EINDHOVEN UNIVERSITY OF TECHNOLOGY

Department of Mathematics and Computer Science

Examination 2XN26 of Real-time Systems (2IN26)
on Monday, January 27^{th} 2014, 14.00h-17.00h.

First read the entire examination. There are 6 exercises in total. Grades are included between parentheses at all parts and sum up to 10 points. *Motivate all your answers.* Good luck!

1. (1.0) A terminal, such as a sensor and an actuator, can be modeled as a component with three interface elements for control as illustrated in Figure 1. Briefly describe

   (a) the three elements of the interface;
   Answer: The control interface consists of *commands*, *observers*, and *events*; see slides Specification concepts.

   (b) the relevance of these elements for both sensors and actuators;
   Answer: The interface of a sensor typically contains a *command* (for initialization) and *observers* and/or *events* to inform the control system about state-changes. The interface of an actuator typically contains *commands*; see slides Specification concepts.

   (c) the relation with event-triggered and time-triggered tasks.
   Answer: a so-called *event* will (re-) activate an event-triggered task, whereas a so-called *observer* is particularly useful for a polling (i.e. time-triggered) task.

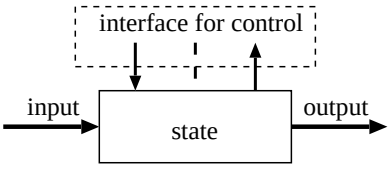

Figure 1: A component-model of a terminal.

2. For real-time systems, *priority* $\neq$ *importance*.

   (a) (0.5) Explain why the (relative) priority of a task may be higher than its (relative) importance for fixed-priority scheduling.
   Answer: The priority is typically used as a means to make a task-set schedulable, e.g. RM and DM are optimal priority assignments.

   (b) (0.5) When there is a mismatch between priority and importance, this may give rise to problems. Give an example of such a problem.
   Answer: Upon an overload, e.g. a less important task with a higher priority than a more important task needs more computation time than assumed, the more important task may miss its deadline.

   (c) (0.5) Describe a way to mitigate such problems.
   Answer: Overloads due to consumption exceeding the worst-case consumption time can be limited to the task experiencing the overload by using *reservations* (or servers) per task. This resolves this inter-task overload problem, and leaves the intra-task problem to the task itself. As an alternative, you can split tasks to remain optimal; see RTS.C6-Resource reservation.

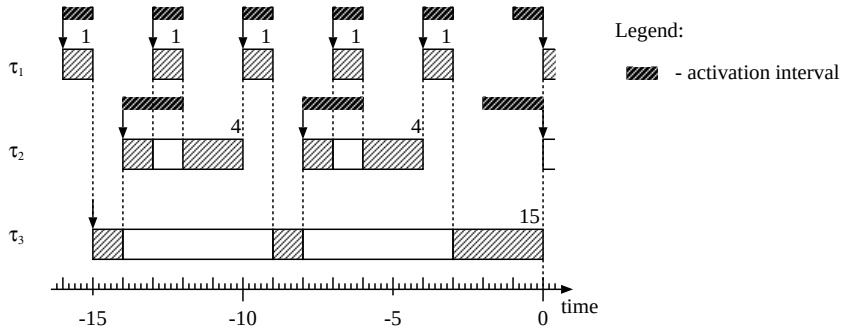

Figure 2: Timeline with an optimal instant for task τ_3.

3. 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	3	1	1	1	1
τ_2	6	6	4	2	3
τ_3	23	25	7	3	5

(a) (0.5) Determine the smallest positive solution of (1) for task τ_3 using

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

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 $WR_3 = 40$, which is larger than $WD_3 = 25$.

(b) (0.5) The period $T_3 = 23$ of τ_3 is *smaller* than the deadline $D_3 = 25$. Is the value determined above therefore indeed the worst-case response time WR_3 of τ_3? Motivate your answer.
Answer: The utilization U of the set of tasks is equal to $\frac{1}{3} + \frac{3}{6} + \frac{5}{23} > \frac{1}{3} + \frac{3}{6} + \frac{5}{25} > 1$. The task set is therefore not schedulable and the worst-case response of the lowest priority task τ_3 can become arbitrarily large. Assuming all three tasks start at the same time, the smallest solution of (1) presents the response time of the first job of τ_3.

(c) 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 2. Note that $BR_3 = 15$. Further note that because $U > 1$, we would get additional interference of a previous job when the time line would be extended towards the left.

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 $WR_3 = 40$ as initial value for the iterative procedure to determine the best-case response time BR_3, we find $BR_3 = 15$. Note that because $U > 1$, the result of the calculation will only hold for the specific instantiation shown in Figure 2.

(d) (0.5) Is task τ_3 schedulable? Motivate your answer.
Answer: No, because $U > 1$.

4. This exercise concerns servers.

   (a) *Periodic servers*

      i. (0.5) Two types of periodic servers have been presented, *idling* and *gain-time consumption*. Explain the difference between both types in your own words.
      Answer: See RTS.B4-Policies-2-FP-servers.

      ii. (0.5) Is the choice of the type of periodic server of influence on worst-case response time calculations? Motivate your answer.
      Answer: No, because the worst-case characteristics of both servers are identical; see RTS.Exercises-5.

   (b) *Schedulability of a system with servers*
   Let a system S consist of a set $\mathcal{T}$ of n hard real-time tasks τ_i with $1 \leq i \leq n$ and a server at highest priority. Assume $D_i \leq T_i$ and scheduling based on FPPS.

      i. (1.0) Describe a worst-case schedulability condition of system S with a *deferrable server* DS based on worst-case response-time analysis. *Hint*: use equation (1).
      Answer: $\forall_{1 \leq i \leq n} WR_i \leq WD_i \ \wedge \ C_{DS} \leq T_{DS}$, where $J_{DS} = T_{DS} - C_{DS}$, T_{DS} is the period of DS and C_{DS} is the capacity of DS. We can now simply add a term $\lceil \frac{x+J_{DS}}{T_{DS}} \rceil C_{DS}$ to the RHS of (1) to determine WR_i for all i.

      ii. (0.5) Describe a best-case schedulability condition of a system with a *polling server* PS based on best-case response-time analysis. *Hint*: use equation (2).
      Answer: $\forall_{1 \leq i \leq n} BR_i \geq BD_i$. We can now simply reuse (2) to determine BR_i for all i, because $BC_{PS} = 0$.

5. A so-called *deadlock* may occur when tasks share resources.

   (a) (0.5) Describe the notion of a deadlock and illustrate it with an example.
   Answer: See slides RTS.B4-Policies-3 and Fig. 7.13 in the 3^{rd} edition of the book.

   (b) (1.0) Give two resource access protocols that prevent a deadlock, and illustrate for one of these protocols how it works using the same example.
   Answer: See slide 8 of RTS.B4-Policies-3 and book.

6. This question concerns "Expected reading".
   The original analysis for CAN was refuted in "R.I. Davis, A. Burns, R.J. Bril, and J.J. Lukkien, *Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised*, Real-Time Systems, 35(3): 239-272, April 2007."

   (a) (0.5) What was wrong in the original analysis and how was it resolved?

   (b) (0.5) Which messages are at risk due to the flaw?

   Answer: See paper.