University of Guelph

ENGG*2410: Digital Design

Combinational Logic Design “Trip-Genie” via Schematic Capture

Group (# 8)

Layal Alzaydi (1041786)
Josiah Varghese (1043468)
Giuliana Lowry (1013417)
Mazen Hassan (0984322)

Preformed On: September 30th, 2019

Due Date: October 8th, 2019

Problem Statement

The problem introduced in this lab is that an owner of a small company is trying to find a more efficient way for his employers to sell his products. The problem is that his salespeople often make poor choices on routes between cities and spend more time on the road than they do selling his products. The goal for this lab is to use figure 1 and find the shortest route connecting any two cities. The group analysed the problem and broke it down by drawing sketches that shows the fastest routes between each pair of cities. These sketches were then used to find a solution for the owners problem.

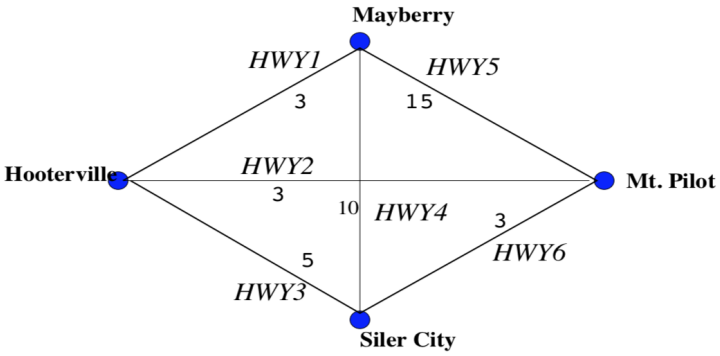

Figure 1: Proto-board used in the Lab.

Assumptions and Constraints

Some of the assumptions and constraints used in this lab are as follows:

- Assumed that Xilinx would perform any optimizations needed
- No hand optimizations were done
- Only AND/OR gates were to be used
- Schematic drawings were used to solve the problem

System Overview & Justification of Design

The purpose of this lab is to find a solution for a given problem. The solution includes formulating the right circuit diagram. The circuit is then tested, debugged and verified so that the Xilinx software simulation and hardware implementation are correct.

During this lab, the group had to design a circuit diagram for the trip genie logic using Schematic drawings. First a truth table was formed from the given problem statement then Boolean equations were generated from the table. From these equations, a circuit diagram was derived using AND/OR gates. The trip genie Schematic included four inputs (name of cities) and six outputs (highway numbers). The Schematic was built so that when two switches are flipped to indicate the starting and ending cities, the LEDs would light for those highways that are on the fastest route connecting the two cities.

After checking that the Schematic was designed correct, a UCF file was then created for all the possible cases. There are 16 possible cases which are shown in figure 2. The FPGA board was then programmed and tested. Everything worked fine, with the LED lighting up to the corresponding highways.

Truth Table

- C1→ Hootervill
- C2→ Mayberry
- C3→ Mt.Pilot
- C4→ Siler City

C1 Input	C2 Input	C3 Input	C4 Input	HWY1 output	HWY2 output	HWY3 output	HWY4 output	HWY5 output	HWY6 output
0	0	0	0	0	0	0	0	0	0
0	0	0	**1**	0	0	0	0	0	0
0	0	**1**	0	0	0	0	0	0	0
0	0	**1**	**1**	0	0	0	0	0	**1**
0	**1**	0	0	0	0	0	0	0	0
0	**1**	0	**1**	**1**	0	**1**	0	0	0
0	**1**	**1**	0	**1**	**1**	0	0	0	0
0	**1**	**1**	**1**	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0
1	0	0	**1**	0	0	**1**	0	0	0
1	0	**1**	0	0	**1**	0	0	0	0
1	0	**1**	**1**	0	0	0	0	0	0
1	**1**	0	0	**1**	0	0	0	0	0
1	**1**	0	**1**	0	0	0	0	0	0
1	**1**	**1**	0	0	0	0	0	0	0
1	**1**	**1**	**1**	0	0	0	0	0	0

Figure 2: Truth Table

Logic Functions

From the Truth table (Figure 2) above the following information was derived:

- $F_1 = HWY_1$

  $$F_1 = (\overline{C_1} \cdot C_2 \cdot \overline{C_3} \cdot C_4) + (\overline{C_1} \cdot C_2 \cdot C_3 \cdot \overline{C_4}) + (C_1 \cdot C_2 \cdot \overline{C_3} \cdot \overline{C_4})$$

- $F_2 = HWY_2$

  $$F_2 = (\overline{C_1} \cdot C_2 \cdot C_3 \cdot \overline{C_4}) + (C_1 \cdot \overline{C_2} \cdot C_3 \cdot \overline{C_4})$$

- $F_3 = HWY_3$

  $$F_3 = (\overline{C_1} \cdot C_2 \cdot \overline{C_3} \cdot C_4) + (C_1 \cdot \overline{C_2} \cdot \overline{C_3} \cdot C_4)$$

- $F_6 = HWY_6$

  $$F_6 = (\overline{C_1} \cdot \overline{C_2} \cdot C_3 \cdot C_4)$$

Circuit Diagram

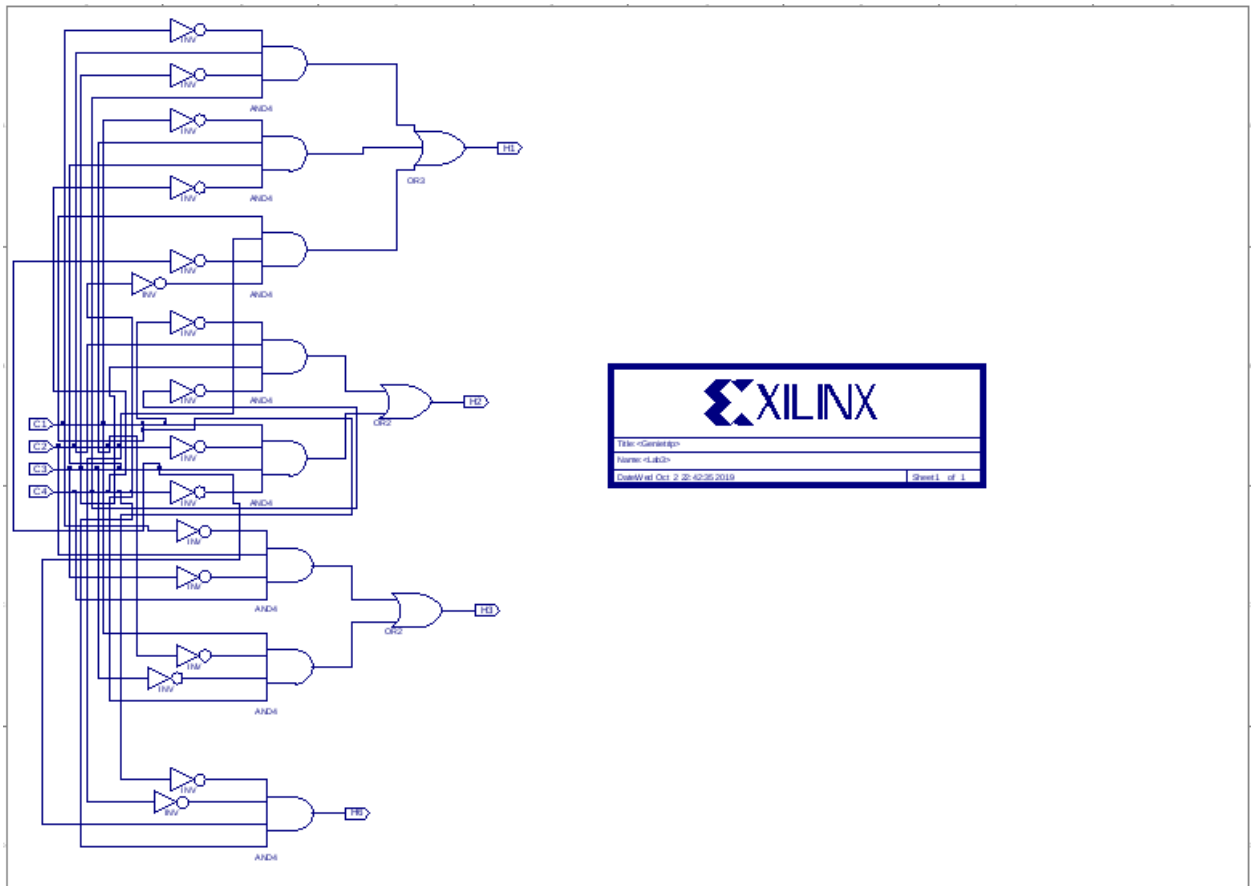

Figure 3: Circuit Diagram

Symbol Diagram

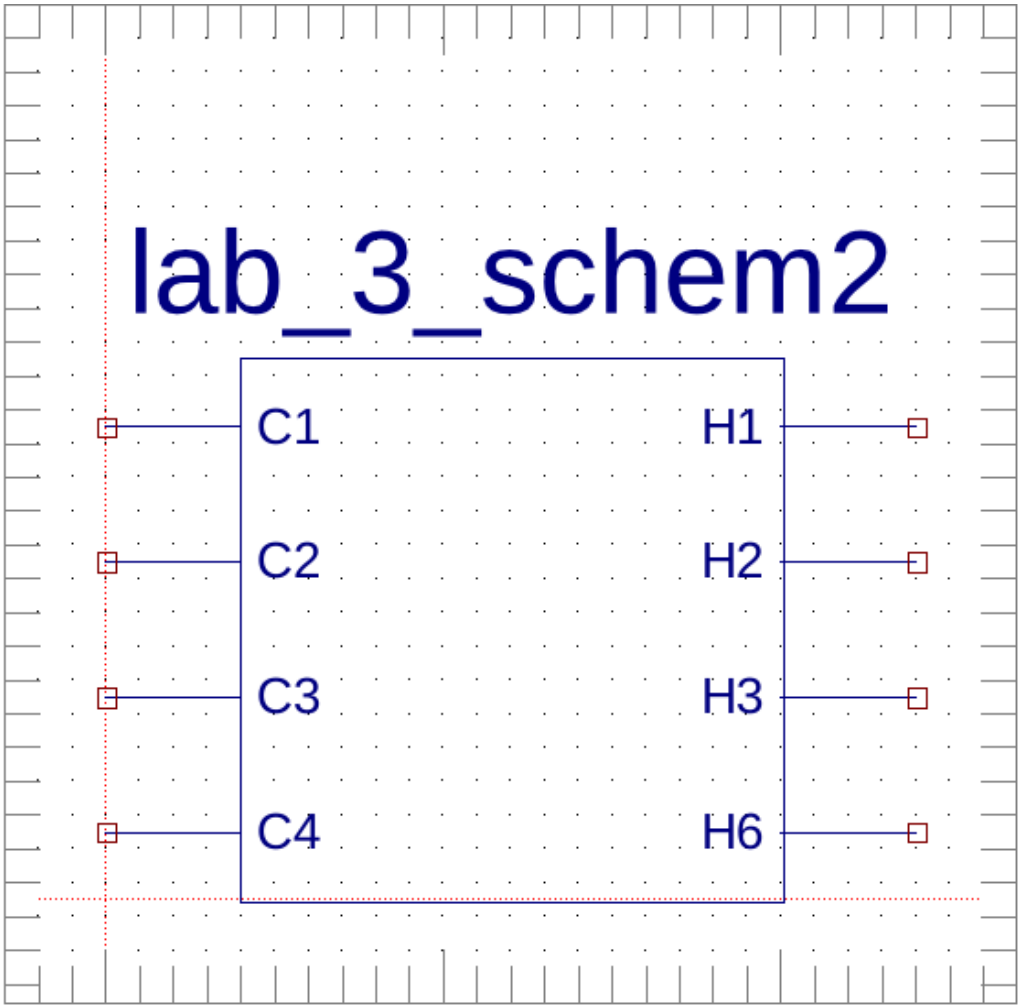

Figure 4: Symbol Diagram

Simulation

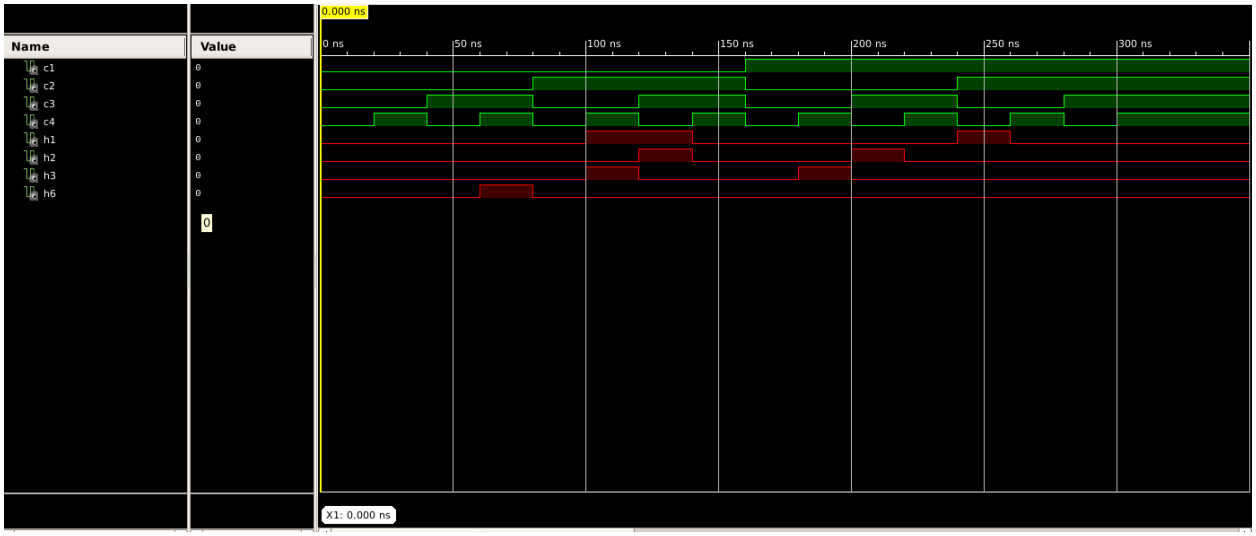

Figure 5: Simulation

The simulation above (Figure 5) is the simulation of The Trip Genie. The four cities are represented as the inputs (c1, c2, c3. c4) and the highways listed as the outputs (h1, h2, h3, h6). The fastest route from each of the respective cities to another city is shown by the simulation.

Test Bench

```
57  -- *** Test Bench - User Defined Section ***
58     tb : PROCESS
59     constant period: time:= 20ns;
60
61     BEGIN
62     C1  <= '0';
63     C2  <= '0';
64     C3  <= '0';
65     C4  <= '0';
66     WAIT for period;
67     assert (H1 = '0' AND H2 = '0' AND H3 = '0' AND H6 = '0')
68     report "test failed for input combination 0000" severity error;
69
70     C1  <= '0';
71     C2  <= '0';
72     C3  <= '0';
73     C4  <= '1';
74     WAIT for period;
75     assert (H1 = '0' AND H2 = '0' AND H3 = '0' AND H6 = '0')
76     report "test failed for input combination 0001" severity error;
77
78     C1  <= '0';
79     C2  <= '0';
80     C3  <= '1';
81     C4  <= '0';
82     WAIT for period;
83     assert (H1 = '0' AND H2 = '0' AND H3 = '0' AND H6 = '0')
84     report "test failed for input combination 0001" severity error;
85
86     C1  <= '0';
```

```
87    C2  <= '0';
88    C3  <= '1';
89    C4  <= '1';
90    WAIT for period;
91    assert (H1 = '0' AND H2 = '0' AND H3 = '0' AND H6 = '1')
92    report "test failed for input combination 0001" severity error;
93
94    C1  <= '0';
95    C2  <= '1';
96    C3  <= '0';
97    C4  <= '0';
98    WAIT for period;
99    assert (H1 = '0' AND H2 = '0' AND H3 = '0' AND H6 = '0')
100   report "test failed for input combination 0001" severity error;
101
102   C1  <= '0';
103   C2  <= '1';
104   C3  <= '0';
105   C4  <= '1';
106   WAIT for period;
107   assert (H1 = '1' AND H2 = '0' AND H3 = '1' AND H6 = '0')
108   report "test failed for input combination 0001" severity error;
109
110   C1  <= '0';
111   C2  <= '1';
112   C3  <= '1';
113   C4  <= '0';
114   WAIT for period;
115   assert (H1 = '1' AND H2 = '1' AND H3 = '0' AND H6 = '0')
116   report "test failed for input combination 0001" severity error;
```

```
118     C1  <= '0';
119     C2  <= '1';
120     C3  <= '1';
121     C4  <= '1';
122     WAIT for period;
123     assert (H1 = '0' AND H2 = '0' AND H3 = '0' AND H6 = '0')
124     report "test failed for input combination 0001" severity error;
125
126     C1  <= '1';
127     C2  <= '0';
128     C3  <= '0';
129     C4  <= '0';
130     WAIT for period;
131     assert (H1 = '0' AND H2 = '0' AND H3 = '0' AND H6 = '0')
132     report "test failed for input combination 0001" severity error;
133
134     C1  <= '1';
135     C2  <= '0';
136     C3  <= '0';
137     C4  <= '1';
138     WAIT for period;
139     assert (H1 = '0' AND H2 = '0' AND H3 = '1' AND H6 = '0')
140     report "test failed for input combination 0001" severity error;
141
142     C1  <= '1';
143     C2  <= '0';
144     C3  <= '1';
145     C4  <= '0';
146     WAIT for period;
147     assert (H1 = '0' AND H2 = '1' AND H3 = '0' AND H6 = '0')
```

```
148     report "test failed for input combination 0001" severity error;
149
150     C1  <= '1';
151     C2  <= '0';
152     C3  <= '1';
153     C4  <= '1';
154     WAIT for period;
155     assert (H1 = '0' AND H2 = '0' AND H3 = '0' AND H6 = '0')
156     report "test failed for input combination 0001" severity error;
157
158     C1  <= '1';
159     C2  <= '1';
160     C3  <= '0';
161     C4  <= '0';
162     WAIT for period;
163     assert (H1 = '1' AND H2 = '0' AND H3 = '0' AND H6 = '0')
164     report "test failed for input combination 0001" severity error;
165
166     C1  <= '1';
167     C2  <= '1';
168     C3  <= '0';
169     C4  <= '1';
170     WAIT for period;
171     assert (H1 = '0' AND H2 = '0' AND H3 = '0' AND H6 = '0')
172     report "test failed for input combination 0001" severity error;
173
174     C1  <= '1';
175     C2  <= '1';
176     C3  <= '1';
177     C4  <= '0';
178     WAIT for period;
179     assert (H1 = '0' AND H2 = '0' AND H3 = '0' AND H6 = '0')
180     report "test failed for input combination 0001" severity error;
181
182     C1  <= '1';
183     C2  <= '1';
184     C3  <= '1';
185     C4  <= '1';
186     WAIT for period;
187     assert (H1 = '0' AND H2 = '0' AND H3 = '0' AND H6 = '0')
188     report "test failed for input combination 0001" severity error;
189
190     WAIT; -- will wait forever
191     END PROCESS;
192 -- *** End Test Bench - User Defined Section ***
193
194 END;
```

Figure 6: Test Bench

Advantages of Using FPGAs Versus Small Scale Integrated Circuits to Implement Logic Designs

Using an FPGA allows for the implementation of more complex designs in a shorter period of time when compared to a Small Scale Integrated Circuit. An FPGA eliminates the need for the designer to wire a breadboard resulting in less time required to debug the hardware. If the board is not behaving as expected, it is likely due to issues with the logic design itself, which can include an improperly drawn schematic or an improperly written UCF file. The designer can then focus on developing and testing the logic design instead of having to set up and verify the hardware.

Advantages of Using Computer Aided Design in the form of Xilinx ISE Foundation Tools

Drawing a schematic in Xilinx provides the advantage of having the ability to check the schematic for errors before implementing the design onto an FPGA. If the software reports errors or warnings after attempting to verify the schematic, the debugging process can begin immediately. It informs the designer that there is a problem with either the schematic itself, the boolean function(s), or the truth table. The designer can then ensure there are no errors in the schematic before writing the test bench.

Simulating the design before programming the FPGA allows for the outputs and inputs to be verified before writing the UCF file and testing the board. If there are any discrepancies in the outputs, the software will report the set of inputs that lead to the discrepancies. The designer can then go back and know exactly which part of the schematic is incorrect which aids in the debugging process. If the input combinations in the simulation do not match those in the truth table, the designer can go back and add them to the test bench and re-run the simulation.

Results

A truth table was made using the given problem statement and diagram of the highways and cities. Using the true outputs in the truth table, four boolean functions were derived, each corresponding to F1, F2, F3, and F6 respectively. A schematic was first drawn by hand on paper using the boolean functions and then drawn in Xilinx. Using the truth table, a test bench was written for the simulation. A simulation was run and all outputs and inputs were checked to ensure that they matched the values listed in the truth table. A UCF file was written to link the input labels, the switches and the output labels to the LEDs. The board was programmed and then tested to ensure it functioned as expected.

Procedure

The inputs (C1, C2, C3, and C4) were controlled with four switches on the FPGA board. Input C1 was set to Hooterville, C2 was set to Mayberry, C3 was set to Mt.Pilot, and C4 was set to Siler City. When a switch was flipped up, the corresponding input would be set to 1. When a switch was down, the input would be set to 0. The outputs (F1, F2, F3, and F6) each corresponded to an LED above the switches. F1 was set to highway one, F2 was set to highway two, F3 was set to highway three, and F6 was set to highway six. Highways four and five were never true because there was no situation in which they would make up the shortest route and therefore were not assigned an LED.

When the number of switches flipped up was not equal to two, all outputs were zero. When the switches for Mt. Pilot and Siler City were flipped on, the LED for highway six turned on. When the switches for Mayberry and Siler City were flipped on, the LEDs for highways one and three turned on. When the switches for Mayberry and Mt. Pilot were flipped on, the LEDs for highways one and two turned on. When the switches for Hooterville and Siler City were flipped on, the LED for highway three turned on. When the switches for Hooterville and Mt. Pilot were flipped on, the LED for highway two turned on. When the switches for Hooterville and Mayberry were flipped on, the LED for highway one turned on.

Error Analysis

The final results worked as expected and the outputs worked according to the information given. There were errors in the beginning which were caused by not choosing the highways that were the fastest route between the cities, but it was resolved before starting the schematic design. There were unexpected errors when making the schematic and led to errors when creating the testbench. This issue was caused by the Xilinx program and could not be solved, this resulted in designing the schematic from scratch again which resolved the previous issue and gave the proper testbench result. There was also an error when assigning values to the testbench for the outputs this was due to two extra values which were not supposed to be used in the testbench but was accidently added, this was resolved by removing the unwanted values.

Recommendations

For the future in order to be more efficient during the labs, there is a need to familiarize with the Xilinx program. This is to improve time efficiency when working during lab times so that the lab can progress much faster in the future. This can also help in understanding errors and implement solutions as soon as possible and do better in future labs.

Conclusion

In conclusion, the purpose and objectives in this lab was met. Using the information given, a truth table, boolean functions and circuit diagrams for the fastest routes to each city was obtained. The circuits were tested and had no errors, it also produced an accurate results representing the problem given. During the lab, there were errors that would have resulted in a failed lab result but the group was able to adjust to the circumstances and work through it successfully. Overall, the results produced was understandable and was able to fulfil the objectives of the lab.