ENGG 125.03 Laboratory Activity 1A: Logic Gates

Activity Sheet; Submitted: 27 April 2021

Mallari, Kevin Luis T.
BS Computer Engineering
Electronics, Computer, and Communications Engineering
Ateneo de Manila University
Philippines

***Abstract*—The laboratory activity focuses on making the student familiar with the Verilog programming language and the Quartus environment. This was done in two (2) parts. The first part of the activity involves the use of simple logic gates and operations. The specific operations used were NOT, AND, OR, NOR, and XOR. For the second part of the activity, a combination of gates were used. The principles were the same with the first part. However, now, multiple gates are used in one line of code. The outputs of all the equations in the activity were simulated using Quartus and verified using truth tables.**

Keywords - Arduino, Touch Sensor, Ultrasonic Sensor

I. Computations/Solutions

To make sure that the simulation is in line with the equations, the solutions for the different equations will be computed. The solutions will be shown using truth tables. The truth tables are evaluated based on the equations and information given.

Firstly, the truth table for the logic gates required in part 1 is shown. Part 1 is about majority circuits, wherein the circuit outputs HIGH if more than 50% of the inputs are HIGH. Otherwise, the output is LOW. Two kinds of majority circuits are done in part 1: 3-input majority circuit and 4-input majority circuit. The truth table for both are seen below.

A	B	C	OUT
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Fig. 1. 3-input Majority Circuit Truth Table

A	B	C	D	OUT
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Fig. 2. 4-input Majority Circuit Truth Table

Secondly, the truth table part 2 is shown. Part 2 is about an alarm system with three (3) input variables. The output alarm (A) will be HIGH if the enable input (E) is HIGH and either the door trigger (D) or window trigger (W) is triggered. The triggered state of the door and window is LOW.

E	D	W	A
0	0	0	0
0	0	1	0

0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Fig. 3. House Alarm Truth Table

Lastly, a basic combinational circuit was simulated for part 3. The truth table is given in the activity, but it will also be shown here for reference.

W	X	Y	Z	f
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Fig. 1. Basic Combinational Circuit Truth Table

II. HDL Code

The first block of code presented is the code for the two majority circuits. Based on the truth table, the equation for the three (3) bit majority circuit was solved to be:

$$f = AB + BC + AC$$

This is the equation that the code was based on. Moreover, the equation for the four (4) bit majority circuit was also solved using the truth table. It was solved to be:

$$f = AB + AC + AD + BC + BD + CD$$

```
module majority(Out1, Out2, A, B, C, D);
// 3-bit Majority Circuit

input A, B, C;
output Out1;

assign Out1 = (A & B) | (A & C) | (B & C);

// 4-bit Majority Circuit

input D; // adding D as an input
output Out2;

assign Out2 = (A & B) | (A & C) | (B & C) |
(A & D) | (B & D) | (C & D);

endmodule
```

Fig. 4. Code for 3 bit and 4 bit Majority Circuit

As previously stated, the second part of the activity is about a house alarm system. The output alarm (A) will be HIGH if the enable input (E) is HIGH and either the door trigger (D) or window trigger (W) is triggered. The triggered state of the door and window is LOW. From this description, the equation of the system can be solved.

$$A = ED' + EW'$$

```
module house_alarm(A, E, D, W);

input E, D, W; // enable, door, window
output A; // alarm

assign A = (E & ~D) | (E & ~W);

endmodule
```

Fig. 5. Code for House Alarm System

Lastly, the code for the given basic combinational circuit was constructed. It is shown in figure 6.

```
module basic_combo(f, W, X, Y, Z);
```

```
input W, X, Y, Z;
output f;

assign f = (~X & ~Z) | (X & Z) | (W & Y);

endmodule
```

Fig. 6. Code for Basic Combinational Circuit

III. Output Circuit

The following figures display the circuits of the code constructed. Furthermore, they are also accurate representations of the truth tables that were previously made.

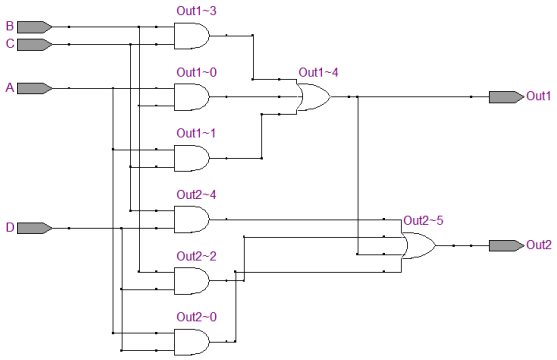

Fig. 7. Circuit for the Majority Circuits

There are two outputs in the circuit in figure 1: one for the 3-bit version and one for the 4-bit version of the majority circuit. The software, Quartus, was able to simplify the circuit such that there are only six (6) AND gates used for the entirety of the circuit.

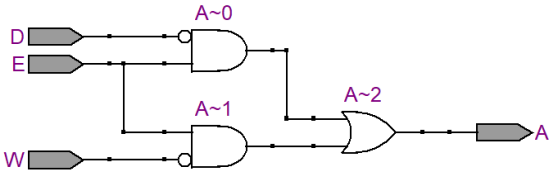

Fig. 8. Circuit for the House Alarm

As seen in the circuit diagram the appropriate inputs and the specific output was used.

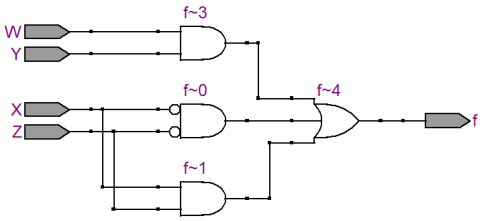

Fig. 9. Circuit for the Basic Combinational Circuit

Following the equation and code of the combinational circuit, this was the circuit produced by the software.

IV. Screenshot(s) of Simulations and VWF file

The screenshots of the simulations are presented below.

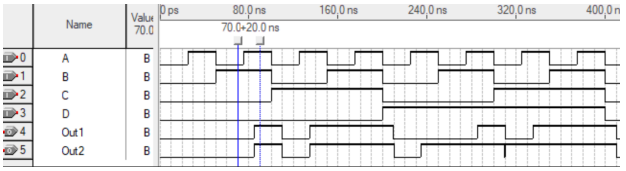

Fig. 10. Simulation for the Majority Circuits

The screenshot above shows the simulation for both the majority circuits.

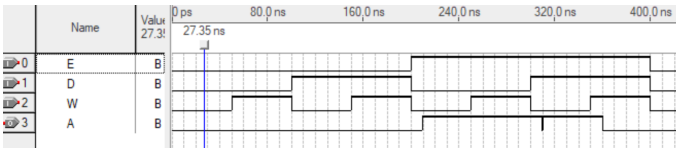

Fig. 11. Simulation for the House Alarm System

The screenshot above shows the simulation for the house alarm system.

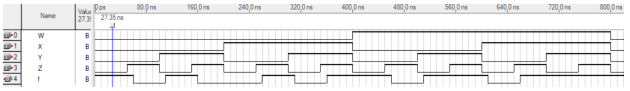

Fig. 12. Simulation for the House Alarm System

As seen in the screenshots, the simulations follow the truth tables for the most part. There are a few differences, however. For example, there is a delay in the result of the outputs by about 10 ns. This is because of the propagation delay. Another noticeable flaw in the simulation is the random spike up/down on the outputs that lasts for an extremely short time. This is most probably due to the inputs not instantaneously changing when transitioning from a HIGH value to a LOW value, and vice versa.