Lab Number 3 – Basic Adders

(To be performed the week of September 10, 2012)
ANALYSIS AND DESIGN OF HALF, FULL, AND RIPPLE-CARRY ADDERS

(100 POINTS)

PURPOSE/OUTCOMES

To introduce you to the basic components of arithmetic circuits. After completing this lab, you will have demonstrated an ability to design half-adders and full-adders, capture your designs using Quartus II, verify your designs using Qsim, construct your designs on a solderless breadboard, and test your implementations using an IDL-800. You will also use full-adders to implement a two-bit ripple-carry adder.

BACKGROUND

The two-input exclusive-OR (XOR) gate that you studied in Lab 1 is useful in the implementation of adders and other circuits and can be generalized to more than two variables. Logic symbols and equations for XOR2, XOR3, and XOR4 gates are shown below.

\[
\begin{align*}
\text{XOR2:} & \quad f = a \oplus b = \sum m(1,2) \\
\text{XOR3:} & \quad f = a \oplus b \oplus c = \sum m(1,2,4,7) \\
\text{XOR4:} & \quad f = a \oplus b \oplus c \oplus d = \sum m(1,2,4,7,8,11,13,14)
\end{align*}
\]

Note that the minterms in the lists are all of the minterms that contain an odd-number of 1's for the given number of variables. As such, these functions are sometimes referred to as odd-parity functions.

Another useful function is the majority function which is defined as equaling logic-1 when a majority of its inputs are equal to logic-1 and, otherwise, equals logic-0. Minterm lists for MAJORITY2, MAJORITY3, and MAJORITY4 functions are shown below.

\[
\begin{align*}
\text{MAJORITY2:} & \quad \text{MAJORITY2} = \sum m(3) \\
\text{MAJORITY3:} & \quad \text{MAJORITY3} = \sum m(3,5,6,7) \\
\text{MAJORITY4:} & \quad \text{MAJORITY4} = \sum m(7,11,13,14,15)
\end{align*}
\]

Half-adders (HA) and full-adders (FA) are one-bit binary adders that can be cascaded to implement multi-bit adders and subtractors. Logic symbols and truth tables for half and full adders are given below.
PRELAB WORK – Must be completed prior to your lab session

1. Design implementations of half-adders and full-adders that utilize XOR2, AND, OR, NOT and/or NAND gates. Minimize the number of ICs needed in each design.

2. Capture and verify your designs using Quartus II and Qsim.

3. Write Verilog code that describes your full-adder from part 1 design.

4. Capture and verify your Verilog code with Quartus/Qsim.

5. Write Verilog code for a full adder.

LAB ASSIGNMENT

1. Construct and exhaustively test your full-adder implementation from pre-lab part 1. Recall that exhaustive testing means to apply all possible input combinations.

2. Construct a second full-adder and connect the two full-adders to form a two-bit ripple-carry adder. Exhaustively test the ripple-carry adder. The carry-in for the least-significant bit should be logic-0 and the carry-out for the most-significant should be displayed as an output.

Note: Record both prelab and inlab work in your laboratory notebook.