ECE 4550/5550G Project 2

Due Friday April 20, 2018 at 11:59pm (300 points)

Objective

To gain experience in writing and debugging a simulator
that implements (1) resource access control, and (2) overload management.

1 Logistics

You are to work in a team of 1-2 members. You and your teammate should be able to independently describe the solutions.

2 Resource Access Control

Extend your RM simulator from Project 1 to use the priority ceiling protocol (PCP) and the immediate ceiling priority protocol (ICPP). Your program should read from two input files. The first input file is exactly in the same format as the input file required by the GTA in the first part of Project 1. The second input file specifies the resource usage of the different tasks and has the following format

```
task_num
resource_num
resource_ID number_used taskID1 taskID2 ...
...
```

An example is as follows.

```
5
3
0 1 2
1 2 0 1
2 1 3
```

In the above example, there are 5 tasks and 3 resources. The first resource is used by the task with ID of 2. The second resource is used by two tasks (with IDs of 0 and 1), and so on.

For simplicity, you may assume that a task will not need more than one resource, i.e., nested critical sections need not be considered. In addition, you may assume that the critical section length is 50% of the total task execution time and occurs exactly in the middle. For instance, if a task's execution time is 2 time units, the critical section will be 1 time unit long, start 0.5 time units in, and end 0.5 time units before the task completes.

To an output file, your program will show the corresponding RM schedule and indicate

- when a job obtains the lock to enter a critical section;
- the current priority of the job (this should be updated as needed);
- the job whose priority the currently executing job inherits from;
- deadline misses (if any).

The output format of the schedule is up to you but should be easy to understand.

3 Overload Management

Modify your EDF simulator to use the highest value density first (HVDF) scheduling algorithm. Both the inputs and output files should be in the same format as those specified in the first part of Project 1.

4 Implementation on an RTOS (Optional + Extra Credits)

Again, this is a risky, time-consuming, but rewarding experience. If you wish to take this option, please email me first.

5 Submission

Your submission should include the source code (plus any other files necessary for compilation such as a makefile) bundled as a single zip file.

6 Honor Code

Please remember that you must turn in your own work. Failure to do so will result in a zero for the assignment, as well as in you being reported to the Office of Undergraduate Academic Integrity.

7 Rubrics

- (120 points) Correct PCP functionality
- (50 points) Correct ICPP functionality
- (110 points) Correct HVDF functionality
- (20 points) Ease of use, e.g., formatting and user-friendliness