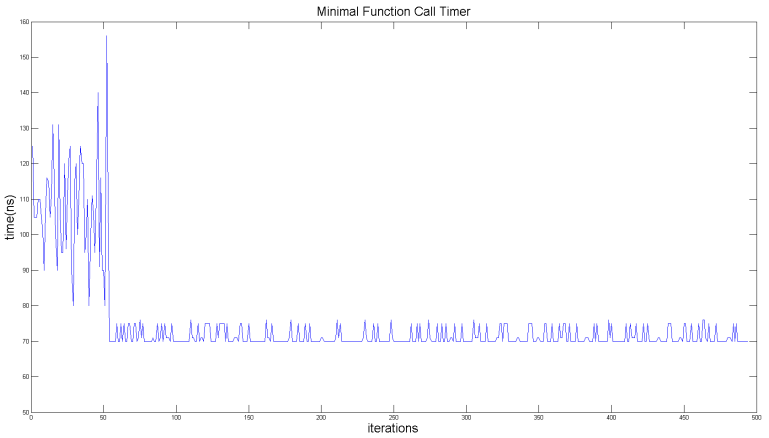
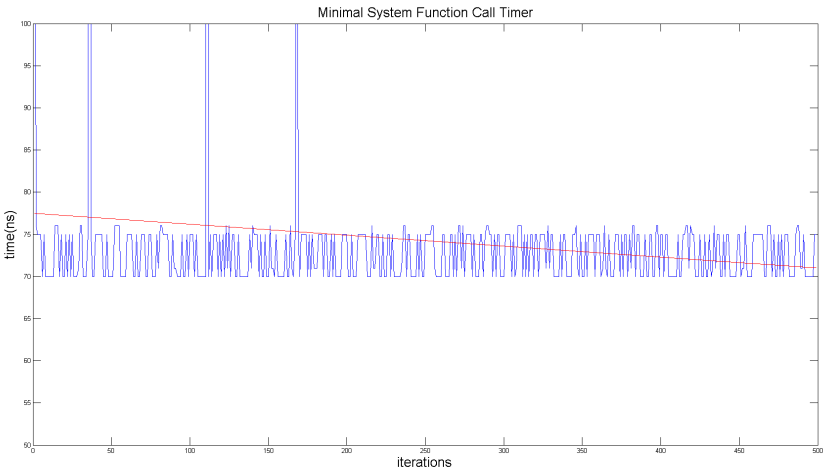
CMPT300 Assignment 2 Report

by Ka Shun (Jason) Chan
ksc19@sfu.ca

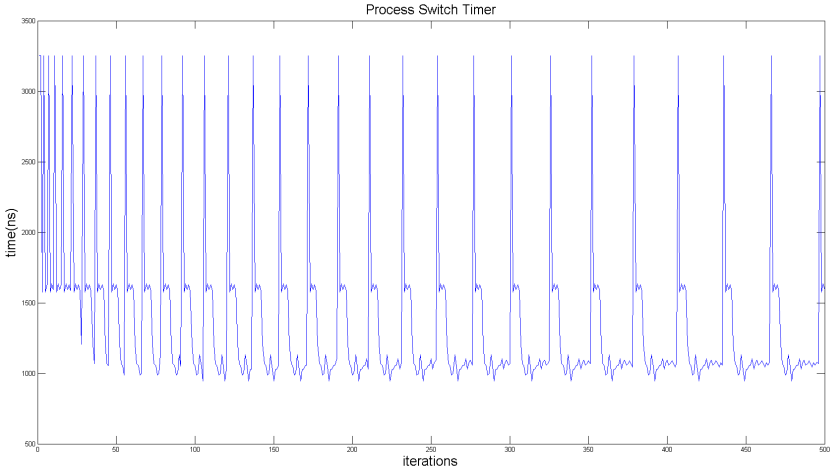
2) Minimal function call is found to cost around 70ns after running 500 iterations. As the graph has shown, the first 50 iterations cost significantly more time. This is due to the operating system loading the function call into the CPU cache and thus speeding up access over time. The 70ns speed is close to the maximum rate of processors executing instructions.



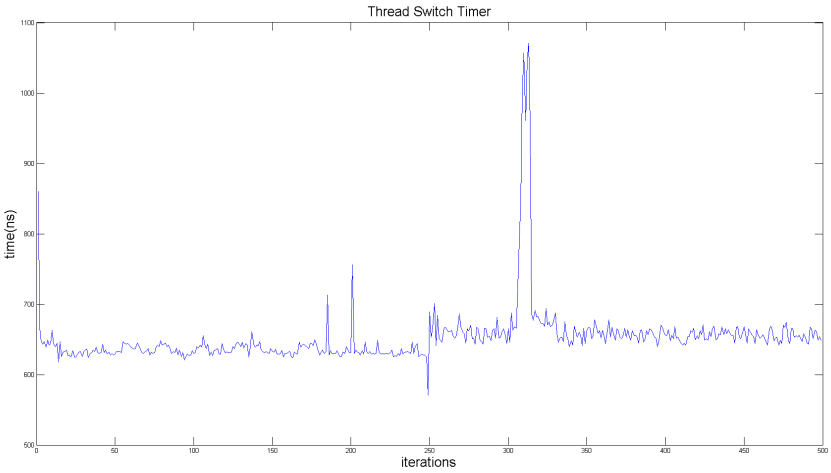


3) getpid() is used to find the minimal cost of a system call. Other than a few spikes, the cost of system call is 70 – 75ns throughout iteration 1 to 500. Unlike function call, system call does not require caching in kernel space thus there is no stabilization period.

4) A pipe that writes and reads a 1 byte message is implemented in order to measure the time it takes to switch a process. CPU affinity is set manually to ensure the process runs in a single core environment. The time is obtained by measure the putting the clock where parent and child processes read/write and divide the time by 2 (because 2 process switches happens during that period).



500 iterations are recorded in the execution and the resulting time range from 900 to 3200ns. Process switch cost significantly more than the function calls and system calls because of the overhead it requires to saving/loading state data into PCB as two processes has different address space



5) Thread switching cost is measured by creating two threads and putting two clocks right after the conditional number (state) changes and the mutex lock in the other thread is unlocked. The resulting time of thread switching after 500 iterations has the range from 600 to 700ns. As expected, thread switching costs a lot less than process switching. The reason is that threads share all resources and there is no need to save/load PCB.