Nizam Ercan 22302317

EEE 102 Section 2

28.02.2024

EEE102 Lab 3: Combinational Logic Circuit

1- Purpose:

The aim of this lab is to design a combinational logic circuit for a real-life scenario and implement this design on a real breadboard with counters and logic gates in the form of chips by researching and examining datasheets for the chips we are using. Also, we aimed to observe the output(s) of our logic circuit in LEDs.

2- Design Specifications

The scenario I want to investigate and simulate is a system with one hungry wolf and 3 bunnies which are a mother bunny, a father bunny and a baby bunny. The hungry wolf wants to eat the baby bunny whereas the mother and the father bunny want to protect their baby from the evil wolf. In the nighttime, the wolf falls asleep which means that the baby bunny is safe. However, in the daytime, at a completely non expected time the wolf attacks the cave where the bunnies live. If, in the time of the attack, the baby bunny isn't in the cave the wolf searches for the baby bunny for a little while but eventually lefts not harming the father or the mother bunny. However, if baby bunny is present in the cave in the time of the attack, the father and the mother bunny (if they are present in the cave) try to save their baby from getting eaten. However, the wolf is too strong for a single bunny to defeat. Therefore, the only way for the wolf to be beaten and not eat the baby bunny is when both the parents are in the cave.

The inputs and the outputs for this scenario can be shown as:

1- x_1: Time of the day (0: night, 1: daytime)
2- x_2: Presence of the baby bunny (0: not in the cave, 1: in the cave)
3- x_3: Presence of the father bunny (0: not in the cave, 1: in the cave)
4- x_4: Presence of the mother bunny (0: not in the cave, 1: in the cave)
5- f: fate of the baby bunny (0: not eaten by the wolf, 1: eaten by the wolf)

We can draw the truth table for these inputs and outputs according to our scenario as you can see in Table 1:

x_1	x_2	x_3	x_4	f
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	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Table 1: Truth table

We can define a logic function to see a way to represent this truth table with logic gates. I am writing the logic function in canonical sum of products form as you can see below:

$$f(x_1, x_2, x_3, x_4) = \sum (m_{12}, m_{13} m_{14}) = x_1 . x_2 . \overline{x_3} . \overline{x_4} + x_1 . x_2 . \overline{x_3} . x_4 + x_1 . x_2 . x_3 . \overline{x_4}$$

As you can see, the expression for the logic function is quite complex to implement in real life with a breadboard and logic gates. Therefore, I need to simplify this expression by either using algebraic manipulation or using a Karnaugh map. I am using a Karnaugh Map to simplify my logic function as it gets rid of the possibility of making an error while doing algebraic manipulation. You can see the Karnaugh map I drew in Table 2.

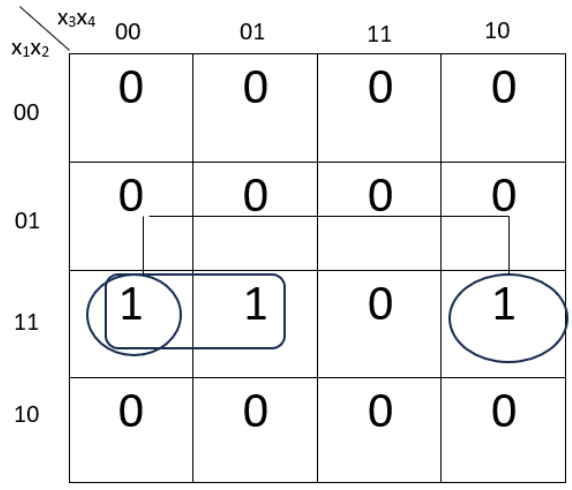

Table 2: The Karnaugh Map

From the Karnaugh map, we can simplify our canonical expression as you can see below:

$$f(x_1, x_2, x_3, x_4) = x_1 x_2 \overline{x}_3 + x_1 x_2 \overline{x}_4$$

I am still not satisfied with the current state of the function. Therefore, I am using a little bit of algebraic manipulation for some expression that would require less logic gates.

$x_1 x_2 \overline{x}_3 + x_1 x_2 \overline{x}_4 = x_1 x_2 (\bar{x}_3 + \bar{x}_4) = x_1 . x_2 . (\overline{x_3 . x_4})$ by Demorgans's Theorem.

This final expression for my circuit only requires a two input NAND gate and a three input AND gate.

3- Methodology:

The lab manual wants me to use a 4-bit counter as a way to show all the potential inputs in my circuit. Therefore, I used a 74HC163 type 4-bit counter in count mode. I also used a SN74HC00 type quadruple 2-input NAND gate for the inputs x_3 and x_4. Unfortunately, there wasn't any 3 input and gates in the lab. Because of that, I used a SN74HC08 type quadruple 2-input AND gate for inputs x_1 and x_2 then connected the output of x_1 and x_2 and the output of the NAND gate at another 2 inputs of the same AND gate (SN74HC08 is a quadruple AND gate which means that I can use it as 4 independent AND gates.) which gave me the output of the overall logic function as an output. Then I showed this output with an LED. You can see the final logic circuit I implemented on my breadboard in Figure 1.

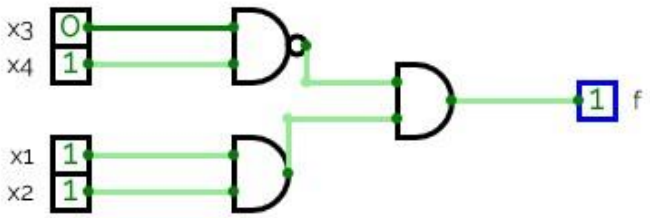

Figure 1: my logic circuit

4- Resuts:

In order to use the 74HC163 type 4-bit counter, I connected its pins to the ground or to the power supply according to the count mode of the counter which is available in the datasheet provided in Moodle as you can see in Figure 2. You can also see the locations of the pins in the chip in Figure 3. I connected (MR)', CEP, CET and (PE)' pins to the positive terminal of my power supply, CP pin to my wave generator and Q_1, Q_2, Q_3 and Q_4 output pins to 4 LEDs which are connected to a resistor and the ground. I opened my wave generator and created a square wave which resulted with the output LEDs to start counting from 0 to 15.

OPERATING MODE	INPUTS						OUTPUTS	
	$\overline{MR}$	CP	CEP	CET	$\overline{PE}$	D_n	Q_n	TC
reset (clear)	l	↑	X	X	X	X	L	L
parallel load	h h	↑ ↑	X X	X X	l l	l h	L H	L (1)
count	h	↑	h	h	h	X	count	(1)
hold (do nothing)	h h	X X	l X	X l	h h	X X	q_n q_n	(1) L

Figure 2: Count mode pins

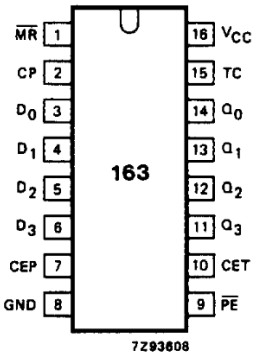

Figure 3: Counter pin locations

After checking that my counter is working properly, I set up my SN74HC00 type NAND gate by using its datasheet as you can see from Figure 4. I firstly connected its Vcc port to the power supply and the GND port to the ground. Then, I connected Q_0 which represents x_4 in the truth table to 1A input port and Q_1 which represents x_3 in my circuit to 1B input port. In the end, this gave me the signal for $\overline{x_3 x_4}$ in 1Y output port.

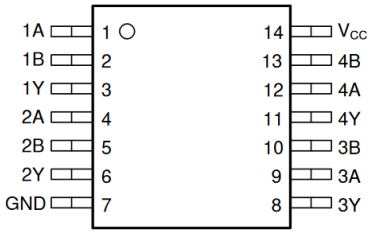

Figure 4: NAND gate pins

After implementing the NAND gate, I set the SN74HC08 type quadruple 2-input AND gate by examining the datasheet as you can see in Figure 5. Similar to the NAND gate, I connected the Vcc port to my power supply and GND pin to the ground. Then, I connected Q_2, which represents x_2, to 4B input port and Q_3, which represents x_4 to 4A input port. This gave me the signal for x_2. x_4 in 4Y output port. After that, I connected the 4Y output port to 3B input port and the output of the

NAND gate 1Y port to 3A input port. This gave $x_1.x_2.(\overline{x_3.x_4})$ in 3Y output port which is the output of my logic function.

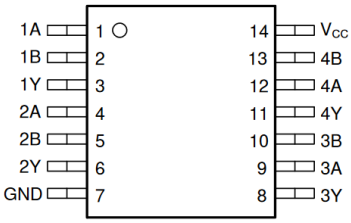

Figure 5: AND gate pins

Finally, connected 3Y output port to an LED, which is also connected to the ground. Additionally, I connected my oscilloscope probe to 3Y output port to display the output waveform. This finished the circuit as you can see in Figure 6.

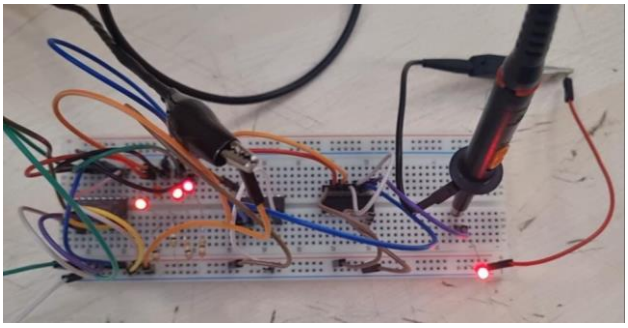

Figure 6: Working logic circuit

In order to see whether my circuit is working properly or not I checked the times when my output LED was turned on and investigated the output waveform in the oscilloscope which you can see in Figure 7. You can see that the wave in the oscilloscope is accurate according to the truth table which indicates that my circuit is working properly.

Figure 7: Output waveform in the oscilloscope

5- Conclusion:

In this lab, I acquired practical skills in assembling a counter and various logic gates on a breadboard, following the instructions outlined in their own specific datasheets. This involved

creating a logic circuit that emulates a real-world scenario. Using my previous knowledge, I constructed a truth table to map out inputs and outputs, which I then translated into a logic function using the canonical sum of products form. Using minimization techniques such as algebraic manipulation and Karnaugh maps, I simplified my logic function. From my prior experience with oscilloscopes, I verified the functionality of my circuit by using it. Despite encountering minor issues like oscilloscope noise, the overall performance of my circuit was precise and reliable.

6- References:

1- https://www.ti.com/lit/ds/symlink/sn74hc08.pdf?ts=1709032948468&ref_url=https%253A%252F%252Fwww.google.com%252F

2- https://www.ti.com/lit/ds/symlink/sn74hc00.pdf?ts=1709018086402&ref_url=https%253A%252F%252Fwww.google.com%252F