EE-281 Logic Design Lab

Lab #3

Introduction to Verilog and the Spartan3/Basys2

## Name: Ajin Sunny and Derek King

## Date of Experiment: February 9th, 2015

## Date of Completion of Report: February 14th, 2015

**Introduction**

In this lab we will be learning how to use Verilog to design a circuit. In addition, this lab will familiarize us with the Spartan3 programmable logic board. This knowledge will be helpful to us as the following labs will include more and more Verilog programming.

**Experiment and Description**

**Part #1**

We will design a logic circuit that will implement a digital version of rock, paper, scissors. We will assign each hand a 2 digit binary value, and display the hand selected on the seven segment display.

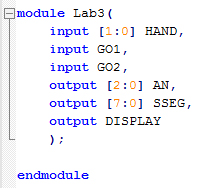
The following is the truth table for the output on the seven segment display based on the input switches.

Truth Table:

|  |  |  |
| --- | --- | --- |
| SW1 | SW2 | F(SW1, SW2) SSEG OUTPUT |
| 0 | 0 | r |
| 0 | 1 | P |
| 1 | 0 | S |
| 1 | 1 | off |

The following Verilog module (Fig 1) that we used to implement this above design:

Fig 1



**Part #2**

In this part of the lab, we were asked to implement a single digit binary to decimal converter using a single seven segment display and four switch inputs. The range of values is between 0000 and 1111 where the following truth table is the logic we will implement:

Truth Table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SW1 | SW2 | SW3 | SW4 | F(SW1, SW2, SW3, SW4) |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 2 |
| 0 | 0 | 1 | 0 | 3 |
| 0 | 0 | 1 | 1 | 4 |
| 0 | 1 | 0 | 0 | 5 |
| 0 | 1 | 0 | 1 | 6 |
| 0 | 1 | 1 | 0 | 7 |
| 0 | 1 | 1 | 1 | 8 |
| 1 | 0 | 0 | 0 | 9 |
| 1 | 0 | 0 | 1 | o |
| 1 | 0 | 1 | 0 | o |
| 1 | 0 | 1 | 1 | o |
| 1 | 1 | 0 | 0 | o |
| 1 | 1 | 0 | 1 | o |
| 1 | 1 | 1 | 0 | o |
| 1 | 1 | 1 | 1 | o |

As one can see, the output is simply the binary represented of the four input switches. Since we are only using one seven segment display, we can only display digits 0-9. When the binary value is larger than nine, we output a ‘o’ to the display to signify that there is an overflow.

The following Verilog module (Fig 2) and UCF file (Fig 3) were the bits of code we used to model the above scenario:

Fig 2

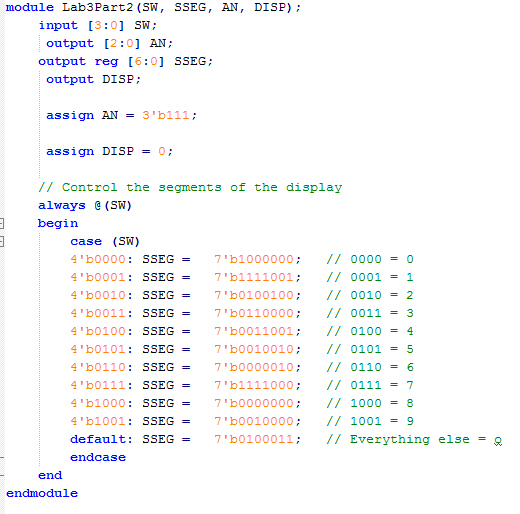
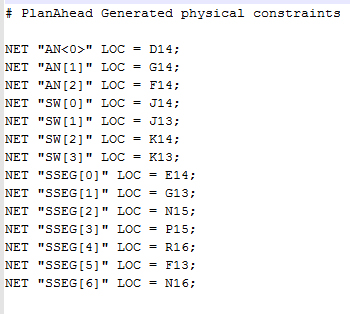


Fig 3



With the above implementation, our Spartan3 programmable logic board implemented the truth table mentioned earlier.

**Conclusions**

In conclusion, this experiment taught us how to properly code in Verilog. Verilog is a very powerful tool because it allows us to design the output of the circuit in words without needing a truth table. In addition, Verilog can easily be translated and tested on a programmable logic board, like the Spartan3 we used in this lab. As labs get more and more complex, we will need Verilog since it is a powerful tool we can use to simplify our designs. This lab was a great introduction to the software behind Verilog as well as the process involving implementing a piece of Verilog code.