch is 3484.827 ns/trial
Time Interval = 34828516 ns
The cost of Context Switch is 3482.852 ns/trial
[aungkhaing@cselx12 OS]$ ./a.out
Time Interval = 34550828 ns
Time Interval = 34522006 ns
The cost of Context Switch is 3452.201 ns/trial
The cost of Context Switch is 3455.083 ns/trial
[aungkhaing@cselx12 OS]$ ./a.out
Time Interval = 34637205 ns
The cost of Context Switch is 3463.720 ns/trial
Time Interval = 34587790 ns
The cost of Context Switch is 3458.779 ns/trial
[aungkhaing@cselx12 OS]$
```

Obstacle

While working on this project, we ran into some problems that took us quite a lot of time to figure out. At the beginning, we tried to use gettimeofday() to measure the time, then multiplied it by 1 billion to get nanosecond. However, we realize that, we accidently round up the runtime which caused the result wasn't accurate. Therefore, we change it to clock_gettime() to be able to measure runtime in nanosecond.

On the other hand, for the context switch, after running it we received two different results. We assume that this problem happens because the test requires two processors, so it is the same as running the program twice for parent and child.

Gavin Le - U54163659
Aung Khaing - U43209795
February 3, 2020

Conclusion

In conclusion, we were able to finish the project within the given time for this project. It took us about 3-4days to finish the project. Through this project, we understand what a system call is and how to use some of the system calls, for example, clock_gettime(), gettimeofday(), fork(), pipe(), etc. However, the result is just an estimation. There are some substances that may cause inaccurate results, such as the runtime of for-loop.