CSCI 247 Computer Systems I

Assignment Exercise 1

Objectives

Gain familiarity with Multithreaded applications. In particular, understand concepts of thread scheduling, priorities and synchronization,

Submitting Your Work

Submit your C program files as the Assignment Exercise 1 Submission item on the course web site. You must submit your program along with an explanation of the results by the deadline in 2 weeks.

Goal

The Linux operating system allows for 3 scheduling policies (SCHED_OTHER, SCHED_FIFO, SCHED_RR) to accommodate the various scheduling needs of applications.

The goal of this assignment is to design a program that will create 9 threads (the design will ensure that future scalability for a higher or lower number of threads are easily achievable).

The first 3 threads will use the SCHED_RR policy, the next 3 will use the SCHED_FIFO policy and the last 3 will use the SCHED_OTHER policy.

Each of these 9 threads will wait for the "main" thread to signal that they should start running a "task". This task should be a function call with non-yielding and non-blocking instructions that take about 5s to 10s to complete. The thread will call this task a certain number of times. You are at liberty to choose how many times this task should be run, but at a minimum you should run it 3 times.

Once a thread executes this task for a predetermined number of times, the thread should output statistics on its run, similar to what is shown below before exiting:

```
Thread# [5] id [0x7f9d0a247700] exiting…
DisplayThreadSchdStats:
    threadID      = 0x7f9d0a247700
    policy        = SCHED_FIFO
```

```
priority        = 62
startTime       = Sun Oct 15 12:29:49 2017
endTime         = Sun Oct 15 12:30:26 2017
Task start TimeStamp  in micro seconds [1508095789066499]
Task end   TimeStamp in micro seconds [1508095801582953]   Delta [12516454]us
Task end   TimeStamp in micro seconds [1508095814150624]   Delta [12567671]us
Task end   TimeStamp in micro seconds [1508095826671818]   Delta [12521194]us
```

Finally you should study the statistics displayed and provide an explanation for the latency seen by each thread.

Assignment Hints

Task 1: Scheduling Policies in Linux

Research the different Scheduling policies available in Linux and understand their applications. This and this are good sites to get familiar with the Linux Operating System.

Task 2: Create a single thread

Use the man pages to see how to create a thread using the "**pthread_create**" API. Create a generic thread function that the created thread can run.

Task 3: Create a structure to store the thread information

Understand the thread parameters that you would need to store to generate the statistics required for this Assignment and create a **typedef structure** to store that data. You can then create an array of this structure for the number of threads you plan to create.

Task 4: Understanding condition variables

Get familiar with the "**pthread_mutex_lock**", "**pthread_cond_wait**" and "**pthread_cond_signal**", to be able schedule your created thread on demand from your main thread.

Task 5: Create a Task function

Create a function that will be non-yielding and non-blocking and will take about 5s to 10s to complete. Just a counter of an unsigned int should serve this purpose and call this function from your thread routine.

Task 6: Learn to use system counters

Being able to use system counters is essential to monitor system performance. Learn to use the "**clock_gettime**" API to get count in micro seconds. Also get familiar with the "**time**" API for counters with less resolution. Use these counters to measure how long your task function is taking. Adapt your thread data structure to include counter values are different points in your code (capture and store counter value at the end of each task run).

Task 7: Understand thread parameters

Get familiar with "**pthread_setschedparam**" and "**pthread_getschedparam**" to set and get thread policies and priorities. Write a function to display thread parameters (including information you have stored about the thread).

Task 8: Increase the number of threads

Increase the number of threads to "9" and set the FIFO, RR and OTHER polices as required in this assignment. Display the counter results and analyze the data you collect. The more details you can provide to explain the data you collect the better. Analysis should include information on the maximum number of threads in given policy that can run simultaneously and why.