ECE380: Pre-Lab #05: Adders/Subtractor

In this lab, you will use the Quartus II software package to design and simulate three adder designs. The requirements for this lab include 1) Completing Quartus Prime designs and printing necessary circuit schematics, 2) Providing simulation results, 3) Downloading the design to Altera's DE1 board, and 4) Submitting the laboratory report.

Prelab Tasks

1. Review: a) Addition of unsigned number; b) Signed numbers, signed addition and subtraction, arithmetic overflow; and c) Design of adder/subtractor circuits.

2. Review the following three design methods for adders:

Design A: Using a single *dataflow* VHDL, design a 4-bit adder for adding two 4-bit signed numbers (X and Y). Use adder4a.vhd code shown below:

File name: adder4a.vhd

```
--File adder4a.vhd
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;

ENTITY adder4a IS
  PORT(          Cin            :       IN STD_LOGIC;
                 X,Y            :       IN SIGNED(3 DOWNTO 0);
                 S              :       OUT SIGNED(3 DOWNTO 0);
                 Cout, Over:    OUT STD_LOGIC);
END adder4a;

ARCHITECTURE Behavior OF adder4a IS
  SIGNAL Sum,sum1,sum2 : SIGNED(4 DOWNTO 0);
BEGIN
  sum1 <= ('0'& X);
  sum2 <= ('0' & Y);
  Sum <= sum1 + sum2 + Cin;
  S <= Sum(3 DOWNTO 0);
  Cout <= Sum(4);
  Over <= Sum(4) XOR (X(3) XOR (Y(3) XOR Sum(3))); -- overflow
END Behavior;
```

Design B: Implement a 4-bit ripple carry adder using *structural* VHDL. Please review the ripple carry adder for the design. You will need two VHDL files 'fulladd.vhd' and 'adder4b.vhd' to implement the design. The 'fulladd.vhd' file will implement a single-bit full adder. The 'adder4b.vhd' file will create four instances of the single-bit fulladd design with two 4-bit data inputs X and Y. The circuit is to have a 4-bit data output 'S'. Your design should also generate an output carry '*Cout*'.

You need to create test vectors in functional simulations to test your design for Designs A and B, in the prelab. Please refer to Appendix: Test Vectors.

Design C: Implement a 4-bit adder using the LPM_ADD_SUB module in Quartus II in a schematic file, file adder4c.bdf. The circuit is to have two 4-bit data inputs X and Y, a carry-in bit *Cin*, and an *add_sub* control input. The X and Y data values are to be SIGNED. The circuit is to have a 4-bit data output 'S', an output carry '*Cout*', and an overflow output '*Over*'.

3. Please complete Designs A and B (Quartus II project, VHDL files, and functional simulations) before your lab session.

Appendix: Test Vectors for Adders

You can create test vectors when generating functional simulations in University Program WVF editor.

Step 1. You need to establish a project and complete your design via either VHDL programming or schematic capture. After successful compilation, launch University Program WVF editor for function simulations.

Step 2. In the Waveform Editor, you need to obtain input and output nodes of the circuit. You can specific the value for single bit signals, for example *Cin* in Design A. So far, the procedure is the same with the previous lab. Please refer to Lab 01 and 02 for detailed procedures.

Please use the vector form of the signals, X, Y, and S in this lab, when available.

Step 3. For the input vector signals, X, Y, it is possible to specify an arbitrary value. To do so, select the vector signal and highlight the desired signal interval, press the **Arbitrary Value** icon in the toolbox (the icon is marked with a question mark). This brings up a dialogue window. You can select "*Hexadecimal*" as its Radix and specify a particular number for that vector, either X or Y in this lab.

Please refer to Quartus Simulation Tutorial, supplement_quartusii_simulation_vhdl.pdf, distributed along with this prelab, for more details.

It is nearly impossible to test all the input possibilities for these adders. But you need to cover test vectors in a wide range of input/output scenarios, for example addition with *Cin*=0, addition with *Cin*=1, addition generating *Cout*=0, addition generating *Cout*=1, addition generating *Over*=0, addition generating *Over*=1, etc. The following table provide a set of test vectors for Designs A and B. 0x denotes a hexadecimal number. Please test these test vectors for Designs A and B in functional simulations. Output values are marked in red, to show that they are the expected values, which should not to be specified.

Input	*Cin*	1	0	1	0	1	0	1	0
Input	X	0x2	0x7	0xC	0xB	0x7	0x7	0x9	0x9
Input	Y	0x3	0xE	0x2	0xE	0x2	0xE	0x2	0xE
Output	S	0x6	0x5	0xF	0x9	0xA	0x5	0xC	0x7
Output	*Cout*	0	1	0	1	0	1	0	1
Output	*Over*	0	0	0	0	1	0	0	1

Pre-Lab (30%)	**Score**	**TA initial**
Design A (15%)		
Design B (15%)		

Report (70%)	
10% Introduction	
10% Procedures	
20% Results	
30% Conclusions	

Lab Grade	