(Till tentamensvakten: engelsk information behövs)

Exam

Embedded Systems I, DVA431
Västerås, 2015-11-02

Teachers:	Moris Behnam, tel: 021-107094 Saad Mubeen, tel: 021-103191 Adnan Causevic, tel: 021-107059 Guillermo Rodriguez-Navas, tel: 021-101356
Exam duration:	08:10 – 12:30
Help allowed:	calculator, language dictionary, ruler
Points:	90 p + extra lab points

Grading:

Swedish grades:		ECTS grades:	
0 – 54	→ failed	0 – 54	→ failed
55 – 76 p	→ 3	55 – 65	→ D
77 – 90 p	→ 4	66 – 80	→ C
91 – 100 p	→ 5	80 – 90	→ B
		91 – 100	→ A

Instructions:

- Answers should be written in English.
- Short and precise answers are preferred. Do not write more than necessary.
- If some assumptions are missing, or if you think the assumptions are unclear, write down what do you assume to solve the problem.
- Write clearly. If I cannot read it, it is wrong.

Good luck!!

Assignment 1: (18 points)

a) Many Embedded Systems are real-time systems. Explain what a real-time system is and give two examples. (3p)

b) Comment on the following statement: “real-time systems have to be fast”. (3p)

c) The build process of a microcontroller typically goes through 3 phases: compilation, linking and relocation, which are performed respectively by the Compiler, the Linker and the Locator. Describe why the relocation phase is needed and how it is implemented by the Locator. (6p)

d) Explain the main differences between an Application Specific Integrated Circuit (ASIC) and a Field Programmable Gate Array (FPGA). In which conditions is it better to use an ASIC than an FPGA? (6p)

Assignment 2: (18 points)

a) Diodes and Zener diodes are used for protecting electronic boards and IC’s. Describe how they are used and what type of protection they provide. Use circuit schematics to illustrate your answer. (6p)

b) Explain the concept of the “Open collector”/”Open Drain” mode to drive a digital output. Describe its role in the WIRED AND configuration and its working principle in the TWI communication bus. (6p)

c) Explain the main advantage and the main disadvantage of interrupt nesting. Illustrate your answer with one example for each situation. (6p)

Assignment 3: (6 points)

a) Explain the difference between partitioned and global scheduling in the case of multi-core systems. (3p)

b) Assume that your job is to develop the software for resource-constrained embedded systems. Briefly explain one resource optimization technique that you would use for such systems. (3p)

Assignment 4: (12 points)

Consider a real-time task set consisting of four tasks, A,B,C,D, that share three resources protected by semaphores S1,S2,S3. The tasks have different priorities and they are released for execution at different release times (see table below). All tasks use their semaphores as illustrated in the column “execution sequence” below (clock ticks are counted relative to the start of the system). The execution times of tasks are A=5 ticks, B=5 ticks, C=4 and D = 7 ticks as illustrated in the table below (in the “execution sequence” column). The deadlines of the tasks are relative to the start of the execution trace (that is, relative to time 0).

Task	Priority	Deadline relative	Release time	Execution sequence						
A	4 (highest)	15	9		S2	S2	S3			
B	3	18	6			S1	S3			
C	2	20	3			S2				
D	1 (lowest)	21	0			S1	S1	S1	S1	
			Clock tick:	0	1	2	3	4	5	6

For example, we can see in the table that task B has the next highest priority, prio(B)=3, it is released at time t=6, and, once released, it will execute like described below:

- *tick 6+0*: tries to execute one clock tick without any semaphores.
- *tick 6+1*: tries to continue execution for one more clock tick without any semaphores.
- *tick 6+2*: tries to lock semaphore S1, and if ok, it enters its critical section with S1.
- *tick 6+3*: tries to lock semaphore S3, and if ok, it enters its critical section with S3. It then releases S3 and S1 at the end of the tick.
- *tick 6+4*: tries to execute one clock tick without any semaphores.

The same reasoning applies to all other tasks.

Note that the execution scenarios for the tasks will be equal to the ones illustrated in the table above *only under the assumption* that the required semaphores are *free* when requested by a task, and the task is not pre-empted by a high-priority task. However, from the release times above we can see that the tasks will interfere with each other. Besides, the semaphores will not be always available when requested by the tasks.

Assume the release times of the tasks, their priorities and the execution sequences from the table above:

a) If *Priority Ceiling Protocol* PCP is used, will all tasks meet their deadlines? Draw the actual execution trace. You should run your trace from time t=0 until all of the tasks have completely executed once their execution sequence. (7p)

b) In the execution trace that you drew in (a), at what priority does the task D execute at (i) time = 2.5, (ii) time = 8, and (ii) time = 10? (3p)

c) Did you encounter priority inversion problem in (a)? If you answer yes then specify the time interval where the priority inversion occurs. If you answer no then explain why not? (2p)

Assignment 5: (18 points)

Assume three periodic tasks τ_1,τ_2 and τ_3 that communicate among each other by sending messages among their instances. The following is given:

Task τ_1:
- Has execution time 2 ms and period 12 ms.
- Sends 2 messages to a message queue MSGQ during each instance (job).
- All the messages are sent at the end of execution of each job.

Task τ_2:
- Has execution time 1 ms and a period 6 ms.
- Sends 2 messages in the message queue during each instance (job).
- All the messages are sent at the end of execution of each job.

Task τ_3:
- Has execution time 2 ms and a period 4 ms.
- Receives 2 messages from the message queue during each instance (job).
- All the messages are received at the end of execution of each job.

MSGQ:
- The queue contains the copy of the messages (not pointers).
- Has First In First Out (FIFO) order for inserting the messages.
- When a task reads a message from the message queue, the message is removed from the queue.

Questions:

a) Assume that the tasks are scheduled using the **Rate Monotonic** algorithm. What is the minimum possible size of the message queue (counted in number of messages) such that we are able to guarantee there will always be enough space in the queue for $\tau 1$ and $\tau 2$ to insert their messages? Motivate your answer by drawing an execution trace up to one hyper period and showing the number of messages in the queue after the execution of each task instance. (6P)

b) Assume that the tasks are scheduled using the **Shortest Job First** algorithm. What is the minimum possible size of the message queue (counted in number of messages) such that we are able to guarantee there will always be enough space in the queue for $\tau 1$ and $\tau 2$ to insert their messages? Motivate your answer by drawing an execution trace up to one hyper period and showing the number of messages in the queue after the execution of each task instance. (6P)

c) Assume that the tasks are scheduled using the **Earliest Deadline First** algorithm. What is the minimum possible size of the message queue (counted in number of messages) such that we are able to guarantee there will always be enough space in the queue for $\tau 1$ and $\tau 2$ to insert their messages? Motivate your answer by drawing an execution trace up to one hyper period and showing the number of messages in the queue after the execution of each task instance. (6P)

Assignment 6: (18 points)

a) Explain discontinuous behaviour and how it relates to testing? (4p)

b) Describe Beizer's Maturity Model. (7p)

c) What is the difference between specification based and implementation based test design techniques? (7p)

Assignment 7: (extra lab points)

You do not need to do anything here. This is for the extra points earned at the labs. Your extra lab points will be automatically added to your total exam score.
