Prof. Bernd Finkbeiner, Ph.D.
Dipl.-Inf. Rüdiger Ehlers
Dipl.-Inf. Hans-Jörg Peter
Michael Gerke, B.Sc.

Reactive Systems Group
Saarland University
Due: Thursday, 15th January 2009
before the lecture, i.e., 2:10pm

Embedded Systems 08/09 – Problem Set 7

Problem 1 - Priority Ceiling Protocol (20 pts.)

Consider the Priority Ceiling Protocol. Using this protocol, give a picture describing a run of three tasks on one processor:

- Task 1 has the highest priority. Task 1 arrives at time t=5. Task 1 consists of normal computation for 1 time unit, followed by critical section 1 for 1 time unit, followed by normal computation for 1 time unit.
- Task 2 has lower priority than Task 1. Task 2 arrives at time t=2. Task 2 consists of normal computation for 1 time unit, followed by critical section 2 for 2 time units, followed by normal computation for 1 time unit, followed by critical section 3 for 1 time unit, followed by normal computation for 1 time unit.
- Task 3 has the lowest priority. Task 3 arrives at time t=0. Task 3 consists of normal computation for 1 time unit, followed by critical section 3 for 2 time units, followed by normal computation for 1 time unit.

Your picture should depict which task is executed (and the type of computation: either normal or the critical section the task is in) in the processor at which point in time, covering the interval from time t=0 to time t=13. You should also explicitly give any changes in the priority of the Tasks.

Problem 2 - Scheduling (50 pts.)

Consider the following scheduling problem $1 \mid sync \mid \overline{R}$:

> Using a uniprocessor machine, find a schedule for n aperiodic tasks $J_1, \ldots, J_n$ with computation times $C_1, \ldots, C_n$ that minimizes the average response time
>
> $$\overline{R} = \frac{1}{n} \sum_{i=1}^{n} (f_i - a_i) \,,$$
>
> where f_i is the time at which task i finishes its execution. Assume that all tasks arrive at time 0, that is, $a_i = 0$, for all $i = 1, \ldots, n$.

(a) Devise a scheduling algorithm that is optimal for the scheduling problem $1 \mid sync \mid \overline{R}$. (20 pts.)

(b) Formally prove that your algorithm is optimal. *Hint:* If your algorithm is deterministic, then you can start by proving the following lemma: Let S be a task-set, let $\sigma(S)$ be *the* schedule obtained by your algorithm, and let τ be any schedule. Then there exists a sequence of schedules $\tau_0, \ldots, \tau_k$ such that (i) $\tau_0 = \tau$, (ii) $\tau_k = \sigma(S)$, and (iii) the average response time of τ_{i+1} is not greater than the average response of τ_i, for $i = 0, \ldots, k-1$. (30 pts.)

Problem 3 - Multi-processor scheduling (30 pts.)

Consider the following task sets:

- Task set 1:

Arrival time	Computation time	Deadline
0	2	4
0	3	4
0	1	2
0	5	6

- Task set 2:

Arrival time	Computation time	Deadline
0	3	5
0	4	6
0	2	7
0	5	8

Please answer the following questions:

1. For **each** of the task sets, either give a feasible schedule **for two processors** or prove that there exists no feasible schedule. (10+10 pts.)

2. For **each** task set, also answer the question whether the result (i. e. whether the task set is schedulable or not schedulable) you have obtained in part one of this problem could be different if the arrival times would be different. (5+5 pts.)