EINDHOVEN UNIVERSITY OF TECHNOLOGY

Department of Mathematics and Computer Science

Examination Real-time Architectures (2XN26)
on Tuesday, November 1^{st} 2011, 14.00h-15.30h.

First read the entire examination. There are 5 exercises in total. Grades are included between parentheses at all parts and sum up to 10 points. Good luck!

1. This exercise concerns general aspects of real-time systems.

   (a) (0.5) Timing requirements can be classified as either *functional* requirements or as *quality* requirements. When should which classification be used?

   (b) (0.5) Is "real time" the same as "fast"? Motivate your answer.

   (c) (0.5) The so-called 4+1 view model of Kruchten consists of a *logical view*, *development view*, *process view*, *deployment view* and *scenarios*. Which view is the *least* relevant for real-time systems and why?

   (d) (0.5) Give at least three characteristics of embedded systems.

   Answer: See RTS.A1-Introduction.

2. A control system monitors its environment by means of sensors and responds via actuators. Monitoring can be performed by either *time*-triggered tasks or *event*-triggered tasks.

   (a) (0.5) Explain the difference between *time*- and *event*-triggered tasks in your own words.

   (b) Let the deadline for a response of the control system for a given event be given by D^{System}. Assume computation and actuation is done by a (single) time-triggered task τ.

      i. (0.5) Draw a worst-case situation for the arrival of the task τ.

      ii. (0.5) Give a *schedulability condition* for task τ, based on D^{System}, and the period T^{τ} and deadline D^{τ} of τ.

   Answer: See RTS.D0-Water-Vessel.

3. To implement periodic tasks, μC/OS-II provides

   - *relative one-shot* timers, i.e. `void OSTimeDly(INT16U ticks)`, where `ticks` are the number of clock ticks; and
   - primitives to set and get time, i.e. `void OSTimeSet(INT32U ticks)` and `INT32U OSTimeGet(void)`.

   (a) (0.5) Explain the notions of *relative* timer and *one-shot* timer in your own words.

   (b) (1.0) Is the functionality provided appropriate to implement periodic tasks?
   Hint: discuss jitter and drift.

   Answer: See RTS.Exercises-3.

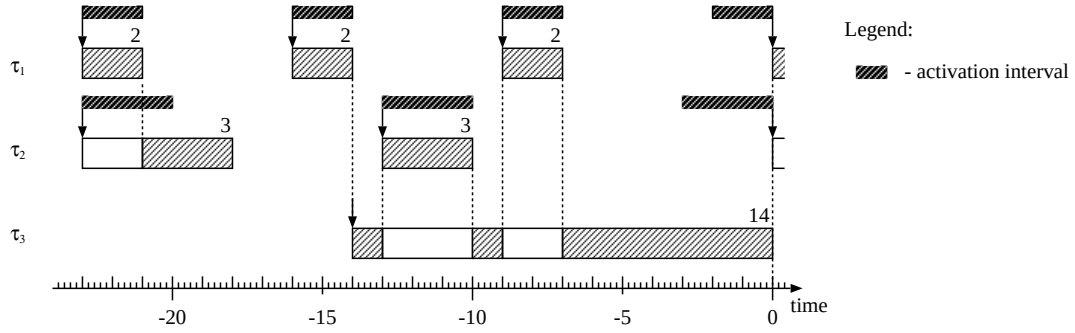

Figure 1: Timeline with an optimal instant for task τ_3.

4. Consider three tasks that are scheduled by means of fixed-priority pre-emptive scheduling (FPPS), where τ_1 has highest and τ_3 has lowest priority, with arbitrary phasing and characteristics as given in the following table.

	T	WD	BD	AJ	WC	BC
τ_1	7	5	2	2	2	2
τ_2	10	7	2	3	3	3
τ_3	31	25	11	5	9	9

(a) Consider the following recursive equation.

$$x = B_i + WC_i + \sum_{1 \le j < i} \left\lceil \frac{x + AJ_j}{WT_j} \right\rceil WC_j. \tag{1}$$

i. (0.5) Determine the smallest positive solution of (1) for task τ_3.
Answer: Note that the *lowest* priority task is *never* blocked. Hence, $B_3 = 0$ irrespective of potential resource sharing. Using an iterative procedure, we find $WR_3 = 26$.

ii. (0.5) Is this value the worst-case response time WR_3 of τ_3?
Answer: Yes, because $WR_3 + AJ_3 = 31 \le T_3 = 31$, hence a job of τ_3 never experiences interference of a previous job of τ_3.

(b) Determine the best-case response time of task τ_3

i. (0.5) by drawing a time line with an optimal instant for τ_3.
Answer: See Figure 1. Note that $BR_3 = 14$.

ii. (0.5) by means of the following recursive equation.

$$x = BC_i + \sum_{1 \le j < i} \left(\left\lceil \frac{x - AJ_j}{BT_j} \right\rceil - 1 \right)^+ BC_j \tag{2}$$

Answer: Using $WR_3 = 26$ as initial value for the iterative procedure to determine the best-case response time BR_3, we find $BR_3 = 14$.

(c) (0.5) Is task τ_3 schedulable?
Answer: No, because $WR_3 = 26 > WD_3 = 25$.

5. This exercise concerns resource access protocols (RAPs).

(a) (1.0) Transitive priority adjustment may occur when applying the Priority Inheritance Protocol (PIP). Describe at least three conditions that need to hold for transitive priority adjustment to occur.
Answer: We essentially need to describe conditions that may give rise to transitive blocking: (1) at least three tasks, e.g. τ_h, τ_m, and τ_l, with distinct priorities, where τ_h has highest and τ_l has lowest priority; (2) at least two mutual exclusive resources, e.g. R_1 and R_2; (3) τ_h uses a resource R_1, τ_l uses R_2, and τ_m uses R_1 and R_2 in a nested fashion, i.e. it first locks R_1 and than locks R_2 before releasing R_1.

(b) (0.5) The Priority Ceiling Protocol (PCP) and the Stack Resource Policy (SRP) are both based on resource ceilings. They differ with respect to the *moment* blocking occurs. Describe the difference in your own words.
Answer: Blocking occurs upon an attempt to

- *lock* a resource under PCP, i.e. *resource access*;
- *start* execution (of a job) under SRP, i.e. *on preemption*.

See RTS.B4-Policies-3-RAP.

(c) (0.5) SRP differs from PIP, HLP, and PCP in an essential aspect when it concerns priorities. Explain that difference in your own words.
Answer: The priority of a task is not changed under SRP. Instead, the priority in combination with the *system ceiling* determines which tasks may preempt under SRP. As an alternative, one may observe that SRP can also be used for *dynamic* priorities, whereas the other policies only apply for *fixed* priorities.

(d) (0.5) PIP is stated to be a *transparant* RAP. Explain transparency in your own words.
Answer: See book Buttazzo p. 230.