Abdulrahman Aldhubaiban
Tracy Jackson Jr.
Feb 1, 2019

OS Project 1 Report

Introduction

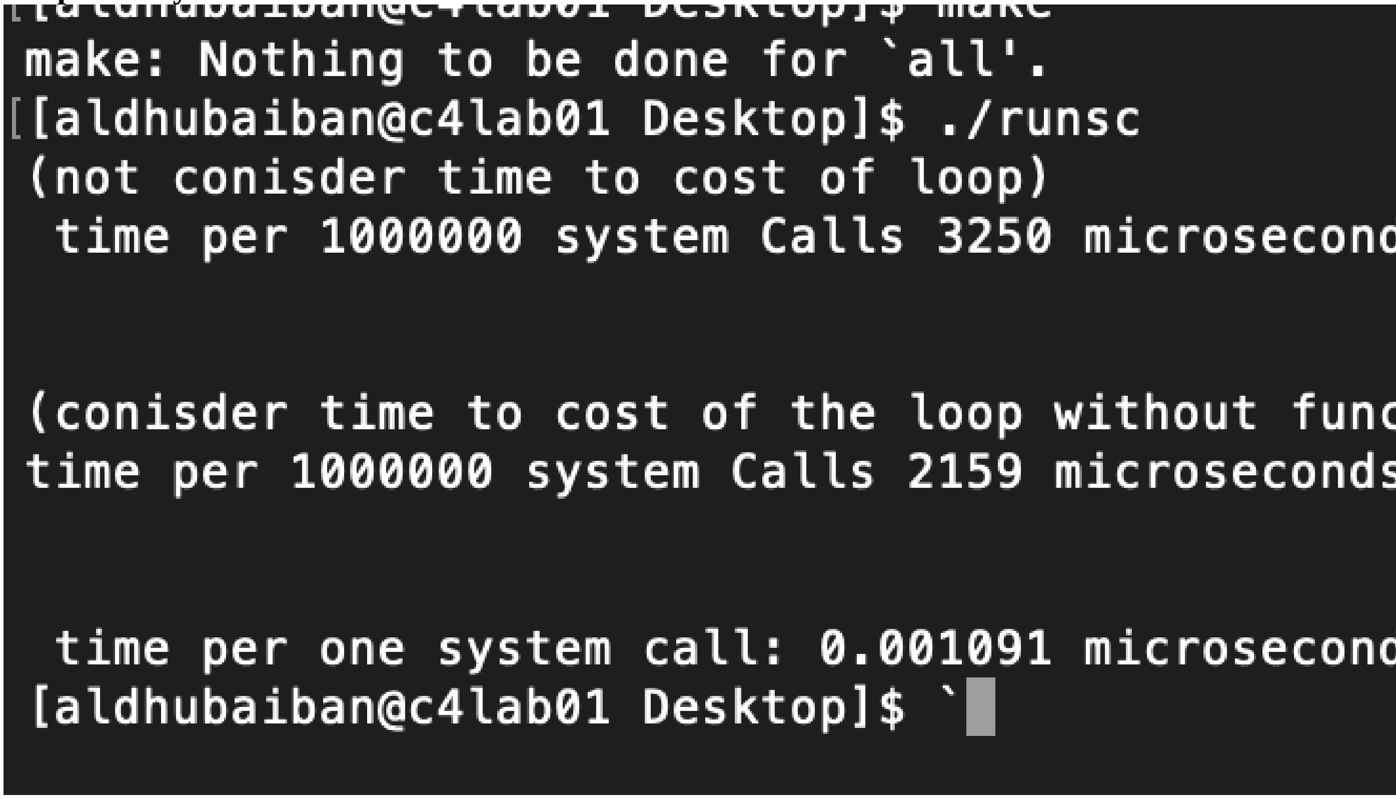
For Project 1, we have two C files we need to create, which is the system_call.c and context_switch.c. The system_call.c measure the cost of a system call.The context_switch.c is to measure the cost of context switch between two process under a single CPU.

For this to happen, we must use a timer function such as gettimeofday() in order to measure the time that needs to do a task.

describes your approach

1-Calling all the needed libraries to measure time
2- measure the time before starting the loop that do the needed task
3- we using a loop to get N number of trials
4- after the loop ended we get the ending time
5- then we use a difference between end and start divided by number of N to get average of each trial

Output for system call



```
[[aldhubaiban@c4lab01 Desktop]$ make
make: Nothing to be done for `all'.
[[aldhubaiban@c4lab01 Desktop]$ ./runsc
(not conisder time to cost of loop)
 time per 1000000 system Calls 3250 microsecond


(conisder time to cost of the loop without func
time per 1000000 system Calls 2159 microseconds


 time per one system call: 0.001091 microsecond
[aldhubaiban@c4lab01 Desktop]$ `
```

Approach of the problem 1

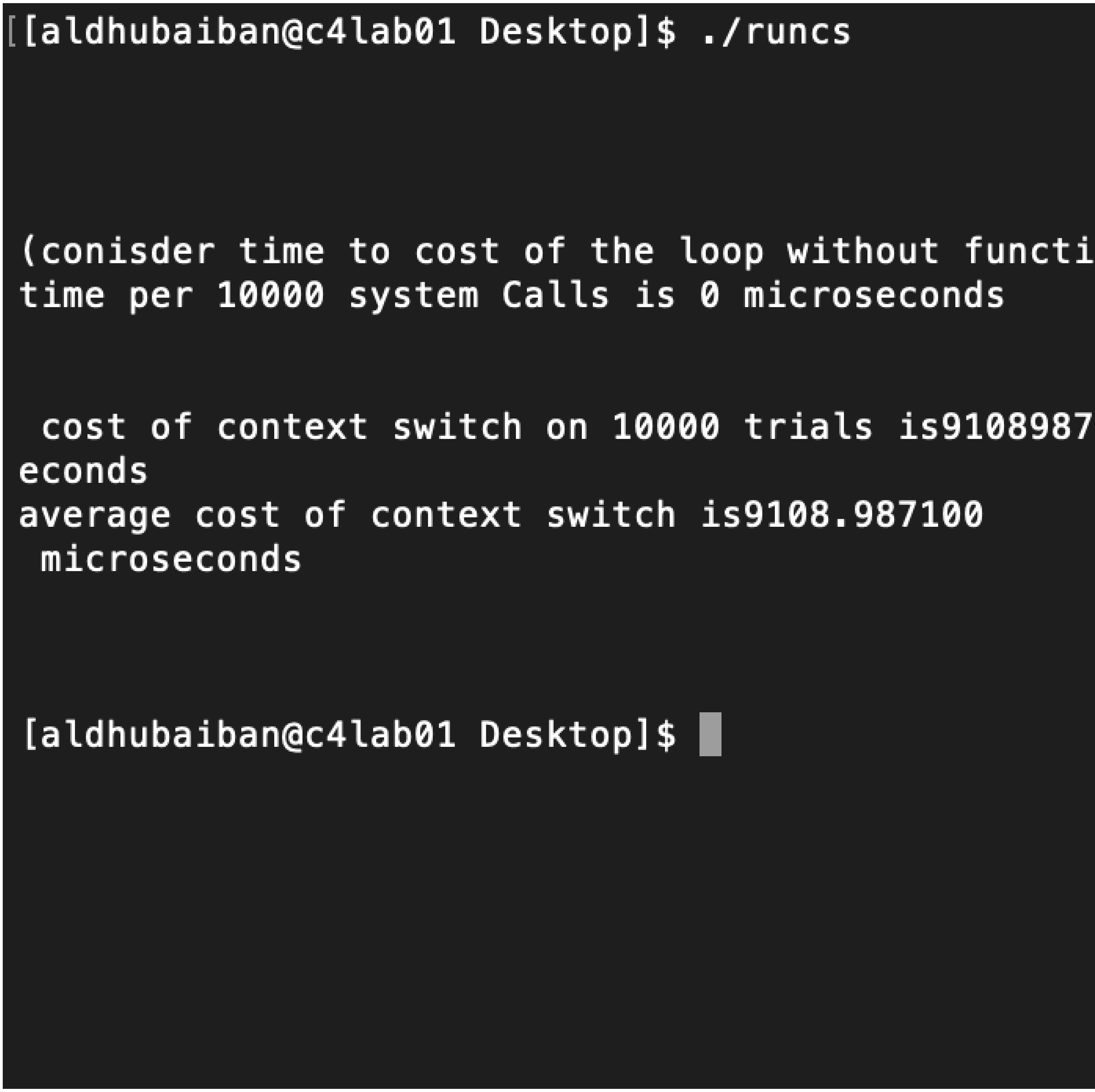
System call used: getpid()

SAMPLE size= 1000000

Empty_loop cost some time so i subtract it from getpid();

A system call per one system = (loop with getpid()- empty_loop)/ SAMPLE size

Output for context_switch



```
[[aldhubaiban@c4lab01 Desktop]$ ./runcs



 (conisder time to cost of the loop without functi
 time per 10000 system Calls is 0 microseconds


  cost of context switch on 10000 trials is9108987
 econds
 average cost of context switch is9108.987100
  microseconds



 [aldhubaiban@c4lab01 Desktop]$ 
```

Approach of the problem 2

Function used: read and write through pipe

SAMPLE size= 10000

Empty_loop cost less than 1 microsecond for 10000 sample. So i didn’t consider it to be an issue.
We measured our time inside the loop therefore i had to store all the values of each trial inside of an array then i had to use another loop in order to get the values in one variable.

Limitation and challenges

The limitation of our project is that we wasn’t getting the most efficient time from our input sample size loop, so we had to make it larger. The challenge we had was that multiple times we were inputting numbers, our outputs were displaying zeros and our last challenge was we had to declare an array to get trials of all the numbers inside the for loop. So we create second for loop to add all the times inside the loop.

Conclusion

In conclusion, we had to read the book to understand each part of the project. The most important part of this project was that we had to run the program multiple times to get the most accurate time. Understanding what each process does was the way to get the full description on how the context switch is supposes to run.