CG2271 Real-Time Operating Systems

Tutorial 5

In this tutorial, we are going to cover many of the important aspects of Scheduling and Concurrency.

Q1. Assume three periodic tasks T1 = (3, 9, 9), T2 = (5, 18, 18), and T3 = (4, 12, 12) where the information represents (WCET, Deadline, Period). Demonstrate that a feasible schedule based on fixed priorities exists or prove that one cannot exist.

Q2. Given a set of tasks with the following specifications:

Task	C_i	P_i
T1	2	6
T2	3	8
T3	4	15

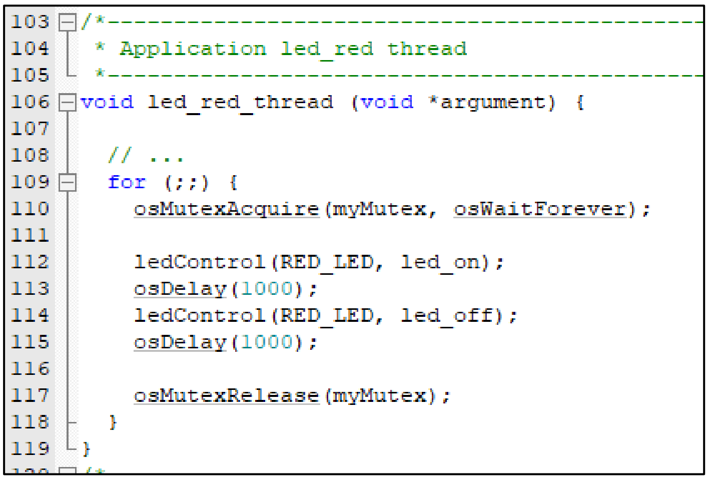
a) What is the CPU utilization of this set of tasks?

b) Is this set of tasks RMS schedulable under the utilization bound criteria?

c) Schedule the tasks using RMS

Q3. In the Lab, you explored the use of Mutexes to control access to the shared resource, the RGB LED.

This is the snippet of the thread with the use of the mutex.



```
/*----------------------------------------------------
 * Application led_red thread
 *----------------------------------------------------
void led_red_thread (void *argument) {

  // ...
  for (;;) {
    osMutexAcquire(myMutex, osWaitForever);

    ledControl(RED_LED, led_on);
    osDelay(1000);
    ledControl(RED_LED, led_off);
    osDelay(1000);

    osMutexRelease(myMutex);
  }
}
```

The second parameter for osMutexAcquire() is a timeout value which is currently set to osWaitForever. What if that value is now changed to 0.

a) What is the significance of this change?

b) Show how the code in the thread must be modified with the timeout value changed to 0?

THE END