Exam Embedded Software TI2720-C

Monday, January 30th 2012

Time: 3 hours

In order to avoid misunderstanding on the syntactical correctness of code fragments in this examination, we will always assume that we are dealing with pseudo-code, although we might have syntactically correct code in some cases. We assume that the required variables, semaphores, tasks, timers, etc. are always declared and initialized correctly.

Please answer **each** of the 20 questions in **less than 100 words**. Each answer will be awarded 0.5 points if fully correctly answered, 0 points otherwise. The exam mark is computed by summing the points for all answers.

1. Name three general characteristics of embedded systems.
2. What is an FPGA? Name one advantage and one disadvantage of using an FPGA in a design versus using a dedicated microcontroller in the same design.
3. Define what interrupts are, and describe what happens when an interrupt is triggered.
4. What is an interrupt vector table and precisely what information does it contain?
5. Assume that event X triggers a specific interrupt. Assume that after a "disable interrupt" command, event X occurs several times, after which an "enable interrupt" command follows. Explain what happens immediately after the "enable interrupt" command.
6. What is the status of the interrupts when a processor starts? Motivate why this is the case.
7. What is a critical section?
8. What does the liveness property of embedded software actually mean?
9. What is the worst response time for the Function-Queue architecture?
10. What is a "context" and how does it relate to the task switching mechanism?
11. Assume that two tasks with the same priority are in the "ready" state. What happens next and how can this situation be dealt with by an RTOS?
12. Define reentrant functions and describe their characteristics.
13. Give a pseudocode example of mutual exclusion occurring between the critical sections of two tasks.
14. Define what priority inversion is and illustrate such a scenario using pseudocode or a time diagram.
15. How does the mechanism of non-busy-waiting work in a RTOS? Comment on its accuracy.

16. Using semaphore functionality inside an interrupt routine in a RTOS without any precautions may crash the system. Please give such an example (time diagram or pseudocode) and comment on how to avoid it.
17. Define and characterize the out-of-band collection technique in the context of network instrumentation.
18. Give two reasons for which time-slicing should not be used in a RTOS for embedded systems.

Consider the following piece of code:

```
volatile static long int lSecondsToday;

void interrupt vUpdateTime()
{
    ++lSecondsToday;
}

long lGetSeconds()
{
    long lReturn;
    lReturn = lSecondsToday;
    while (lReturn!=lSecondsToday)
        lReturn = lSecondsToday;
    return (lReturn);
}
```

19. Does this code suffer from the shared-data bug? Explain your answer.
20. How does the behavior of the previous fragment of code change if the word "volatile" would be removed from the first line of code?