Name: Id#

COE 202, Term 162

Fundamentals of Computer Engineering

Quiz# 4 Solution

Date: Sunday, April 16

# 

**Q1.** In designing a combinational circuit that computes the function *f(X) =X2* – X for a 3-bit 2’s complement signed number *X,* ***where the output f(X) is an un-signed integer*:**

1. How many bits do we need for the output? [2 points]

X that produces that largest f(x) is -4. In this case f(x) = 16+4 = 20. So, the number of bit needed for the output is 5 bits.

1. Obtain the truth table for this circuit. [4 points]

|  |  |  |  |
| --- | --- | --- | --- |
| X2 X1 X0 | Decimal value of X | Decimal  value of f(X) | F4 F3 F2 F1 F0 |
| 0 0 0 | 0 | 0 | 0 0 0 0 0 |
| 0 0 1 | +1 | 0 | 0 0 0 0 0 |
| 0 1 0 | +2 | 2 | 0 0 0 1 0 |
| 0 1 1 | +3 | 6 | 0 0 1 1 0 |
| 1 0 0 | -4 | 20 | 1 0 1 0 0 |
| 1 0 1 | -3 | 12 | 0 1 1 0 0 |
| 1 1 0 | -2 | 6 | 0 0 1 1 0 |
| 1 1 1 | -1 | 2 | 0 0 0 1 0 |

1. Obtain simplified Boolean expressions of the circuit outputs in SOP form. [4 points]

F4 F3 F1 F0 Can be obtained directly from the truth table (no minimization can be done)

F4 = X2 X1’ X0’

F3 = X2 X1’ X0

F1 = X1

F0 = 0

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | X1’ | |  | |
|  | 00 | 01 | 11 | 10 |
| X2’0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |

K-map for F2:

F2 = X2’ X1 X0 + X2 X1’ + X2 X0’

**Q2.**

1. What is the **minimum** number of bits needed to represent integers in the range from –100 to +100 using sign-magnitude representation? [2 points]

**8-bits**

1. Show the binary representations of **+49** and **–49** using **10-bits** signed-magnitude, 1’s complement and 2’s complement representations (record your answers in the table below). [4 points]

|  |  |  |  |
| --- | --- | --- | --- |
| Decimal | Binary Signed-magnitude representation | Binary Signed-1’s complement representation | Binary Signed-2’s complement representation |
| - 49 | **1\_000\_110\_001** | **1\_111\_001\_110** | **1\_111\_001\_111** |
| + 49 | **0\_000\_110\_001** | **0\_000\_110\_001** | **0\_000\_110\_001** |

1. Perform the following operations on **6-bits** signed numbers **using 2’complement representation**. Check for overflow and mark clearly any overflow occurrences. [4 points]

|  |  |
| --- | --- |
| (1) 011100 – 011111  **= 011100 + 100001**    **= 111101**  Overflow: Yes/**No** | (2) 101111 + 100110  **= 010101**      Overflow: **Yes**/No |