Nathan Tomlin & Layth Maloul
CISP 2410, Project 1

SECTION 1 - OBJECTIVES

The objective of this project is to create a working hex display in Logisim using the correlation between special gray code and binary numbers. The result should be a hex display capable of displaying values 0-9 as well as A-F. We do this by using circuit design patterns in Logisim to achieve the desired output, as well as our knowledge of Boolean equations and K-Maps.

The general approach to the design that we had was simple. We knew that we had to break this project down into multiple sections and subsections, and work from the base up to achieve a working product. First, we restructured the truth table given in the assignment briefing so that it was in order from 0000 to 1111. Then, we broke down the truth table and split it up into 4 separate truth tables for clarity. Each truth table contained all inputs of X and 1 output for each individual Y. From there, we derived a Boolean equation for each respective Y output (Y0, Y1, Y2, &Y3) in sum of products form. We then used K-Maps to simplify each individual equation into something far smaller and far easier to implement into a circuit design. Once we simplified the equations, we went over to Logisim to implement the circuit design. In Logisim, each equation had AND gates for each product, and one OR gate to sum all of the products in that equation together. From there we tested each individual equations output, and made sure that it matched its respective Y output for each of the inputs. Finally, we tied each of the OR gates from the equations into the Y output lines going into the splitter, giving us the finished product.

SECTION 2 – TRUTH TABLE

HEX INPUT	X_3	X_2	X_1	X_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0	0
F	0	0	0	1	1	1	1	1
7	0	0	1	0	0	1	1	1
C	0	0	1	1	1	1	0	0
9	0	1	0	0	1	0	0	1
A	0	1	0	1	1	0	1	0
8	0	1	1	0	1	0	0	0
B	0	1	1	1	1	0	1	1
1	1	0	0	0	0	0	0	1
E	1	0	0	1	1	1	1	0
6	1	0	1	0	0	1	1	0
D	1	0	1	1	1	1	0	1

2	1	1	0	0	0	0	1	0
3	1	1	0	1	0	0	1	1
5	1	1	1	0	0	1	0	1
4	1	1	1	1	0	1	0	0

SECTION 3 – LOGIC EXPRESSIONS

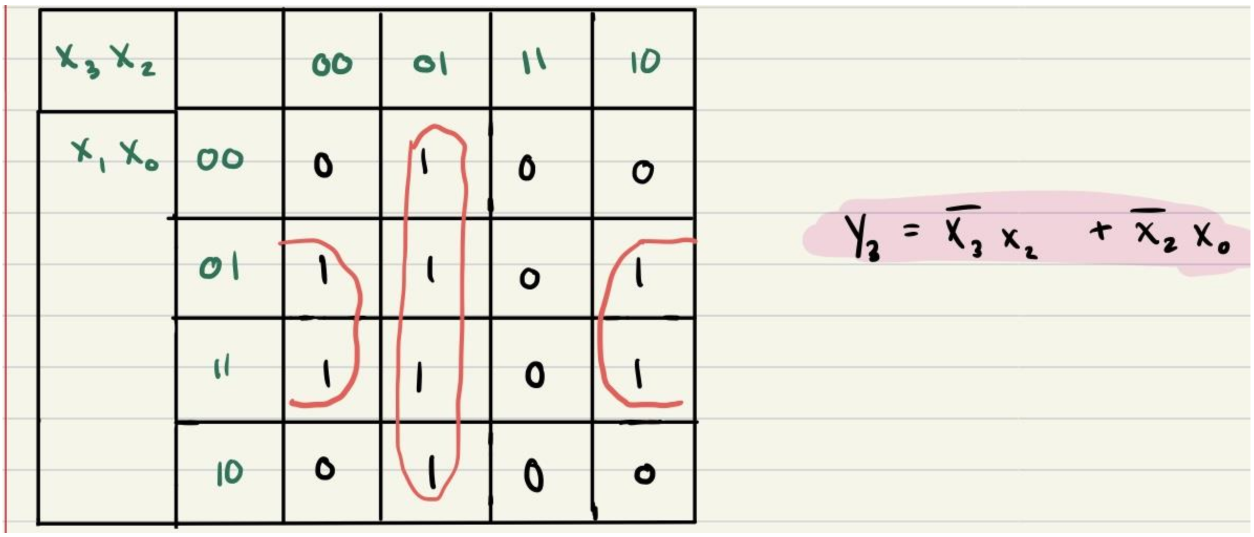
Y3

HEX INPUT	X_3	X_2	X_1	X_0	Y_3
0	0	0	0	0	0
F	0	0	0	1	1
7	0	0	1	0	0
C	0	0	1	1	1
9	0	1	0	0	1
A	0	1	0	1	1
8	0	1	1	0	1
B	0	1	1	1	1
1	1	0	0	0	0
E	1	0	0	1	1
6	1	0	1	0	0
D	1	0	1	1	1
2	1	1	0	0	0
3	1	1	0	1	0
5	1	1	1	0	0
4	1	1	1	1	0

For the Y3 equation, we looked at all the inputs where Y = 1. We then took all the inputs that equal one then AND them together. We repeat this for all other outputs of Y = 1, and then sum all of them together to achieve a sum of products equation for Y3.

Y3 = (X3 ’X2’ X1’ X0) + (X3’ X2’ X1 X0) + (X3’ X2 X1’ X0’) + (X3’ X2 X1’ X0) + (X3’ X2 X1 X0’) + (X3’ X2 X1 X0) + (X3 X2’ X1’ X0) + (X3 X2’ X1 X0)

Next, we simplified using the K-MAP shown below to get the final equation for Y3.



This gives us the final simplified equation for Y3, which is: **Y3 = (X3’X2) + (X2’X0)**

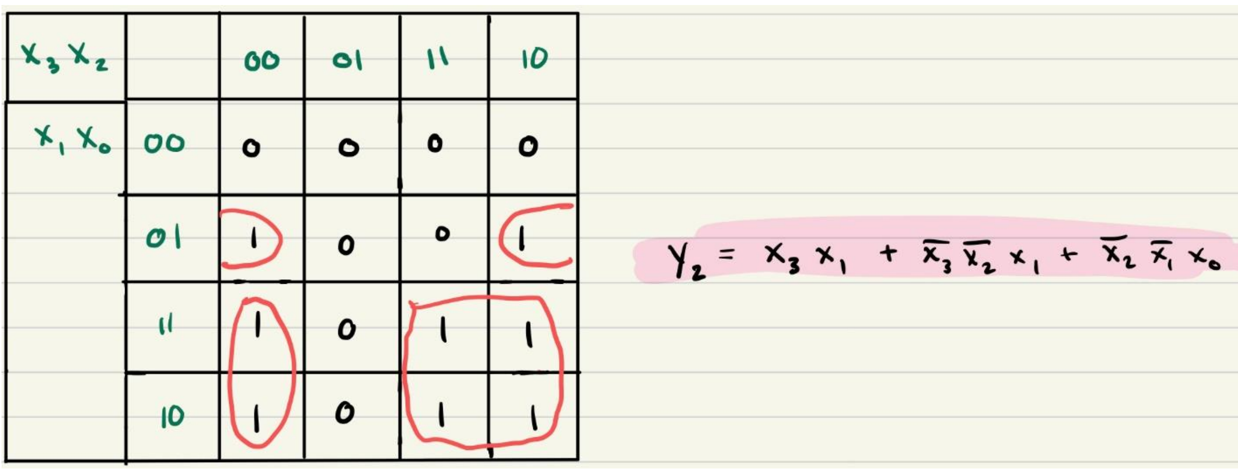
<u>Y2</u>

HEX INPUT	**X_3**	**X_2**	**X_1**	**Y_2**
0	0	0	0	0
F	0	0	0	1
7	0	0	1	1
C	0	0	1	1
9	0	1	0	0
A	0	1	0	0
8	0	1	1	0
B	0	1	1	0
1	1	0	0	0
E	1	0	0	1
6	1	0	1	1
D	1	0	1	1
2	1	1	0	0
3	1	1	0	0
5	1	1	1	1
4	1	1	1	1

For the Y2 equation, we looked at all the inputs where Y = 1. We then took all the inputs that equal one then AND them together. We repeat this for all other outputs of Y = 1, and then sum all of them together to achieve a sum of products equation for Y2.

Y2 = (X3’ X2’ X1’ X0’) + (X3’ X2’ X1’ X0) + (X3’ X2’ X1 X0) + (X3 X2’ X1’ X0) + (X3 X2’ X1 X0’) + (X3 X2’ X1 X0) + (X3 X2 X1 X0’) + (X3 X2 X1 X0)

Next, we simplified using the KMAP shown below to get the final equation for Y2.



This is the final simplified equation for Y2, which is:

Y2 = (X3 X1) + (X3’ X2’ X1) + (X2’ X1' X0)

Y1

HEX INPUT	X_3	X_2	X_1	Y_1
0	0	0	0	0
F	0	0	0	1
7	0	0	1	1
C	0	0	1	0
9	0	1	0	0
A	0	1	0	1
8	0	1	1	0
B	0	1	1	1
1	1	0	0	0
E	1	0	0	1
6	1	0	1	1
D	1	0	1	0
2	1	1	0	1
3	1	1	0	1
5	1	1	1	0
4	1	1	1	0

For the Y1 equation, we looked at all the inputs where Y = 1. We then took all the inputs that equal one then AND them together. We repeat this for all other outputs of Y = 1, and then sum all of them together to achieve a sum of products equation for Y1.

Y1 = (X3’ X2’ X1’ X0) + (X3’ X2’ X1 X0’) + (X3’ X2 X1’ X0) + (X3’ X2 X1 X0) + (X3 X2’ X1’ X0) + (X3 X2’ X1 X0’) + (X3 X2 X1’ X0’) + (X3 X2 X1’ X0)

Next, we simplified using the KMAP shown below to get the final equation for Y1.

X_3X_2		00	01	11	10
X_1X_0	00	0	0	1	0
	01	1	1	1	1
	11	0	1	0	0
	10	1	0	0	1

$$Y_1 = \overline{X_2}\,\overline{X_1}\,X_0 + \overline{X_2}\,X_1\,\overline{X_0} + \overline{X_3}\,X_2\,X_0 + X_3\,X_2\,\overline{X_1}$$

This is the final simplified equation for Y1, which is

$Y1 = (X2' X1' X0) + (X2' X1 X0') + (X3' X2 X0) + (X3 X2 X1')$

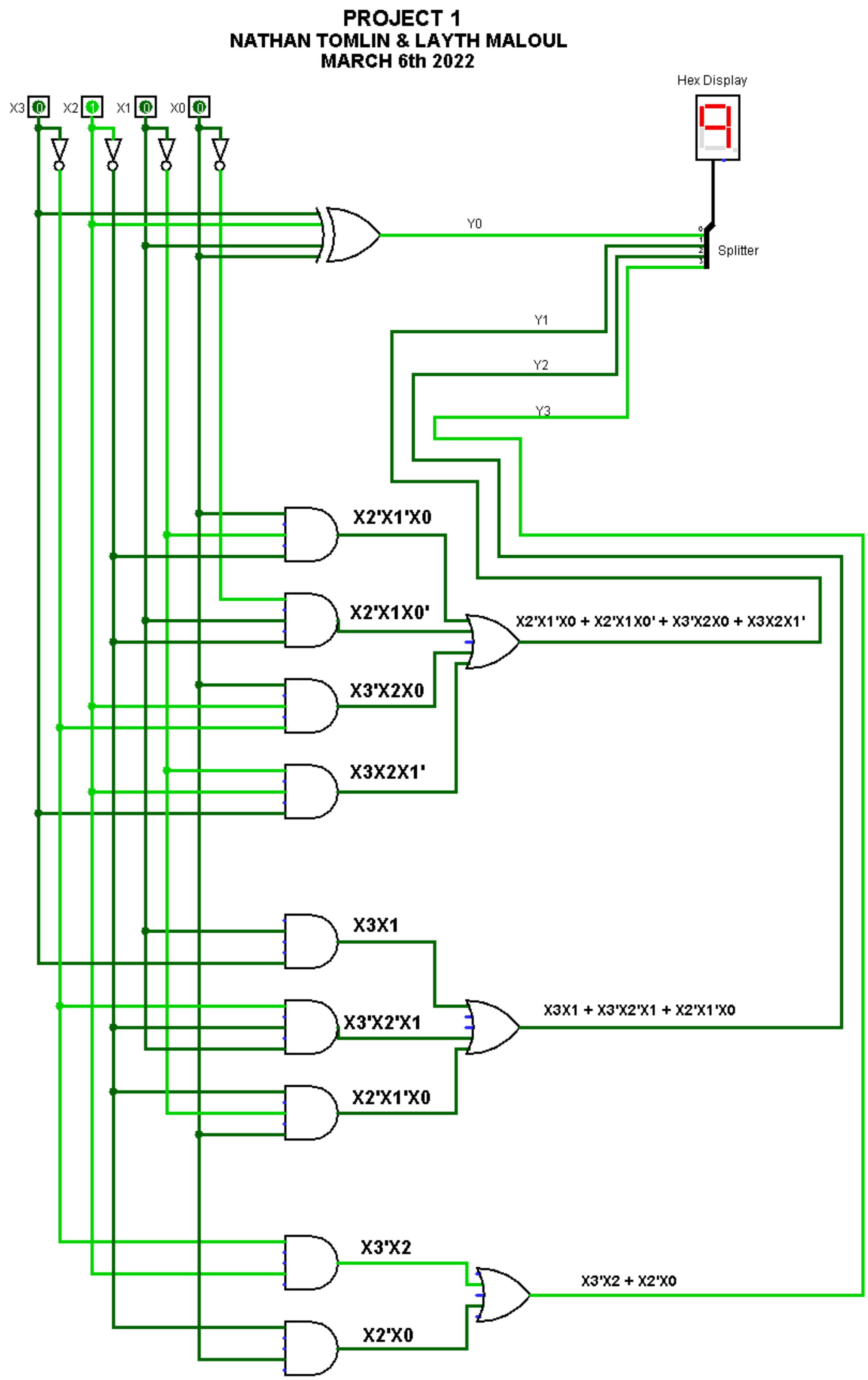
Y0

HEX INPUT	X_3	X_2	X_1	Y_0
0	0	0	0	0
F	0	0	0	1
7	0	0	1	1
C	0	0	1	0
9	0	1	0	1
A	0	1	0	0
8	0	1	1	0
B	0	1	1	1
1	1	0	0	1
E	1	0	0	0
6	1	0	1	0
D	1	0	1	1
2	1	1	0	0
3	1	1	0	1
5	1	1	1	1
4	1	1	1	0

For Y0, the equation was given in the assignment details.

Therefore, **$Y0 = X3 \oplus X2 \oplus X1 \oplus X0$**

SECTION 4 – CIRCUIT DESIGN



Section 5 – Conclusions

The outcome of our design was a fully functional combinational logic circuit with the ability to display hexadecimal numbers 0-9 & A-F based on the four inputs X3, X2, X1 and X0. All aspects of this design are fully functional and fortunately there are no problems/aspects of the design that do not work properly. This project helped me further solidify my understanding of combinational logic and any subsequent steps required to get to that point, such as KMAPS and truth tables. I feel as though this project did a wonderful job of tying all of the previous chapters together into one final hands on activity. The project required us to use extensive thought combined with fundamental logic skills in order to be able to apply everything we've learned so far.