Försättsblad till skriftlig tentamen vid Linköpings Universitet

Cover page for written exam at Linköping University

Datum för tentamen Date of exam	**2011-08-15**
Sal Room	**U4**
Tid Time	**08:00 - 12:00**
Kurskod Course code	**TDDI11**
Provkod LADOK code	**TEN1**
Kursnamn/benämning Course name	**Programmering av inbyggda system** Embedded software
Institution Department	**IDA**
Antal uppgifter som ingår i tentamen Number of assignments	**9** 9 assignments for a total of 40 points
Antal sidor på tentamen (inkl. försättsbladet) Number of pages including cover	**3**
Jour/Kursansvarig Responsible/Examiner	**Klas Arvidsson** klas.arvidsson@liu.se
Telefon under skrivtid Phone during exam	**013 - 28 21 46** 013 - 28 21 46
Besöker salen ca kl. Time of exam visit	**Ca 1h efter tentastart** About 1h after exam start
Kursadministratör Course administrator	**Gunilla Mellheden** 013 - 28 22 97, 070 - 597 90 44, gunilla.mellheden@liu.se
Tillåtna hjälpmedel Allowed aids	**Ordlista och enkel miniräknare (+,-,*,/)** Dictionary and simple pocket calculator (+,-,*,/)
Övrigt Other information	*Precise, explained and clearly motivated assumptions, statements and reasoning raise the impression and are required for highest score.* ***Solve at most one assignment per sheet.*** ***If in doubt, state clearly your interpretation of the question, your assumptions, and answer according to that.*** Preliminary graded: U < 50% < 3 < 67% < 4 < 84% < 5 Grades may be raised or lowered based on overall impression. Results available within 10 working days.
Typ av papper Paper to use	**Rutigt, linjerat eller blankt** No preference
Antal anmälda Number of exams	**7**

1. (2p)

Embedded development sets many non-functional requirements on the system, such as unit cost or battery life. List four other *common* and *important* such requirements.

2. (3p)

Explain with a clear example the concept of mutual exclusion and the errors that may occur if not properly handled.

3. (3p)

An 8-bit processor have two 8-bit registers for 8-bit arithmetic operations, `reg1` and `reg2`. It also have the following assembly operations (the first operand is the destination):

```
mov [ADR] regX  ;; Move from register to memory
mov regX  [ADR] ;; Move from memory to register
add regX  regY  ;; Addition, regX = regX + regY
adc regX  regY  ;; Addition, regX = regX + regY + carry
;; Example, X represent the base address of some data
mov reg1 [X+1] ;; Move second byte of X to reg1
```

Base addresses A, B and C each store one 16-bit integer.

a) Show assembly code to perform the *16-bit addition* C = A + B. Assume that `reg1` and `reg2` are free, and that no other registers exist. Both A and B must remain unmodified. (2p)

b) What happens if the 16-bit addition overflows? How is it detected? Compare to what happens if a regular 8-bit addition overflows. State all assumptions. (1p)

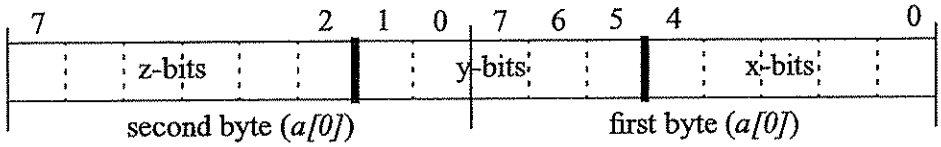
4. (4p)

Code can be optimized for several reasons, with several goals.

a) Your code require too much memory (ROM). *Explain* what you can do, and what you shall think of in order to reduce the size of the compiled code. At least three things. (3p)

b) *Mention* two other goals of optimization. (1p)

5. (4p)



An array *a* of two 8-bit data (*a[0]* and *a[1]*) is outlined above. It is used to store three 5-bit integers. Use the C-language to demonstrate how to assign the content of variable `b` (the low 5 bits) to *y-bits*. All *x-bits* and *z-bits* must of course remain the same.

6. (5p)

a) *Describe* four criteria that are said to distinguish an embedded system. We say it's an embedded system when some or all of them is true.

b) Give an example of an embedded application where at least two of the distinguishing criteria are found. *Motivate.*

7. (5p)

The spiral model is a development methodology that can be used.

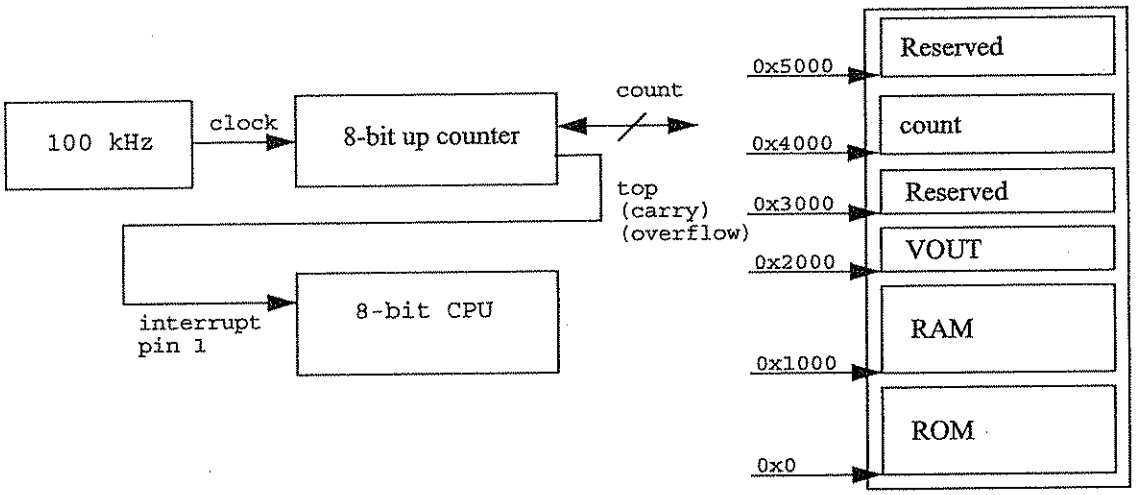
a) *Draw* a simple figure. *List* the iterations and phases taken when the methodology is used. (2p)

b) *Describe* how it works. What is the central idea? (2p)

c) *Mention* a methodology (*any* methodology) specially appealing to embedded systems developement and briefly *state* why. (1p)

8. (6p)

A system outlined below have an output VOUT that controls an output voltage. The voltage is turned on (12.0V out) by writing 1 to VOUT, and turned off (0.0V out) by writing 0 to VOUT.



a) Explain theoretically how to get an *average* voltage of 7.0V on the output. (2p)

b) Write a C-code solution to a). You may use polling, but an interrupt solution gives the chance to demonstrate more knowledge in c). (You may answer c) anyway, but it is easier with b) to relate to.) (2p)

c) Imagine an interrupt solution to b). *Explain* where in the memory you place your pieces of code and how you set up the system to find it. *State all assumptions.* (2p)

9. (8p)

A system have an input signal that from time to time is high. The high period may be short: 40, 50 or 60ms; or it may be long: 80, 90 or 100ms. Every high period is followed by an at least 30ms long low period. The system shall reliably count when the signal is *high and long* (80 ms or longer). The output of the system is a count of the total number of long high periods that have occured.

a) State and motivate how you select the state machine period. (2p)

b) Design a synchrounous state machine that solve the problem. (5p)

c) State if your machine is of mealy or moore type. (1p)