EE-281 Logic Design Lab

Lab #7

Memory Elements, a real alarm, a counter, and hazards

## Name: Ajin Sunny and Derek King

## Date of Experiment: March 23rd, 2015

## Date of Completion of Report: March 27th, 2015

# **Introduction**

In this Lab we will study Memory elements and Hazards by designing the logic using digital logic gates. In Digital Logic hazards are mainly caused due to delay in the timing which is caused when there is a change in the input which causes a change in the output due to the delay caused. In this Labs we will be performing two experiments, the first experiment involves us to build a simple counter and check the stability of the counter , if they are unstable we will have to debounce it to maintain the stability. Secondly we will have to build a circuit which contains a static hazard and fix this static hazard.

In the second part of this Lab we will perform another experiment which involves us to build a rock paper scissor game using the Spartan 3 FPGA board. The game will be played between the human(player) and the Spartan 3 FPGA board(computer).

# **Experiment Description**

**Experiment 1**

In the first part of the experiment we first built simple counter using the 7490TTL chip from our kit and used the protoboard presented to us on the lab station to check the results of the counter. The results from the counter were not stable. The LED’s on the protoboard displayed instability which results in proving that the counter had bouncing outputs.

In the second part of this experiment we used the debounce switch to debounce the output. After connecting the input of the counter to the debounce switch , we observed the output of the result from LED’s on the protoboard and we observed that the results were stable and the LED had a stable response.

Next we were asked to build a circuit which has a static hazard in it and correct this static hazard. We chose to build the logic of a multiplexer which has a static hazard in it. Below shown is a circuit diagram for a multiplexer with static hazard.

Output = X1X2 + X1’ X3

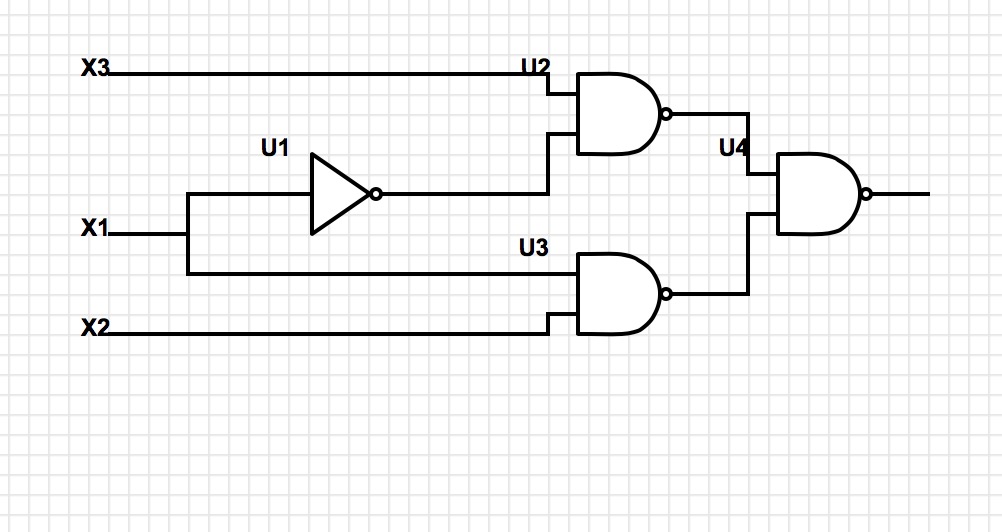


Figure 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| X2X3  X1 | 00 | 01 | 11 | 10 |
| 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 |

Figure 1 shows a multiplexer circuit which contains a static hazard caused by the delay in the input and this static hazard will be reduced making some changes to minimization of the logic, this can be done by looking into the KMAP and truth table of the multiplexer circuit.

|  |  |  |  |
| --- | --- | --- | --- |
| X1 | X2 | X3 | F |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

Now if you look at the above Truth table and the KMAP we find a redundant case in which the adjacent one’s of different color is not included in the minimized circuit in order to avoid the static hazard we will have to include this redundant case. Therefore the final equation of the output will be :

Output = X1X2 + X1'X3 + X2X3

Now if you carefully observe the redundant case is **X2X3 .** This case avoids the static hazard present in the circuit. Therefore the final circuit can be drawn as shown below.

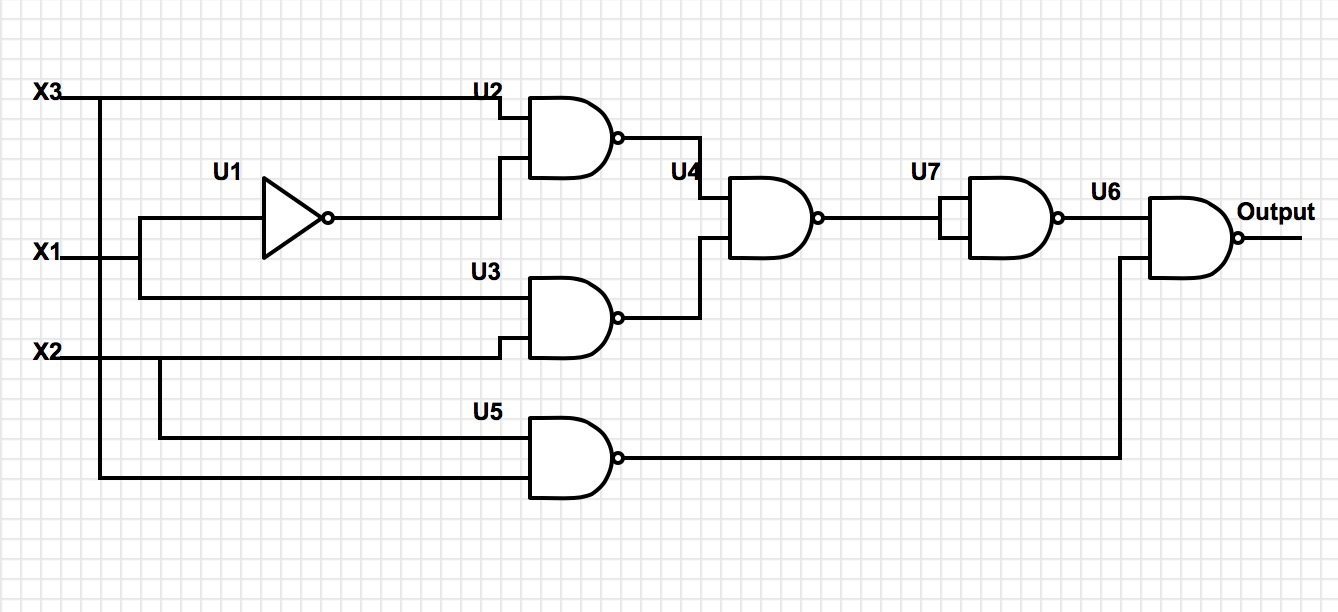
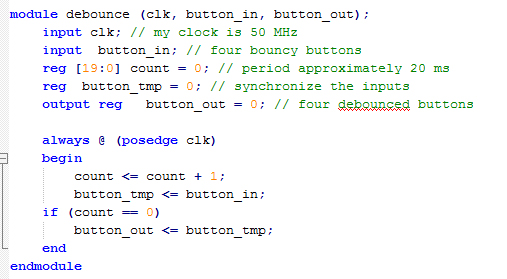


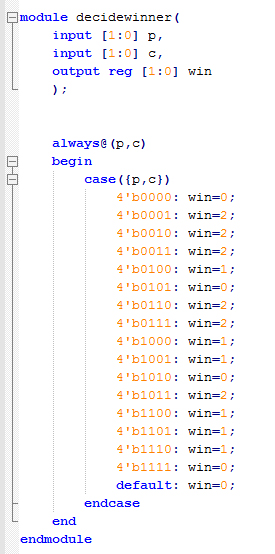
Figure 2.1

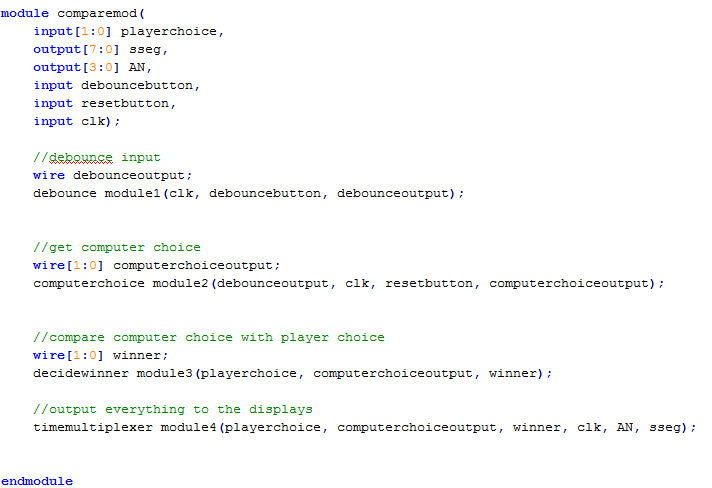
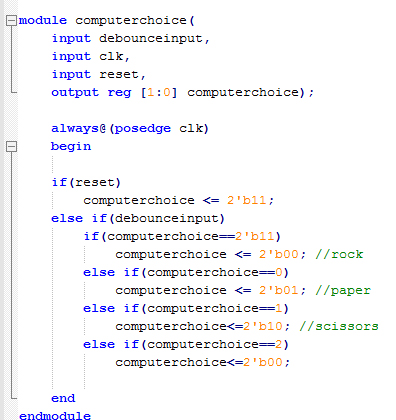
**Experiment 2**

In the second part of this experiment we were asked to improve upon the design we did in Lab #5. In lab #5, we implements a rudimentary rock, paper, scissors game in when the Spartan3 board compared the winner based on what selections were made on the set of switches. For this part, the goal was to add debounce to a button, that when pressed would start cycling between rock, paper, and scissors and when released, locked in the value and used that as a ‘random’ value for the computer. Then that random value and player selected value using switches would be compared and a winner would be displayed. In addition, three displays must be used, requiring a time multiplexor circuit to control the displays.

Below is the Verilog code that we used to implement Lab #7, complete with every function described above:







# **Results**

In part one, adding the redundant term was crucial in order to avoid a static hazard. We will need to keep this in mind as we move forward and build more complex circuits. The result for part two was as we expected. The computer’s ‘random’ choice and the player’s choice were both evaluated to determine a winner. The game played exactly as we expected it to.

# **Conclusion**

In conclusion, this lab helped us to understand the possible errors that are involved with both static and dynamic hazards. We have learned tools that we can use to avoid these hazards in the future in case they arise. Part two of the lab taught us how to manage a larger Verilog project. It improved our understanding of how Verilog worked and left us with confidence that we can work on projects of increasing magnitude.