OLLSCOIL NA hÉIREANN, CORCAIGH
THE NATIONAL UNIVERSITY OF IRELAND, CORK

COLÁISTE NA hOLLSCOILE, CORCAIGH
UNIVERSITY COLLEGE, CORK

Summer Examination 2012

Second Science

CS2502: Logic Design

Professor Ian Gent
Professor James Bowen
Dr Frank Böhme

Answer all questions.

90 minutes

1. These questions deal with *majority decoders*. Recall, a majority decoder is a combinational circuit with an odd number of inputs and one output Y. The output Y follows the *predominating input value*. If the number of inputs that assume 1 is higher than the number of inputs that assume 0, then Y assumes 1 and vice versa.

   a) Recall also the two canonical notations for switching functions, i.e. the *canonical sum of products* (CSOP) and the *canonical product of sums* (CPOS). Consider the case of the majority decoder with *three inputs*. Should one of these canonical forms be preferred over the other because of a shorter expression? If so, which one (CSOP or CPOS) should be preferred? Justify your answer. (*3 marks*)

   b) How can the answer obtained in a.) be extended to the *general case*, i.e. to majority decoders with 5, 7, 9, 11 and more input lines? (*3 marks*)

2. The questions deal with the *Associative Law*. Recall that the AND as well as the OR operations observe the Associative Law. For instance, in case of OR that law states that $(A + B) + C = A + (B + C)$.

   a) Does the XOR (exclusive OR) operation also observe the Associative Law? Justify your answer. (*3 marks*)

   b) Does the NAND (NOT AND) operation also observe the Associative Law? Justify your answer. (*3 marks*)

   c) Does the NOR (NOT OR) operation also observe the Associative Law? Justify your answer. (*3 marks*)

3. These questions deal with the *simplification of a certain type of switching functions*. These switching functions are *similar to majority decoders*, i.e. their output depends on the number of input values with the logic value 1.
When you answer these questions, it might be helpful to recall that one particular method to calculate a simplified form (SSOP or SPOS) is using a Karnaugh-map.

a) Consider a switching function f of 4 inputs, A, B, C, D. The output $Y = f(A, B, C, D)$ should assume 1 if and only if exactly 3 of the 4 input variables assume 1. For example, we have $f(1, 0, 0, 1) = 0$ because only 2 (and not 3) of the 4 input values are 1. But $f(1, 1, 1, 0) = 1$ because in this case there are exactly three inputs with the logic value 1.
Can the CSOP of f be simplified into a shorter expression? Answer with 'yes' or 'no' and justify your answer. (*X marks*)

b) Now we generalize the specification of f from question a) as follows: We consider any switching function with more than two inputs that assumes 1 if and only if a certain predefined number of inputs assumes the logic value 1.
In other words, we consider any switching function that is specified by two numbers, n and m. n is the number of input variables and we require $n > 2$. m is a certain number between 0 and n, i.e. $0 \leq m \leq n$. The switching function shall assume 1 if and only if exactly m input values assume the logic value 1.
Can the CSOP of such a switching function be simplified into a shorter expression ('yes' or 'no') and if so, under which condition? Justify your answer. (*X marks*)

c) Now we consider another fully specified case, similar to a). Again, we have 4 inputs A, B, C, D and one output $Y = f(A, B, C, D)$. But this time Y shall assume 1 if *2 or 3* input values assume 1. Otherwise, Y shall be 0. For example,

$$f(0, 0, 0, 1) = 0, \quad f(1, 1, 0, 0) = 1, \quad f(1, 1, 0, 1) = 1, \quad f(1, 1, 1, 1) = 0$$

Calculate the *Simplified Product of Sums* (SPOS) notation of $f(A, B, C, D)$. (*X marks*)
There is only one single solution.

d) Use the Theoreme of De Morgan to rewrite the answer from c) into two other expressions. One expression should only contain NAND (and NOT) operations and the other expression should only contain NOR (and NOT) operations. (*X marks*)

4. The following questions deal with a *sequential majority decoder* which should be implemented as a *Moore-machine* with one binary input X and one binary output Y. Its input-output behaviour is described as follows: The output Y always depends on the three values of X which occurred at the *most recent three clock steps*. Whenever a clock step occurs, the output Y assumes that value which *occurred more frequently* during these three recent clock steps.

   For reasons of simplicity, we don't pay attention to the output at the very first two clock steps. The machine has exactly *8 states* that should correspond to the binary number composed by the input values of the most recent three clock steps.
   For example, if the last sequence of input values was $1, 1, 0$ (0 being most recent), then the machine is in state 6 (because the binary represenation of 6 is 110) and the output is $Y = 1$ because during the last three clock steps the input value $X = 1$ occurred more often (twice) than the input value $X = 0$ (once).

   a) Determine the *output table* of this Moore-machine, i.e. a table that shows how the output depends on the current state. (*3 marks*)

   b) Determine the *state transition table* of this Moore-machine. (*3 marks*)

   c) Give an example of an input sequence, that causes *alternating output*, i.e. $Y = 0, 1, 0, 1, 0, 1, \ldots$. (*2 marks*)

   d) How many flip-flop circuits would a circuit realization of this machine contain? Justify your answer. (*2 marks*)