Lebanese American University

Logic Design Lab Final Project

2021-2022

Razan Houdaifa

Rebecca Bou Zerdan

Karem Khaddaj

Samih Al Haj Ibrahim

Mouhamad Al Achi

Table of Content:

Table of Tables:

Table of Figures:

Abstract:

Nearly all societies have been constantly trying to design smart cities to ride the wave of revolution within the digital age, thus public toilets have been a major issue when it comes in revolutionizing these cities. This has aggravated the need for smart toilets to manage its operation taking into consideration the ethical and social prospects. The purpose of this project is to build a smart toilet conforming to the demands of Jbeil's municipality and ensuring its smooth operation with every citizen utilizing it. Several sectors were considered which included maintenance, cleanliness, adaptivity, safety, and privacy. The toilet was tested and built by relying on a structured distribution of thorough research, analysis, and virtual implementations to ensure timely and effective results. The results show that although the project's demands are quite problematic, they can be solved using logical analysis techniques and built using specific electrical components. Our analysis found evidence for the need of IC circuits and logic concepts representing different functions and electrical tools to build such project. As expected, certain difficulties were faced throughout the process varying from testing to virtual and hardware implementation. Overall, the findings proved our goal and served as steps for our implementation ensuring our smart toilet delivery in a power consumption and cost-effective manner.

Keywords:

Smart toilet, implementation, testing, IC circuits.

Introduction:

In this final project, we were asked to design a circuit controller for a smart toilet. This smart toilet is suitable for normal and handicap people were it it has push buttons that enables the user to show if he wants to use the toilet or wash his hands only. Coins should be given to use this smart toilet and a maintenance buzzer is present as well. The user can't enter the toilet unless the green Led light is on indicating the the toilet is empty and ready to use. If the yellow light is on, this indicates that the toilet is in cleaning mode and if the red LED is on this indicated that the toilet is is in engaged mode. In addition, an on orange LED light signifies that the toilet is under maintenance. If the user decides to wash his hands only, after his use of this smart toilet, no cleaning is required. However, if he decides to use the toilet and pushes the corresponding button, the toilet will undergo cleaning after his use and will undergo a deep cleaning after 10 uses. As the user enters and uses the toilet and wants to leave, he should flush the toilet and press its corresponding valve, use the soap, use the sink, and then use the paper towel in this specific sequence. Failure in doing these mentioned steps in this order will oblige him to repeat them for him to unlock the toilet door and leave. These actions will be indicated by 4 led lights inside the toilet where the door won't be unlocked unless 4 of them is on.

Equipment Needed:

- Constructor Virtual de Circuitos software
- Various ICs:

  3 3-input AND gates (74LS11)

  4 2-input AND gates (74LS08)

  4 2-input OR gates (74LS32)

  4 NOT gates (74LS04)
- 2 7 segment displays
- 2 BCD to 7 segment decoders (74LS47)
- 1 555 timers
- Switches/ buttons
- 1 5V battery
- 10 300Ω resistors
- 8 LED lights of colors:
- 24 chairs for chips
- 1 soldering tool
- Cables/wires
- Breadboard

- Altra Quartus II software
- Various Flip-Flops: JK Flipflops – D Flipflops….

  7 JK flip flops (74LS109)

Normal Cleaning- Counter:

In the normal cleaning, the logic behind it is the following:

We will implement a counter that counts from 0 to 9, (i.e., the toilet will be used 10 times) the counter will count the times we used the toilet and will repeat eventually.

The following implementation will have in the present state (4bits), then we will use 4 flipflops. We considered JK-Flipflops as our type of flipflops used, since their hardware happen to be less complex than D-Flipflops due to the huge number of don't cares we'll get in the state table of the JK flipflops.

The following **state table** for the normal cleaning counter is as follows:

Present State				**Next state**				**Output**	**Flip Flop outputs**							
A3	A2	A1	A0	A3	A2	A1	A0	Y	JA3	KA3	JA2	KA2	JA1	KA1	JA0	KA0
0	0	0	0	0	0	0	1	1	0	X	0	X	0	X	1	X
0	0	0	1	0	0	1	0	1	0	X	0	X	1	X	X	1
0	0	1	0	0	0	1	1	1	0	X	0	X	X	0	1	X
0	0	1	1	0	1	0	0	1	0	X	1	X	X	1	X	1
0	1	0	0	0	1	0	1	1	0	X	X	0	0	X	1	X
0	1	0	1	0	1	1	0	1	0	X	X	0	1	X	X	1
0	1	1	0	0	1	1	1	1	0	X	X	0	X	0	1	X
0	1	1	1	1	0	0	0	1	1	X	X	1	X	1	X	1
1	0	0	0	1	0	0	1	1	X	0	0	X	0	X	1	X
1	0	0	1	0	0	0	0	0	X	1	0	X	0	X	X	1
1	0	1	0	X	X	X	X	X	X	X	X	X	X	X	X	X
1	0	1	1	X	X	X	X	X	X	X	X	X	X	X	X	X
1	1	0	0	X	X	X	X	X	X	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X	X	X	X	X	X	X

Table 1:Truth table for normal cleaning

We'll use K-maps to get the equations:

K-Maps:

Y:

	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	X	X	X	X
10	0	1	X	X

Figure 1: K-map for Y

Y=A3A0

JA0 :

	00	01	11	10
00	1	X	X	1
01	1	X	X	1
11	X	X	X	X
10	1	X	X	X

Figure 2: K-map for JA0

JA0=1

KA0:

	00	01	11	10
00	X	1	1	X
01	X	1	1	X
11	X	X	X	X
10	X	1	X	X

Figure 3: K-map for KA0

KA0= 1

JA1:

	00	01	11	10
00	0	1	X	X
01	0	1	X	X
11	X	X	X	X
10	0	0	X	X

Figure 4: : K-map for JA1

JA1=A3'A0

KA1:

	00	01	11	10
00	X	X	1	0
01	X	X	1	0
11	X	X	X	X
10	X	X	X	X

Figure 5: : K-map for KA1

KA1: A0

JA2:

	00	01	11	10
00	0	0	1	0
01	X	X	X	X
11	X	X	X	X
10	0	0	X	X

Figure 6: : K-map for JA2

JA2=A1A0

KA2:

	00	01	11	10
00	X	X	X	X
01	0	0	1	0
11	X	X	X	X
10	X	X	X	X

Figure 7: : K-map for KA2

KA2=A1A0

JA3:

	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	X	X	X	X
10	X	X	X	X

Figure 8:: K-map for JA3

JA3=A2A1A0

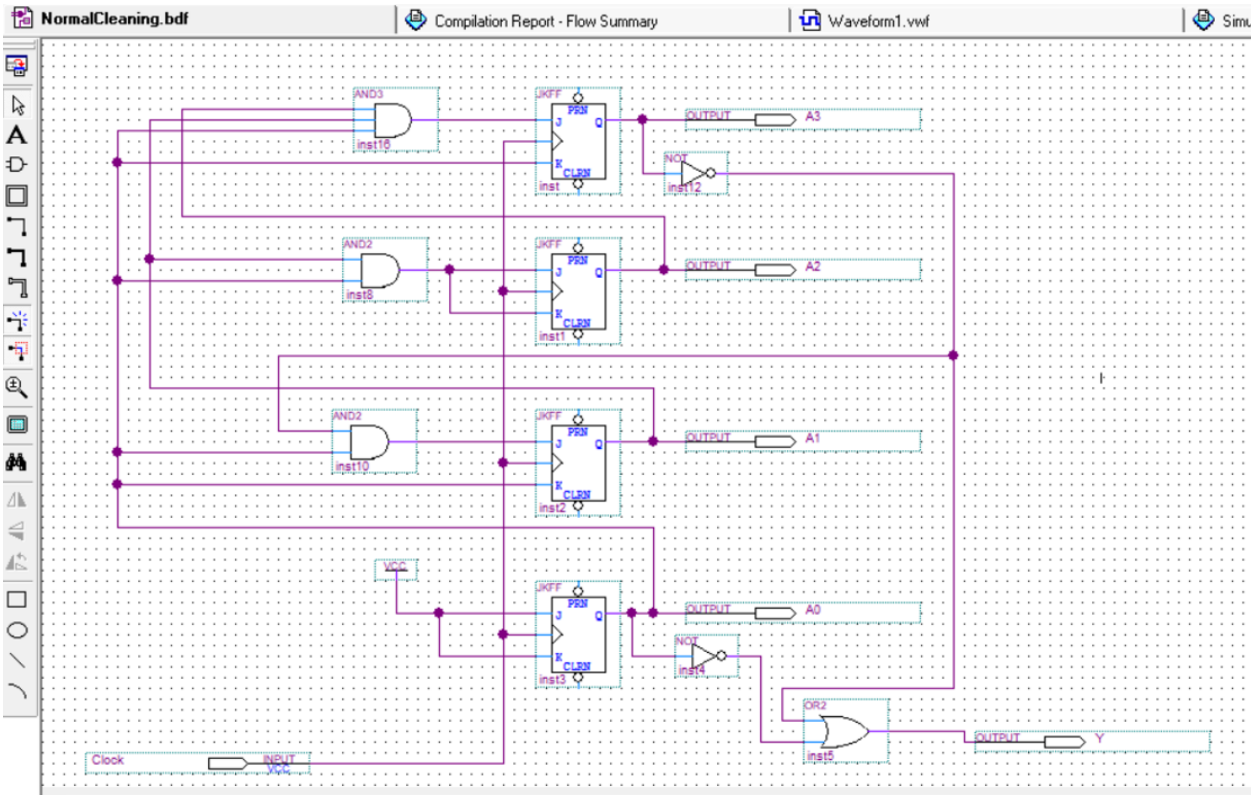
KA3:

	00	01	11	10
00	X	X	X	X
01	X	X	X	X
11	X	X	X	X
10	0	1	X	X

Figure 9: : K-map for KA3

KA3=A0

<u>Now Moving to the implementation on Quartus:</u>



In this Quartus implementation, we used 4 JK flip flops, since it's easier for implementation because there's a significant number of Don't cares in the Truth Table. We took the equations that we derived from the K-maps and connected the inputs accordingly. This is the original implementation without any minimization. We started with this design, then we tried to minimize it as much as possible. We took this Quartus implementation as a reference to check how much minimization we did, but most importantly to check if the results align with the

simplified design. In the original design, we had 3 inputs AND gate and one OR gate, we minimized and used only one AND gate instead.

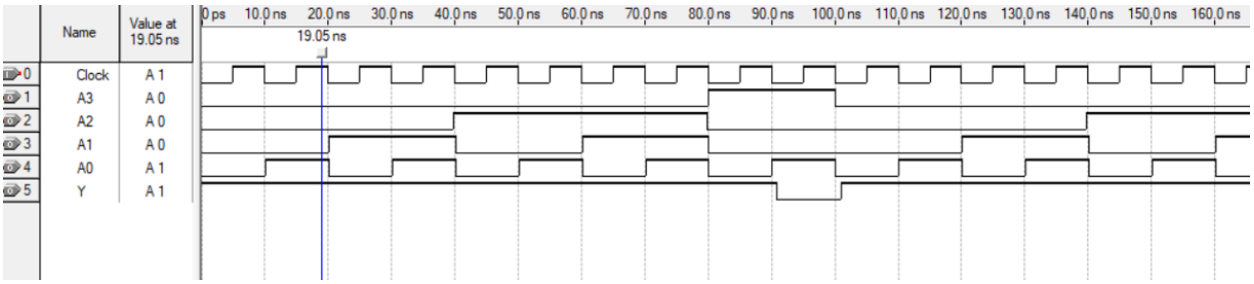


Figure 10-Normal Cleaning

We simulated our design using Altera Quartus, and this is the waveform we obtained. We notice that when the output Y reaches the 9th state, it turns down to 0, and that's a valid result, since the counter is designed to reach the 9th state and then reset to 0.

At first, the clock, A3, A2, A1, A0 and Y are all 0, then the clock gets to 1, but all the other inputs stay same. On the falling edge of the clock, A0 turns to 1 at 10 ns and then turns to 0 at 20ns and it stays like that (between 0 and 1 each 10ns). A1 turns to 1 at 20ns (when A0 turns to 0) and then turns to 0 at 40 ns and stays terminating between 0 and 1 each 20 ns. A2 turns to 1 at 40 ns and stays 1 till 80 ns, where it turns to 0, and stays terminating between 0 and 1 for 40 ns, then it turns to 1 at 140 and stays 1 for 4 ns. A3 stays 0 till 80 ns, where it turns to 1 and stays 1 till 100 ns.

Deep-Cleaning:

In this part, we can use exactly the same implementation from before however, we just invert the output to obtain the result needed.

Thus, we will use the same state table as before, but we'll create another output (named X) for the deep cleaning that occurs once every 10 uses…

Present State				Next state				Output	Flip Flop outputs								Output
A3	A2	A1	A0	A3	A2	A1	A0	Y	JA3	KA3	JA2	KA2	JA1	KA1	JA0	KA0	X
0	0	0	0	0	0	0	1	1	0	X	0	X	0	X	1	X	0
0	0	0	1	0	0	1	0	1	0	X	0	X	1	X	X	1	0
0	0	1	0	0	0	1	1	1	0	X	0	X	X	0	1	X	0
0	0	1	1	0	1	0	0	1	0	X	1	X	X	1	X	1	0
0	1	0	0	0	1	0	1	1	0	X	X	0	0	X	1	X	0
0	1	0	1	0	1	1	0	1	0	X	X	0	1	X	X	1	0
0	1	1	0	0	1	1	1	1	0	X	X	0	X	0	1	X	0
0	1	1	1	1	0	0	0	1	1	X	X	1	X	1	X	1	0
1	0	0	0	1	0	0	1	1	X	0	0	X	0	X	1	X	0
1	0	0	1	0	0	0	0	0	X	1	0	X	0	X	X	1	1
1	0	1	0	X	X	X	X	X	X	X	X	X	X	X	X	X	X
1	0	1	1	X	X	X	X	X	X	X	X	X	X	X	X	X	X
1	1	0	0	X	X	X	X	X	X	X	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Table 2: Truth table for deep cleaning

Using k-maps to get the function X:

X:

A3A2 \ A1A	00	01	11	10
00	1	1	1	1

01	1	1	1	1
11	X	X	X	X
10	1	0	X	X

Figure 11: K-map for X

X= A3' + A0'

Quartus Implementation:

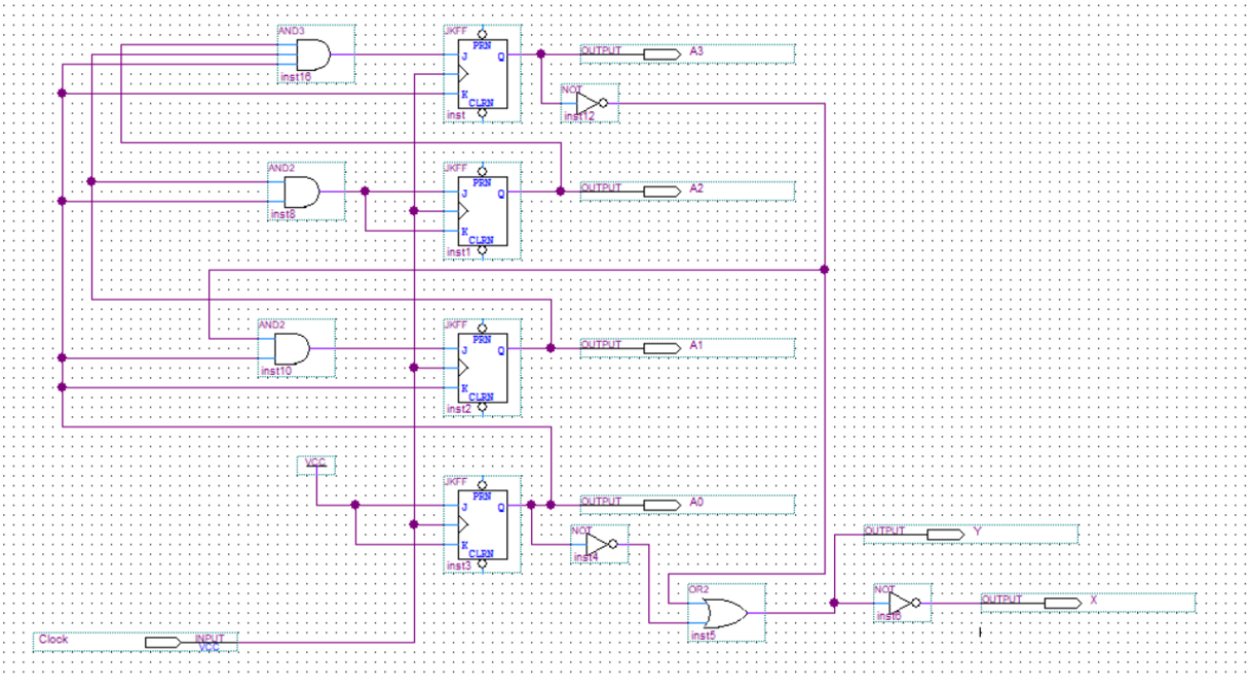


Figure 12: Circuit design for deep cleaning

The same design was used for the normal cleaning counter; however, we just added an inverter to get the output needed for the deep cleaning (named X).

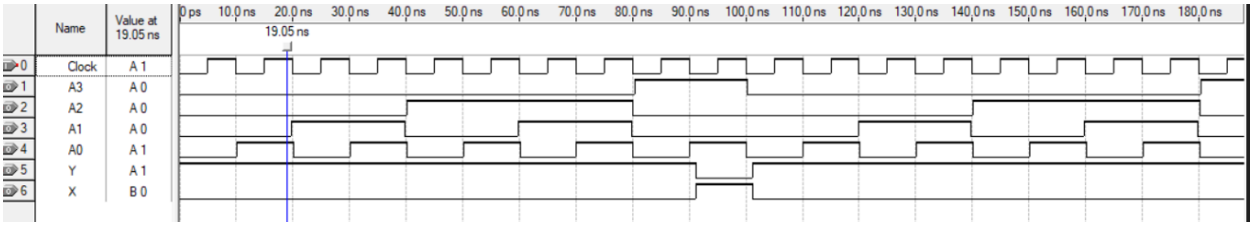


Figure 13-Deep Cleaning and Normal Cleaning

As clearly shown in the waveform stimulation, the results of X and Y are completely opposite, hence we can deduce that our goal is achieved in building 2 counters using the same design on Quartus and getting our valid results.

We also implemented the deep cleaning design using D-Flip Flops in order to justify our choose for the JK flipflops instead of D to get a less complex hardware.

The design used with D-flipflops is the following:

"Very complex and hard design"

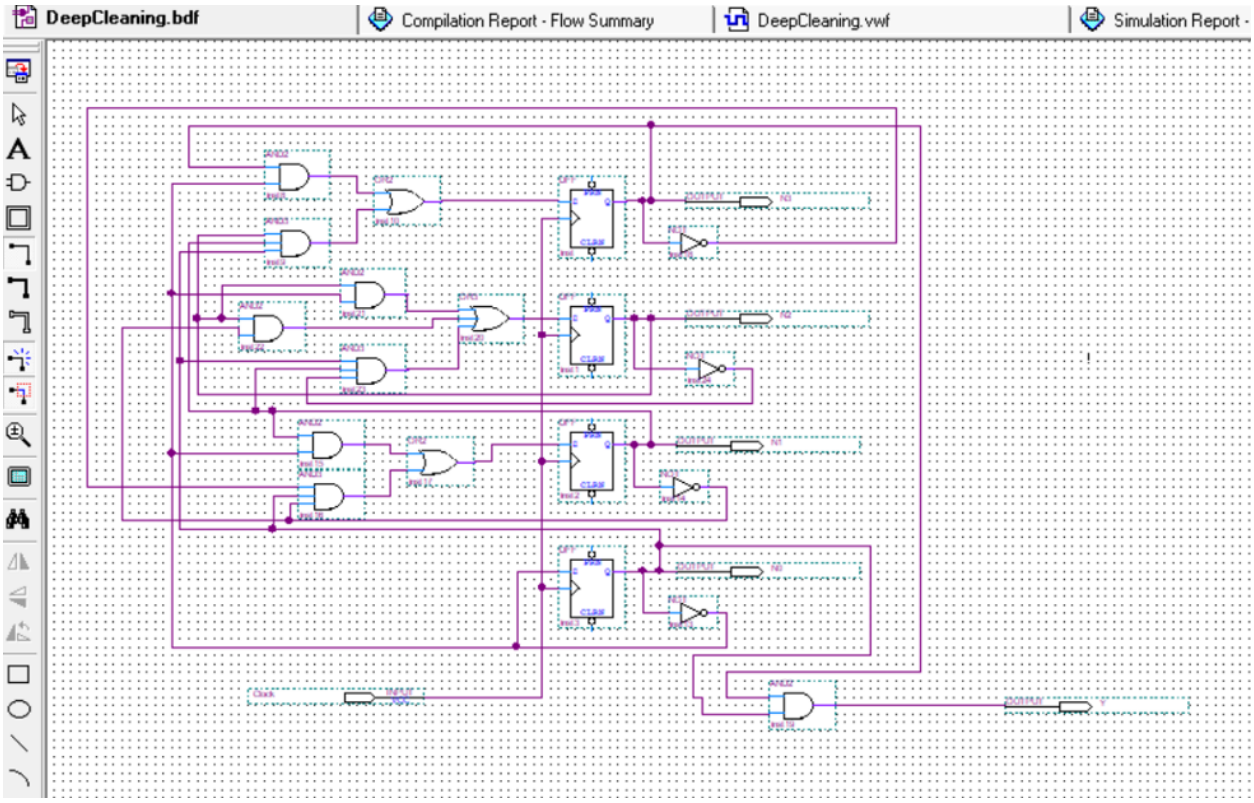


Figure 14: Circuit design for deep cleaning using D-flip flops

Clearly, we used a lot more gates, and clearly it is harder hardware to implement on breadboard.

But in order to ensure that our results are also correct using this design we did the stimulation too and we obtained the same results needed.

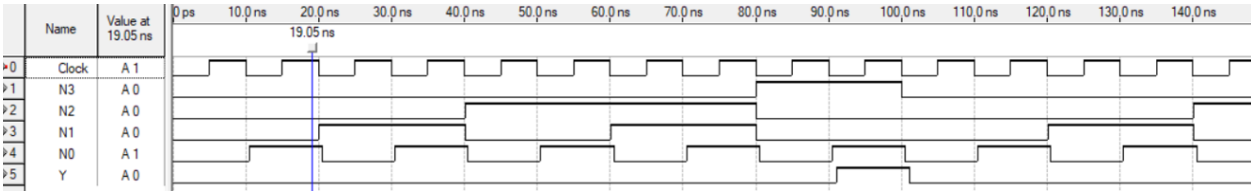


Figure 15-Deep Cleaning

The output only turns to 1 once at the 9th state since it is a deep cleaning. Therefore, the results ,match exactly the results we wanted using JK flipflops.

Conclusion: Using JK- flipflops will have a simpler and a much cheaper design. Thus, it is obvious that we will use the JK- flipflops in our design.

Breadboard:

In the following breadboard we implemented both the normal cleaning, and the deep cleaning using 2 chips holding 4 JK flipflops, 1 NOT gate, and 1 AND gate.

The following pictures are taken every second on the breadboard…

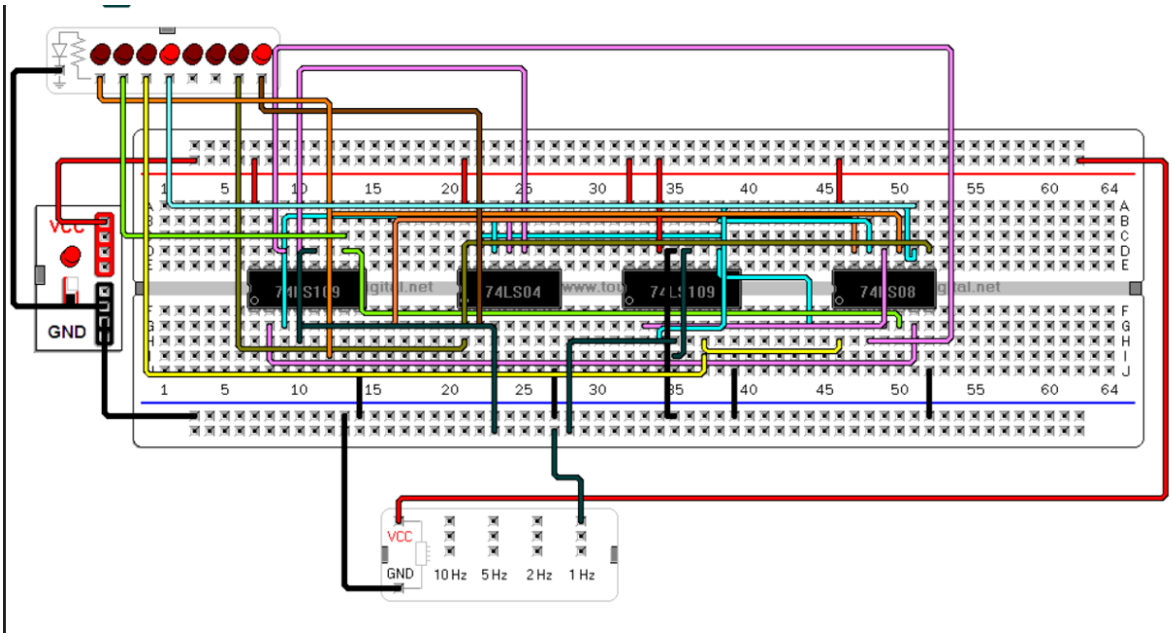


Figure 16-sec 1 (Normal Cleaning)

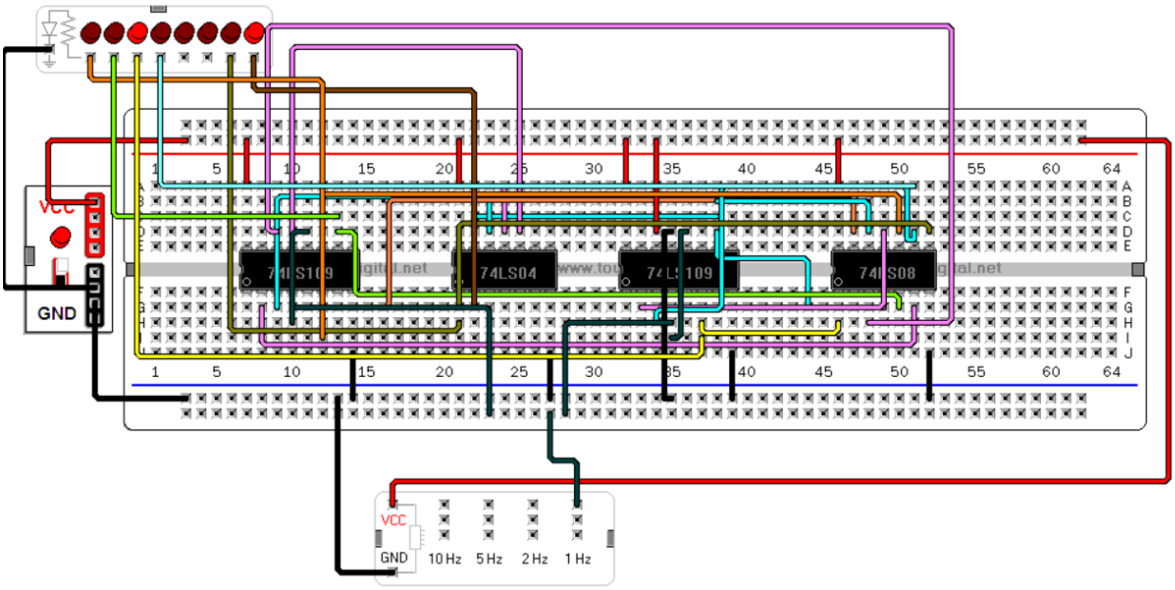


Figure 17-sec 2

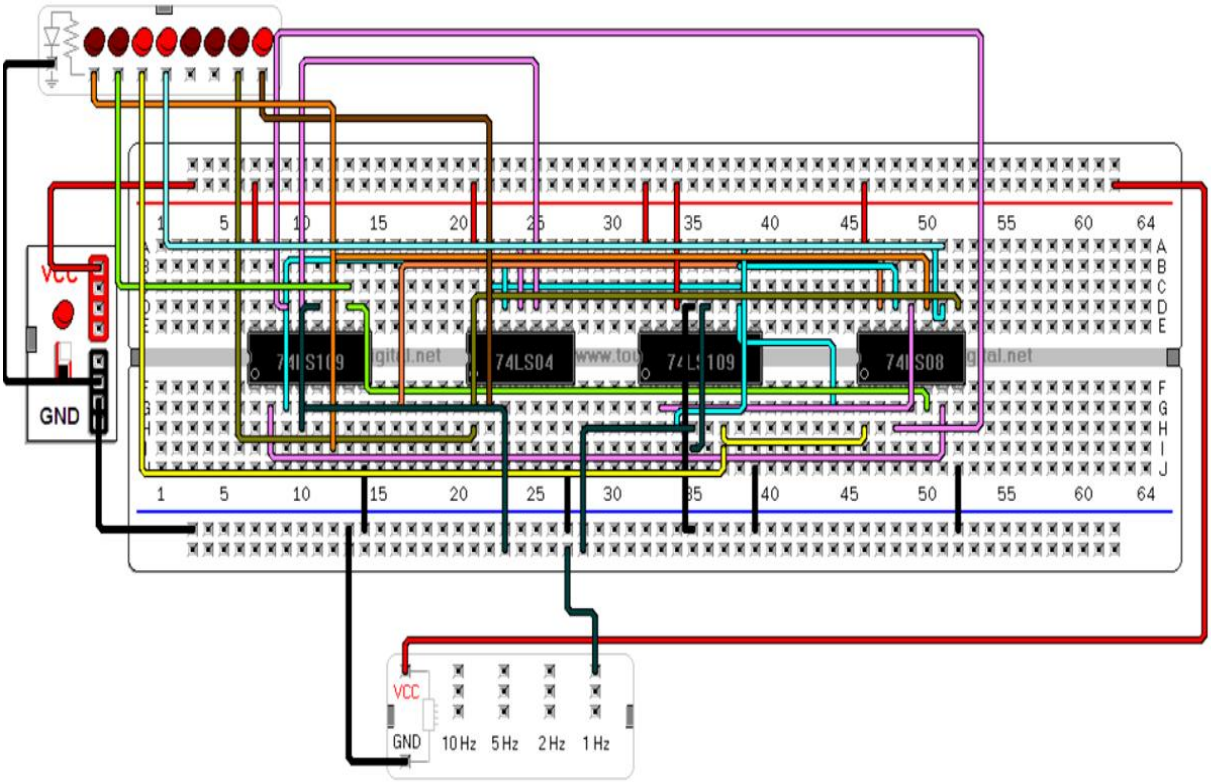


Figure 18-sec 3

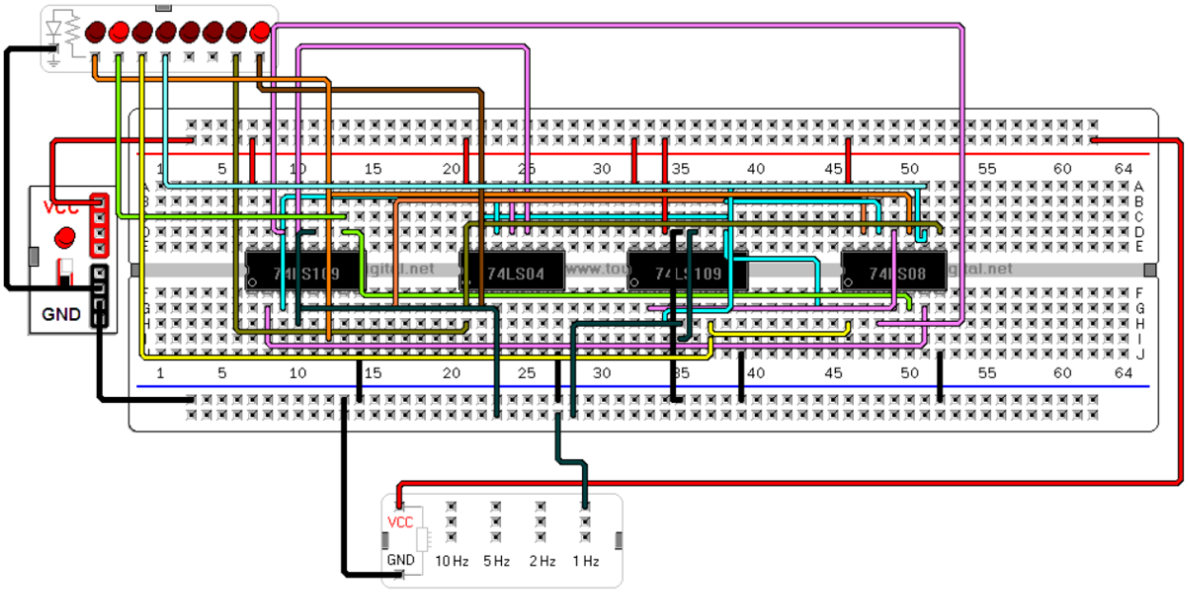


Figure 19-sec 4

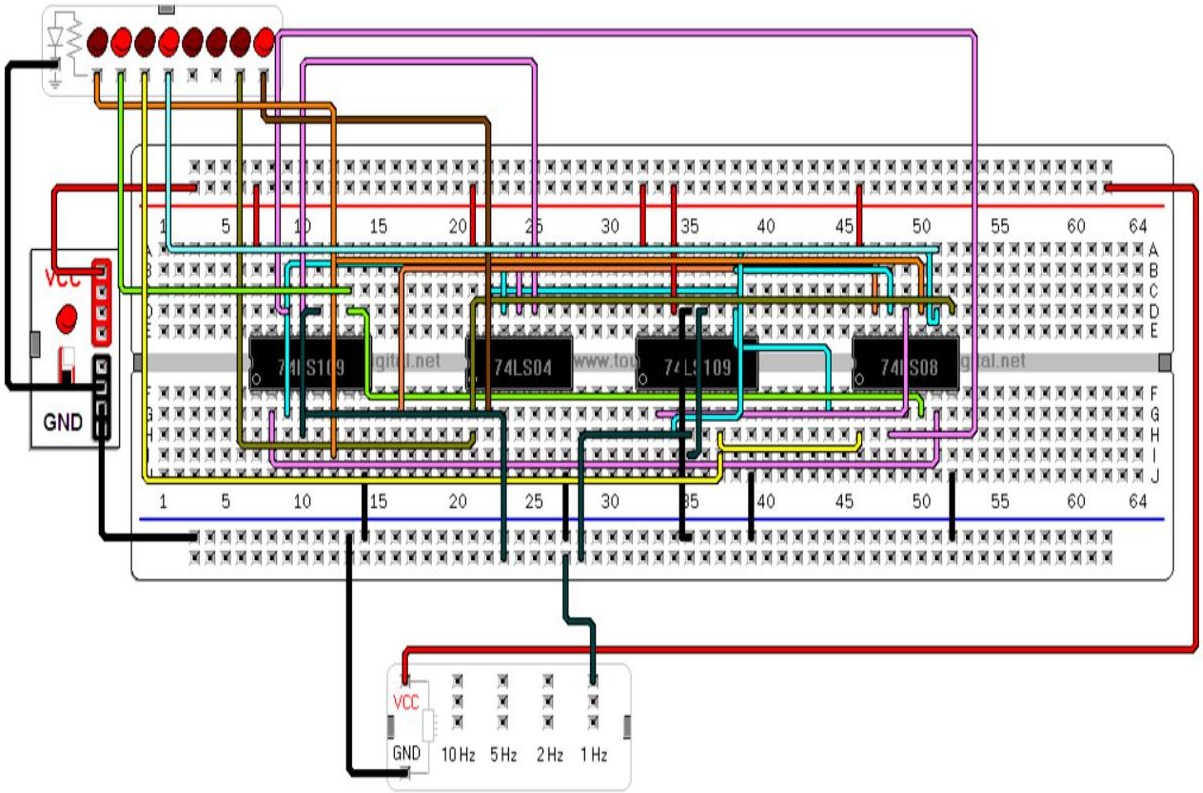


Figure 20-sec 5

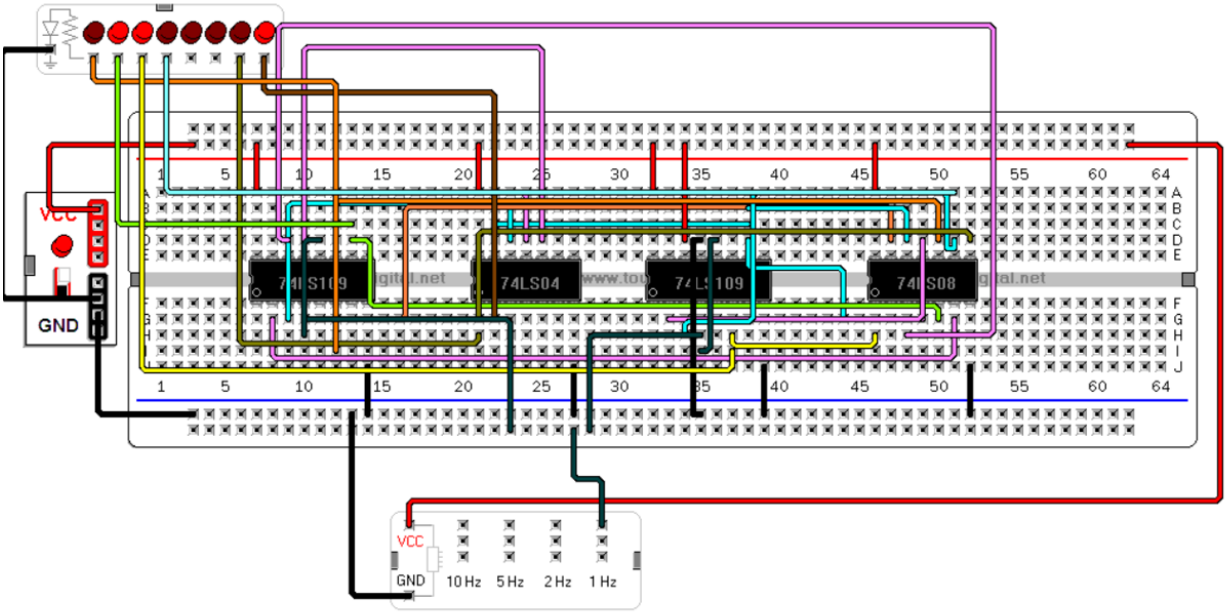


Figure 21-sec 6

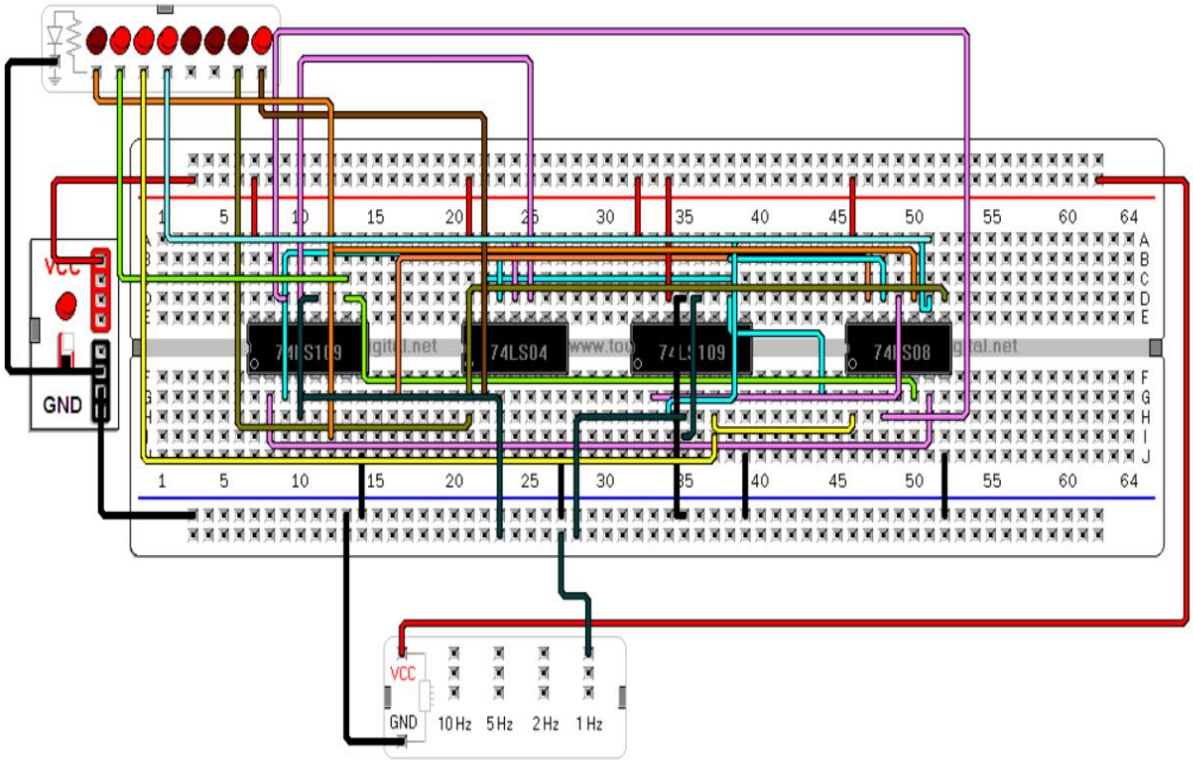


Figure 22- sec 7

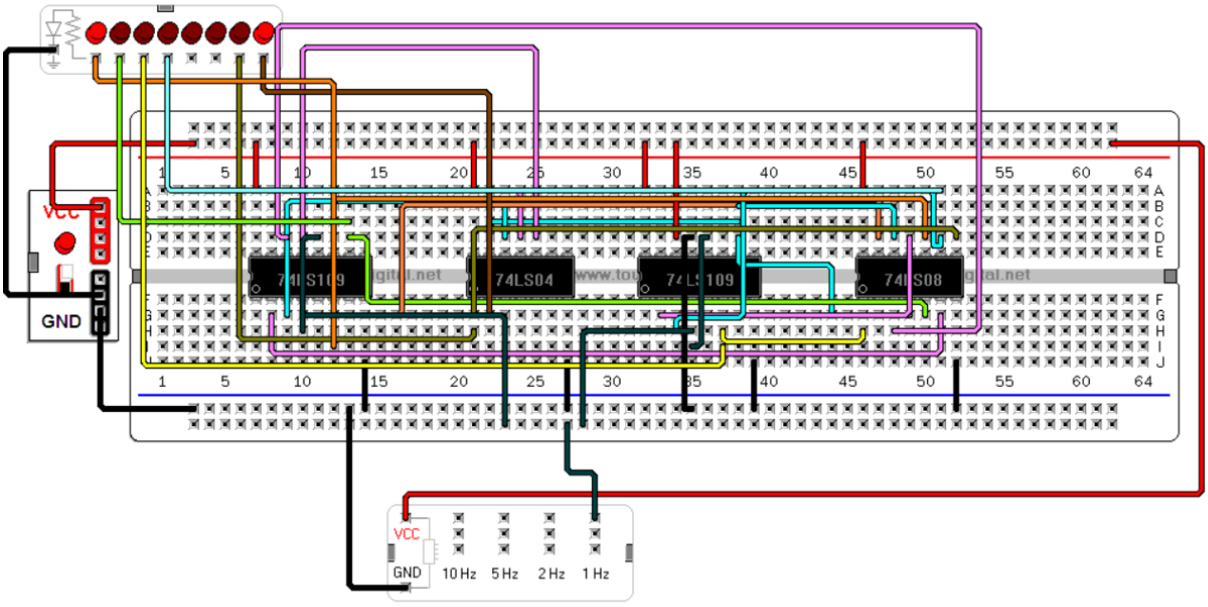


Figure 23- sec 8

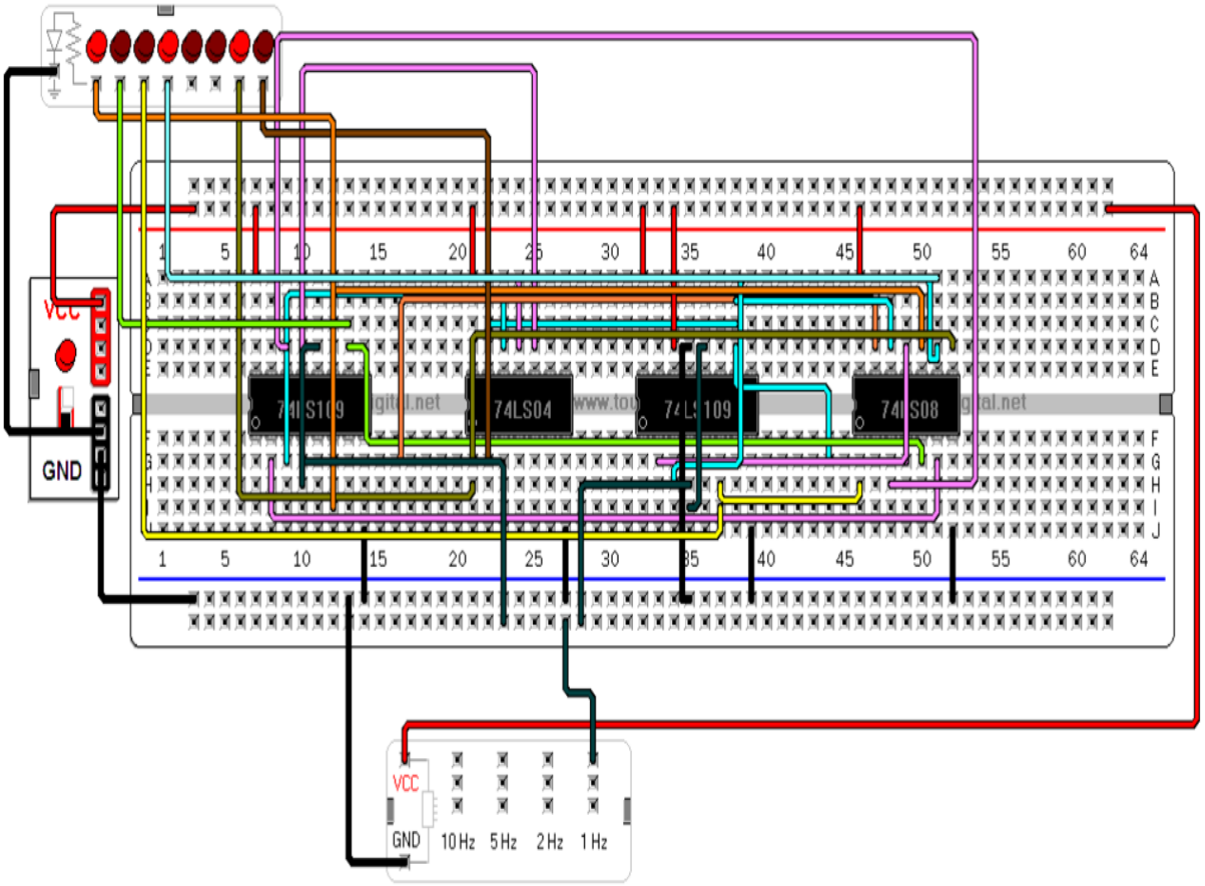


Figure 24- sec 9 (deep cleaning)

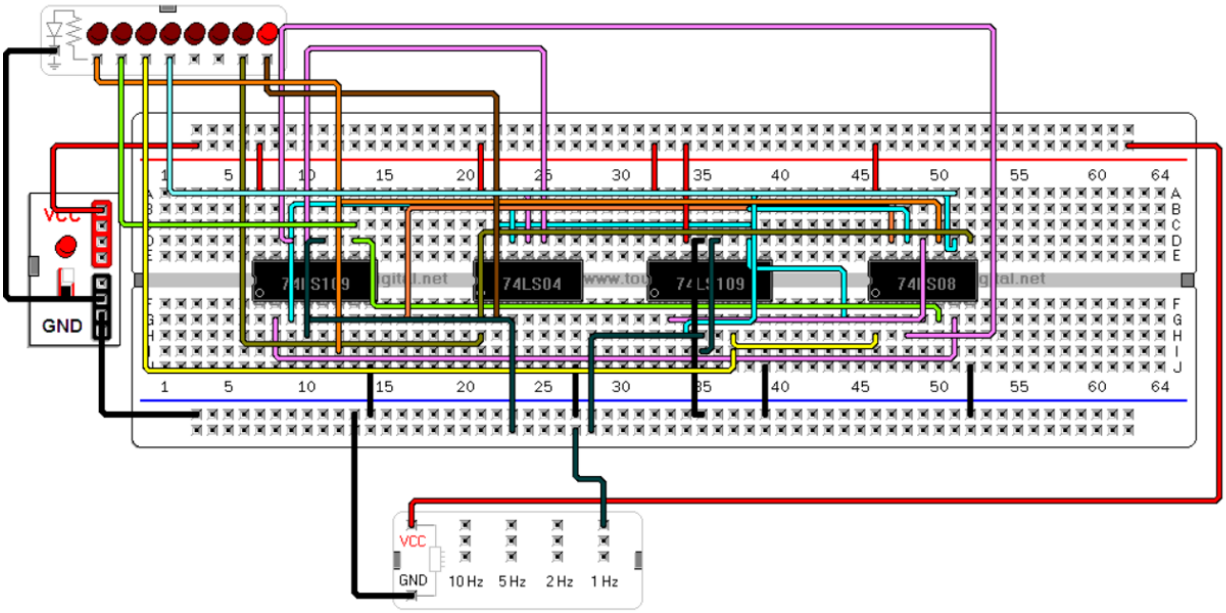


Figure 25- normal cleaning again

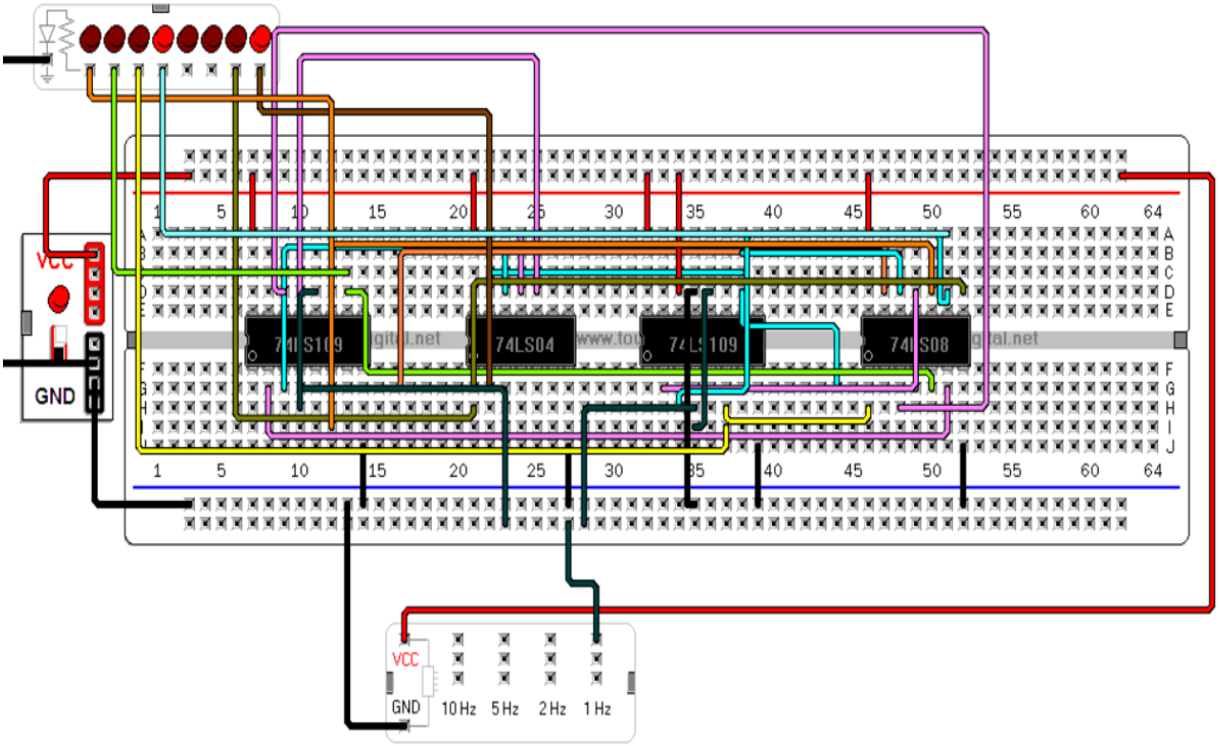


Figure 26- Repeating (sec 1 normal cleaning)

The output of the deep cleaning is quite the opposite to the output of the normal cleaning, which makes sense, since whenever there is deep cleaning (every 10 times we use the bathroom) there will be no normal cleaning and vice versa. Hence, we implemented both counters together.

Analysis of the breadboard:

On this breadboard, we used 4 JK flipflops (2 chips), a NOT gate, and an 2-inout AND gate. We connected each to Vcc and to the ground.

We used a clock, and we connect 1Hz to the flipflops, thus The time will be 1 sec between each state and the next one, since T=1/f = 1 seconds.

Then we connected each connection by looking at the outputs we get before for each function keeping a look on each chip datasheet.

The type of a chip used here is 74LS109 which contains 2 JK flipflops, however one thing is that it accepts J and K' as inputs. (K' instead of K in this chip datasheet), so we managed to get the function of K and then inverting it before plugging it in each flipflop as an input.

Door Lock:

Description:

In this part our aim is to ensure the application of the 4 set steps in order which are pressing the flush valve, using the soap, using the sink, and using paper towels. When these steps are applied in order, the door will open, and certain light bulbs will be lit. After each of the steps there is a corresponding light bulb that lights up inside the bathroom and finally, we have outer red-light bulb which lights when the door is opened or unlocked and turns off when the door is locked. To achieve our result, we applied it using a Moore state machine to switch from state to state and thus we used three JK-flip flops.

Inputs:

2 bits: 00,01,10,11 (each representing one input corresponding for each step for going from state to state.

States:

3 bits are used to represent each state, 5 states are taken 000, 001, 010, 011, 100.

Outputs:

L1: pressing the flush valve

L2: using the soap

L3: using the sink

L4: using paper towels

Red Light: if door locked opened or not (unoccupied or occupied toilet)

State Diagram:

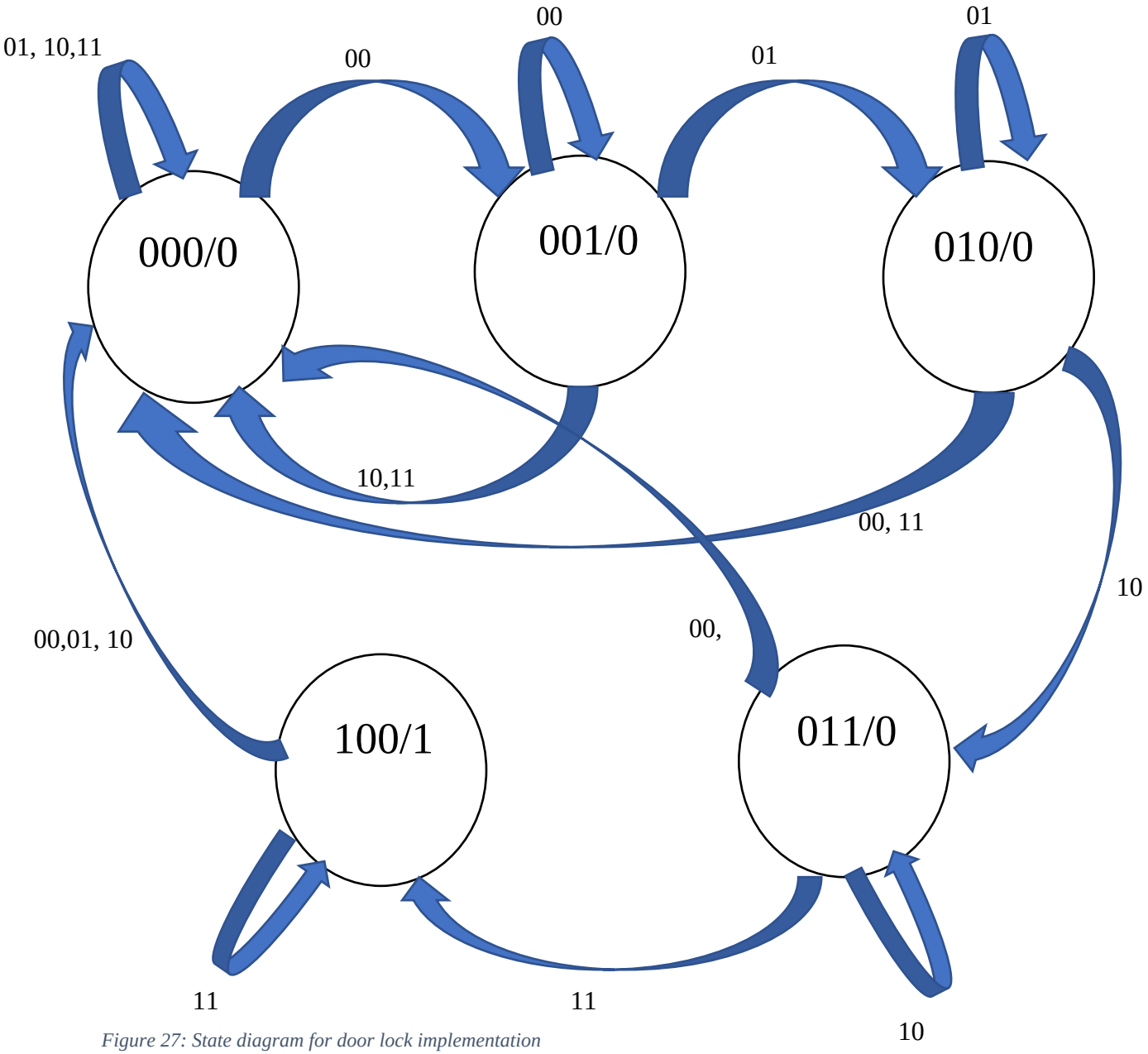


Figure 27: State diagram for door lock implementation

State Table:

Current State			Inputs		Next State			Output					Flip flops					
A	B	C	D	E	A+	B+	C+	O	L1	L2	L3	L4	JA	KA	JB	KB	JC	KC
0	0	0	0	0	0	0	1	0	1	0	0	0	0	×	0	×	1	×
			0	1	0	0	0	0	0	0	0	0	0	×	0	×	0	×
			1	0	0	0	0	0	0	0	0	0	0	×	0	×	0	×
			1	1	0	0	0	0	0	0	0	0	0	×	0	×	0	×
0	0	1	0	0	0	0	1	0	1	0	0	0	0	×	0	×	×	0
			0	1	0	1	0	0	1	1	0	0	0	×	1	×	×	1
			1	0	0	0	0	0	0	0	0	0	0	×	0	×	×	1
			1	1	0	0	0	0	0	0	0	0	0	×	0	×	×	1
0	1	0	0	0	0	0	0	0	0	0	0	0	0	×	×	1	0	×
			0	1	0	1	0	0	1	1	0	0	0	×	×	0	0	×
			1	0	0	1	1	0	1	1	1	0	0	×	×	0	1	×
			1	1	0	0	0	0	0	0	0	0	0	×	×	1	0	×
0	1	1	0	0	0	0	0	0	0	0	0	0	0	×	×	1	×	1
			0	1	0	0	0	0	0	0	0	0	0	×	×	1	×	1
			1	0	0	1	1	0	1	1	1	0	0	×	×	0	×	0
			1	1	1	0	0	1	1	1	1	1	1	×	×	1	×	1
1	0	0	0	0	0	0	0	0	0	0	0	0	×	1	0	×	0	×
			0	1	0	0	0	0	0	0	0	0	×	1	0	×	0	×
			1	0	0	0	0	0	0	0	0	0	×	1	0	×	0	×
			1	1	1	0	0	1	1	1	1	1	×	0	0	×	0	×
1	0	1	0	0	×	×	×	×	×	×	×	×	×	×	×	×	×	×
			0	1	×	×	×	×	×	×	×	×	×	×	×	×	×	×
			1	0	×	×	×	×	×	×	×	×	×	×	×	×	×	×
			1	1	×	×	×	×	×	×	×	×	×	×	×	×	×	×
1	1	0	0	0	×	×	×	×	×	×	×	×	×	×	×	×	×	×
			0	1	×	×	×	×	×	×	×	×	×	×	×	×	×	×
			1	0	×	×	×	×	×	×	×	×	×	×	×	×	×	×
			1	1	×	×	×	×	×	×	×	×	×	×	×	×	×	×
1	1	1	0	0	×	×	×	×	×	×	×	×	×	×	×	×	×	×
			0	1	×	×	×	×	×	×	×	×	×	×	×	×	×	×
			1	0	×	×	×	×	×	×	×	×	×	×	×	×	×	×
			1	1	×	×	×	×	×	×	×	×	×	×	×	×	×	×

Table 3: State table for door lock implementation

Functions:

Functions obtained using Online K-Map solver since we have 5 Variables:

O=ADE+BCDE

L1= B’CD’+BDE’+BCD+ADE+A’B’D’E’+BC’D’E

L2=BDE’+BCD+ADE+B’CD’E+ BC’D’E

L3=BDE’+BCD+ADE

L4=ADE+BCDE

JA=BCDE

KA=D’+E’

JB=CD’E

KB=D’E’+DE+CD’

JC=BDE’+A’B’D’E’

KC=E+B’D+BD’

Quartus:

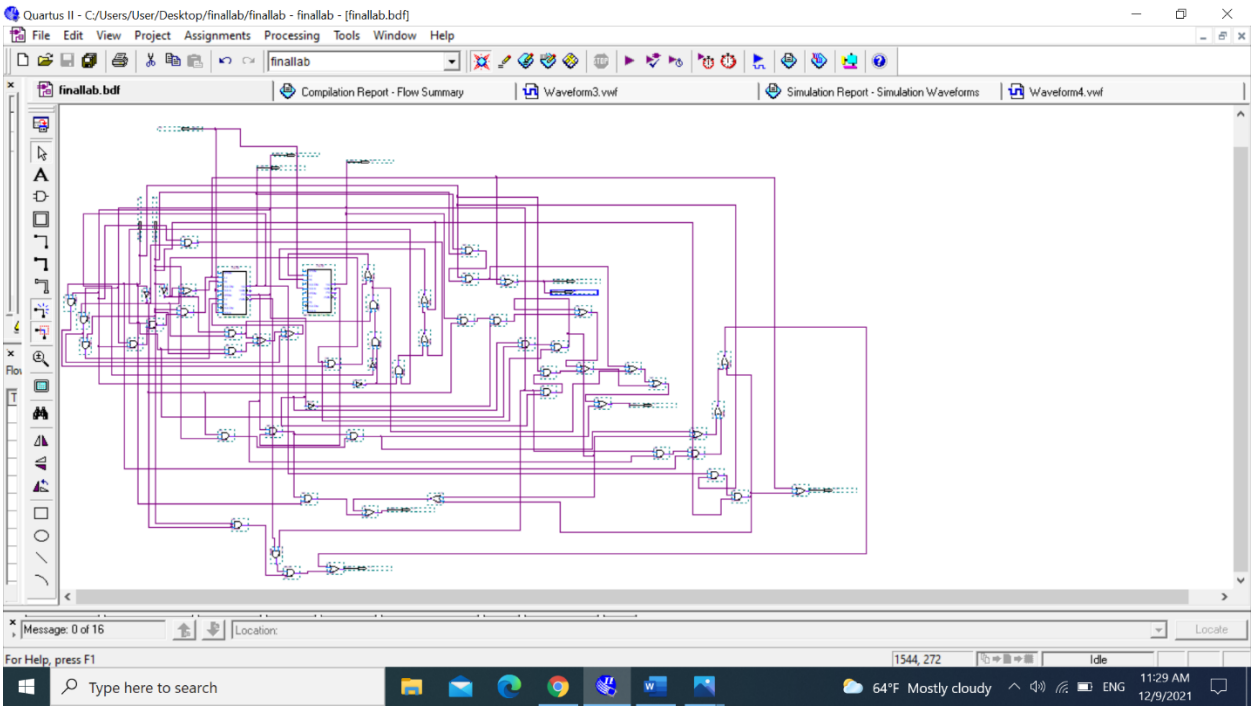


Figure 28: Circuit design for door lock implementation

How was it designed:

To implement our design, we used 3 JK flip flops (2 chips each consisting of two of them 2-input OR gates, 2-input AND gates, and NOT gate. According to the functions above, we constructed

the circuit by connecting the resulting combination of each circuit to the corresponding pins of the flip flops. Then, we connect the J and K input functions to the chip and connect the output of each JK flip flop Q considering it as the bit of each state, so every flip flop leads to one bit of the state. The clocks of all three flip flops are connected to function at the same rhythm. We add the input pins of D and E and the output pins of O and the lights L1, L2, L3, L4, and the red-Light bulb. As well we add output pins on the outputs of the flip flops to keep track of the states when testing the circuit using the simulation tool on quartus.

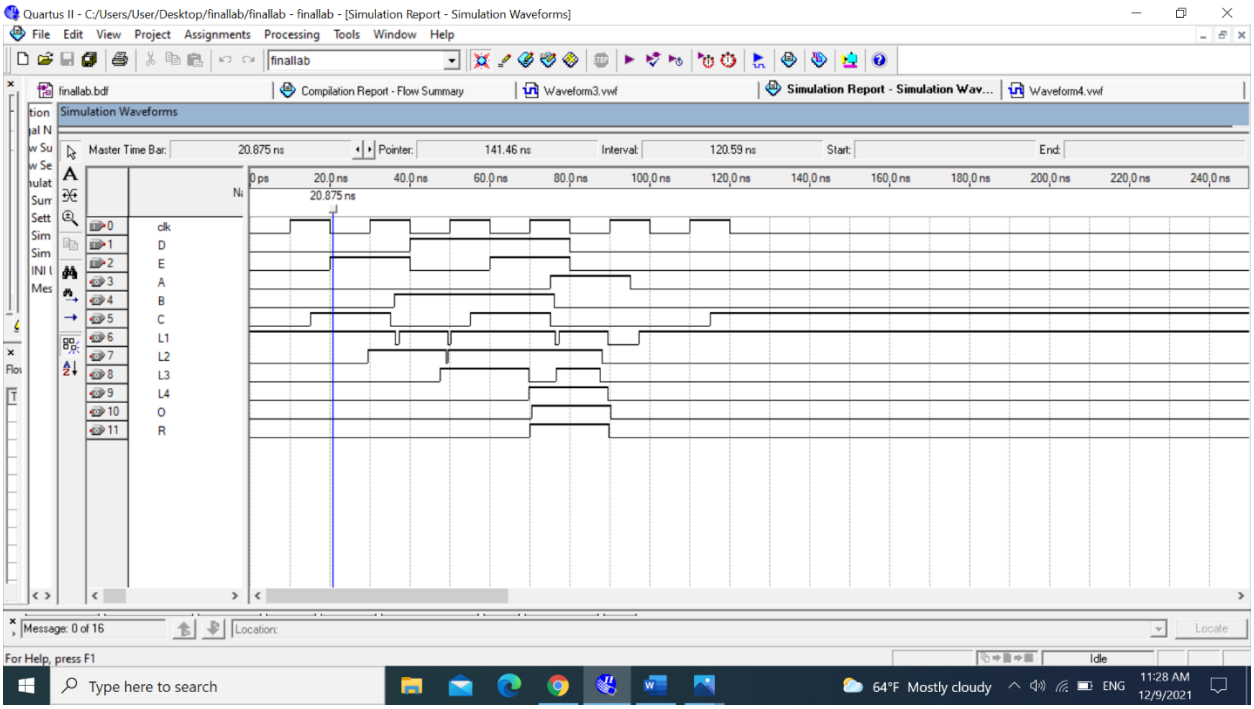


Figure 29: Simulation of door lock circuit design

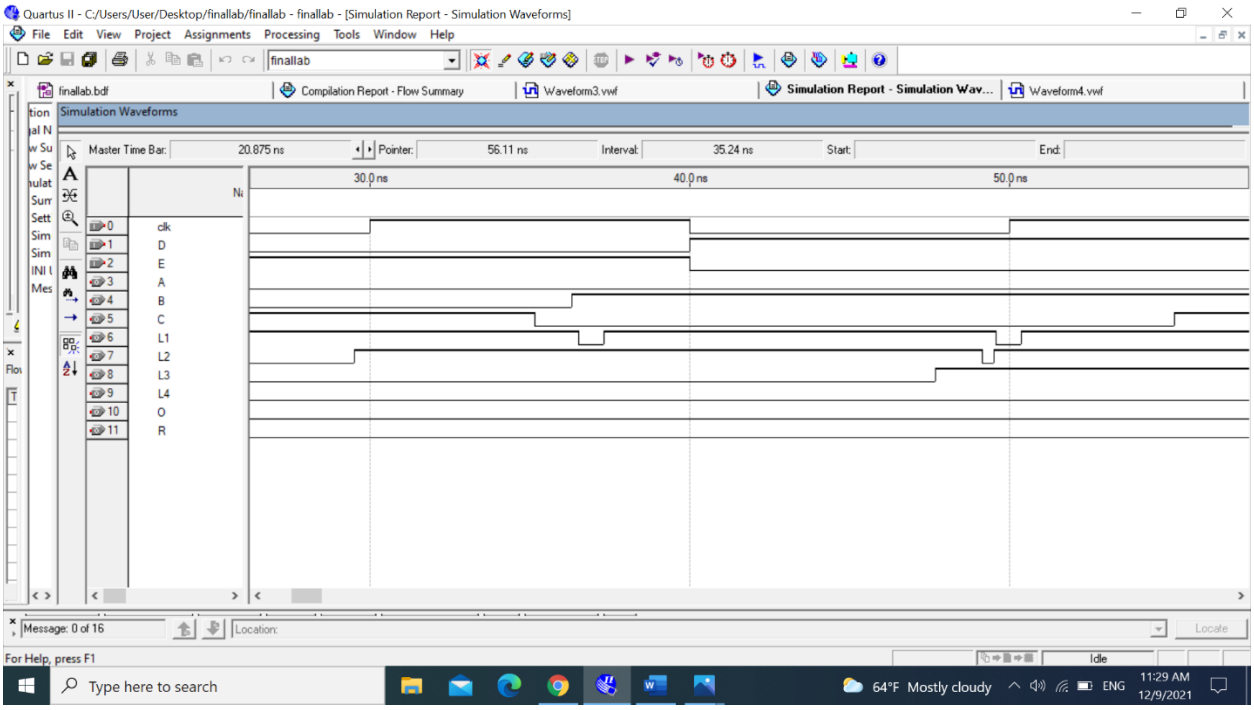


Figure 30: Simulation of door lock circuit design (a closer look)

In the above simulation we are testing for all results of the design by referring to every state and input.

For state 000, when the input is 10 and we find out that L1, L2, L3, L4, and the red light are 0 our findings in the truth table. Inputing 10 at the state 001 and at 100 will be at state 000 and the 0s for the first step L1, second step L2, third step L3, and fourth step L4 indicate nothing happening. Also we notice that the lock light bulb denoted by R(considering it the outer red light) is 0 so the toilet door is locked and the toilet is still occupied.

For state 001, when the input is 00 and we find out that L1 is 1 and L2, L3, L4, and the red light are 0 our findings in the truth table. Inputing 00 at the state 000 and at 001 will be at state 001, the 1 for the first step L1, the 0s for the second step L2, third step L3, and fourth step L4 indicate the results. Also we notice that the lock light bulb denoted by R(considering it the outer red light) is 0 so the toilet door is locked and the toilet is still occupied.

For state 010, when the input is 01 and we find out that L1 and L2 are 1 and L3, L4, and the red light are 0 supporting our findings in the truth table. Inputing 01 at the state 001 and at 010 will be at state 010, the 1 for the first step L1 and second step L2 , the 0s for the third step L3, and fourth step L4 indicate the results. Also we notice that the lock light bulb denoted by R(considering it the outer red light) is 0 so the toilet door is locked and the toilet is still occupied.

For state 011, when the input is 10 and we find out that L1, L2, and L3 are 1 and L4 and the red light are 0 supporting our findings in the truth table. Inputing 10 at the state 010 and at 011 will be at state 011, the 1 for the first step L1, second step L2, and the third step L3 , and the 0s for

fourth step L4 indicate the results. Also we notice that the lock light bulb denoted by R(considering it the outer red light) is 0 so the toilet door is locked and the toilet is still occupied.

For state 100, when the input is 11 and we find out that L1, L2, L3, L4, and the red light are 1 our findings in the truth table. Inputing 11 at the state 011 and at 100 will be at state 100 and the 1s for the first step L1, second step L2, third step L3, and fourth step L4. Also we notice that the lock light bulb denoted by R(considering it the outer red light) is 1 so the toilet door is unlocked and the toilet becomes unoccupied.

Unfortunately, the simulation contined numerous delays of which include between 30ns and 40ns which is starting at approxametly at 37.58ns and another inlcude one between the interval 70ns and 80ns which starts at approximately 78.6ns.

Breadboard:

How it was designed:

To implement our design, we used 3 JK flip flops (2 chips each consisting of two of them 74LS109), 2 3-input AND gates(74LS11), 1 2-input OR gate(74LS32) , 2-input AND(74LS08) gate, and 1 NOT gate(74LS04). According to the functions above, we constructed the circuit by connecting the resulting combination of each circuit to the corresponding pins of the flip flops. We start by connecting each chip to the ground and VCC according to the corresponding data sheets of every chip. Then, we connect the J and K input functions to the chip and connect the output of each JK flip flop Q considering it as the bit of each state, so every flip flop leads to one bit of the state. We should note that the chip contains K’ so we should include an inverter to get K when needed. The clocks of all three flip flops are connected to function at the same rhythm. After being connected all together, they are connected to the frequency of 1Hz. The VCC of switches and frequency generator are connected and the ground lights, switches, and frequency generator are connected.

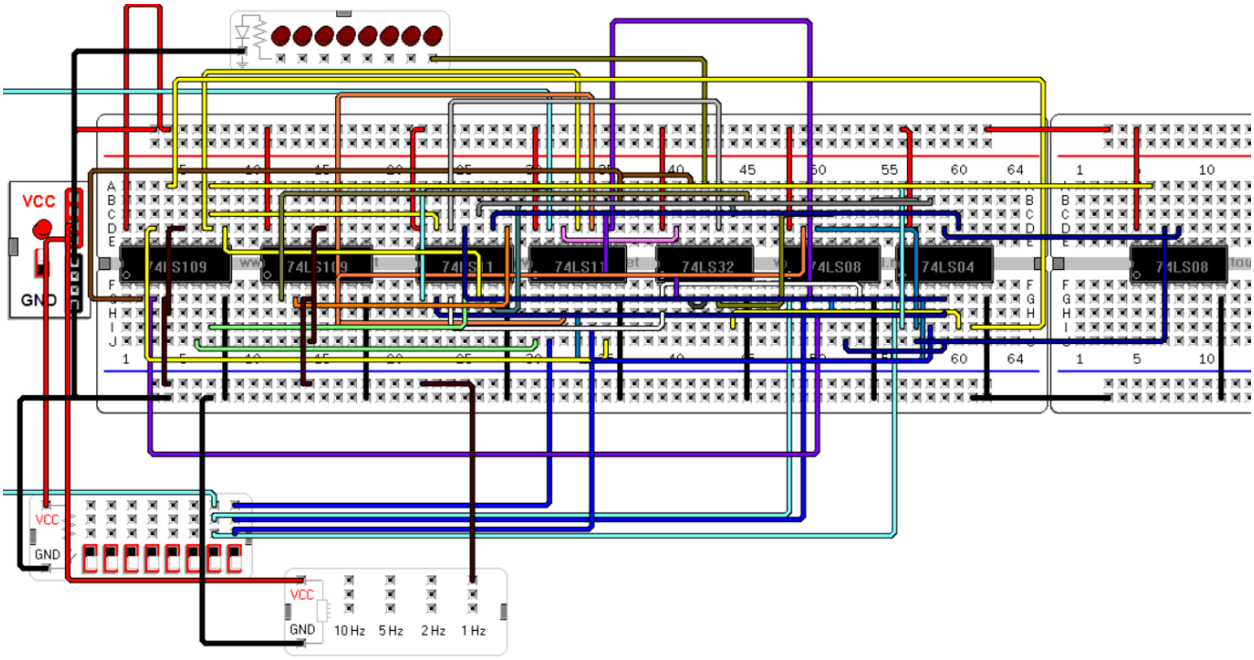


Figure 31: Breadboard representing input 00

When we set the switches of the input to 00, clearly the results conform with the ones presented in the truth table where for any state varying form 000 up till 100, the output will be zero in all cases which conforms with all the cases stated in the truth table. So, the light bulb connected to test the door lock is still off which implies the lock is still closed.

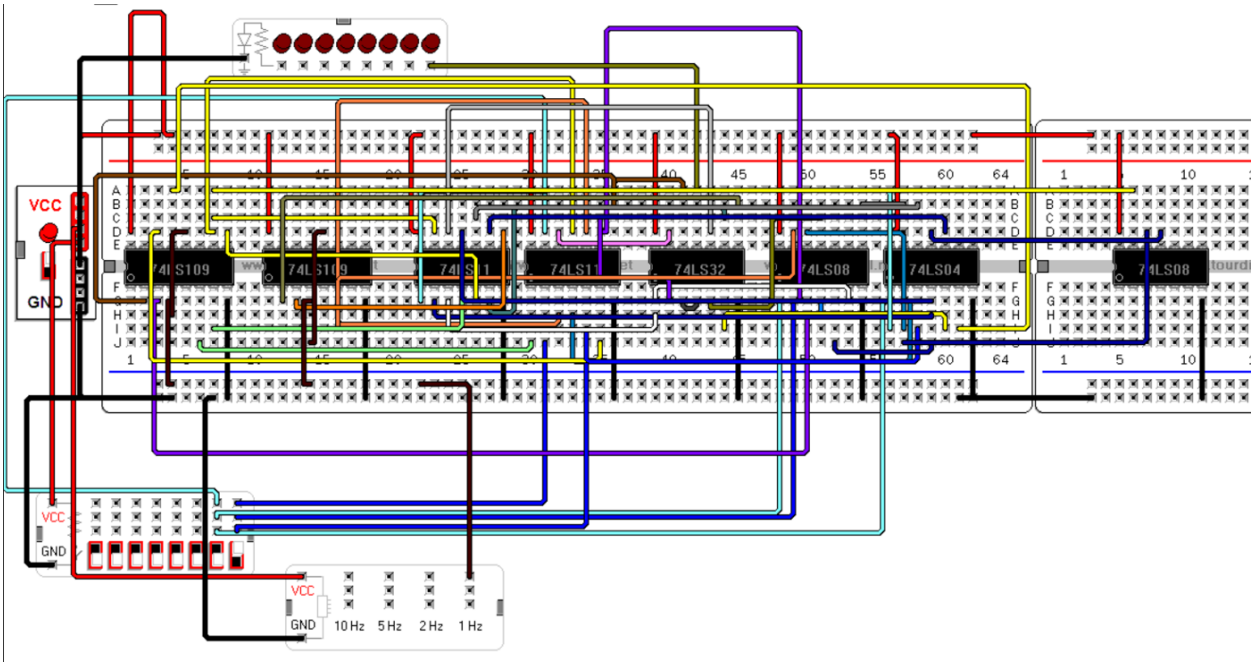


Figure 32: Breadboard representing input 01

When we set the switches of the input to 01, clearly the results conform with the ones presented in the truth table where for any state varying form 000 up till 100, the output will be zero in all cases which conforms with all the cases stated in the truth table. So, the light bulb connected to test the door lock is still off which implies the lock is still closed.

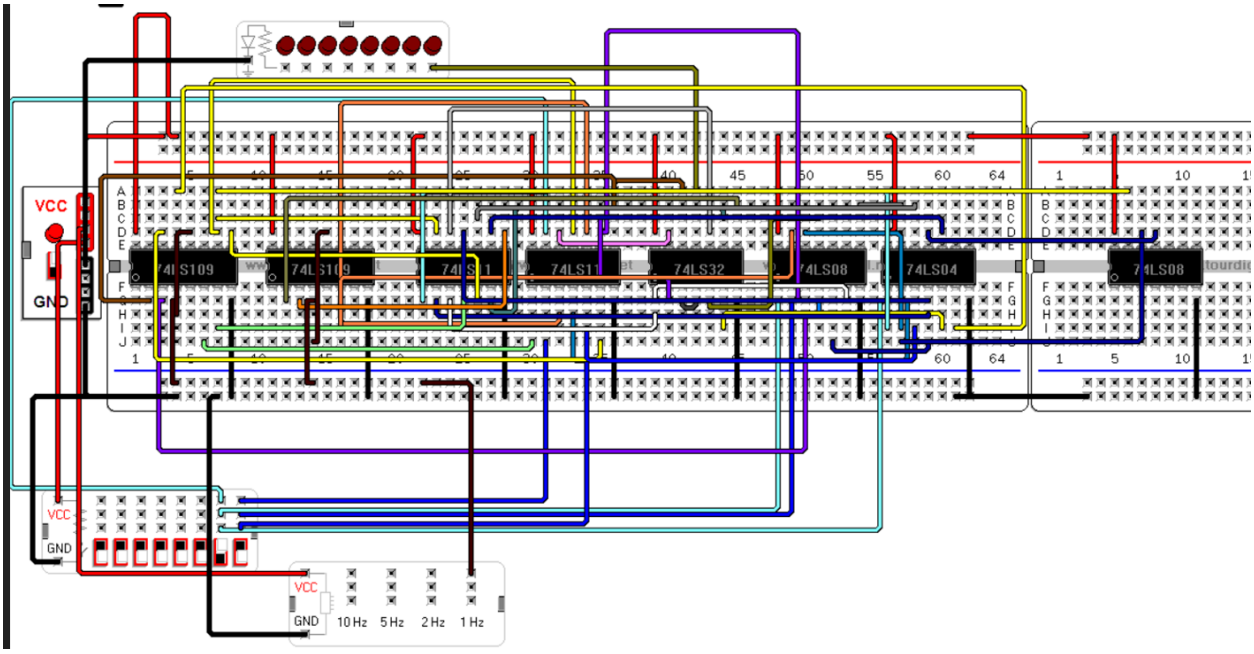


Figure 33: Breadboard representing input 10

When we set the switches of the input to 10, clearly the results conform with the ones presented in the truth table where for any state varying form 000 up till 100, the output will be zero in all cases which conforms with all the cases stated in the truth table. So, the light bulb connected to test the door lock is still off which implies the lock is still closed.

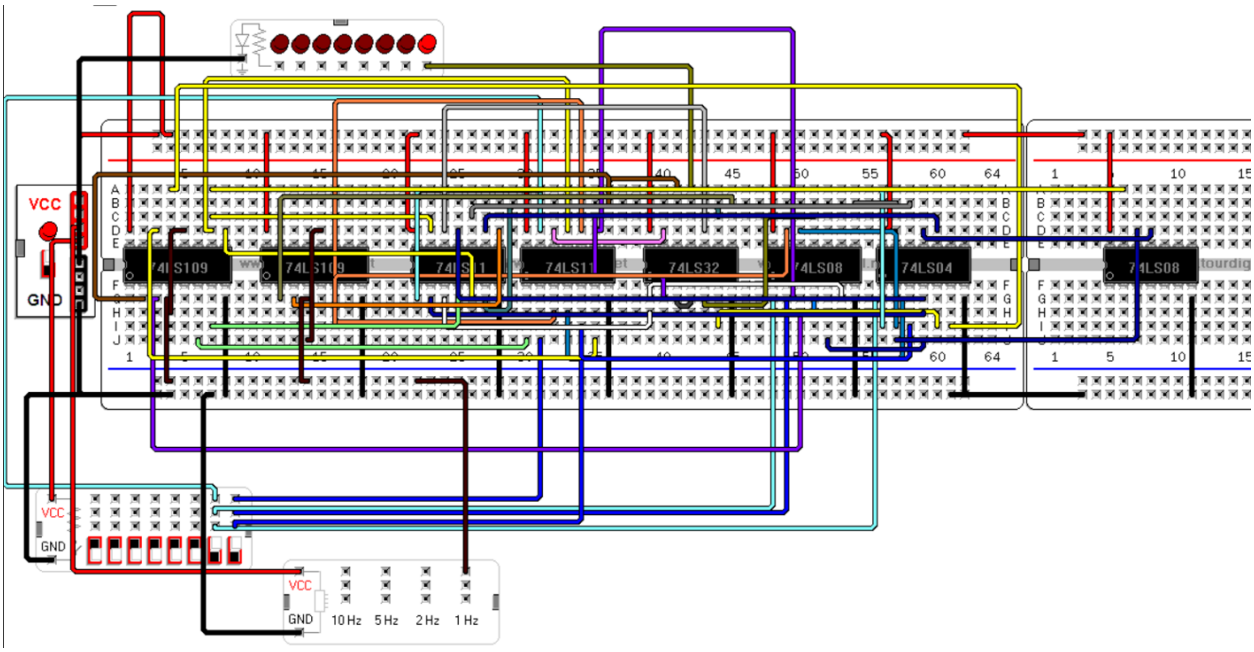


Figure 34: Breadboard representing input 11

When we set the switches of the input to 11, clearly the results conform with the ones presented in the truth table where for any state varying form 000 up till 011, the output will be zero in all cases which conforms with all the cases stated in the truth table. So, the light bulb connected to test the door lock is still off which implies the lock is still closed. When the state becomes 100, the light bulb lightens up which indicates that we have reached this state and thus the door lock opens.

We have chosen some cases to prove that our logic is right and that it conforms with the truth table.

Now we have connected the lights to test if they conform with our truth table:

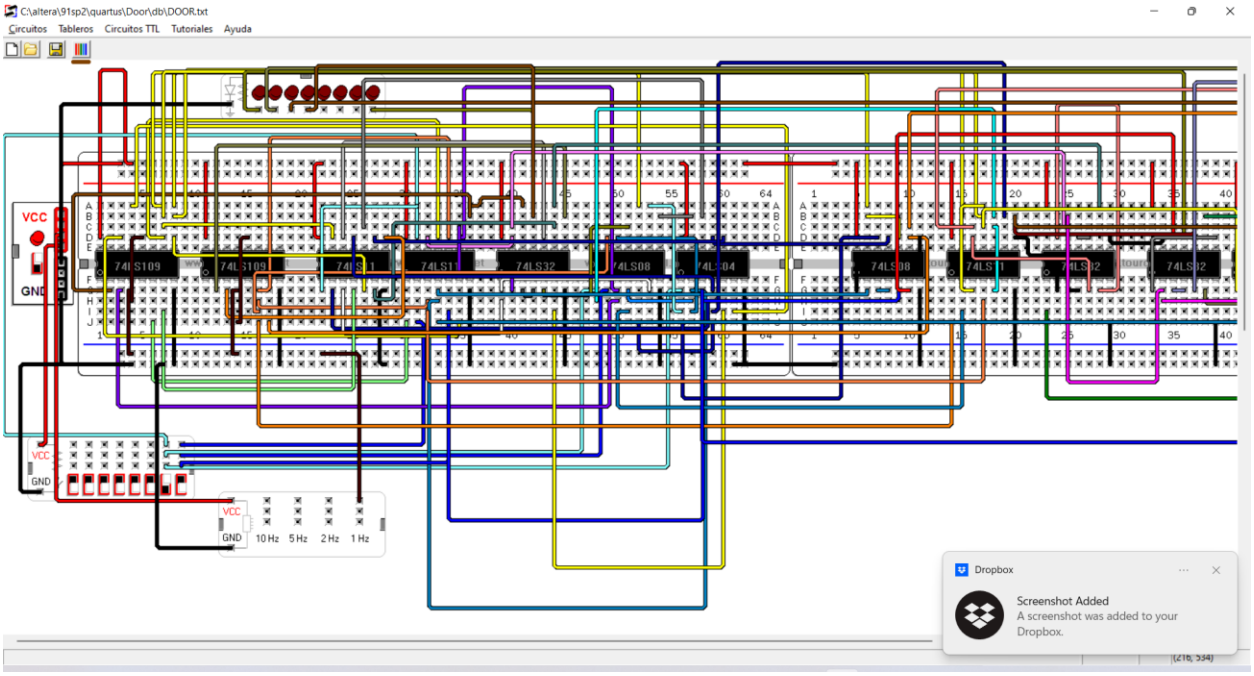


Figure 35: Breadboard representing state 000

In the above figure we are testing the the state is 000, we took as exmaple the case of the input 10 and we find out that all light bulbs are off which conforms with our findings in the truth table. This case is taken to test state 000.Inputing 10 at the state 001 and at 100 will return to the initial state 000 and emitting no light from the light bulb corresponding to the first step L1. Also we notice that the lock light bulb which may be indicated by the light bulb(considering it the outer

red light) is off so the toilet is still occupied and the door remains locked.

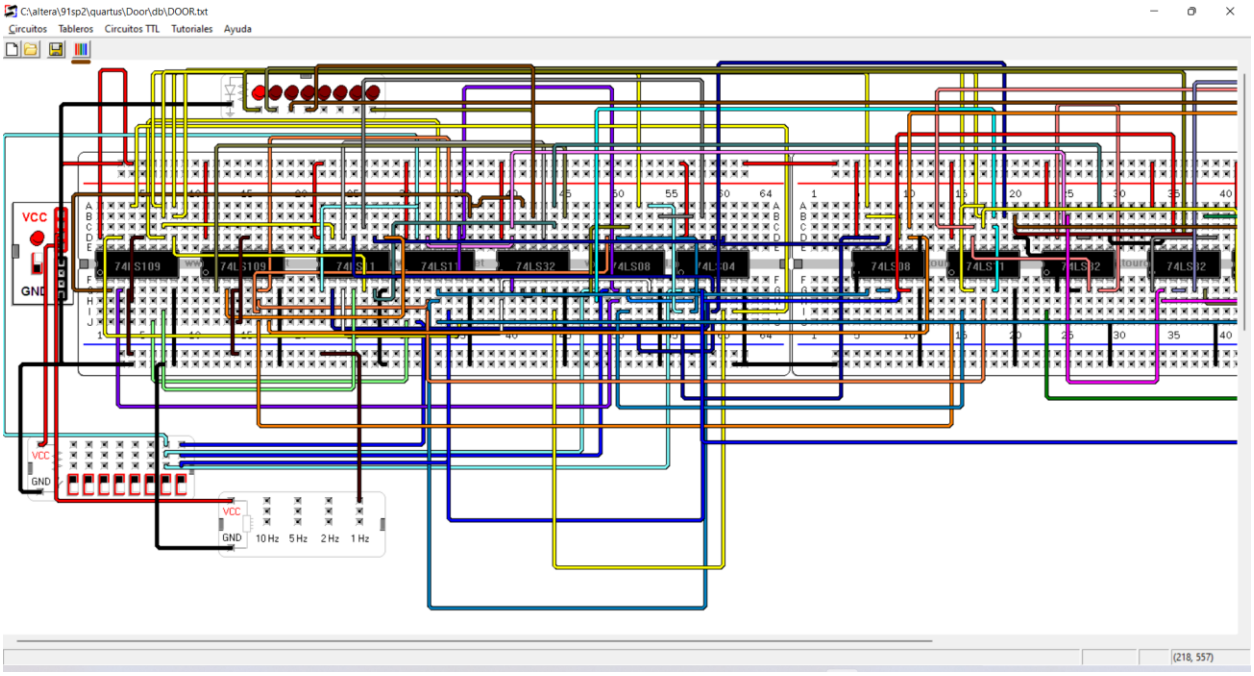


Figure 36: Breadboard representing state 001

In the above figure we are testing the the state is 001, we took as exmaple the case of the input 00 and we find out that L1 is lit up which conforms with our findings in the truth table. This case is taken to test state 001.Inputing 00 at the state 000 and at 001 will be at state 001 emitting light from the light bulb corresponding to the first step L1. Also we notice that the lock light bulb which may be indicated by the light bulb(considering it the outer red light) is off so the toilet is

still occupied and the door remains locked.

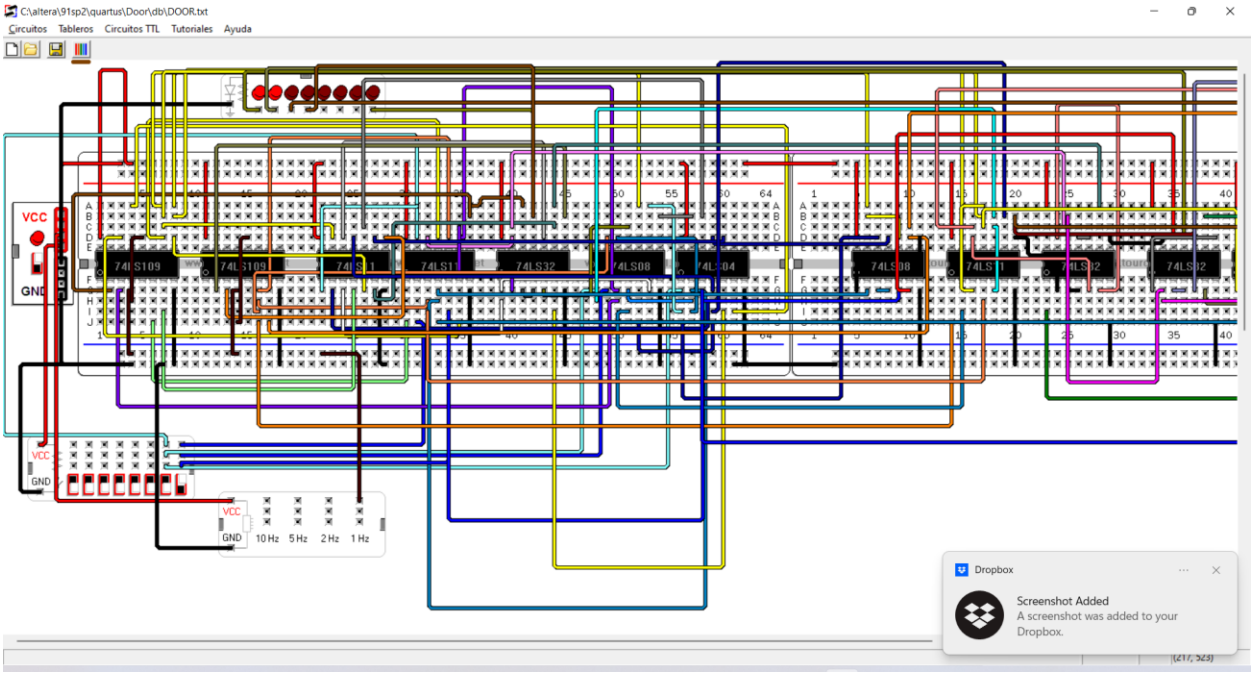


Figure 37:Breadboard representing state 010

In the above figure we are testing the the state is 010, we took as exmaple the case of the input 01 and we find out that L1 and L2 are lit up which conforms with our findings in the truth table. This case is taken to test state 010.Inputing 10 at the state 001 and at 010 will be at state 010 and emitting light from the light bulb corresponding to the first step L1 and second step L2. Also we notice that the lock light bulb which may be indicated by the light bulb(considering it the outer red light) is off so the toilet is still occupied and the door remains locked.

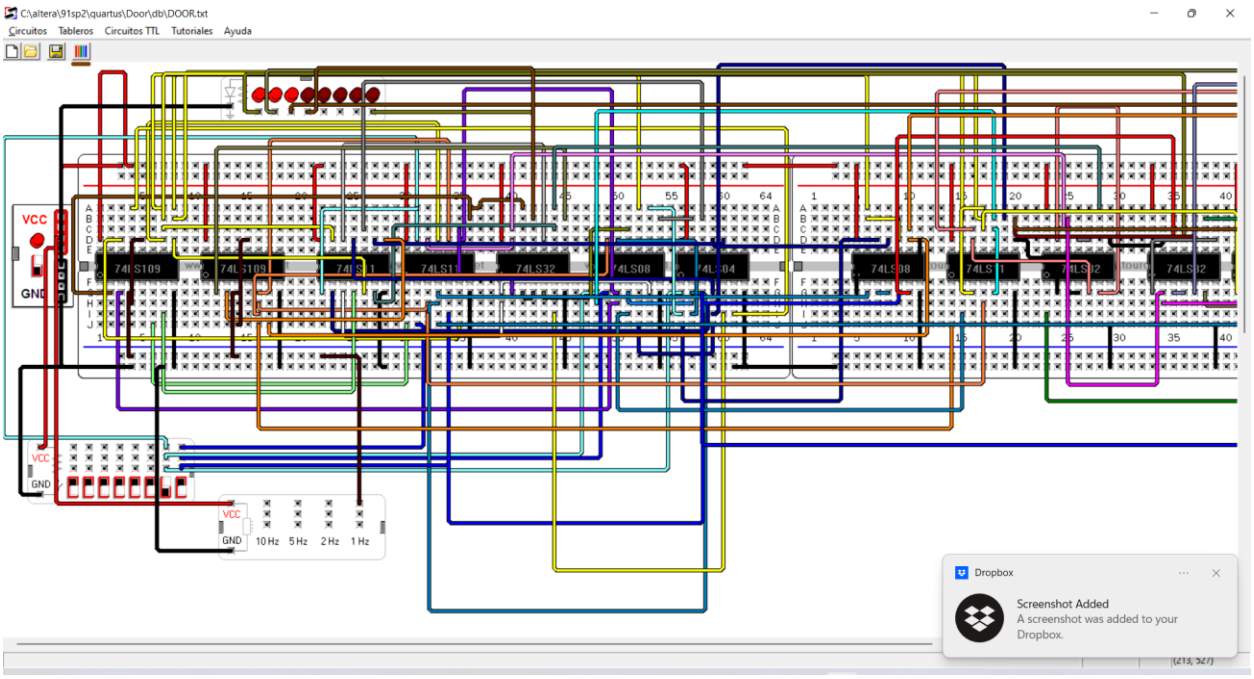


Figure 38:Breadboard representing state 011

In the above figure we are testing the the state is 011, we took as exmaple the case of the input 10 and we find out that L1, L2, and L3 are lit up which conforms with our findings in the truth table. This case is taken to test state 011. Inputing 10 at the state 010 and at 011 will be at state 011 and emitting light from the light bulb corresponding to the first step L1, second step L2, and third step L3. Also we notice that the lock light bulb which may be indicated by the light bulb(considering it the outer red light) is off so the toilet is still occupied and the door remains locked.

‘

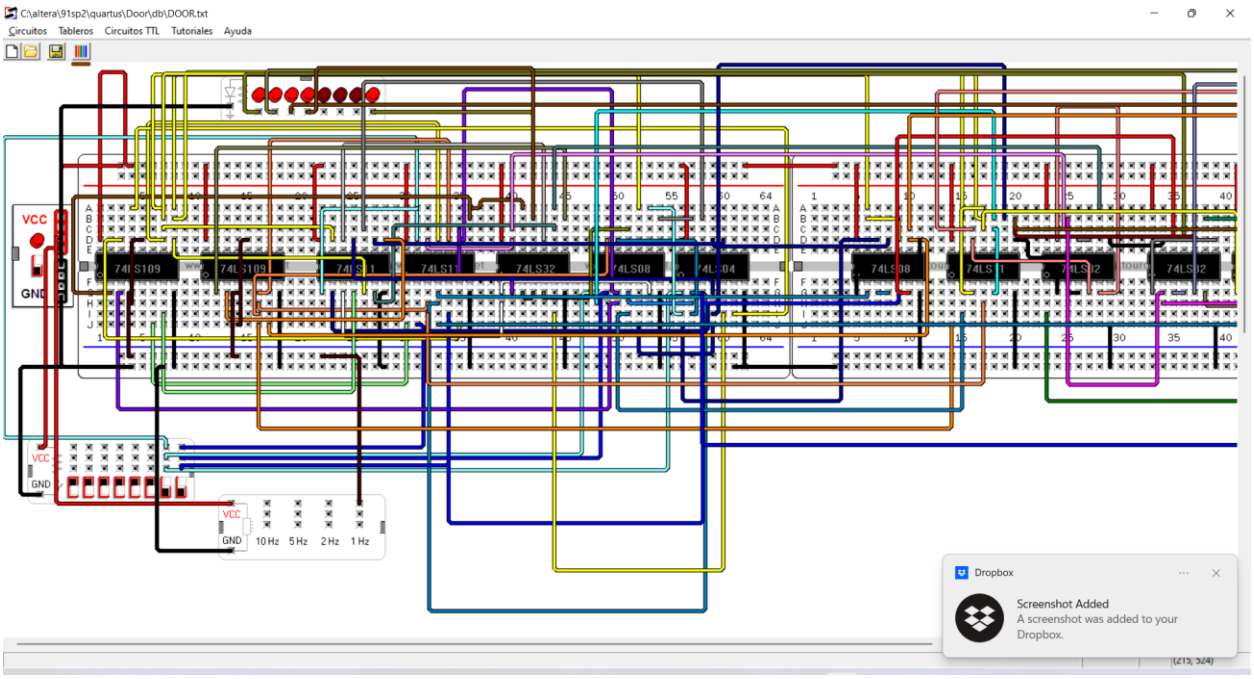


Figure 39:Breadboard representing state 100

In the above figure we are testing the state is 100, we took as example the case of the input 11 and we find out that L1, L2, L3, L4, and the red light corresponding to the door lock are lit up which conforms with our findings in the truth table. This case is taken to test state 100. Inputing 11 at the state 011 and at 100 will be at state 100 and emitting light from the light bulb corresponding to the first step L1, second step L2, third step L3, and fourth step L4. Also we notice that the lock light bulb which may be indicated by the light bulb(considering it the outer red light) is on so the toilet door is unlocked and the toilet becomes unoccupied.

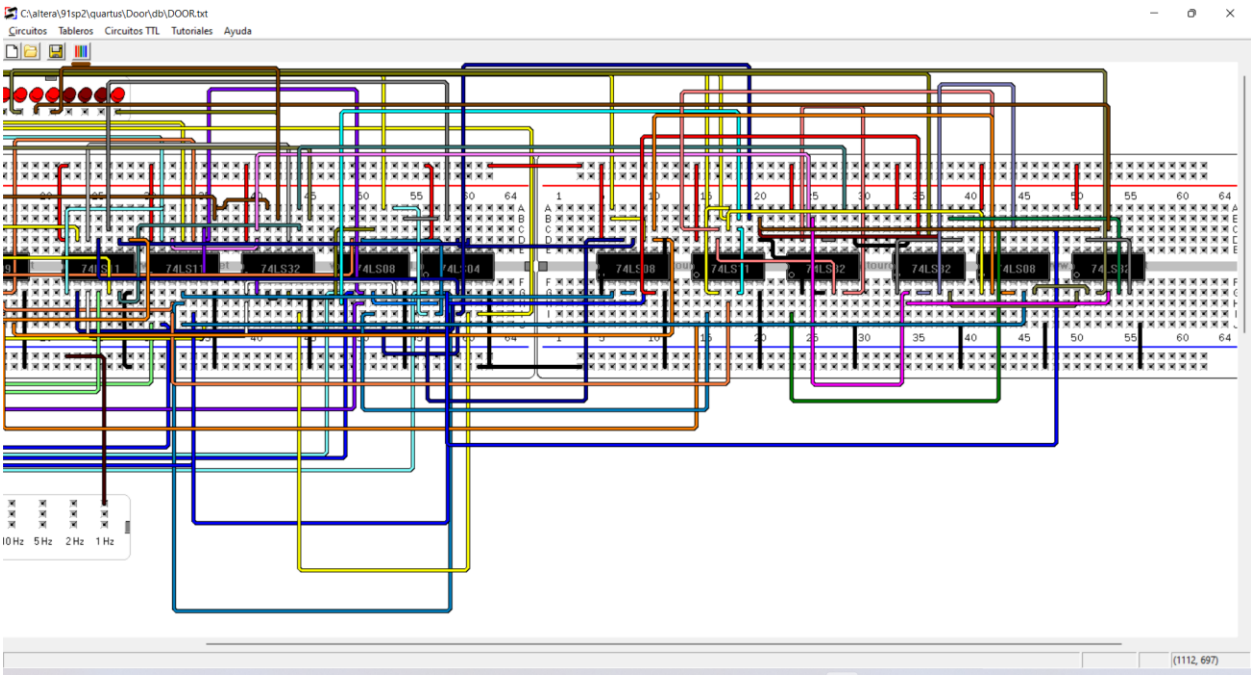


Figure 40: Breadboard representing continuation of design on sate 100

This is added to show the continuation of circuit.

Counter from 20 to 0:

Inputs:

Number of inputs in this 20 second counter is 5 inputs and the number of outputs is also 5.

State table:

Present State					Next State					Outputs									
A	B	C	D	E	A	B	C	D	E	JA	KA	JB	KB	JC	KC	JD	KD	JE	KE
0	0	0	0	0	1	0	1	0	0	1	X	0	X	1	X	0	X	0	X
0	0	0	0	1	0	0	0	0	0	0	X	0	X	0	X	0	X	X	1
0	0	0	1	0	0	0	0	0	1	0	X	0	X	0	X	X	1	1	X
0	0	0	1	1	0	0	0	1	0	0	X	0	X	0	X	X	0	X	1
0	0	1	0	0	0	0	0	1	1	0	X	0	X	X	1	1	X	1	X
0	0	1	0	1	0	0	1	0	0	0	X	0	X	X	0	0	X	X	1
0	0	1	1	0	0	0	1	0	1	0	X	0	X	X	0	X	1	1	X
0	0	1	1	1	0	0	1	1	0	0	X	0	X	X	0	X	0	X	1
0	1	0	0	0	0	0	1	1	1	0	X	X	1	1	X	1	X	1	X
0	1	0	0	1	0	1	0	0	0	0	X	X	0	0	X	0	X	X	1
0	1	0	1	0	0	1	0	0	1	0	X	X	0	0	X	X	1	1	X
0	1	0	1	1	0	1	0	1	0	0	X	X	0	0	X	X	0	X	1
0	1	1	0	0	0	1	0	1	1	0	X	X	0	X	1	1	X	1	X
0	1	1	0	1	0	1	1	0	0	0	X	X	0	X	0	0	X	X	1
0	1	1	1	0	0	1	1	0	1	0	X	X	0	X	0	X	1	1	X
0	1	1	1	1	0	1	1	1	0	0	X	X	0	X	0	X	0	X	1
1	0	0	0	0	0	1	1	1	1	X	1	1	X	1	X	1	X	1	X
1	0	0	0	1	1	0	0	0	0	X	0	0	X	0	X	0	X	X	1
1	0	0	1	0	1	0	0	0	1	X	0	0	X	0	X	X	1	1	X
1	0	0	1	1	1	0	0	1	0	X	0	0	X	0	X	X	0	X	1
1	0	1	0	0	1	0	0	1	1	X	0	0	X	X	1	1	X	1	X

Table 4: State Table for counter from 20 to 0

Deriving Functions:

By using 3D K-maps:

JA= B'C'D'E'

KA=C'D'E'

JB=A

KB=C'D'E'

JC= D'E'

KC=D'E'

JD= CE' + BE' + AE' = E'(C+B+A)

KD=E'

JE=D + C + B + A

KF=1

Quartus:

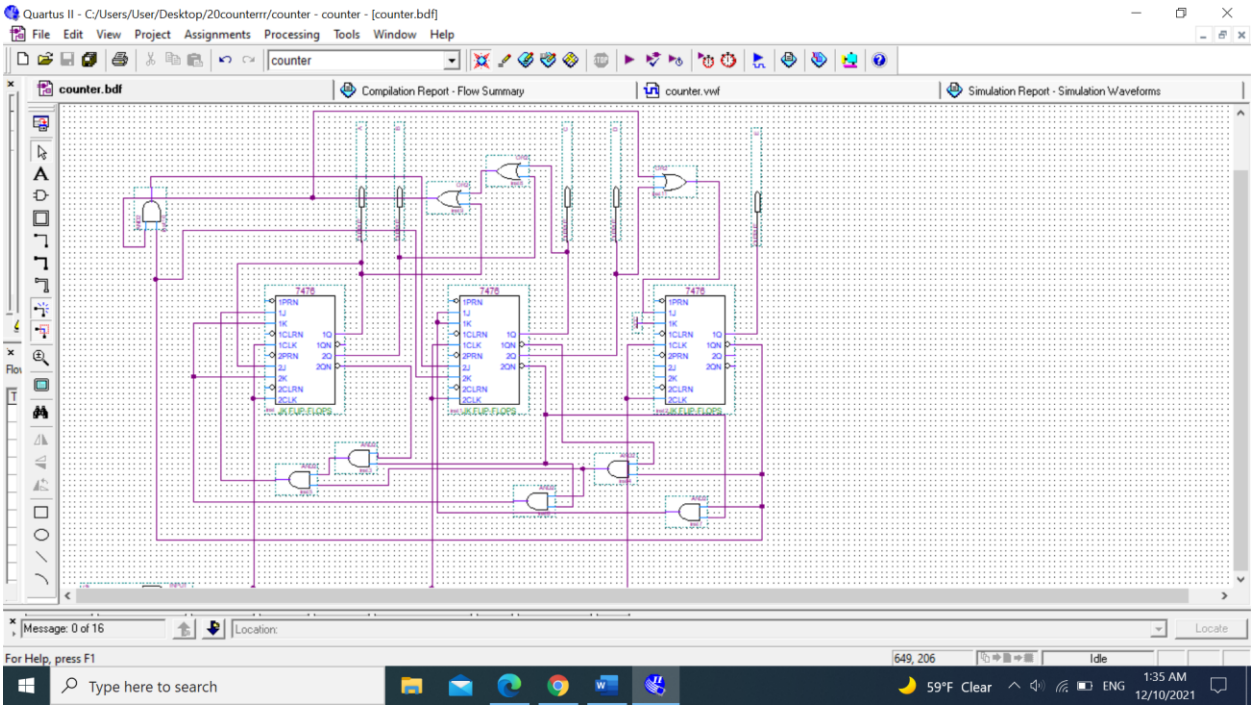


Figure 41: Circuit Design for counter from 20 to 0

How was it designed?

To implement our design, we used 3 JK flip flops (2 chips each consisting of two of them 2-input OR gates and 2-input AND gates. According to the functions above, we constructed the circuit by connecting the resulting combination of each circuit to the corresponding pins of the flip flops. Then, we connect the J and K input functions to the chip and connect the output of each JK flip flop Q considering it as the bit of each state, so every flip flop leads to one bit of the state. The clocks of all three flip flops are connected to function at the same rhythm. As well we add output pins on the outputs of the flip flops to keep track of the states when testing the circuit using the simulation tool on Quartus.

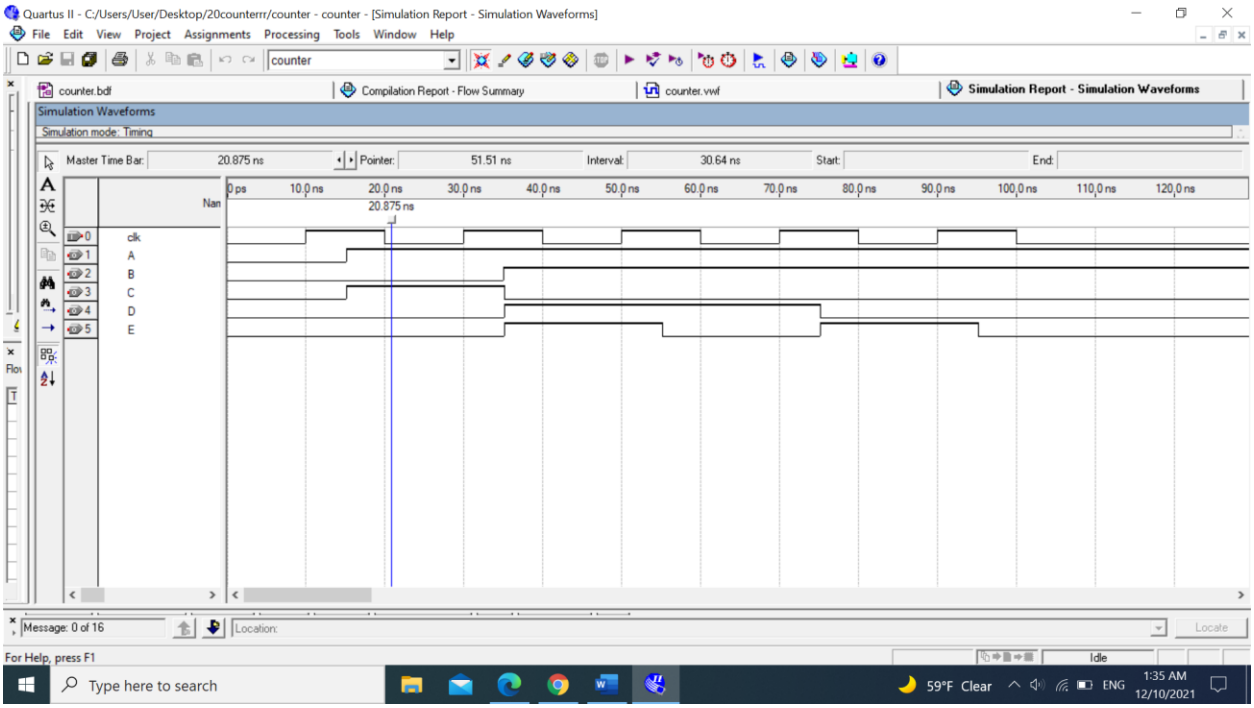


Figure 42: Simulation for counter from 20 to 0

By tracking some of the cases according to the states we notice that the results conform to the truth table presented above. This indicates that the design is logical and conforms to the requirements of the project.

Unfortunately we notice certain delays especially a 15.136ns and another slight delay at 34.987ns.

Breadboard:

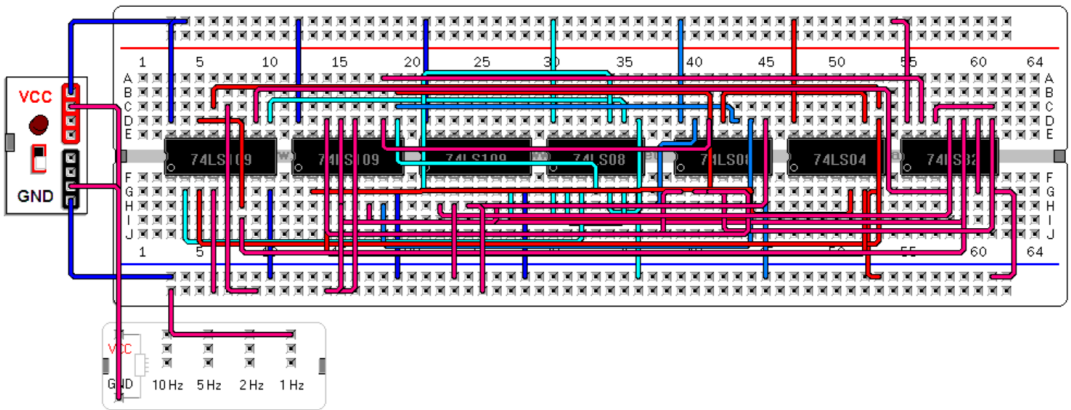


Figure 43: Circuit Design from 20 to 0

In this 20 to 0 counter, we needed 5 flip-flops jk so we used 3 74LS109 chips since each has 2 flipflips.In order to connect them we should first get the equations of j and k that are gotten above previously. First we connected the inverse output of the second flip-flop B' with inverse output of 3rd flip-flop C' to a and chip. This output was treated as an input to another and gate where D' or inverse output other fourth flip-flop is added. Similarly the output is used as an input with E' inverse output of last flip-flop and the output is connected to JA. The same procedure is done to get C'D'E' where inverted of these outputs are put as inputs to 2 and gates and finally the output is connected to KA.

Simply connect output of first flip-flop to JB and connect C'D'E' done previously to KB. Add both D' and E' to an and gate and the output should be connected to JC and KC. To an or gate get C and B as an input and add the output to another or gate with A. connect the output with E' in and gate. Connect the corresponding output to JD. Connect KD to E' and the output of the previous or gate use it as input to another or gate with input D and connect output to JE. Connect KF to 1.

It is important to note that all of the K inputs must be inverted in order to use this 74109 since no K pin is available but K' is present.

It is also very important to note that all flip flips are connected to the same clock and all of the chips have their pin 7 connected to ground and pin 16 connected to Vcc.

Counter from 10 to 0:

In this part we designed a counter that counts from 10 to 0. We used here 4 JK-flipflops, i.e 2 chips of type 74LS109. So, the logic we used here is that we started our current state from 0000 (first state), and our next state was 1001, then on the next state it goes to 1000, then to 0111…. Until reaching 0000. Therefore, it will go from 9 to 0 as a countdown.

States:

4 bits are used to represent each state, states are taken 0000, 1001, 0110, 0111, 0101, 0100, 0011, 0010, 0001.

State Table:

Current State				Next State				Flip Flops							
A	B	C	D	A+	B+	C+	D+	JA	KA	JB	KB	JC	KC	JD	KD
0	0	0	0	1	0	0	1	1	×	0	×	0	×	1	×
1	0	0	1	1	0	0	0	×	0	0	×	0	×	×	1
1	0	0	0	0	1	1	1	×	1	1	×	1	×	1	×
0	1	1	1	0	1	1	0	0	×	×	0	×	0	×	1
0	1	1	0	0	1	0	1	0	×	×	0	×	1	1	×
0	1	0	1	0	1	0	0	0	×	×	0	0	×	×	1
0	1	0	0	0	0	1	1	0	×	×	1	1	×	1	×
0	0	1	1	0	0	1	0	0	×	0	×	×	0	×	1
0	0	1	0	0	0	0	1	0	×	0	×	×	1	1	×
0	0	0	1	0	0	0	0	0	×	0	×	0	×	×	1
All remaining permutations are associated as dont cares.															

Deriving Functions:

Using K-Maps:

JA=B’C’D’

KA=D’

JB=AD’

KB=C’D’

JC=BD’+AD’

KC=D’

JD=KD=1

Breadboard:

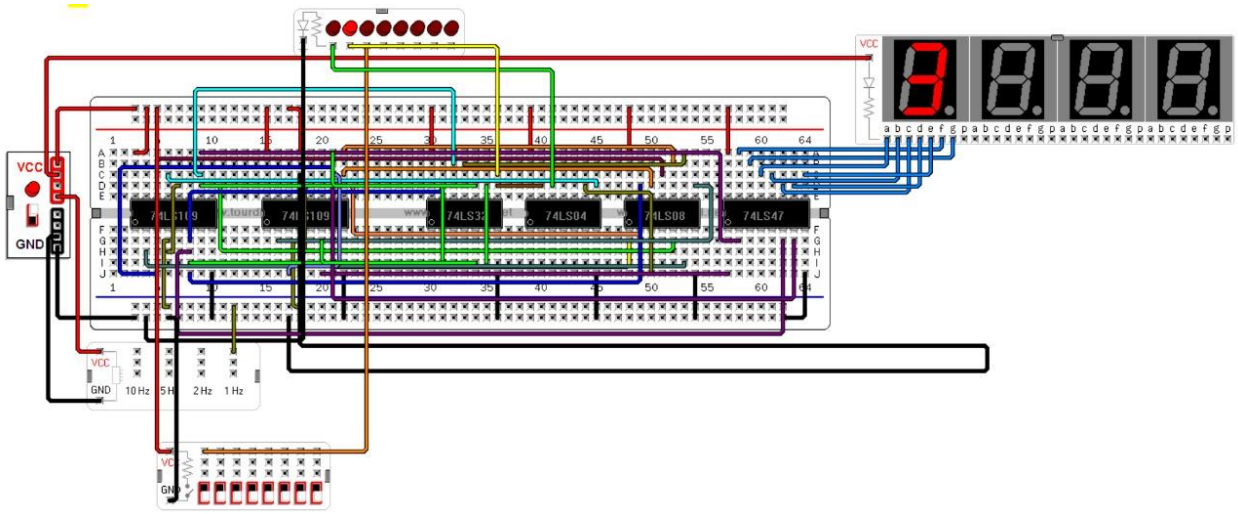


Figure 44: Circuit Design representing a case of countdown from 9 to 0

So in this part, we used 4 Jk Flipflops, 1 OR GATE , 1 NOT gate, 1 and Gate and a BCD to segment display. We used the previous logic and the following functions derived to implement it on breadboard...

The display will show the countdown starting from 9 to 0, and repeating all over again...

Financial Study:

In this project we made sure to make as low cost as possible. We went to buy our chips from the most affordable shop in Tripoli and all of the chips and equipment needed costed (including the extra material we used) 40$.We also made sure to minimize our equations as much as possible to reduce gates and chip usage leading to a lower cost. Common equations were built once and used as many as needed too were we made sure not to build the same equation twice unless needed for a less complicated circuit. All of these efforts were effective in making our circuit affordable by avoiding building common equations, going to the most affordable shop, reducing chip uses as soon as possible, and minimized equations.

Delay Calculation:

In this project, upon using parallel loads, only the equation of highest delay is counted while in series loads all of the delays of the gates are added with no exception. Accordingly our expected delay of the whole circuit is expected to be as follows:

We noticed some delays in the above implementation on breadboard. We've encountered 68 delays in total…

Power Consumption:

Regarding the power consumption of our lab project, only 1 clock was used for the entire circuit and only 1 voltage source was used of 5V which shows how our project does need a lot of power to function where only 1 clock and voltage source were used for the entire design.

Problems Faced:

When executing this lab project, most of the common problem we faced was in building the circuit on breadboard and soldering. Soldering wasn't as simple as expected and building and connected chips to each other needed a lot of focus and time. Most of the theoretical work on Quartus and the virtual Breadboard was direct and doable which ensures that the problems we faced during building the circuit were only due to the huge amount of focus and time needed to complete it successfully with infinite number of tries.

Advantages:

What makes our circuit design special is that straightforward and direct techniques were used where most of our work was a result of a state table and deriving equations out of them. Our lab project is easy to understand and straight to the main goal which is to build a smart toilet for the Jbeil municipality. Our project was built with low cost costing only 40$ including the extra material which means our project even cost less than 40$.All of these elements ensure that our project has a lot of advantages compared to the others.

Conclusion:

By using and applying all the techniques and skills we acquired during this semester which include state machines, sequential circuits, flip flops etc.., we were able to successfully design the circuit controller of the smart toilet that Jbeil municipality is deciding to build. By using Quartus, breadboard, state tables and state tables different circuit controllers are built and complemented together to design such a toilet that is suitable for normal and handicap people that has a cleaning mode after 1 use and a deep cleaning mode after 10 uses and requires the user to flush the toilet and use the soap, sink, paper towels in order in order to get out and get the door unlocked were LED lights were use in many cases to make it simpler for the user to visualize what's needed next and makes it more clear for people about how this smart toilet in Jbeil works.