ENGPHYS 2E04

Design project

April 23rd, 2021

Professor: Dr. Matthew Minnick

TA: Curtis Goosney

Name: Roxxannia Wang

MacID: Wangs302

Student Number: 400237438

Table of Contents

Introduction

In the final design project of 2E04 course, a 7 Segment Display will be used to display from BCD using a decoder to generate the decimal output. The decimal output will the student number “400237438”. With the previous knowledge from Topic 6 digital logic analysis and Topic 7 sequential logic analysis, this project will employ the knowledge from these topics to produce an excitation table from state transition tables, using Karnaugh mapping to design a fata and flipflop implementation that produces a desired timing diagram for the sequence of numbers that are going to be outputted on the display. This project will also be divided into 3 main segments which the circuit design will be analytically calculated, run through Multisim Simulation and physically constructed to check the validity and the reliability of the circuit design.

Analytical Logic

Step 1: Number Conversion

The first step is to convert the Decimal System number to Binary System in order for the 7-Segment display to recognize the inputs. In my case, my student number is 400237438. The following table shows the conversion of the student number to 4-digit binary numbers. From left to right, each digit of binary number is assigned to a Q bit. The last bit, q4, is not part of the binary system, however, a memory bit to record the repeated number assignments. For the repeat numbers that appear for the first time, the memory bit is going to be 1 and when it appears the second time, it is going to be 0. (For the numbers do not repeat, they are going to be assigned to 1 as well instead of being an "X" to simplify the circuit.

Decimal Number	**4-Digit Binary Number**				**Memory**
	Q3	**Q2**	**Q1**	**Q0**	**Q4**
4	0	1	0	0	1
0	0	0	0	0	1
0	0	0	0	0	0
2	0	0	1	0	1
3	0	0	1	1	1
7	0	1	1	1	1
4	0	1	0	0	0
3	0	0	1	1	0
8	1	0	0	0	1

Table 1: Decimal Number to Binary Number Conversion

Noticed that in the circuit, "4", "0" and "3" are the ones that are repeating.

Step 2: JK Excitation Table

Since sequential logic requires JK flip-flops, the JK Excitation Table is used to determine the inputs for J and K by looking at the digit change from current state to the next state. All four possible combinations are listed below:

Q_n	Q_{n+1}	**J**	**K**
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Table 2: JK Excitation Table

Step 3: Determining JK inputs

The next state of the student number is just the next combination of all the digit in order, with the first digit in the current state goes to the last place in the sequence. With the JK excitation tables, the inputs for J and K can be determined as following.

Current State						Next State						JK Inputs									
#	**Q3**	**Q2**	**Q1**	**Q0**	**Q4**	**#**	**Q3**	**Q2**	**Q1**	**Q0**	**Q4**	**J3**	**K3**	**J2**	**K2**	**J1**	**K1**	**J0**	**K0**	**J4**	**K4**
4	0	1	0	0	1	0	0	0	0	0	1	0	X	X	1	0	X	0	X	X	0
0	0	0	0	0	1	0	0	0	0	0	0	0	X	0	X	0	X	0	X	X	1
0	0	0	0	0	0	2	0	0	1	0	1	0	X	0	X	1	X	0	X	1	X
2	0	0	1	0	1	3	0	0	1	1	1	0	X	0	X	X	0	1	X	X	0
3	0	0	1	1	1	7	0	1	1	1	1	0	X	1	X	X	0	X	0	X	0
7	0	1	1	1	1	4	0	1	0	0	1	0	X	X	0	X	1	X	1	X	1
4	0	1	0	0	0	3	0	0	1	1	0	0	X	X	1	1	X	1	X	0	X
3	0	0	1	1	0	8	1	0	0	0	1	1	X	0	X	X	1	X	1	1	X
8	1	0	0	0	1	4	0	1	0	0	0	X	1	1	X	0	X	0	X	X	0

Table 3: Transition Table

Step 4: K-mapping

To determine the Boolean expression of each J and K inputs, K-mapping is used. In the following K-mapping systems, SOP expansion will be used for the simplicity as there are less scattered “1”s than “0”s. A size of 2^n rectangle will be made to cover every 1 and the goal is to use as few and as big of rectangles as possible.

J0 Boolean Expression: $J0 = \overline{Q4}Q2 + Q4Q1$

	Q4 = 0						**Q4 = 1**			
$^{Q3Q2}/_{Q1Q0}$	**00**	**01**	**11**	**10**		$^{Q3Q2}/_{Q1Q0}$	**00**	**01**	**11**	**10**
00	0	X	X	X		**00**	0	X	X	1
01	1	X	X	X		**01**	0	X	X	X
11	X	X	X	X		**11**	X	X	X	X
10	X	X	X	X		**10**	0	X	X	X

K0 Boolean Expression: $\overline{Q4} + Q4Q2$

	Q4 = 0						**Q4 = 1**			
$^{Q3Q2}/_{Q1Q0}$	**00**	**01**	**11**	**10**		$^{Q3Q2}/_{Q1Q0}$	**00**	**01**	**11**	**10**
00	X	X	1	X		**00**	X	X	0	X
01	X	X	X	X		**01**	X	X	1	X
11	X	X	X	X		**11**	X	X	X	X
10	X	X	X	X		**10**	X	X	X	X

J1 Boolean Expression:$\overline{Q4}$

	Q4 = 0						**Q4 = 1**			
$^{Q3Q2}/_{Q1Q0}$	**00**	**01**	**11**	**10**		$^{Q3Q2}/_{Q1Q0}$	**00**	**01**	**11**	**10**
00	1	X	X	X		**00**	0	X	X	X
01	1	X	X	X		**01**	0	X	X	X
11	X	X	X	X		**11**	X	X	X	X
10	X	X	X	X		**10**	0	X	X	X

K1 Boolean Expression: $\overline{Q4} + Q4Q2$

	Q4 = 0						**Q4 = 1**			
$^{Q3Q2}/_{Q1Q0}$	**00**	**01**	**11**	**10**		$^{Q3Q2}/_{Q1Q0}$	**00**	**01**	**11**	**10**
00	X	X	1	X		**00**	X	X	0	0
01	X	X	X	X		**01**	X	X	1	X
11	X	X	X	X		**11**	X	X	X	X
10	X	X	X	X		**10**	X	X	X	X

J2 Boolean Expression: $Q4Q3 + Q4Q0 = Q4(Q0 + Q3)$

	Q4 = 0						**Q4 = 1**			
$^{Q3Q2}/_{Q1Q0}$	**00**	**01**	**11**	**10**		$^{Q3Q2}/_{Q1Q0}$	**00**	**01**	**11**	**10**
00	0	X	0	X		**00**	0	X	1	0
01	X	X	X	X		**01**	X	X	X	X
11	X	X	X	X		**11**	X	X	X	X
10	X	X	X	X		**10**	1	X	X	X

K2 Boolean Expression:$\overline{Q4} + Q4\overline{Q1}$

	Q4 = 0						**Q4 = 1**			
$Q3Q2/Q1Q0$	**00**	**01**	**11**	**10**		$Q3Q2/Q1Q0$	**00**	**01**	**11**	**10**
00	X	X	X	X		**00**	X	X	X	X
01	1	X	X	X		**01**	1	X	0	X
11	X	X	X	X		**11**	X	X	X	X
10	X	X	X	X		**10**	X	X	X	X

J3 Boolean Expression: $\overline{Q4}Q1$

	Q4 = 0						**Q4 = 1**			
$Q3Q2/Q1Q0$	**00**	**01**	**11**	**10**		$Q3Q2/Q1Q0$	**00**	**01**	**11**	**10**
00	0	X	1	X		**00**	0	X	0	0
01	0	X	X	X		**01**	0	X	0	X
11	X	X	X	X		**11**	X	X	X	X
10	X	X	X	X		**10**	X	X	X	X

K3 Boolean Expression: $K3 = 1$

	Q4 = 0						**Q4 = 1**			
$Q3Q2/Q1Q0$	**00**	**01**	**11**	**10**		$Q3Q2/Q1Q0$	**00**	**01**	**11**	**10**
00	X	X	X	X		**00**	X	X	X	X
01	X	X	X	X		**01**	X	X	X	X
11	X	X	X	X		**11**	X	X	X	X
10	X	X	X	X		**10**	1	X	X	X

J4 Boolean Expression: $\overline{Q4}\,\overline{Q3}\,\overline{Q2}$

	Q4 = 0						**Q4 = 1**			
$Q3Q2/Q1Q0$	**00**	**01**	**11**	**10**		$Q3Q2/Q1Q0$	**00**	**01**	**11**	**10**
00	1	X	1	X		**00**	X	X	X	X
01	0	X	X	X		**01**	X	X	X	X
11	X	X	X	X		**11**	X	X	X	X
10	X	X	X	X		**10**	X	X	X	X

K4 Boolean Expression: $Q4(\overline{Q3}\,\overline{Q2}\,\overline{Q1} + Q3)$

	Q4 = 0						**Q4 = 1**			
$\boldsymbol{Q3Q2}/\boldsymbol{Q1Q0}$	**00**	**01**	**11**	**10**		$\boldsymbol{Q3Q2}/\boldsymbol{Q1Q0}$	**00**	**01**	**11**	**10**
00	X	X	X	X		**00**	1	X	0	0
01	X	X	X	X		**01**	0	X	1	X
11	X	X	X	X		**11**	X	X	X	X
10	X	X	X	X		**10**	0	X	X	X

To summarize the Boolean expressions above:

J0	$\overline{Q4}Q2 + Q4Q1$	K0	$\overline{Q4} + Q4Q2$
J1	$\overline{Q4}$	K1	$\overline{Q4} + Q4Q2$
J2	$Q4(Q0 + Q3)$	K2	$\overline{Q4} + Q4\overline{Q1}$
J3	$\overline{Q4}Q1$	K3	1
J4	$\overline{Q4}\,\overline{Q3}\,\overline{Q2}$	K4	$Q4(\overline{Q3}\,\overline{Q2}\,\overline{Q1} + Q3)$ $Q4(\overline{Q3}\,\overline{Q2}\,\overline{Q1} + \overline{Q3}\,Q2Q1)$

Table 4: Boolean Expression Summarization

Since the JK – flipflop takes J and $\overline{K}$ as inputs, the following table converts the table above to the final JK inputs.

$\overline{K0}$	$\overline{\overline{Q4} + Q4Q2} = Q4\,\overline{Q4Q2} = Q4(\overline{Q4} + \overline{Q2}) = Q4\,\overline{Q4} + Q4\,\overline{Q2} = \mathbf{0} + \boldsymbol{Q4\,\overline{Q2}}$ $= \boldsymbol{Q4\,\overline{Q2}}$
$\overline{K1}$	$\overline{\overline{Q4} + Q4Q2}$ (same as above)
$\overline{K2}$	$\overline{\overline{Q4} + Q4\overline{Q1}} = Q4\overline{Q4\,\overline{Q1}} = Q4(\overline{Q4} + Q1) = Q4\,\overline{Q4} + Q4\,Q1 = \mathbf{0} + \boldsymbol{Q4\,Q1} = \boldsymbol{Q4\,Q1}$
$\overline{K3}$	$\mathbf{0}$
$\overline{K4}$	$\overline{Q4(\overline{Q3}\,\overline{Q2}\,\overline{Q1} + \overline{Q3}\,Q2Q1)} = \overline{Q4} + \overline{\overline{Q3}\,\overline{Q2}\,\overline{Q1} + \overline{Q3}\,Q2Q1}$ $= \overline{Q4} + \overline{\overline{Q3}\,\overline{Q2}\,\overline{Q1}}\;\overline{\overline{Q3}\,Q2Q1} = \overline{Q4} + (Q3 + Q2 + Q1)(Q3 + \overline{Q2} + \overline{Q1})$ $= \boldsymbol{Q3 + Q2\,\overline{Q1} + \overline{Q2}Q1}$

Table 5: Boolean Expression of K Conversion

$J0$	$\overline{Q4}Q2 + Q4Q1$	$\overline{K0}$	$Q4\,\overline{Q2}$
$J1$	$\overline{Q4}$	$\overline{K1}$	$Q4\,\overline{Q2}$
$J2$	$Q4(Q0 + Q3)$	$\overline{K2}$	$Q4\,Q1$
$J3$	$\overline{Q4}Q1$	$\overline{K3}$	0
$J4$	$\overline{Q4}\,\overline{Q3}\,\overline{Q2}$	$\overline{K4}$	$Q3 + Q2\,\overline{Q1} + \overline{Q2}Q1$

Table 6: Boolean Expression Summarization with K Conversion

Step 5: Circuit Design

$J0$	$\overline{Q4}Q2 + Q4Q1$	2 AND, 1 OR
$J1$	$\overline{Q4}$	/
$J2$	$Q4(Q0 + Q3)$	1 AND, 1 OR
$J3$	$\overline{Q4}Q1$	1 AND
$J4$	$\overline{Q4}\,\overline{Q3}\,\overline{Q2}$	2 AND
$\overline{K0}$	$Q4\,\overline{Q2}$	1 AND
$\overline{K1}$	$Q4\,\overline{Q2}$	(Same as above, no need for extra gates)
$\overline{K2}$	$Q4\,Q1$	1 AND
$\overline{K3}$	0	/
$\overline{K4}$	$Q3 + Q2\,\overline{Q1} + \overline{Q2}Q1$	2 AND, 2 OR

Table 7: Circuit Design (Original)

Therefore, the circuit requires 5 JK Flipflops, 10 AND Gates, and 4 OR Gates

This requires 3 74HC109N, 3 74HC08 and 1 74HC32.

Step 5.1: Alternative Designs

There are also other ways to write the Boolean Expressions above

Using NAND Gates only:

$J0$	$\overline{\overline{\overline{Q4}Q2}\,\overline{Q4Q1}}$	3 NAND
$J1$	$\overline{Q4}$	/
$J2$	$\overline{\overline{Q4Q0}\,\overline{Q4Q3}}$	3 NAND
$J3$	$\overline{\overline{\overline{Q4}Q1}}$	2 NAND
$J4$	$\overline{\overline{\overline{Q4}\,\overline{Q3}\,\overline{Q2}}}$	3 NAND
$\overline{K0}$	$\overline{\overline{Q4\,\overline{Q2}}}$	2 NAND
$\overline{K1}$	$(Q4\,\overline{Q2})$	(Same as above, no need for extra gates)
$\overline{K2}$	$\overline{\overline{Q4\,Q1}}$	2 NAND
$\overline{K3}$	0	/
$\overline{K4}$	$\overline{\overline{\overline{\overline{Q3}}\,\overline{Q2\,\overline{Q1}}}\,\overline{\overline{Q2}Q1}}$	5 NAND

Table 8: Circuit Design (NAND)

Therefore, 20 NAND Gates is required to complete this circuit, meaning 3 74HC109N, 5 74HC00 are required for this design.

Using NOR Gates and other simplifications:

$J0$	$\overline{Q4}Q2 + Q4Q1$	2 AND, 1 OR
$J1$	$\overline{Q4}$	/
$J2$	$Q4(Q0 + Q3)$	1 AND, 1 OR
$J3$	$\overline{Q4 + \overline{Q1}}$	1 NOR
$J4$	$\overline{Q4 + Q3 + Q2}$	2 NOR
$\overline{K0}$	$\overline{\overline{Q4} + \ Q2}$	1 NOR
$\overline{K1}$	$\overline{\overline{Q4} + \ Q2}$	(Same as above, no need for extra gates)
$\overline{K2}$	$\overline{\overline{Q4} + \overline{Q1}}$	1 NOR
$\overline{K3}$	0	/
$\overline{K4}$	$Q3 + Q2\,\overline{Q1} + \overline{Q2}Q1$	2 AND, 2 OR

Table 9: Circuit Design With NOR and MIXED

The ultimate goal to construction this circuit is to have the least number of gates and the least number of ICs used. The following table shows the final construction of simplified design.

$J0$	$\overline{Q4}Q2 + Q4Q1$	2 AND, 1 OR
$J1$	$\overline{Q4}$	/
$J2$	$Q4(Q0 + Q3)$	1 AND, 1 OR
$J3$	$\overline{Q4}Q1$	1 AND
$J4$	$\overline{Q4}\,\overline{Q3}\,\overline{Q2}$	2 AND
$\overline{K0}$	$\overline{\overline{Q4} + Q2}$	1 NOR
$\overline{K1}$	$\overline{\overline{Q4} + \ Q2}$	(Same as above, no need for extra gates)
$\overline{K2}$	$\overline{\overline{Q4} + \overline{Q1}}$	1 NOR
$\overline{K3}$	0	/
$\overline{K4}$	$Q3 + Q2\,\overline{Q1} + \overline{Q2}Q1$	2 AND, 2 OR

Table 10: Final Circuit Design

Therefore, the circuit requires 5 JK Flipflops, 7 AND, 4 ORs, 2 NOR Gates

Physical components:

74HC109N (JK)	3
74HC08 (AND)	2
74HC32 (OR)	1
74HC02 (NOR)	1

Table 11: Physical Built Implementation

Multisim Simulation

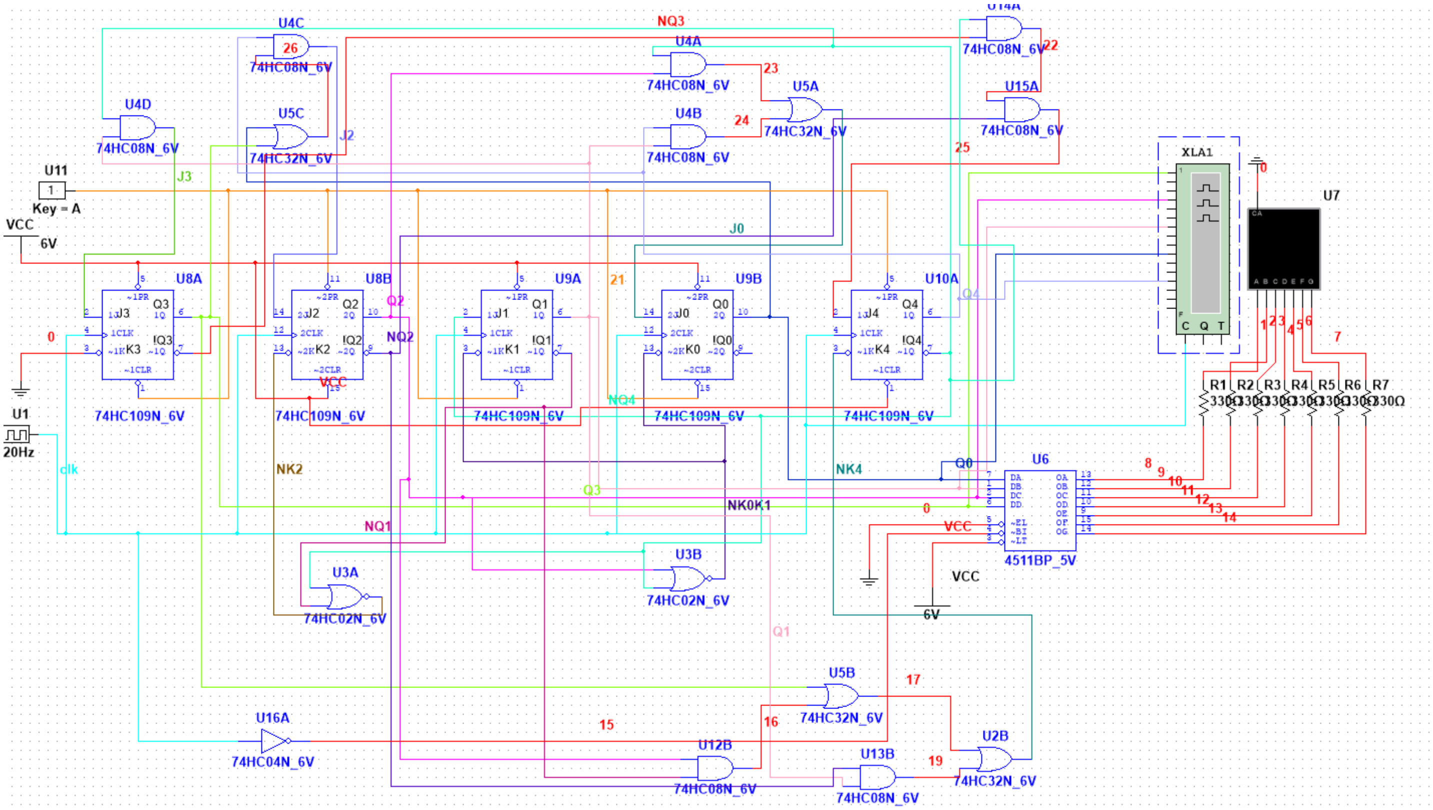
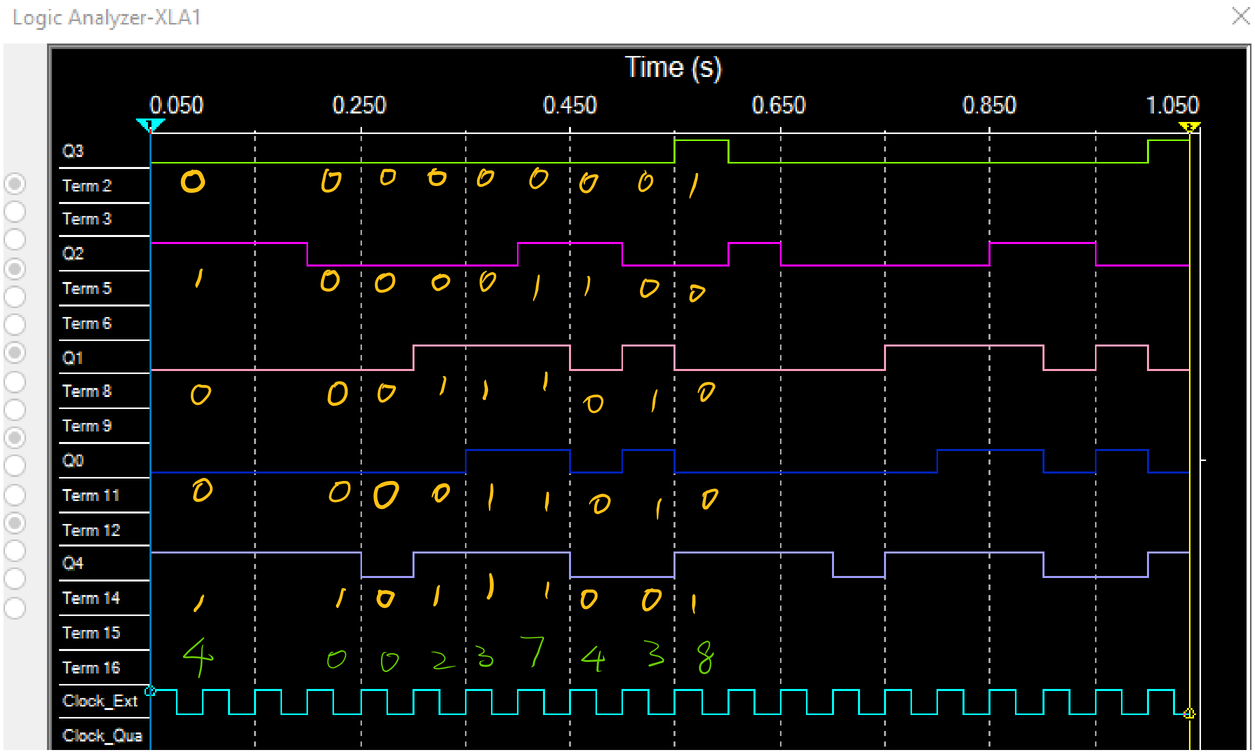


Figure 1: Multisim Setup

<u>Step 1: Multisim Design</u>

With the final circuit design that was written in Table _ in the Analytical logic section, the setup in Multisim follows the exact same setup. As shown in figure 5, 5 Flipflops are used, and AND gates, OR gates, NOR gates placing right above or below their corresponding inputs. All the wires are color-coded and labelled with a net name to help distinguish the wires among the complex networks. A BCD decoder is used to convert the inputs from Q3, Q2, Q1 and Q0 to a signal that the 7-segment display can display the corresponding numbers. Each of the inputs to the display is connected to a 330Ω resistor to ensure that the current is matching the threshold of the display. Beside the display, there is a logic analyzer added to the design in order to generate the clock diagram, to verify the actual output to the transition table that were created in Analytical portion. An interactive digital constant is added to have the circuit start with the initial state of 0100 → "4"; it is connected to the not preset pin on JK2. The not preset pin of JK4, which is the memory bit, is also hooked to the digital constant as its initial state is forced to be set at "1". The rest of JKs are set to "0" by connecting the not clear pin to the interactive digital constant. The reset of the not preset and not clear pins are connected to VCC as they do not provide any inputs, always false. An additional NOT gate is added to invert the clock signal for the decoder.

Step 2: Logic Analyzer



As shown in the logical analyzer, the clock diagram gives the sequence of:

01001 → 00001 → 00000 → 00011 → 00111 → 01111 → 01000 → 00110 → 10001

The first 4 digital corresponds to the outputs of Q3, Q2, Q1 and Q0. The last digit does not count in the display, but is also outputted to confirm the analytical result of the memory pin.

According to the clock diagram, the student number is correctly output: 4 0 0 2 3 7 4 3 8.

After 1 round of display is shown through, the sequence restarts from “4” and continues to show the similar

Multisim Demonstration: https://youtu.be/IOiGTvbHWl4

Step 3: Results

The video shows the 7-segment display. The display shows the correct sequence of the student number which matches with the clock diagram. Other than the displaying numbers, there are some flickering between the correct digit displays, however, this could just be due to the states switching as these flickering was not shown on the clock time diagram. Therefore, the flickering can be ignored, and the student number is correctly displayed in Multisim.

The Multisim Results can be summarized into the following table:

Multisim Output					
#	**Q3**	**Q2**	**Q1**	**Q0**	**Q4**
4	0	1	0	0	1
0	0	0	0	0	1
0	0	0	0	0	0
2	0	0	1	0	1
3	0	0	1	1	1
7	0	1	1	1	1
4	0	1	0	0	0
3	0	0	1	1	0
8	1	0	0	0	1

Table 12: Multisim Results

Physical Build

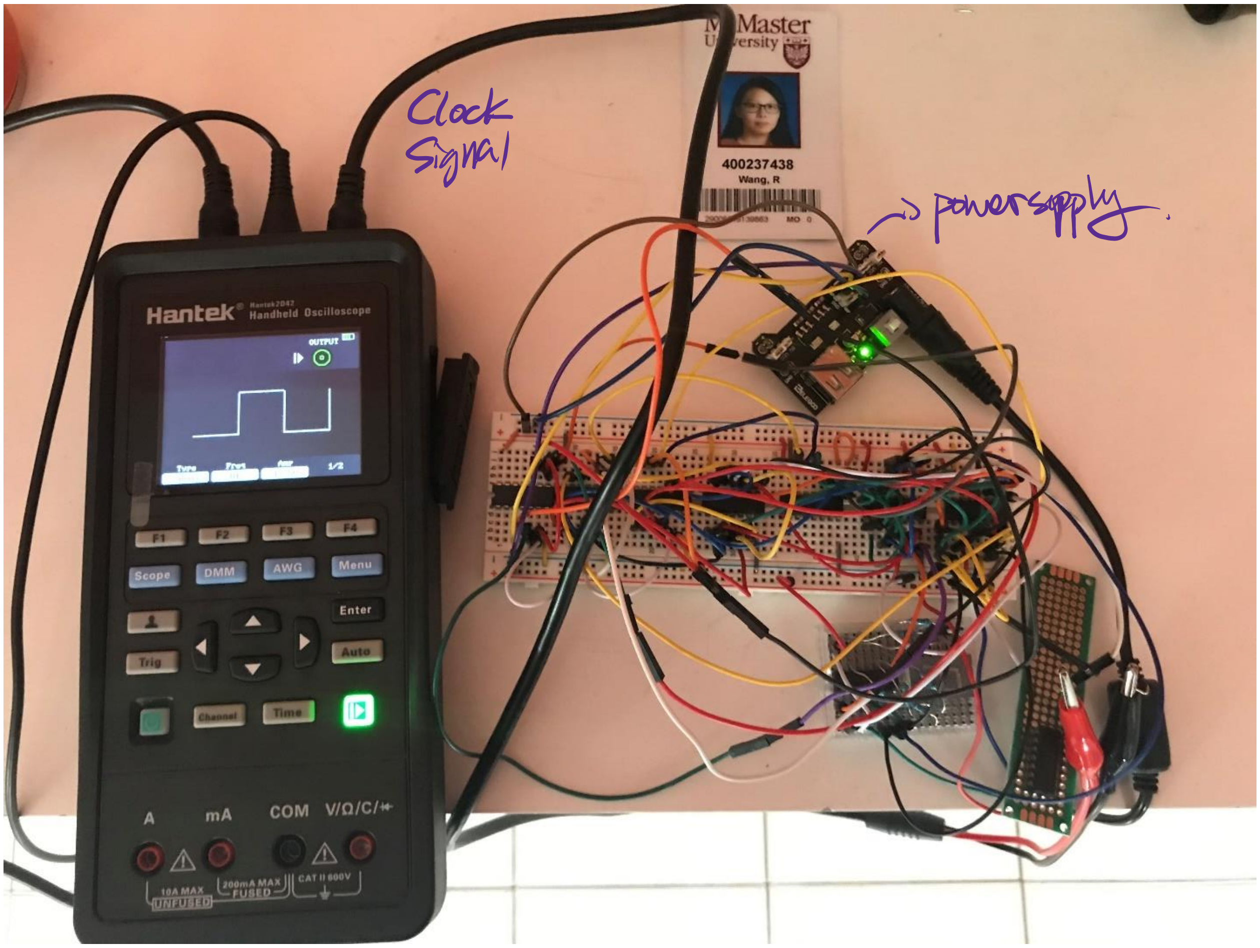


Figure 2: Physical Built Overview

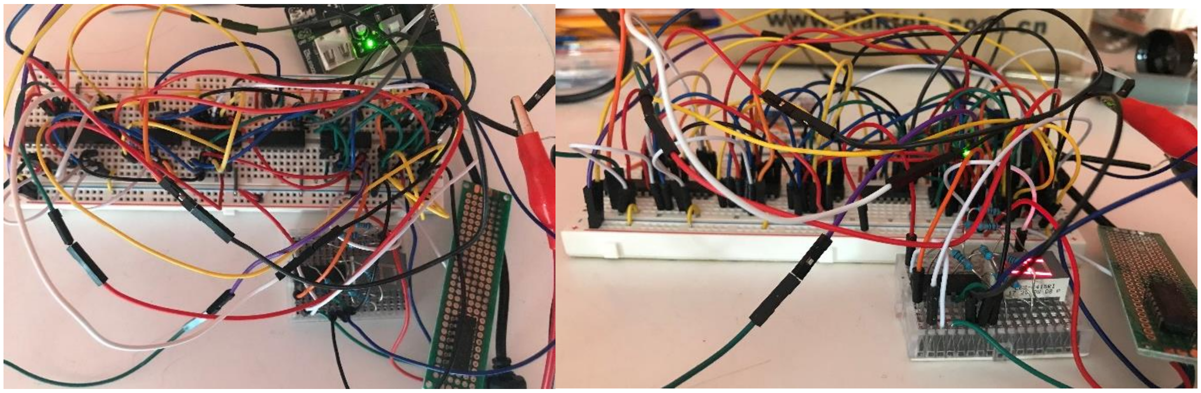


Figure 3: Physical Build Overview Close Up

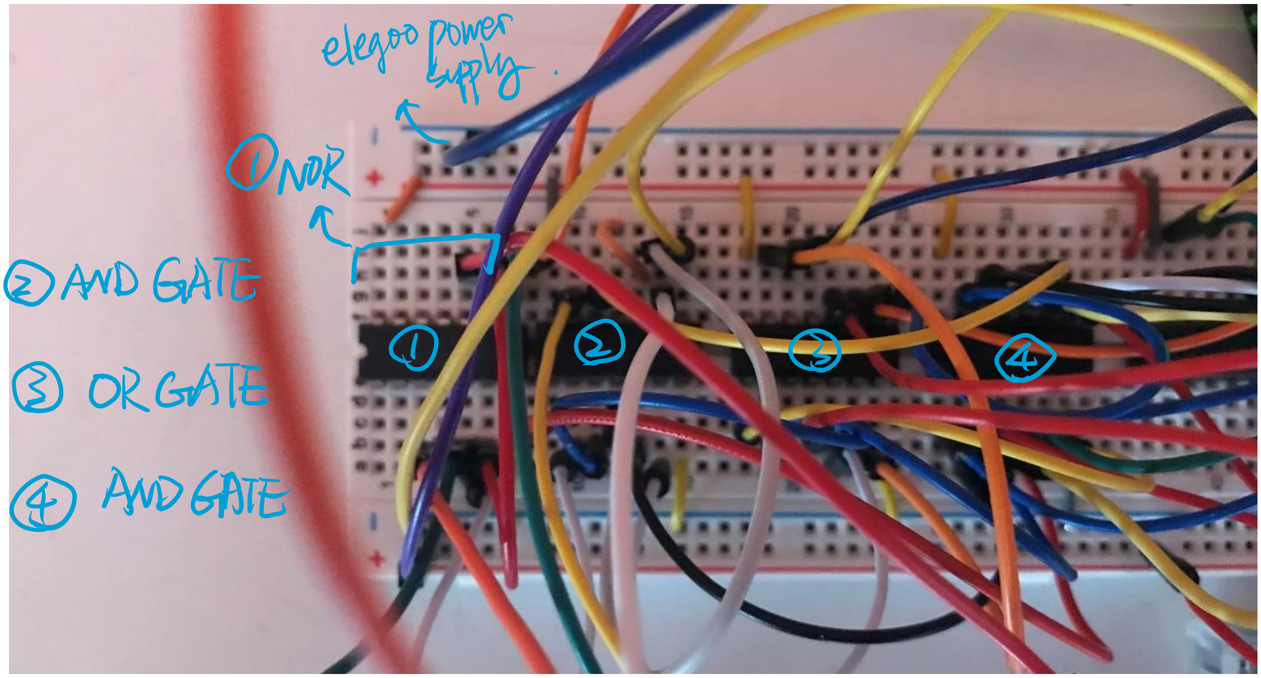


Figure 4: Labelled AND OR NOR Gates

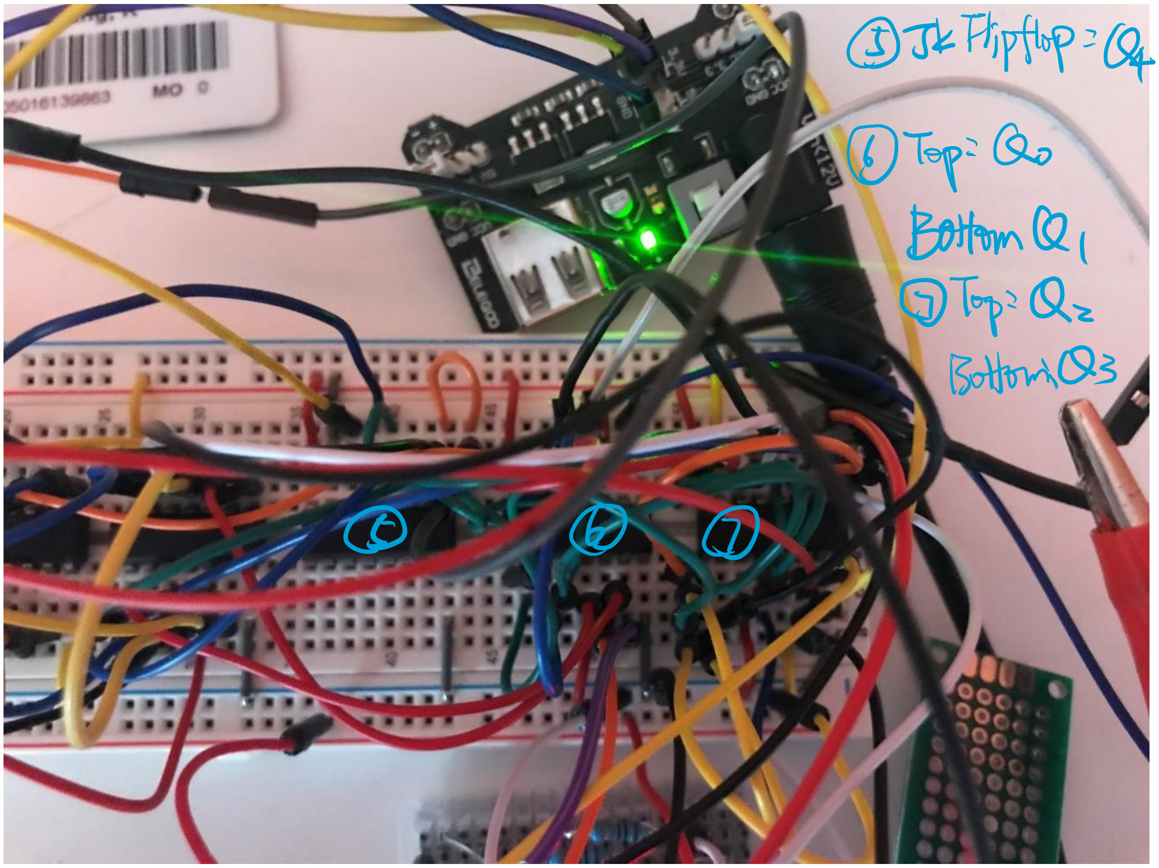


Figure 5: Labelled JK FlipFlops

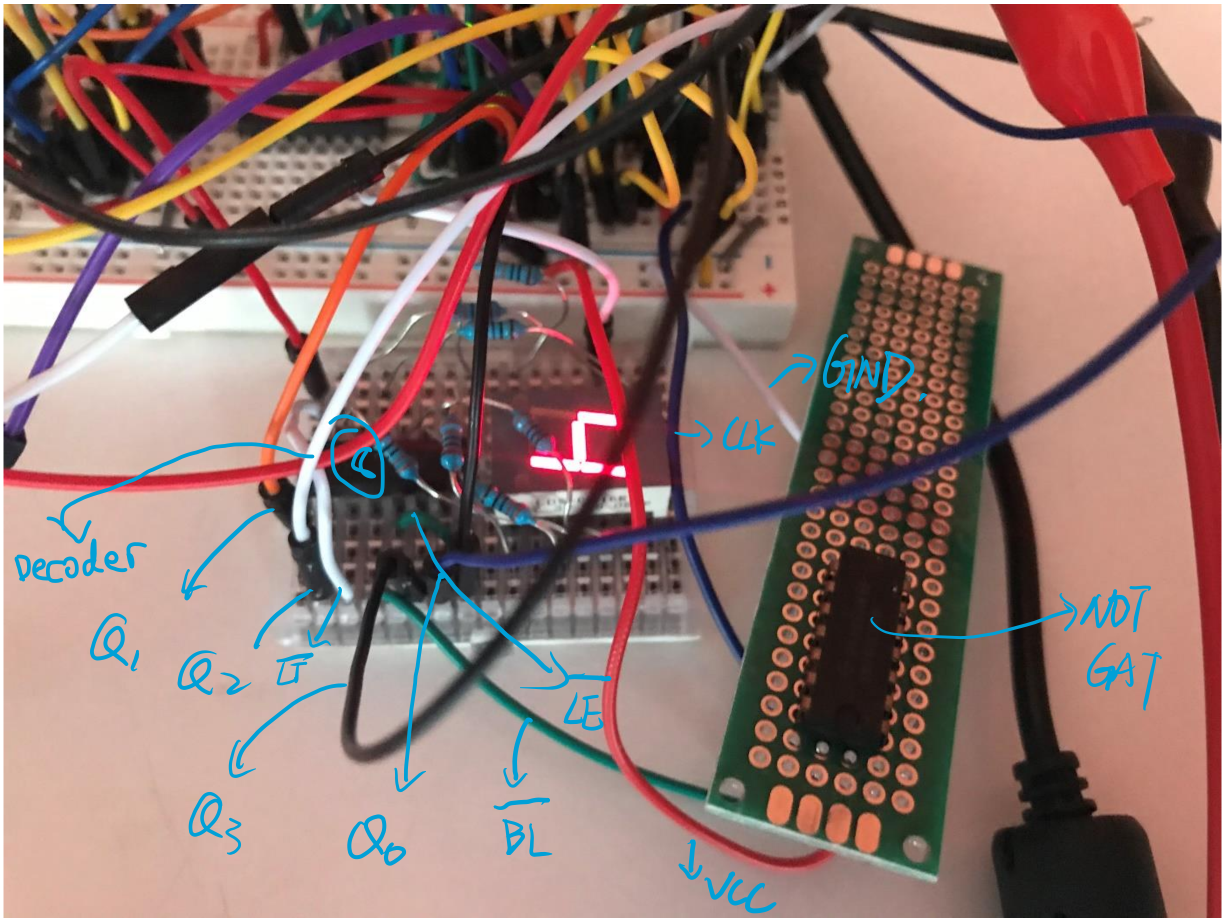
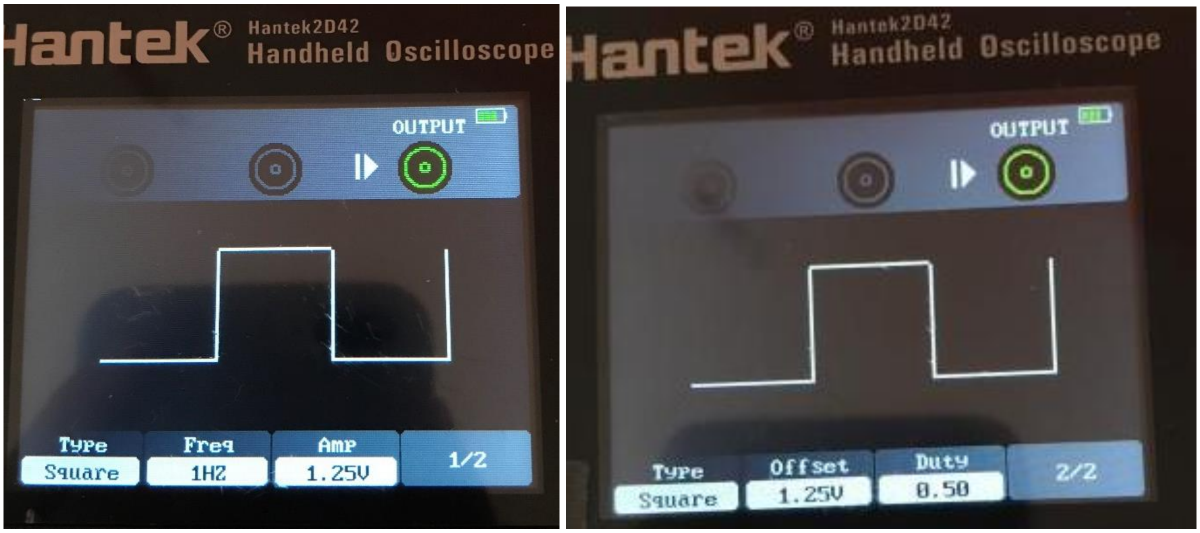


Figure 6: Labelled Display and NOT gate



Type	Freq	Amp	1/2
Square	1HZ	1.25V	

Type	Offset	Duty	2/2
Square	1.25V	0.50	

Figure 7: Clock Signal Setup

The pictures above show the overview of the construction of the circuit. In total, there are 3 74HC109N used, placed on the right-hand side. The AND gates, OR gates and NOR gates are placed on the left-hand side of the breadboard. On the baby breadboard, the BCD decoder and the 7-segment display are placed there. On the green solder board, a NOT gate is soldered onto the board and 4 wires are also soldered onto connect to the breadboard; they are VCC, GND, CLK and the output is the inverted CLK. The power supply comes from the 3.3V pin on Elegoo board. The clock signal is set to 1.25V and an offset of 1.25V as shown in figure 7. The clock pin connects directly to the Hantek output.

Unfortunately, the display did not display the sequence as expected in Multisim even though it was constructed the same way as Multisim. The following are my debugging process and some videos showing the process and the "PARTIALLY" working circuit.

Debugging Process

I have put in a lot of efforts debugging the circuit because I really wanted to work properly. I spent 3 full days debugging the circuit and finally some numbers are showing up.

Step 1: Rebuilding the Circuit

As the circuit did not work on the first try, I recognized that some jumper wires are connected incorrectly and my setup for the clock, and the display were partially incorrect. Therefore, rebuilt the circuit and a "4" is finally showing up on the display at the initial state.

Step 2: Changing ICs and 7-segment display

However, the circuit was working very by chance since the breadboard was very finicky. I also suspected that some IC was not pushed in properly or not connected to the pins really well. In some of the trial, it would show "5" if I tilt the board towards the left, and show "4" or "7" when I tilt the board towards the right. When I place the board horizontally on the desk and push it down a bit, it would show "8". These all indicated that the connection of the wires, ICs and the breadboard were definitely not good. Therefore, I decided to take everything apart and started my 3rd rebuild. I also switched from the 516 to the 416RS display.

Since I suspected some ICs are not functioning properly (and I did find that one of the AND gate was acting funny). I took out some new ICS and replaced the old ones on the board. After changing the ICs, I made sure that the ICs are pushed ALL THE WAY DOWN like displayed in figure 8 below. After pushing the ICs forcefully down for about 15 minutes, I asked a family member to do the same thing just to ensure that all ICs are connected to the breadboard properly. Next, I connected all the VCC pins and GND pins to the "+" rail and the "-" rail.

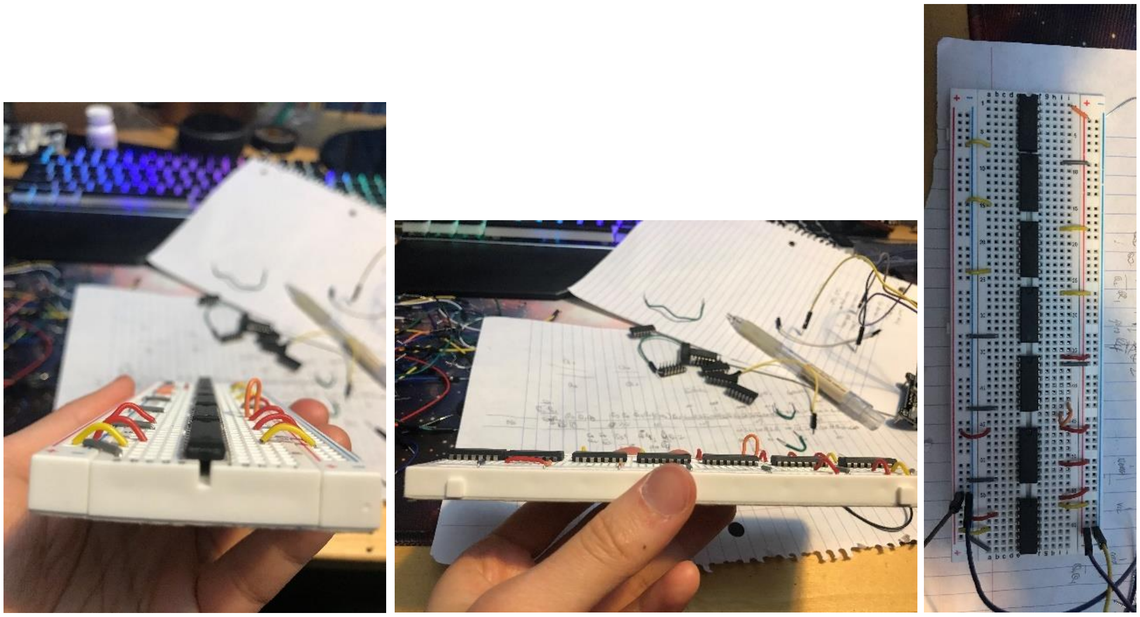


Figure 8: Integrated Circuits Placements

Step 3: Gates Testing

The next part of the debugging is unit testing. I wanted to ensure that all the gates are working properly and not having any defects that can potentially affect the final result. I took the time to check each gate individually. I checked the AND, OR, NOR gates by connecting an LED with a 330Ω resistor. The input for the gates use 2 jump wires iterate through the 4 cases of the possible combination for inputs. All the demo testing videos are listed below:

AND Gate Testing: https://youtu.be/UFlHYkU3eLs

OR Gate Testing: https://youtu.be/-iVfno6co0s

NOR Gate Testing: https://youtu.be/xHwgHSNR0mQ

I went through all the gates and all of them works as expected. One of the NOR gate was working a little bit finicky, and I avoid that set of inputs and outputs in the final build.

Step 4: Display Testing

Next, I wanted to make sure that all pins of the display can turn on the corresponding digit pin on the display. In the video, I grounded the display with a $1K\Omega$ resistor. I used a jump wire that is connected to the 3.3V power supply to touch every pin. As shown in the video, all the pins have turned on correctly.

7 Segment Display Testing: https://youtu.be/zjK8k6K1_U8

Step 5: JK Flipflop Testing

The flipflops are the important parts sequence logic. I used the setup from Topic 7; I set up the click signal with the 2N222 Transistor, and set the amp on Hantek to 1.00V and offset = 0V. the J pin is connected to VCC, $\overline{K}$ connected to GND. $\overline{PR}\ and\ \overline{CLR}$ pin are both set up high, connected to the VCC. The clock pin takes clock signal from the transistor setup. An LED is attached through a resistor to be the output. The LED is expected to be blanking which was shown in the Topic 7 lab video that Dr. Minnick made. In the debugging video, the LED was flashing as expected. I have also gone through every single flipflops to make sure that all of them are at least functioning. Something I noticed when I was testing was that some of the lights does not turn on automatically and I had to just randomly rotate the wire or add extra pressure on the IC in order for the LED to be flashing. I tried to solve the problem by changing to another 74HC109, and used other wires and switched pin holes. But again, sometimes they work perfectly and sometimes, it is really luck.

JK Flipflop Testing: https://youtu.be/0UbGEFObqLQ

Step 6: Confirmation of Correct Wire Connection

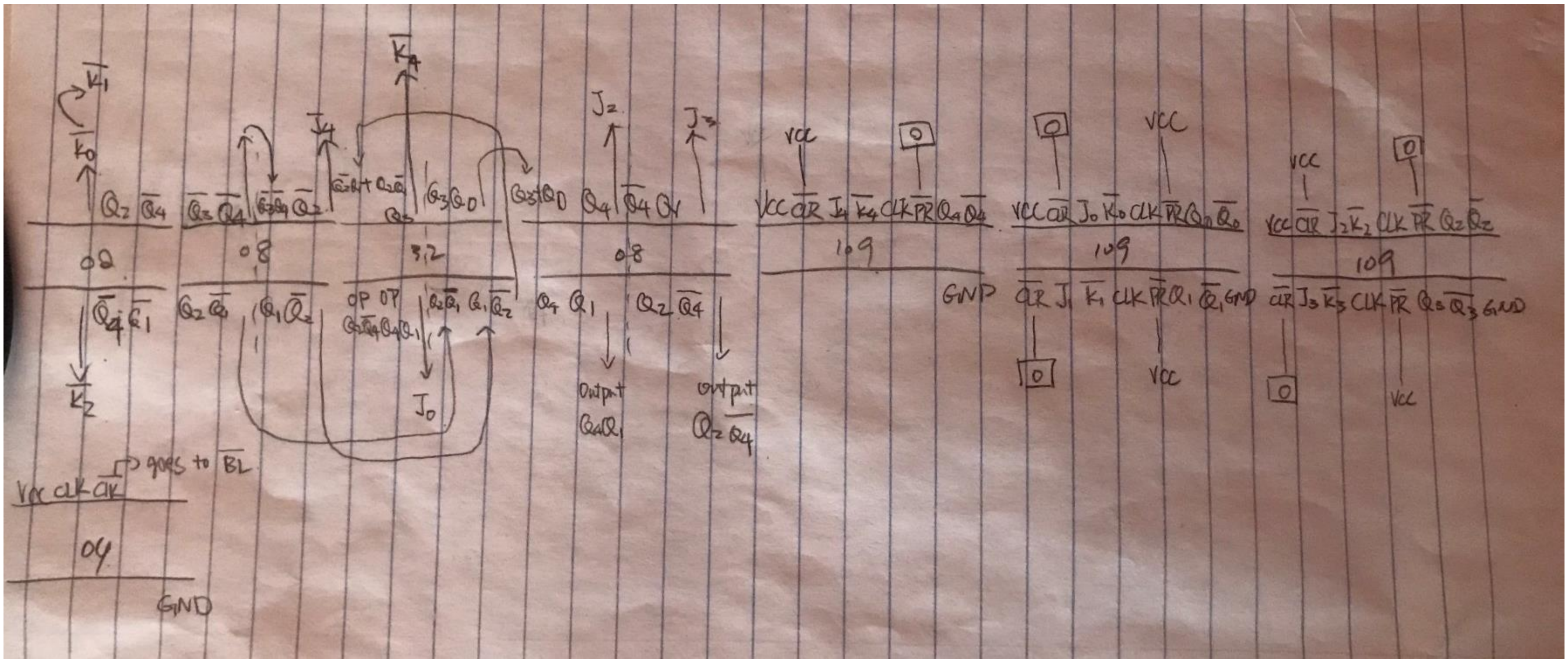

K1
K0
K4
J4
J2
J3
Vcc
Q2 Q4
Q3 Q4
08
08
32
08
109
109
109
GND
Q4 K1
Q2 Q3
Q1 Q2
J0
K2
Q4 Q1
Q2 Q4
output Q4Q1
output Q2Q4
Vcc CLR J4 K4 CLK PR Q4 Q4
Vcc CLR J0 K0 CLK PR Q0 Q0
Vcc CLR J2 K2 CLK PR Q2 Q2
CLR J1 K1 CLK PR Q1 Q1 GND
CLR J3 K3 CLK PR Q3 Q3 GND
0
goes to BL
Vcc clk clr
04
GND

Figure 9: Wire Connection Checking

After testing every single thing I can potentially test and debug, I proceeded to build the circuit. Sadly, the circuit still doesn't work as expected. However, it at least shows a "4" at the initial state and does not flicker to other numbers when I move the board around. I suspected if I have connected any wires incorrectly because there is a huge chance of me doing that due to the number of gates and wires there are. I took a piece of paper and drew out the outline of the circuit. The 7-segment display part was ignored because I made sure that my connections were definitely correct, and I have asked my peers who are also taking the course to check my connection for the display. On the paper, I went through every single wire, and wrote down where they are placed in the gates, and where their outputs went. I checked the handwritten checking with my Multisim and my analytical chart, and I can confirm that all the pins are connected correctly.

Step 7: Final Testing and Demonstration

After debugging everything I can, I still couldn't figure out why the circuit was not working as expected. I have asked for peer help and just checking the wires and build and rebuild the same circuit for 3 days straight. Here are the best results I have every gotten.

Sequence 4 0 2 7 4 3 8: https://youtu.be/8MYxJeRkWSo

Sequence Variation 2: https://youtu.be/l_xsButcuDM

Circuit Explanation: https://youtu.be/RlX6udZpHiw

In the best trial, I have gotten:

4 → 0 → 2 → 7 → 4 → 3 → 8

which at least shows all the numbers that is in my student number and got the number order correctly. However, this does not happen when the circuit is initiated, it happens after going through a bunch of weird loops, and numbers that is not even supposed to be in the sequence, like 5, 6, 9 as shown in the video. Also, sometimes this the "best" sequence does not appear no matter how long I have been running the circuit, and sometimes, it would iterate through the nice sequence just with some weird numbers in between the iteration.

Discussion and Reflection

To be honest, I did not think this project is super hard because I understood the topics before really well. My analytical and Multisim did not take me more than 3 hours to do and my Multisim work on the first try. Everything was going smoothly until the physical build hits me. I still cannot figure out why the circuit would not work as expected although I have debugged through everything and asked for my friend's help. From the analytical section and Multisim, it is proved that my sequential logic for the circuit is definitely correct, and the physical build at least shows all the numbers that are supposed to be in the student number in the correct order but just not consistently showing. I still suspect that the 74HC109s are working weirdly and some connections on the breadboard were just being funny for this whole time. However, I at least learned how to use the 7 segment display theoretically with a simulation software. I am still very frustrated that the circuit does not work properly after 3 days of effort, however, I think it is time to move on since I have done everything to correct it.