Exam TI2720-C/TI2725-C Embedded Software

Wednesday April 16 2014 (18.30 - 21.30)

Koen Langendoen

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.

Further, we assume the following abbreviations to be known:

- RR = Round Robin,
- RRI = Round Robin with Interrupts,
- FQS = Function Queue Scheduling,
- RTOS = Real-Time Operating System,
- IR = interrupt and
- ISR = interrupt service routing.

In this exam, we use the following definitions, unless stated otherwise:

```
void delay(int ms) {
  !! do some CPU computation for ms milliseconds
}

void putchar(char c) {
  while (!! UART tx buffer not empty);
  !! send c to UART tx buffer
}

void puts(char *s) {
  !! print string s using putchar
}
```

To pass this written exam, you need to correctly answer at least 20 questions. The relationship between the number of correctly answered questions and your mark is given below.

#correct	≤ 8	9	10	11	12	13	14	15	16	17	18	19
Mark	1	1.5	2	2.5	3	3	3.5	4	4.5	5	5	5.5

#correct	19	20	21	22	23	24	25	26	27	28	29	30
Mark	5.5	6	6.5	7	7	7.5	8	8.5	9	9	9.5	10

Question 1
Embedded programming is more difficult than “classical” programming because of a. the interaction with hardware b. real-time issues (timing) c. limited visibility (e.g., no display) d. all of the above
Question 2
An embedded program can be coded as a finite state machine where a. interrupts trigger state transitions b. user actions trigger all state transitions c. interrupts must be disabled during state transitions d. state transitions occur when RTOS functions are called
Question 3
Which of the following statements is true regarding the embedded software crises a. It was triggered by the “year 2000” bug b. It was triggered by the increasing speed gap between on- and off-chip communication c. It has been resolved due to “Morse law” generating ever faster hardware d. It refers to the lack of correct code for the increasing number of embedded systems

Question 4
Which of the following statements is correct? An interrupt is a. a synchronous signal form hardware to indicate processor attention b. a synchronous signal from the processor to indicate hardware attention c. an asynchronous signal from hardware to indicate processor attention d. an asynchronous signal from the processor to indicate hardware attention
Question 5
Which of the following statements is correct? An interrupt service routine is supposed to a. disable the non-maskable interrupt b. restore the context and return c. restore the lowest-priority interrupt d. increase the program counter
Question 6
Which of the following statements is correct? An interrupt vector a. contains a pointer to an ISR b. points to a table with interrupt routines c. contains the addresses of interrupts d. contains interrupt routines
Question 7
Which of the following statements is correct? Using interrupts impairs … a. task response time b. higher priority ISR response time c. processor response time d. none of the above

Question 8
Which of the following statements is correct? a. we cannot have a shared data problem in an RR architecture b. we cannot have a shared data problem in an RRI architecture c. we cannot have interrupt vectors in an FQS architecture d. we cannot have atomic sections in an RTOS architecture
Question 9
For which kind of software architecture for embedded systems is the **worst-case** response time for task code equal to the total execution time for all task code plus the execution time for interrupt routines? a. RR b. RRI c. FQS d. RTOS
Question 10
Which of the following architectures is most robust to code changes? a. RR b. RRI c. FQS d. RTOS

Question 11

In an RTOS, tasks can be in state BLOCKED, READY, or RUNNING. Which of the following statements is true?

a. A task can transition directly from BLOCKED to RUNNING
b. A task can transition directly from READY to BLOCKED
c. A task can transition directly from RUNNING to BLOCKED
d. A task can stay in one state for at most one time slice

Question 12

After creation a task enters the RTOS in the state

a. BLOCKED
b. READY
c. RUNNING
d. that depends on its priority relative to the running task

Question 13

Which of the following statements is correct?

a. at most one task can be in the state READY
b. at most one task can be in the state RUNNING
c. only the task with the highest priority can be in the state RUNNING
d. only the task with the lowest priority can be in the state BLOCKED

Questions 14 – 16

Given is the following RTOS (pseudo) code with decreasing priorities T1 > T2 > T3:

```
void f1(void) { delay(100);}
void f2(void) { delay(100);}
void f3(void) { delay(100);}

void T1(void) {
   while (1) {
      OS_Pend(sem1);  // wait for event #1
      OS_Pend(mutex);
      delay(100);
      OS_Post(mutex);
      f1();
}}

void T2(void) {
   while (1) {
      OS_Pend(sem2);  // wait for event #2
      f2();
}}

void T3(void) {
   while (1) {
      OS_Pend(sem3);  // wait for event #3
      OS_Pend(mutex);
      delay(100);
      OS_Post(mutex);
      f3();
}}
```

Question 14

No other events have been treated by the system previously. The events #1, #2, #3 are triggered **periodically**, in this order, 10 ms one after another (example: #1,#2,#3,#1,#2,#3,#1,#2,#3,etc.). Which of the following statements is correct?

a. f1() is executed at least once
b. each of f1() and f2() are executed at least once
c. each of f1(), f2(), and f3() are executed at least once
d. priority inversion occurs between T1 and T3

Question 15

No other events have been treated by the system previously. Imagine that the following sequence of events comes **once**: #3, #2, #1 (timing between events is not known). Which of the following statements is always correct?

a. f1() is executed first, before f2() and f3()
b. f2() is executed first, before f1() and f3()
c. f3() is executed first, before f1() and f2()
d. we cannot make any prediction on the order of execution

Question 16
No other events have been treated by the system previously. Imagine that the following sequence of events comes **once**: #3, #2, #1 (timing between events is not known). The “delay(100)” calls are replaced by “OSTimeDly(10)” calls. Which of the following statements is always correct? a. f1() is executed first, before f2() and f3() b. f2() is executed first, before f1() and f3() c. f3() is executed first, before f1() and f2() d. we cannot make any prediction on the order of execution

Question 17
We assume a system clock tick of 10ms. Which of the following statements is most accurate? Calling the function “OSTimeDly(3)” ... a. causes a delay of exactly 30ms b. causes a delay between 20ms and 30ms c. causes a delay between 30ms and 40ms d. causes a delay between 20ms and 40ms
Question 18
Semaphores can be used for mutual exclusive access to shared resources. In that case a semaphore must be initialized to a. 0 (zero) b. 1 (one) c. n (number of concurrent tasks) d. m (number of resources)
Question 19
In order for priority inversion to occur, it be must be true that a. there are at least two tasks with different priorities b. there are at least three tasks with different priorities c. the RTOS scheduler uses a round-robin policy d. task preemption is disabled
Question 20
Mutual exclusive access can also be accomplished by disabling interrupts. Compared to using mutexes this has the advantage that a. context switching is faster b. programs are more responsive to external events c. deadlocks cannot occur d. none of the above
Question 21
Which of the following statements is correct? The X32 platform ... a. has one interrupt priority level b. has an interrupt controller c. is not equipped with an interrupt vector d. prohibits interrupt preemption

Question 22
A re-entrant function **f** may not use variables in a non-atomic way ... a. no matter where they are located b. unless they are allocated on the heap c. unless they are local to **f** (and allocated on the call stack) d. unless they are declared static in the C implementation file
Question 23
Which of the following statements is correct about RTOSs and interrupts? a. an ISR may call any RTOS function because it will never block the caller b. an ISR must **not** call any RTOS function that might block the caller c. an ISR must **not** call any RTOS function that might block the caller unless it has a small size d. an ISR may call any RTOS function that might block the caller when it has a small size
Question 24
Which of the following statements is correct about RTOS and interrupts? a. an ISR may call any function b. an ISR may only call RTOS functions c. an ISR may never call functions that might cause a context switch d. an ISR may call RTOS functions that might cause a context switch provided that the RTOS knows that an ISR (and not a task) is making the call

Question 25
To master the complexity of a program it can be structured as a set of tasks. Which of the following statements is true? a. the memory footprint grows linearly with the number of tasks b. when doubling the number of tasks, the number of semaphores also doubles c. the context switch time between tasks doubles with every new task d. tasks can share the same stack as mutual exclusion allows only one task to execute a critical section
Question 26
Creating and destroying tasks dynamically is somewhat problematic because a. the RTOS must disable interrupts for too long b. a task's priority may change during its lifetime c. a task may hold a semaphore when being destroyed d. a task may create another task recursively, draining resources managed by the RTOS
Question 27
Running tasks with the same priority complicates an RTOS because a. round robin scheduling requires exclusive access to the heap b. round robin scheduling may lead to oscillating behavior between two high-priority tasks c. response times become unpredictable in overload situations d. the RTOS can no longer access the tasks through a dispatch table

Question 28
When developing code for an embedded system, the software can de structured into HW-dependent and HW-independent code. This is done to ensure that a. debugging can be performed by two independent teams of software developers b. debugging HW independent code becomes feasible on the target platform c. debugging HW independent code becomes feasible on the host platform d. debugging HW dependent code becomes feasible during runtime
Question 29
One approach to facilitate debugging is to **script** test scenarios. Which of the following statements is true? a. scripting can only be used to test HW independent code b. a script can only trigger one interrupt at the same time c. each line of a script must start with a time stamp d. parsing a script can easily be done in C
Question 30
When debugging code for a distributed system, collecting the (debug) output of the different sensor nodes can be arranged in different ways [Beutel:2009}. Which of the following statements is true? a. **online** sniffing requires logging facilities on the sensor nodes themselves b. **offline** sniffing requires logging facilities on the sniffer nodes c. a **wireless** testbed requires **no** physical instrumentation (i.e. wiring) of the sensor node d. a **wired** testbed requires **no** physical instrumentation (i.e. wiring) of the sensor node