EINDHOVEN UNIVERSITY OF TECHNOLOGY
Department of Mathematics and Computer Science

Examination Real-time Architectures (2IN20)
on Tuesday, August 23^{rd} 2005, 14.00h-17.00h.

First read the entire examination. There are 4 exercises in total. Grades are included between parentheses at all parts and sum up to 11 points. You may use slides and papers for reference purposes. Good luck!

1. (a) (0.5) Both *Rate Monotonic* (RM) and *Earliest Deadline First* (EDF) are said to be *optimal*. What is the difference between the optimality criteria?
   Answer See book.

   (b) (0.5) Are "real-time system" and "fast system" synonyms? Motivate your answer.
   Answer See book.

   (c) (1.5) Consider a system with a sporadic server using rate monotonic scheduling (RMS). Does the following extension to the replenishment rule of the sporadic server affect schedulability?

   > "When the *system* becomes idle at time t_{idle}, the capacity of the sporadic server is replenished and planned replenishments are abandoned."

   Motivate your answer.
   Answer The answer is **no**. This follows immediately from the notion of *critical instant*. For RMS, a critical instant for a task is found for the simultaneous release of all higher priority tasks (optionally including the sporadic server). Assume the system is schedulable for a simultaneous release of all tasks, and a fully loaded server. The extension to the replenishment rule does not make it worse, i.e. can not adversely affect schedulability.
   Note that when the sporadic server is replenished taking the extension into account, it may no longer be a *sporadic* server. Hence, the extension is "strange" from a *naming* perspective.

   (d) (0.5) Describe in your own words what the term "temporal protection" means.
   Answer See slide 9 of the lecture on reservations ("RTA.D8-Reservations").

2. Consider a task set consisting of four hard real-time tasks τ_1, τ_2, τ_3, and τ_4, which share three resources R_1, R_2, and R_3. The tasks are scheduled using fixed priority scheduling, where task τ_i has a higher priority than τ_j iff $i < j$, i.e. τ_1 has highest and τ_4 has lowest priority. Figure 1 illustrates a situation of chained blocking when no resource sharing protocol is applied. Draw the timeline (including the dynamic priorities of the tasks) that results when applying the following resource sharing protocols for the example.

   (a) (1.0) Priority Inheritance Protocol.
   Answer See Figure 2. Note that the priority of task τ_i (with $2 \leq i \leq 4$) is temporarilly increased to the priority of task τ_1 when τ_1 blocks on R_{i-1}.

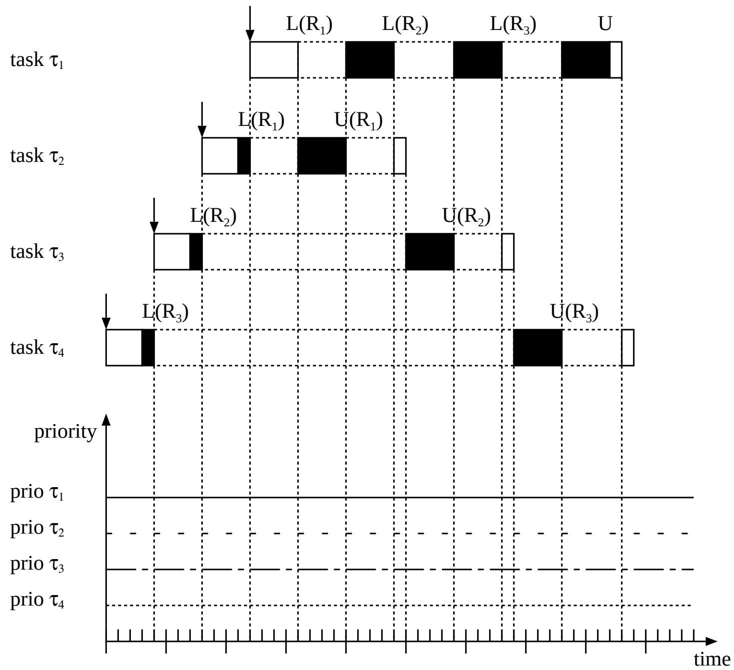


Figure 1: Example of chained blocking.

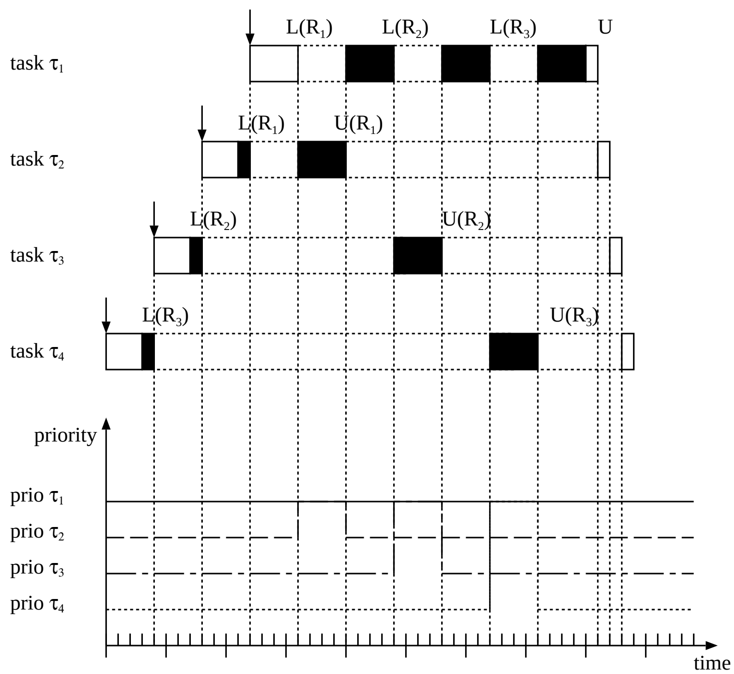


Figure 2: PIP applied to example with chained blocking.

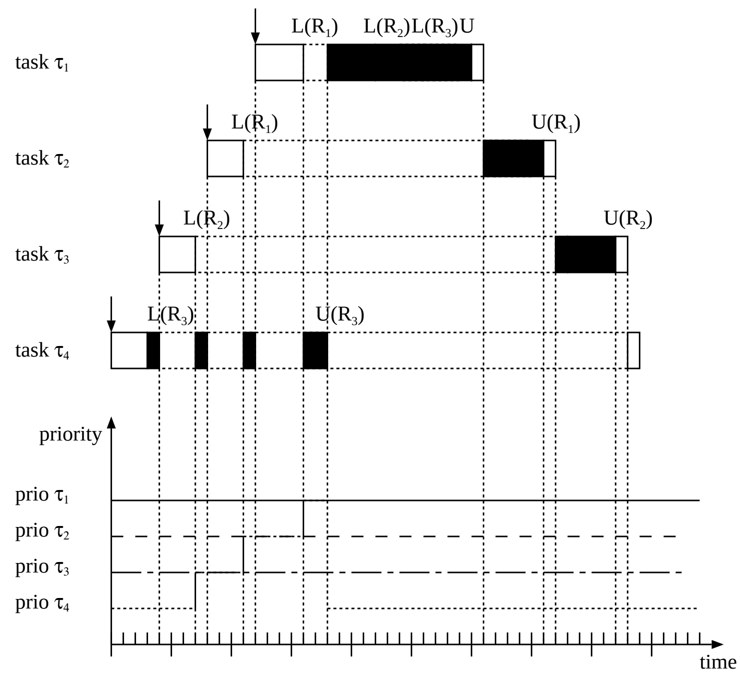


Figure 3: PCP applied to example with chained blocking.

Name	$T_i (= D_i)$	C_i
τ_1	5	2
τ_2	7	2
τ_3	11	2.5
τ_4	20	1

Name	$T_i (= D_i)$	C_i
τ_1	7	4
τ_2	14	3
τ_3	28	5

Table 1: Two sets of tasks, Γ_1 (left-hand side) and Γ_2.

(b) (1.0) Priority Ceiling Protocol.
Answer See Figure 3. Note that the priority of task τ_4 is temporarilly increased to the priority of task τ_i (with $1 \le i \le 3$) when task τ_i blocks on R_{i-1}, and the priority of task τ_4 is decreased when it unlocks R_3.

(c) (1.0) Highest Locker Protocol.
Answer See Figure 4. Note that the priority of task τ_i is temporarilly increased to the highest priority from the moment it locks a resource till the it unlocks it again.

3. Consider Table 1, describing two sets of hard real-time, periodic tasks, Γ_1 and Γ_2. The tasks are characterized by means of their periods T_i and (worst-case) computation times C_i.

   (a) (1.0) Determine whether or not task sets Γ_1 and Γ_2 are schedulable under RMS and arbitrary phasing.

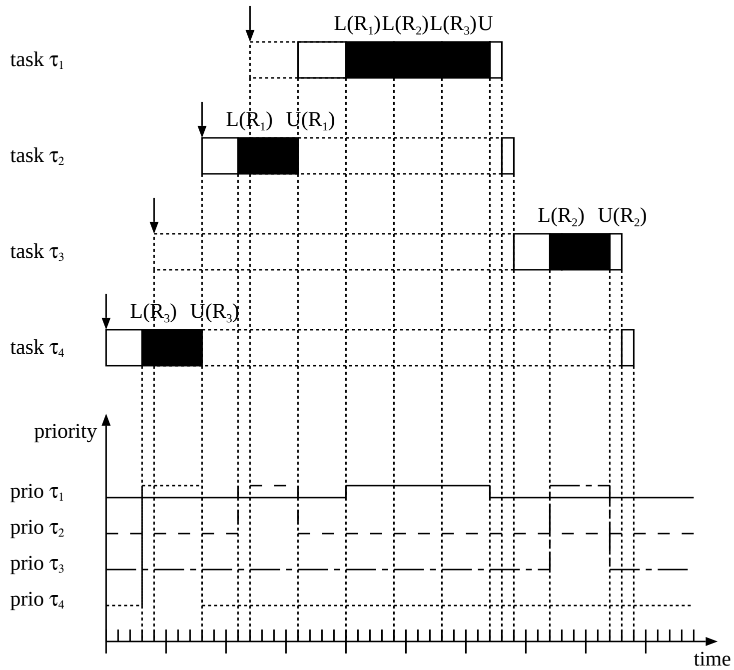


Figure 4: HLP applied to example with chained blocking.

Answer

- Task set Γ_1 is not schedulable (task τ_3 misses its deadline). This becomes immediately clear when a timeline is drawn.
- Task set Γ_2 is schedulable, which also becomes clear by drawing a timeline.

(b) (1.0) Determine for task set Γ_1 and Γ_2 with which factor the processor speed should be increased or decreased to make the task set precisely schedulable. Motivate your answer.

Answer

- When all tasks of Γ_1 are simultaneously released at time 0, task τ_3 needs to execute 2.5 units of time in an interval of length 10 (i.e. till the next activation of task τ_1). As a result, a total computation of 10.5 units of time has to be executed in an interval of length 10. The speed of the processor must therefore be *increased* with a factor $\frac{21}{20}$, effectively reducing the computation times with a factor $\frac{20}{21}$.
- Note that the periods of the tasks of Γ_2 are harmonic, i.e. for every i, j with $i > j$ there exists a $k \in Nat$ such that $T_i = kT_j$. For this particular case, the necessary test $U \leq 1$ is also an exact test. Because $U_2 = \frac{27}{28}$, the speed of the processor can therefore be *decreased* with a factor $\frac{27}{28}$, effectively increasing the computation times with a factor $\frac{28}{27}$.

4. Consider the final design of the truck bed in the presentation “Real-time Systems Design”

given by prof. Gerhard Fohler, and assume that this design will be implemented using online scheduling (FPS).

(a) Assume that the position information provided by the task POSITION is composite data, i.e. it takes multiple instructions to read and write the data. This can give rise to a problem.

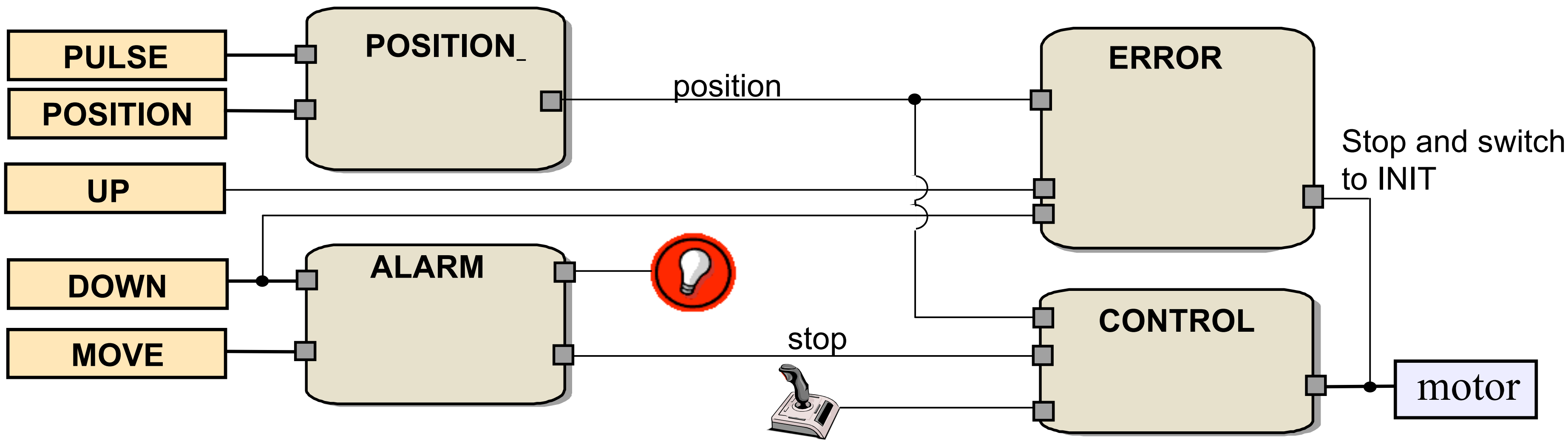
  i. (0.5) Explain the problem that may arise.
  Answer: Reading the data may result in inconsistencies, i.e. accessing the data requires mutual exclusion.

  ii. (1.0) Describe how the problem can be resolved.
  Answer: Protect the (global) data by means of, for example, semaphores. As an alternative, mutual exclusion can be guaranteed with offsets.

(b) (1.0) There is a precedence relation between task ALARM and task CONTROL. Describe two different ways to achieve the precedence relation.
Answer: A first way is to set the priorities right, i.e. the priority of ALARM must be higher than the priority of CONTROL. A second way is to use an appropriate offset, and to make sure that CONTROL never starts before ALARM has completed. There are various other ways to achieve the precedence relation. As an example, one may use a semaphore S on which ALARM performs a V-operation and CONTROL performs a P-operation.

(c) (0.5) Would it be possible to implement task ALARM and CONTROL as a single task? Motivate your answer.
Answer: Yes. Both tasks have the same period and the same deadline, and we can therefore give the single task that period and deadline. By first performing the code for ALARM and subsequently performing the code for CONTROL, we can enforc the required precedence relation. Finally, the behavior of the single task will be "similar" to the behavior of ALARM and CONTROL when ALARM and CONTROL have the same offset and ALARM has a higher priority than CONTROl.

Truck bed – Final design



INIT mode

Period: T=200 ms

INIT task
RT=0 ms
D=200 ms

OPERATE mode

Period: T=2 ms

POSITION task
RT=0 ms
D=2 ms

Period: T=100 ms

ALARM task
RT=0 ms
D=50 ms

→

CONTROL task
RT=0 ms
D=50 ms

Period: T=200 ms

ERROR task
RT=0 ms
D=200 ms

MUTEX: CON, ERR

<u>Estimated WCET:</u>

INIT:	1 ms
POSITION:	1 ms
ALARM:	2 ms
CONTROL:	2 ms
ERROR:	1 ms