King Fahd University of Petroleum and Minerals
College of Computer Sciences and Engineering
Department of Computer Engineering

COE 202 – Digital Logic Design (T121)

Homework # 03 (*due date & time: Saturday 10/11/2012 during class period*)

***** Show all your work. No credit will be given if work is not shown! *****

Problem 1 (15 points): Consider the Boolean function F(A, B, C, D)=Σ m(0, 1, 2, 5, 6, 7, 10, 12, 13, 14, 15).

1. Identify all the *prime implicants* and the *essential prime implicants* of F.
2. Simplify the Boolean function **F** into a minimal sum-of-products expression.
3. Simplify the Boolean function **F** into a minimal product-of-sums expression.

Problem 2 (20 points): Consider the Boolean function **F**(A, B, C, D)=Σ m(0, 10, 15), together with the don't care conditions **d**(A, B, C, D)=Σ m(1, 2, 4, 8, 11, 14).

1. Simplify the Boolean function **F** together with the don't care conditions **d,** into minimal sum-of-products expression.
2. Starting with the sum-of-products expression, implement the function using only **NAND** gates and **Inverters**.
3. Starting with the sum-of-products expression, implement the function using only **NOR** gates and **Inverters**.

Problem 3 (30 points): Design an **all NAND** circuit that accepts two 2-bit unsigned numbers $A = A_1A_0$ and $B = B_1B_0$. The circuit produces A – B when A > B, and produces A + B otherwise. Derive the simplified Boolean expressions of all outputs, and show the logic diagram implementation of the **all NAND** circuit.

Problem 4 (35 points):

(a) If **6-bit registers** are used, show the binary number representation of the decimal numbers (+23), (–23), (+11), and (–11) using the following representation systems:
   i. Signed magnitude system
   ii. Signed 1's complement system
   iii. Signed 2's complement system

(b) Provide the decimal equivalent of each of the following **signed 2's complement** numbers:
   i. 001101
   ii. 010011
   iii. 101101
   iv. 110011

(c) If **6-bit registers** are used, perform the following **signed 2's complement** arithmetic operations on the provided signed 2's complement numbers. For each case, state whether the result is correct or an **overflow** has occurred.
   i. 001101 – 101101
   ii. 010011 – 001101
   iii. 101101 + 110011