Ryan Donahue

2/14/17

Lab 3

Digital Logic (T@11:30)

**Two-Bit Ripple Carry Adder**

**Introduction**

The goal of this lab was to design, verify, and implement a 2-bit binary adder. This circuit accepted two, 2-bit unsigned numbers and produced a three-bit result. First, I made an adder circuit, this dealt with two digits in the same column. It would output the sum of the two and move a “carry digit” to the next column of more significance. Next, I connected two adder circuits to form a 2-bit *ripple carry adder*. The lab introduced hierarchal design.

**Procedure**

Prior to entering lab, I created a circuit diagram that accepted three numbers A\_n, B\_n and a C\_in and outputs two numbers S\_n and C\_out. The circuit took the three 1-bit digits, and adds them into a 2-bit output. S\_n is the least significant column sum, and C\_out (msb) is copied to the C\_in of the next adder, made of an identical adder circuit. Through use of a truth table and k-maps I minimized the logic of addition with a carry out into a viable circuit. Upon entering the lab, I created a new project and drafted this schematic, which is attached to this report.

After creating the schematic, I created a test bench and tested the functionality of the circuit. The operation of addition was successful. The behavioral simulation is attached to this sheet.

Next, using the define area constraints tool, I assigned the board to take inputs from the four leftmost switches and output to the three leftmost LEDs. Once assigned, I performed a post-route simulation. Attached is the post-route simulation of the circuit.

Lastly, I downloaded the design to the Digilent board and verified using the switches and LEDs, Attached is my data sheet with a signature from the TA showing that my adder functioned as described.

**Conclusion**

This lab showed the power of hierarchal designs. Without the reciprocity, making an adder of large numbers would take hours to design and implement. With the simple reciprocal design of a ripple carry adder, this process is as simple as copy and pasting. Overall, I gained confidence with the functionality of Xilinx, practiced logic design and created a fully functioning adder in lab 3.