Andrew Stites

EEE 64 – CpE64 Section 2

Wednesday

Lab 8-9

Samuel Wekanda

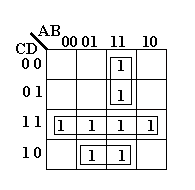
Lab Objective/Goal: Design more combinational circuits using Multisim, gain more experience in using K-maps to get equations for any function, and using ModelSim with Verilog HDL to simulate in a wave format.

Lab Preparations and Challenges:

I created a K-Map utilizing the truth table with 4 inputs and the output only being HIGH if, and only if, there were two HIGH inputs next to one another in the inputs. For instance, 0110 would yield a HIGH output due the two HIGH inputs being in close proximity. In the K-Map, there were a total of 8 “1”s aligned to created three groupings: one being four ones and the other two being two ones. The boolean equation that came of it was F = AB + BC + CD. I used that boolean equation to create the combinational logic design in MultiSim that consisted of three AND gates and one three-inputted OR gate. In Quartus, I created a Verilog HDL file named “labcircuit” and coded my combinational logic design which compiled successfully. I created a second Verilog HDL file named “labcircuit\_tb()” which was used as the testbench for the initial Verilog HDL file named “labcircuit”. Once that compiled successfully, I ran a simulation through Quartus that opened up ModelSim. I compiled the Verilog HDL file named “labcircuit\_tb()” and accessed it under the “work” drop down section in the Library module. I copied the variables to the Wave editor so that they can be shown utilizing green lines that are either HIGH for 1 or LOW for 0. I ran the simulation at “160 ps” to fully simulate all inputs.

Lab Results:

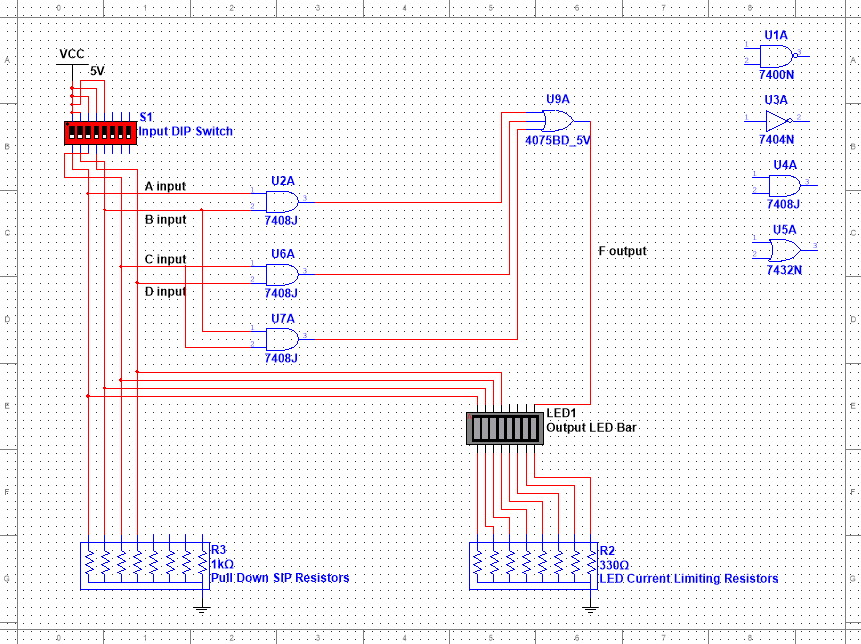
**K-Map and Boolean Algebra:**



F = AB + BC + CD

This K-Map has a total of 8 High inputs that create three groupings allowing for the creation of the boolean equation: F = AB + BC + CD. This boolean equation allows for only close inputs being High to output a High value of 1.

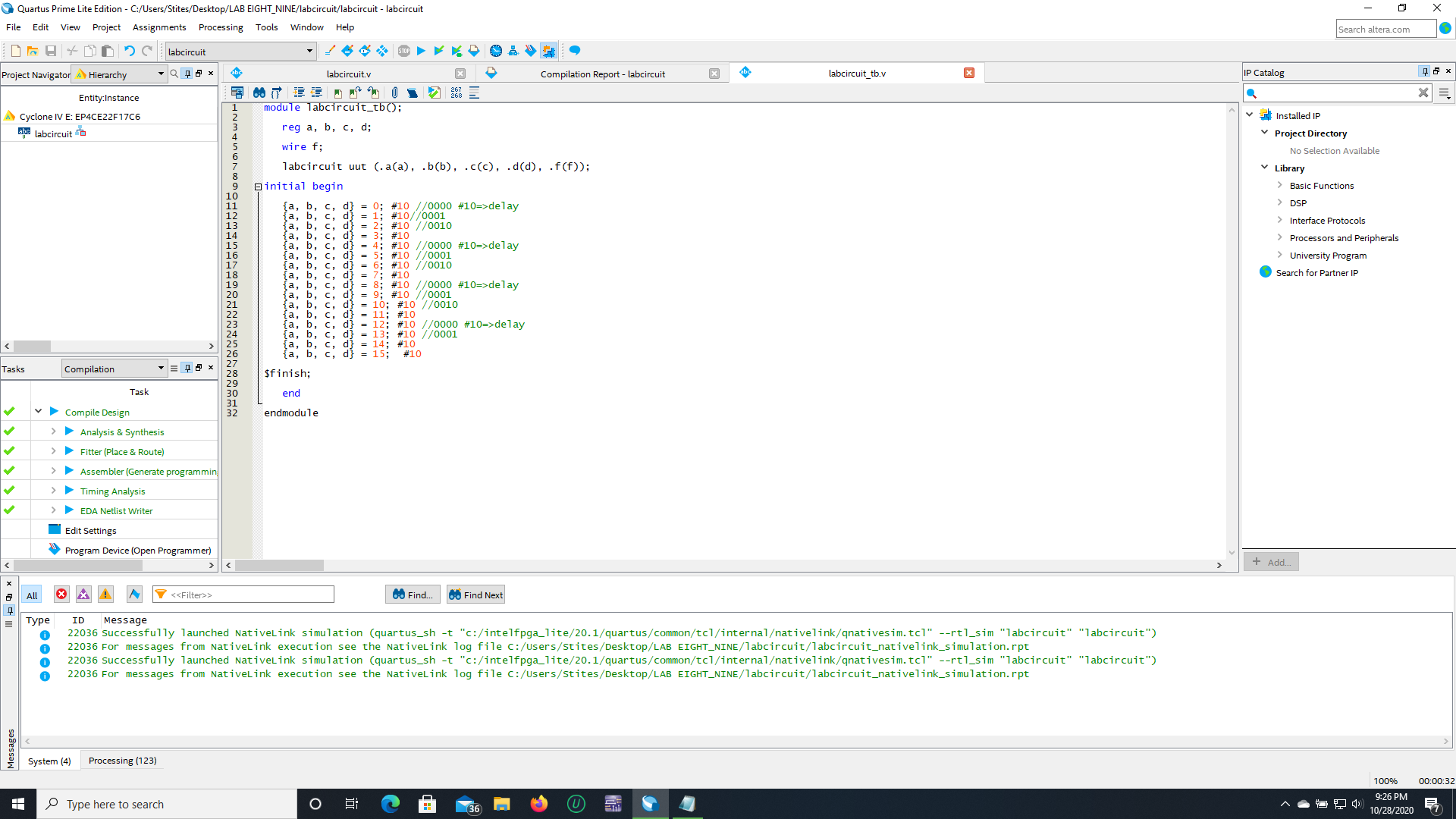
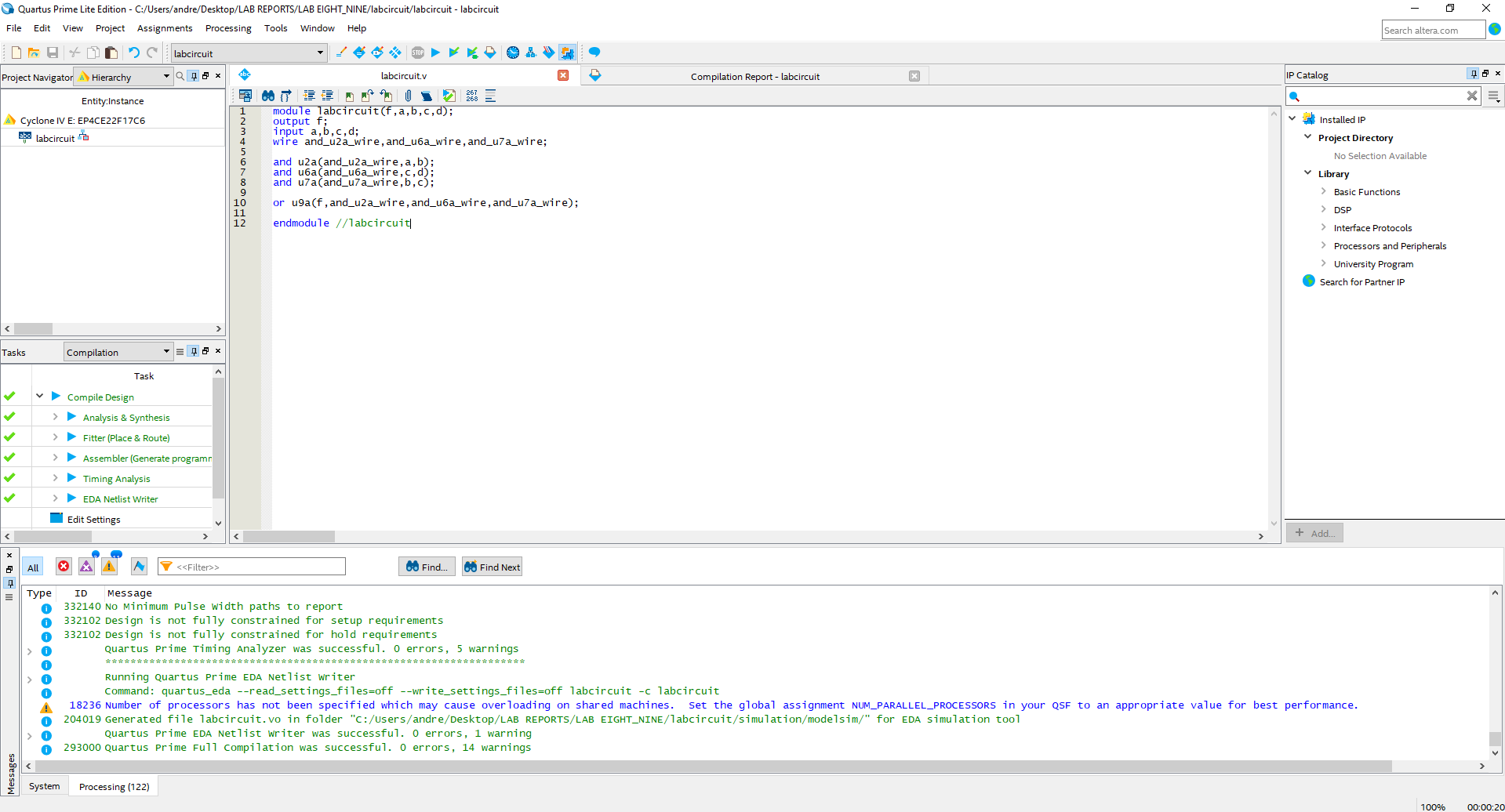
**Multisim Combinational Circuit:**



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | D | F(MultiSim) |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

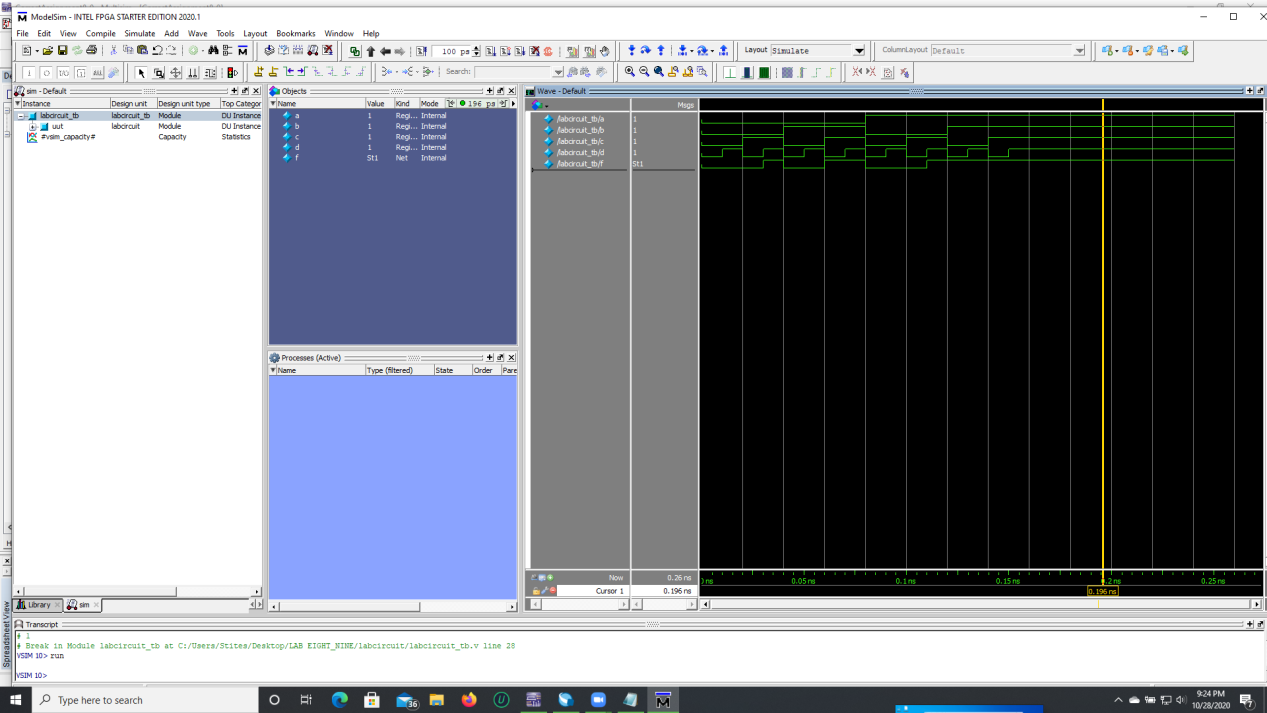
The multisim file shown above portrays the boolean equation of F = AB + CD + BC. The table below shows the outputs only being High if, and only if, there are two High inputs next to each other. For instance, A and B are High, which would result in a High output for F. Conversely, only A High and B Low would result in a Low output for F.

**Quartus with Verilog HDL:**



This portrays the combinational circuit that was created in Multisim written in Verilog HDL. The module is named “labcircuit” with output “f” and inputs “a”, “b”, “c”, “d”. Next, there are wires named “and\_u2a\_wire”, “and\_u6a\_wire”, “and\_u7a\_wire” that symbolize the outputs of the and gates used in the combinational circuit from Multisim. Following suit are the AND gates set-up with the respective wire as the output and the respective input going into the correct AND gate. The final gate is the OR gate that uses the output “f” and the AND gates’ output wires: “and\_u2a\_wire”, “and\_u6a\_wire”, and the “and\_u7a\_wire” as inputs. This leads to the end of the module. The second photo introduces the Verilog HDL code for the testbench needed to simulate a counter from 0000 to 1111 for the input variables.

**ModelSim with Altera Simulated in Waveform:**

****

The components of the combinational circuit have been implemented into the wave editor with the inputs “a” through “d” and the output “f”. This is using the compiled file “labcircuit\_tb()” for the simulation. The testbench allowed for the cycling from 0000 to 1111 in our initial Verilog HDL file that contained our combinational code setup. The wave form is presented portraying the inputs and output with all the lines being green.

**Truth Table for MultiSim and Expected:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A | B | C | D | F(MultiSim) | F(Expected) |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 |

This table portrays the Multisim and expected results for the inputs A through D. As shown, they are identical, which is a desired result.

Conclusion:

These labs took a lot more time and meticulous effort to make sure every part worked properly. The need for every part to compile and run so that the following step could be accomplished made these labs more difficult, but possible. I learned a great amount of Quartus and ModelSim and feel confident regarding the future labs. I initially had an unsatisfactory K-Map that produced a boolean equation that was not reduced. I re-circled the “ones” on the K-Map to get a better equation and was able to get the most reduced equation before starting any coding or simulating. If I fully read through the procedure regarding Lab 9, I believe I would have been able to complete the Wave form simulation quicker instead of a few days.