EXAM
(Tentamen)

TDDI11
Embedded Software

2018-06-01 08:00-12:00

On-call (jour):
Ahmed Rezine, 013 - 28 1938

Admitted material:

- Dictionary from English to another language

General instructions:

- The assignments are **not ordered** according to difficulty.
- You may answer in either English or Swedish.
- Read all assignments carefully and completely before you begin.
- Use a new sheet for each assignment and use only one side.
- Before you hand in, order the sheets according to assignment, number each sheet, and fill in AID-number, date, course code and exam code at the top of the page.
- Write clearly. Unreadable text will be ignored.
- Be precise in your statements.
- **Motivate** clearly all statements and reasoning.
- **Explain** calculations and solution procedures.
- If in doubt about the question, write down your interpretation and assumptions.
- Grading: U, 3, 4, 5. The **preliminary** grading thresholds for p points are:

$0 \leq p < 20$:	U
$20 \leq p < 30$:	3
$30 \leq p < 35$:	4
$35 \leq p \leq 40$:	5

Good Luck!

Question 1, multiple choice. (10 points)

- Use the answer sheet at the end of the exam.
- No motivation or explanation is required for this 10 points question.
- **Zero or more statements may be correct for each question**.
- Tick each statement if and only if it is correct. Ticking a wrong statement or missing to tick a correct statement gives 0 points for that question.

1a) Compared to polling-based programming:

1. Interrupt based programming requires simpler hardware support.
2. Interrupt based programming wastes more CPU cycles to monitor the status of I/O device controllers.
3. Interrupt based programming is harder to maintain and to scale.

1b) A correct real time system

1. may miss some deadlines if it is a **soft** real time system
2. may miss some deadlines if it is a **firm** real time system
3. may miss some deadlines if it is a **hard** real time system

1c) What will be the output from the following C program?

```
#include <stdio.h>
int main() {
        unsigned long int a = 2;
        unsigned long int *b = &a;
        unsigned long int *c = b;
        *b = 0;
        printf("%lu %lu \n", a, *c);
}
```

1. 0 0
2. 2 0
3. 2 2

1d) What will be the output of the following C program?

```
#include <stdio.h>
int main() {
        printf("%d \n", (1 && 2) && 4);
}
```

1. 0
2. 1
3. 7

1e) What will be the output of the following C program?

```
#include <stdio.h>
int main() {
        printf("%d \n", (4 | 2) & 1);
}
```

1. 0
2. 1
3. 6

1f) The foreground/background model
1. is suitable for complex applications with dynamically generated tasks.
2. requires a scheduler and a a multitasking operating system in addition to the foreground and background tasks.
3. results in applications where a main loop repeatedly executes each task in the background.

1g) When used to model an embedded system, a state machine:
1. Needs to have a final state.
2. Should be complete (i.e., all possible transitions for each state should be represented).
3. Can have self-loops (i.e., transitions with the same source and target states).

1h) Concurrent software is ...
1. not needed for embedded systems because they are simpler than multicore systems
2. easy to get wrong and is therefore never used in embedded software.
3. useful in embedded applications and can therefore be found in embedded systems.

1i) A non-recurring engineering cost:
1. Is another name for unit cost
2. Is negligible for newly designed satellites
3. Is a cost for a work that is not necessary but that engineers like to do.

1j) Specification of embedded software:
1. Is usually efficiently and sufficiently carried with a natural language (e.g. English)
2. Does not profit from using state machines
3. Might include errors and deficiencies

Question 2. (4 points)
- Explain the difference between little and big endian representations of a 4 bytes-integer:
- Include clean and simple figures in your explanation.
- Describe, in a sentence or two, a situation where translating from one representation to the other is needed.

Question 3. (5 points)
What is the difference between I/O programming using DMA and polling in terms of required CPU cycles and hardware support. Explain.

Question 4. (6 points)

Consider a task set with two periodic tasks: Task 1 with period T1= 6 and execution time C1= 2 and Task 2 with period T2= 3 and execution time C2= 2. Both tasks are to be run on the same processor using some scheduling algorithm.

1. Give the processor utilization ratio in case the two tasks are scheduled (1pt)
2. Which task would get the highest priority if Rate Monotonic Scheduling (RMS) is used (1pt)
3. Assume non-pre-emptive RMS is used. Can the tasks be scheduled? Explain using a diagram (1 pt).
4. Can non-pre-emptive EDF schedule the tasks? What about pre-emptive EDF? Explain using two diagrams (3pt).

Question 5. (5 points)
Give a Mealy machine (outputs associated to transitions, not states) that takes sequences of 0s and 1s as input. The machine should output 1 when it finished reading an odd sequence of ones (i.e. 1,3,5,7,.. of consecutive ones). It should output 0 otherwise.
Possible runs of your solution:

Input sequence	Output sequence
0100000...	0010000...
1111100...	0000010...
1100000...	0000000...
0011100...	0000010...

Question 6. (5 points)
Describe the sequence of events that occur when a byte arrives at a serial port and interrupt-driven I/O is used to copy the byte to a given address. Recall a serial port uses a control register to inform on whether new data arrived and a data register to contain the data (you can keep the interaction with the serial port at this level of details: namely read status, read data, etc).

Question 7. (2 points)
The following macros is meant to compute the sum of two numbers. This macro is not well written. We still want to use a macro for computing sums of two numbers. Explain what problems may occur if used as currently written and rewrite it to solve these problems.

```
#define sum(x,y) x+y
```

Question 8. (3 points)
In this question, you are allowed to use bit-level operators (i.e. some of “and”, “or”, “shift left”, “shift right”, and “inversion”), no loops, no additions, no divisions, no subtractions and no multiplications. Write a C program that checks if a 16 bits unsinged int is a multiple of 8.

Answer sheet for question 1. Please hand this paper in together with the answers for the other questions (numbered and with AID number).

1a)	( ) 1	( ) 2	( ) 3
1b)	( ) 1	( ) 2	( ) 3
1c)	( ) 1	( ) 2	( ) 3
1d)	( ) 1	( ) 2	( ) 3
1e)	( ) 1	( ) 2	( ) 3
1f)	( ) 1	( ) 2	( ) 3
1g)	( ) 1	( ) 2	( ) 3
1h)	( ) 1	( ) 2	( ) 3
1i)	( ) 1	( ) 2	( ) 3
1j)	( ) 1	( ) 2	( ) 3