Introduction to Digital Systems

REPORT

Lab #1

NAME: **Peng Li**

STUDENT ID: **101047123**

DUE DATE: **Sep 25th 6:00AM**

Purpose

To understand how does the Gates work in Digital Systems. Also, using the data or table haven been created to find equivalent design.

Procedure

First of all, go through the guide of < Logisim Guide for SYSC 2310 Students> which make the Logisim easier to use. Later on, it would find the tutorial implement used two AND gates and one OR gate. Also there is negate AND gate connect to A input and other AND gate has another negate connect to B input. Next, there is a file called “Lab1Star” have been opened in Logisim, which gives input for circuit that we would build in the whole lab.

Secondly, we start our main part to create the first circuit and name “Lab1Mystery” and connect to our test component, after that we can record truth table(Lab1MysteryLog.txt).

From previous truth table, I get some relationship between input/output, and then I can design an equivalent circuit (Mystery circuit) which gives us same output.

At last, there is sub-circuit has been created to prove the original circuit is equivalent Mystery circuit.

Part 0

“Lab1Star” is form by 2 bottoms and 3LED light, one of the bottoms control the LED is light or not, and the other bottom is reset the LED light(all close). Except that, 3 LED light are distinguish input A, input B and input C.

Part 1

During this part, is focus on build a same circuit that appear in PDF. The truth table shows below:

A	**B**	**C**	**X**
0	**0**	**0**	**0**
0	**0**	**1**	**1**
0	**1**	**0**	**1**
0	**1**	**1**	**0**
1	**0**	**0**	**1**
1	**0**	**1**	**1**
1	**1**	**0**	**1**
1	**1**	**1**	**1**

$$\overline{[\overline{A \cdot B} + \bar{B}] \oplus (\overline{A + C})}$$

As table we can see, A, B, C are representing 3 different input and X gives the output.

A	B	C	not(AandB)	[not(AandB)]or(notB)	not(AorC)	X
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	1	1	1
0	1	1	1	1	0	0
1	0	0	1	1	0	1
1	0	1	1	1	0	1
1	1	0	0	0	0	1
1	1	1	0	0	0	1

As above, gives a perfect proof for the output and it is complete truth table for reader to think about their relationship. For next step, I would create a new equivalent circuit, as the final output, which gives me a think of XOR or NXOR gate (it from0,1,1,0).

Part 2

The equivalent design table shows blow:

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

$$\boldsymbol{A + [\,(B + C) \cdot \overline{B \cdot C}\,]}$$

This truth table almost same as the Part 1's table, because this is the one we are looking for that design can gives same output.

A	B	C	BorC	BandC	not(BandC)	(BorC)and[not(BandC)]	Y
0	0	0	0	0	1	0	0
0	0	1	1	0	1	0	1
0	1	0	1	0	1	0	1
0	1	1	1	1	0	0	0
1	0	0	0	0	1	0	1
1	0	1	1	0	1	0	1
1	1	0	1	0	1	0	1
1	1	1	1	1	0	0	1

This complete table shows detail about new design, it gives different way to connect but still same final output. In this part, the first time in my logic is using XOR Gate to connect with OR can give the equal output as first circuit. Unfraternally, this part only allow use AND, OR and NOT gates, so I have to simplify XOR gate. Thus, we get a component with OR gate be one of output and NAND gate for another input, both be part of an AND gate. It respects as **(BorC)and[not(BandC)]**. By the way, NAND can be separate as an AND gate and a NOT gate to meet our requirement.

Part 3

This part is a bonus part, I only add a NXOR gate between end of each circuit's output, and both circuit's output be the two-new input for NXOR gate. The reason is I want to proof two circuit are equal, which has to response to 1. Therefore, there have two options: 1-1, 0-0. As we know only NXOR can gives me 1-1-1 and 0-0-1. The truth table below would be clarifying the reason. In addition, I the first time I was using XOR gate, because all the outputs are 0 can be a poof they have same result, but 0 sometimes response wrong, all wrong could also shows all 0. Thus, I switch to NXOR to be a stronger proof.

X	Y	Z
1	**1**	**1**
0	**0**	**1**
1	**1**	**1**
0	**0**	**1**

Inconclusion

We are achieved our purpose to make two circuit equivalents, and sub-circuit become a strong proof support our result.