# CSE352 Autumn 2013 Homework #3

Instructor: Mark Oskin
TAs: Vincent Lee, Mark Wyse

Due In Class 10/25/2013
Version 1.1

Please write your name and student ID at the top right corner of each page, and staple or paperclip your work together. We are NOT responsible for losing papers that were not stapled or paperclipped together.

Complete the following questions. Please write legibly and try to draw clean diagrams. Spaghetti wiring in circuit diagrams is difficult to grade. We will not grade work that is too heavily encrypted for us to read (i.e. we can't read it, we can't grade it). Please consider typesetting your work if you think that it may not be legible to the grader. You are encouraged to collaborate with your peers but you must turn in your own work. Justice will be enforced if you are caught cheating.

## Problem 1 *Multiplexor Implementation*

1. Design an 8-1 multiplexor using simple 2-input logic gates and/or inverters. Your design should take a 3-input select signal `sel`, and 8 one-bit input signals `x0` to `x7`. It should also output a single output bit `out`. Minimize the delay through the circuit. Use abstraction and hierarchy. (Hint: start with a 2-1 multiplexor).

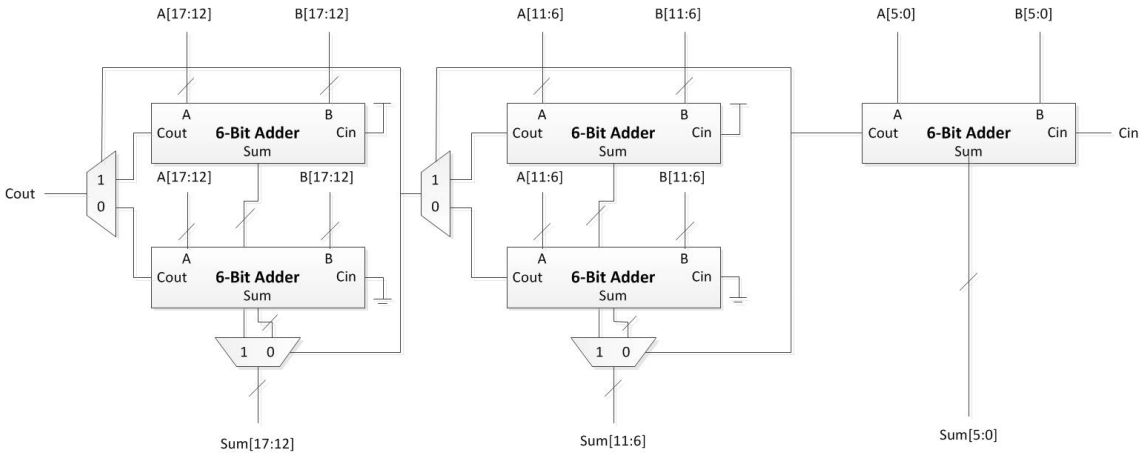
2. Using only the 2-1 multiplexors you designed in the previous part, design a circuit that implements the following truth table given inputs r0, r1, r2, r3, s0, s1, s2, s3:

| s0 | s1 | s2 | s3 | out |
|---|---|---|---|---|
| 1 | x | x | x | r0 |
| 0 | 1 | x | x | r1 |
| 0 | 0 | 1 | x | r2 |
| 0 | 0 | 0 | 1 | r3 |
| 0 | 0 | 0 | 0 | 0 |

You may also use ground or $V_dd$ as inputs. Minimize the amount of hardware required.

## Problem 2 *Carry Select Adders*

Ben Bitdiddle and Alyssa P. Hacker are arguing (again) about the best way to implement an 18-bit carry select adder. Assuming full adders and multiplexors incur the same delay of `n` Ben argues that the fastest implementation would be to use 3 groups of 6 adders and incur a delay of `8n` as shown below:




Alyssa obviously disagrees and thinks that there is an implementation that has a delay of `7n`. Does such an implementation exist? If so show how it can be done.

## Problem 3 *Multipliers*

Using two 12-bit adders, implement a multiplier that multiplies a 4-bit number `x` by 153. Minimize the hardware usage.

## Problem 4 *Field Programmable Gate Arrays*

(a) Implement a 2-LUT which takes a two bit input select input `s`, and four programmable lookup table values `v0, v1, v2,` and `v3`, and outputs a single bit `out` using only simple logic gates. Neatness and simplicity counts.

(b) Using only 2:1 multiplexors and 2-LUTs, design a 4-LUT which takes a 4 bit select input `s`, programmable lookup table values `v0...v15`, and outputs a single bit `out`.

(c) Suppose we want to implement the function$(s_0 + s_1)(s_2 + s_3)$ in our 4-LUT, what values of `v0...v15` must be programmed into this circuit in order for this circuit to implement this function? Assume the 4 bit select is indexed such that s = $[s_3, s_2, s_1, s_0]$.

## **Problem 5** *Bonus Question: MEME ME ME ME ME (Optional)*

Using the image below, generate and submit a meme reflecting how you feel about taking long walks on the beach with your FPGA. The more amusing the better: http://wp.streetwise.co/wp-content/uploads/2012/06/cute-puppy.jpg