ECE380: Pre-Lab #06: Adder Design Tradeoffs

In this lab, you will use the Quartus II software package to design and simulate two 16-bit, and one 4-bit adder designs. The requirements for this lab consist of completing Quartus II designs, downloading designs to the Altera DE1 board, printing circuit diagrams, VHDL files, simulation results, and the laboratory report.

1. **Design A:** Implement a 4-bit subtractor using *structural* VHDL. The circuit will have two 4-bit data inputs (A and B), a 4-bit data outputs (S), a carry-out bit (*Cout*), and an overflow flag. You need two VHDL files (fulladd.vhd) and (sub4b.vhd) to implement the design. VHDL code, fulladd.vhd, will implement a single-bit full adder. The VHDL file, sub4b.vhd, will create four instances of the single-bit full adder. Four NOT gates will be needed to create the subtractor. You need a XOR gate to generate the overflow.

   Devise a set of test input vectors (values) that will be used to verify that the circuit is working properly.

2. Understand carry look ahead adders (CLA) and derive the expression for the sum and carry out bits.

3. **Design B**: Implement a 4-bit CLA using *data-flow* VHDL. The circuit will have two 4-bit signed inputs (A and B), a 4-bit data outputs (S), and a carry out bit (*Cout*). Only use simple assignment operators and basic keywords (AND, OR, NOT, XOR) in VHDL for describing your design. You need a single data-flow VHDL design (adder4c.vhd) to implement the design.

4. **Design C (Bonus 20 points):** Write VHDL code for a BCD-to-seven segment LED display converter with four inputs, h_3-h_0, and a seven-bit output suitable for driving a seven segment LED display on the Altera DE1 board.

   Connect three BCD-to-seven segment LEDs to the subtractor in Design A, to display the inputs, A and B, and the output, S.

   That is the 4-bit signals of the input, A and B, and the output (S), need to serve as the input to the BCD-to-seven segment LEDs.

   For the output S, you may declare the sum S as INOUT:

   ```
           S: inout std_logic_vector(3 downto 0);
   ```

   ```
   ---VHDL code for a BCD-to-seven segment LED display
   LIBRARY ieee;
   USE ieee.std_logic_1164.all ;
   ENTITY DesignA IS
   PORT ( h: IN STD_LOGIC_VECTOR(3 DOWNTO 0) ;
   ```

```
leds: OUT STD_LOGIC_VECTOR(0 TO 6) ) ;
END DesignA ;
ARCHITECTURE Behavior OF DesignA IS
BEGIN
PROCESS (h)
BEGIN
        CASE h IS --abcdefg
                WHEN "0000" => leds<= "0000001" ;--0
                WHEN "0001" => leds<= "1001111" ;--1
                WHEN "0010" => leds<= "0010010" ;--2
                WHEN "0011" => leds<= "0000110" ;--3
                WHEN "0100" => leds<= "1001100" ;--4
                WHEN "0101" => leds<= "0100100" ;--5
                WHEN "0110" => leds<= "0100000" ;--6
                WHEN "0111" => leds<= "0001111" ;--7
                WHEN "1000" => leds<= "0000000" ;--8
                WHEN "1001" => leds<= "0000100" ;--9
                WHEN "1010" => leds<= "0001000" ;--A
                WHEN "1011" => leds<= "1100000" ;--B
                WHEN "1100" => leds<= "0110001" ;--C
                WHEN "1101" => leds<= "1000010" ;--D
                WHEN "1110" => leds<= "0110000" ;--E
                WHEN "1111" => leds<= "0111000" ;--F
                WHEN OTHERS => NULL ;
        END CASE ;
END PROCESS ;
END Behavior ;
```

5. Write a paragraph describing how you would go about creating test vectors (a set of inputs that test the system's functionality), for the designs above.

Pre-Lab (30%)	**Score**	**TA initial**
20% Designs		
10% Paragraph		
Report (70%)		
10% Introduction		
10% Procedures		
20% Results		
30% Conclusions		
Lab Grade (100%)		
Bonus (20 pts)		