Real-Time Systems (CSE 40463/60463) — Fall 2007

Programming Assignment 2

Due date: November 1, 2007

Note: The second programming assignment relies on the code from the first assignment! If your code from the first assignment was flawed or incomplete, you will have to make sure that it works correctly first! However, for this assignment, your code has to support only one scheduler (EDF, RMS, or LST; it is your choice)! If desired (say you did not finish the first assignment), it is ok to ask a colleague to share the code with you, but you have to work on the following extensions independently!

Part 1: Extend your simulator from programming assignment 1 to support the scheduling of sporadic tasks. That is, in addition to the periodic task set, your program must accept any number of sporadic tasks. The parameters of each sporadic task are: arrival, execution time, deadline. As before, your program will print a schedule (up to the desired size of the schedule), clearly indicating when a job is being preempted and when a job misses its deadline. Highlight the execution of sporadic tasks in your schedule (e.g., use a different color, font, etc.). Your summary output will indicate the average response time for each task, the number of preemptions, and the number of deadline misses (for both periodic and sporadic tasks). You do not have to print the response time for a sporadic task if it has not finished before the end of the schedule. If a sporadic task has been rejected, clearly indicate so in the output summary.

Part 2: Extend your simulator to implement the Priority Inheritance Protocol and extend your input parameters such that resource accesses can be specified. You can assume that each task will access the specified resource(s) in every invocation, that every task access at most two resources, and only periodic tasks will access resources. You can also assume that the number of resources is limited to 3. The new input parameters should include, for every available resource (say A, B, C), the time when the resource is accessed and how long it will be accessed. For example, an input line could be: T1(20,5):A(0,0),B(1,3),C(2,1), which means that periodic task T1 has a period of 20, an execution time of 5, and that it accesses resources B and C. B is accessed after 1 time unit of T1 has been executed and will be accessed for 3 units of time; C will be accessed after 2 time units and will be accessed for 1 time units. Note that this is an example for a nested resource access, your program should print an error message and stop if the input is incorrect (i.e., resource accesses are not correctly nested). The new schedule should also indicate when resources (or locks) are accessed, held, or denied.

Submission: Submit the source code and binary (plus any other files necessary for compilation such as a makefile) in the AFS dropbox. Further, for Part 1, include at least 2 examples (task set plus corresponding schedule) with your submission, each example containing at least 2 periodic and 2 sporadic tasks. For Part 2, your task is to use your simulator to find a task set and a corresponding schedule where at least two different cases of priority inheritance take place (you will get extra credit if your example shows a case of transient priority inheritance). If necessary (e.g., compilation difficulties), the instructor may ask you to come to an office hour to discuss your submission.