LUNDS TEKNISKA HÖGSKOLA
Lunds universitet

Institutionen för datavetenskap
Krzysztof Kuchcinski

Tentamen i kursen
EDA380: Konstruktion av inbyggda system
(Design of Embedded Systems)

2007-05-23, kl. 14-19

Sal:
MA 8

Hjälpmedel:
Inga

Resultat anslås:
Senast 2007-06-07

Poänggränser:
Max 40 p., för godkännande krävs ca 20 p.

Jourhavande lärare:
Kris Kuchcinski, tel. 22 23414

The answers to the questions can be written in Swedish or English.

Good luck!
Lycka till!

1 (3 p.)

Discuss and give motivations why implementation of embedded systems using a single processor that implements all functionality in software is often not possible or difficult. Compare this solution to a design that has a processor and specialized hardware connected to it. In your discussion evaluate which architecture is better in respect to the following parameters: performance, cost, power consumption and possibility for future upgrades.

2 (3 p.)

Discuss a typical design methodology for embedded systems. When are different design activities, such as design specification, design partitioning, component allocation, and communication synthesis performed?

3 (3 p.)

Draw a data-flow graph that implements the following computation that has three inputs in_1, in_2 and in_3, and one output out. Use several nodes that implement different functions.

$x = in_1 + in_2$
$y = x + 1$
if ($in_3 > 11$)
  $out = x$
else
  $out = y$

4 (3 p.)

Draw a Petri net that models one task that produces items (modelled as tokens) to a buffer of size three and the other task that consumes these items, that is a producer/consumer model with the bounded buffer of size three items.

5 (6 p.)

Write the synthesizeable behavioral VHDL code that implements the synchronous finite state machine (FSM) depicted in figure 1. The FSM has one input, called `request`, that is of enumeration type with values `(r1, r2, r3)`. The enumeration values represent requests to floors 1, 2 and 3. The machine has also one output, called `go`, that is also of enumeration type with values (n, u1, d1, u2, d2). The values represent `n`- do nothing, `u1`- go one floor up, `u2`- go two floors up, `d1`- go one floor down, and `d2`- go two floors down.

6 (6 p.)

Write the synthesizeable behavioral VHDL code that implements the following C function. The computation should be done using 32 bit integers and it should take several clock cycles. You can assume that the inputs are available when needed on ports `a` and `b`. The output is sent to port `r`.

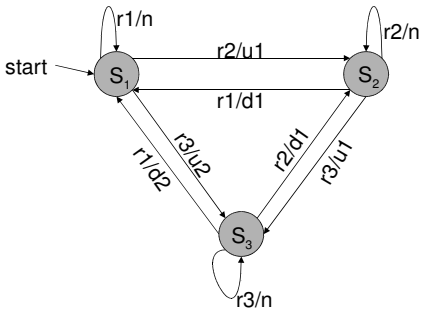

Figure 1: FSM model for elevator controller.

```
int foo(int a, int b) {
  int result = 0;
  while (b>0) {
    result += a;
    b--;
  }
  return result;
}
```

7 (5 p.)

Using list scheduling, make a schedule for a data-flow graph depicted in Figure 2. Assume that you can use two adders and two pipelined multipliers. The adders have 1 clock cycle delay and multipliers have a total delay of 2 clock cycles but each pipeline stage has delay of 1 clock cycle. Answer the following questions:

a) What priorities do you use in your list scheduling?
b) What is the number of clock cycles for execution of this model?

Give the sequence of the steps performed by the list scheduling algorithm.

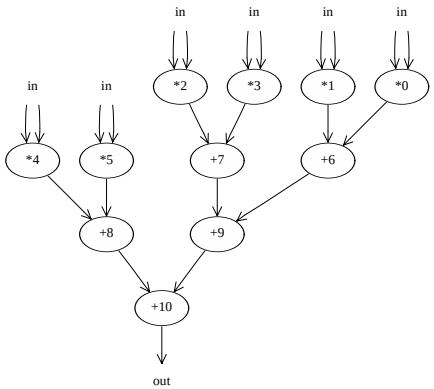

Figure 2: An example of data-flow graph

8 (4 p.)

The classical example of the phenomenon of *priority inversion* is the Mars Pathfinder mission and is known as "Mars Pathfinder Bug". Explain what is priority inversion when rate monotonic priority (RMS) scheduling is used. Discuss a method that avoids priority inversion.

9 (3 p.)

Give the formula for power consumption in CMOS technology and explain different parameters that influence power consumption. Discuss how power consumption of a design can be minimized in relation to these parameters.

Explain why *parallelization* of computations can be used not only to speed-up a design but also to reduce power consumption.

10 (4 p.)

Discuss briefly the main idea of SCAN path testability improvement technique. In the discussion include the following points:

a) the general idea of the SCAN path, and
b) why SCAN path improves test generation and testing time.