Syed Ali Zaidi
zaidis33

Design Project

Table of Contents

Syed Ali Zaidi
zaidis33

Deliverables

Submit a report and video as follows:

1. Analytical:
    a. Write-up of your logic design showing your k-mapping, and logic optimization after k-mapping
    b. (Hint: for your simulation and physical build, your design may start in a state that you didn't consider it could ever be in. You can either design the implementation to be robust enough to deal with these or build in a reset feature to ensure it starts in an expected state).
2. Simulation:
    a. Picture of your Multisim circuit and output timing diagram
    b. Video must include a screen capture of your simulation progressing through your student number
3. Physical build:
    a. Document your build with pictures at various stages along the way and write-up explanations of work to test and debug as you progress through it
    b. Picture of your built circuit implementation with your student card in frame (in the report)
    c. Video must include your built circuit implementation cycling through your student number and a brief explanation of your build and what does what

Introduction

The main goal of the design project is to create an infinite loop of the numbers of my student number on a 7-segment display. The solution of my design project is composed of three types of solutions: analytical, digital, and physical solution. My analytical solution involved making a state-transition table and making the K-maps of every input. I then used techniques like sums of products to get a Boolean expression while also considering the possible optimization techniques. With my digital solution, I created a circuit in NI Multisim which used the expressions from the analytical solution. I also showed the timing diagram for the circuit using the logic analyzer tool. For my physical solution, I replicated the circuit on a breadboard and used another circuit as my clock signal which produced the desired result on the 7-segment display.

Syed Ali Zaidi
zaidis33

Analytical Solution

Student Number

First, I have to convert each number of my student number from decimal to binary form. I will do this because the outputs of the flip-flops (q1, q2, q3, and q4) should resemble each number of my student number.

Student Number = 400313696

Decimal Number	**Binary Number**				**Counter (q5)**
	q1	**q2**	**q3**	**q4**	
4	0	1	0	0	X
0	0	0	0	0	0
0	0	0	0	0	1
3	0	0	1	1	0
1	0	0	0	1	X
3	0	0	1	1	1
6	0	1	1	0	0
9	1	0	0	1	X
6	0	1	1	0	1

The counter bit (or q5) represents the numbers that repeat twice. This will lead to us adding one more flip-flop to keep track of all the numbers that repeat and hence are in different states.

One optimization I came up with was that I only included only one bit for my counter variable. This was only possible because the numbers that repeat in my student number only repeat once. Thus, I represented them by 0 and 1. Not only will this decrease a flip-flop but also will make my K-mapping easier by only having to do 5-bit K-mapping instead of 6-bit which allows for lesser mistakes.

Excitation Table and JK Flip-Flop State Transition Table

Below are the tables I will use to determine the inputs and the outputs. The excitation table will allow me to exactly determine what the inputs will be due to a change in the outputs. For example, for my first number, when q2 goes from 1 to 0, we will know exactly the inputs which will cause that change. This is very useful to go from one specific number to the next.

Q_n	Q_{n+1}	J_n	$\sim K_n$
0	0	0	X
0	1	1	X
1	0	X	0
1	1	X	1

I also used the JK flip-flop state transition table to check my working. If I got the next number from the inputs, then that means that my working is correct. Otherwise, I should debug what is wrong (check Discussion for my debugging approach).

J_n	$\sim K_n$	Q_{n+1}
0	1	Q_n
0	0	0 (reset)
1	1	1 (set)
1	0	$\sim Q_n$ (toggle)

State Transition Table (Preliminary Design)

With all of the necessary tools to start the analysis, I will first build a state-transition table as shown below. As seen, the outputs correspond to the next number in binary form which shows that the table is correct.

Note that for the counter bit, I chose a value instead of X. This is because even though placing an X is more robust, it is not optimized. It will also be a very huge problem when determining the outputs of each flip-flop as an X for both inputs could mean any condition from the truth table thus, I ignored that case.

I have also colour coded all of my rows which not only saves me time trying to find the right row but also prevents mistakes. The colour coding was especially useful when doing the K-mapping as I can just colour the specific block and match it from my state transition table. Moreover, it also proved extremely useful when debugging.

Syed Ali Zaidi
zaidis33

#	Current State				Counter	Outputs					JK Flip-Flop Inputs									
	q1	q2	q3	q4	q5	Q1	Q2	Q3	Q4	Q5	J1	~K1	J2	~K2	J3	~K3	J4	~K4	J5	~K5
4	0	1	0	0	1	0	0	0	0	0	0	X	X	0	0	X	0	X	X	0
0	0	0	0	0	0	0	0	0	0	1	0	X	0	X	0	X	0	X	1	X
0	0	0	0	0	1	0	0	1	1	0	0	X	0	X	1	X	1	X	X	0
3	0	0	1	1	0	0	0	0	1	0	0	X	0	X	X	0	X	1	0	X
1	0	0	0	1	0	0	0	1	1	1	0	X	0	X	1	X	X	1	1	X
3	0	0	1	1	1	0	1	1	0	0	0	X	1	X	X	1	X	0	X	0
6	0	1	1	0	0	1	0	0	1	0	1	X	X	0	X	0	1	X	0	X
9	1	0	0	1	0	0	1	1	0	1	X	0	1	X	1	X	X	0	1	X
6	0	1	1	0	1	0	1	0	0	0	0	X	X	1	X	0	0	X	X	1

We can optimize the table to include only 0 for the value of ~K1 so that it would not have any gates.

#	Current State				Counter	Outputs					JK Flip-Flop Inputs									
	q1	q2	q3	q4	q5	Q1	Q2	Q3	Q4	Q5	J1	~K1	J2	~K2	J3	~K3	J4	~K4	J5	~K5
4	0	1	0	0	1	0	0	0	0	0	0	0	X	0	0	X	0	X	X	0
0	0	0	0	0	0	0	0	0	0	1	0	0	0	X	0	X	0	X	1	X
0	0	0	0	0	1	0	0	1	1	0	0	0	0	X	1	X	1	X	X	0
3	0	0	1	1	0	0	0	0	1	0	0	0	0	X	X	0	X	1	0	X
1	0	0	0	1	0	0	0	1	1	1	0	0	0	X	1	X	X	1	1	X
3	0	0	1	1	1	0	1	1	0	0	0	0	1	X	X	1	X	0	X	0
6	0	1	1	0	0	1	0	0	1	0	1	0	X	0	X	0	1	X	0	X
9	1	0	0	1	0	0	1	1	0	1	X	0	1	X	1	X	X	0	1	X
6	0	1	1	0	1	0	1	0	0	0	0	0	X	1	X	0	0	X	X	1

Now we shall do K-mapping to determine the Boolean expressions of each input.

Syed Ali Zaidi
zaidis33

K-mapping (Preliminary Design)

For my K-mapping, I used the sum of products approach to determine the Boolean expression. The algorithm is that we will enclose the 1s in boxes which have a size of 2^n. Furthermore, the boxes also have to occupy the largest amount of space possible.

For my design, because I am using five flip-flops, the K-mapping will also allow plane-to-plane traversal which I imagined as the two tables stacked on top of one another. I have shown this by using the same colour of boxes.

For J1									
	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	0	0	0	X	00	0	X	0	X
01	0	X	X	1	01	X	X	X	0
11	X	X	X	X	11	X	X	X	X
10	X	X	X	X	10	X	X	X	X

$$J1 = q3 * \overline{q4} * \overline{q5}$$

$$\sim K1 = 0$$

For J2									
	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	0	0	0	X	00	0	X	1	X
01	X	X	X	X	01	X	X	X	X
11	X	X	X	X	11	X	X	X	X
10	X	1	X	X	10	X	X	X	X

$$J2 = q1 + q5 * q3$$

For ~K2

	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	X	X	X	X	00	X	X	X	X
01	0	X	X	0	01	X	X	X	1
11	X	X	X	X	11	X	X	X	X
10	X	X	X	X	10	X	X	X	X

$$\sim K2 = q5$$

For J3

	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	0	1	X	X	00	1	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	1	X	X	10	X	X	X	X

$$J3 = q4 + q5$$

For ~K3

	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	X	X	0	X	00	X	X	1	X
01	X	X	X	0	01	X	X	X	0
11	X	X	X	X	11	X	X	X	X
10	X	X	X	X	10	X	X	X	X

$$\sim K3 = q5 * q4$$

For J4

	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	0	X	X	X	00	1	X	X	X
01	0	X	X	1	01	X	X	X	0
11	X	X	X	X	11	X	X	X	X
10	X	X	X	X	10	X	X	X	X

$$J4 = \overline{q5} * q3 + q5 * \overline{q3}$$

Syed Ali Zaidi
zaidis33

For ~K4

	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	X	1	1	X	00	X	X	0	X
01	X	X	X	X	01	X	X	X	X
11	X	X	X	X	11	X	X	X	X
10	X	0	X	X	10	X	X	X	X

$$\sim K4 = \overline{q5} * \overline{q1}$$

For J5

	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	1q2/q3q	00	01	11	10
00	1	1	0	X	00	X	X	X	X
01	X	X	X	0	01	X	X	X	X
11	X	X	X	X	11	X	X	X	X
10	X	1	X	X	10	X	X	X	X

$$J5 = \overline{q1}$$

For ~K5

	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	1q2/q3q	00	01	11	10
00	X	X	X	X	00	0	X	0	X
01	0	X	X	X	01	X	X	X	1
11	X	X	X	X	11	X	X	X	X
10	X	X	X	X	10	X	X	X	X

$$\sim K5 = q3 * \overline{q4}$$

The expressions are summarized in the table below.

Input	**Boolean Expression**	**Logic Gates**
J1	$q3 * \overline{q4} * \overline{q5}$	3 AND
~K1	0	-
J2	$q1 + q5 * q3$	1 OR, 1 AND
~K2	$q5$	-
J3	$q4 + q5$	1 OR
~K3	$q5 * q4$	1 AND
J4	$\overline{q5} * q3 + q5 * \overline{q3}$	1 OR, 2 AND
~K4	$\overline{q5} * \overline{q1}$	1 AND
J5	$\overline{q1}$	-
~K5	$q3 * \overline{q4}$	1 AND

Preliminary Design Problems

There are a lot of problems associated with this design. As seen from the results, there are a lot of different gates which means that not only will a chance of making a mistake increase but also it will require more space on the breadboard.

In addition, it is not optimized at all. Expressions like J4 could be more simplified and converted to NAND gates which will only require three NAND gates. This might not seem like a huge difference from the AND and OR gates but in reality, the NAND gates have approximately 50% less transistors and the chip has four NAND gates so only one chip can be used.

My approach to this problem also did not consider the product of sums method which would have been more useful in situations where there were less 0s.

Due to all of these factors, I shall now determine expressions which are more optimized.

Syed Ali Zaidi
zaidis33

State Transition Table (Optimized Design)

Below is the state-transition table for my optimized design. The only thing I could optimize in this table was the value of the counter bit that had a value of X. With techniques like trial-and-error, I determined that if I set the counter bit to 0 for all the X values, I will get a column of 0s and Xs for ~K1 and ~K5 which will lead to lesser gates.

I also tried the same thing but now I placed 1s instead of Xs for the counter bit and got the opposite result (Xs and 1s) for the input. Thus, both methods were equally optimized.

#	Current State				Counter	Outputs					JK Flip-Flop Inputs									
	q1	q2	q3	q4	q5	Q1	Q2	Q3	Q4	Q5	J1	~K1	J2	~K2	J3	~K3	J4	~K4	J5	~K5
4	0	1	0	0	0	0	0	0	0	0	0	X	X	0	0	X	0	X	0	X
0	0	0	0	0	0	0	0	0	0	1	0	X	0	X	0	X	0	X	1	X
0	0	0	0	0	1	0	0	1	1	0	0	X	0	X	1	X	1	X	X	0
3	0	0	1	1	0	0	0	0	1	0	0	X	0	X	X	0	X	1	0	X
1	0	0	0	1	0	0	0	1	1	1	0	X	0	X	1	X	X	1	1	X
3	0	0	1	1	1	0	1	1	0	0	0	X	1	X	X	1	X	0	X	0
6	0	1	1	0	0	1	0	0	1	0	1	X	X	0	X	0	1	X	0	X
9	1	0	0	1	0	0	1	1	0	1	X	0	1	X	1	X	X	0	1	X
6	0	1	1	0	1	0	1	0	0	0	0	X	X	1	X	0	0	X	X	0

Now converting all of the Xs in ~K1 and ~K5 to 0s so that the inputs can be connected to ground.

#	Current State				Counter	Outputs					JK Flip-Flop Inputs									
	q1	q2	q3	q4	q5	Q1	Q2	Q3	Q4	Q5	J1	~K1	J2	~K2	J3	~K3	J4	~K4	J5	~K5
4	0	1	0	0	0	0	0	0	0	0	0	0	X	0	0	X	0	X	0	0
0	0	0	0	0	0	0	0	0	0	1	0	0	0	X	0	X	0	X	1	0
0	0	0	0	0	1	0	0	1	1	0	0	0	0	X	1	X	1	X	X	0
3	0	0	1	1	0	0	0	0	1	0	0	0	0	X	X	0	X	1	0	0
1	0	0	0	1	0	0	0	1	1	1	0	0	0	X	1	X	X	1	1	0
3	0	0	1	1	1	0	1	1	0	0	0	0	1	X	X	1	X	0	X	0
6	0	1	1	0	0	1	0	0	1	0	1	0	X	0	X	0	1	X	0	0
9	1	0	0	1	0	0	1	1	0	1	X	0	1	X	1	X	X	0	1	0
6	0	1	1	0	1	0	1	0	0	0	0	0	X	1	X	0	0	X	X	0

Syed Ali Zaidi
zaidis33

K-Mapping (Optimized Design)

Below are the K-maps for each input.

For J1									
	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	0	0	0	X	00	0	X	0	X
01	0	X	X	1	01	X	X	X	0
11	X	X	X	X	11	X	X	X	X
10	X	X	X	X	10	X	X	X	X

$$\boldsymbol{J1 = q3 * \overline{q4} * \overline{q5}}$$

Optimization:

Although this method below might be more optimized, it will later be very difficult to add a separate NOR gate. Overall, it requires more gates, so the original expression is used. The expression also involves a lot of 0s thus the sums of products is considered.

$$J1 = q3 * \overline{q4} * \overline{q5}$$

$$J1 = \overline{\overline{q3 * \overline{q4} * \overline{q5}}}$$

$$J1 = \overline{\overline{q3} + \overline{q4 * q5}}$$

$$\boldsymbol{\sim K1 = 0}$$

Optimization:

This can not be optimized any further.

For J2									
	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	0	0	0	X	00	0	X	1	X
01	X	X	X	X	01	X	X	X	X
11	X	X	X	X	11	X	X	X	X
10	X	1	X	X	10	X	X	X	X

$$J2 = q1 + q5 * q3$$

Optimization:

J2 was converted to a NAND expression to reduce the types of gates being used. A more complex expression would be determined for product of sums, so it was ignored.

$$J2 = q1 + q5 * q3$$

$$J2 = \overline{\overline{q1 + q5 * q3}}$$

$$\boldsymbol{J2 = \overline{\overline{q1} * \overline{q5 * q3}}}$$

For ~K2									
	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	X	X	X	X	00	X	X	X	X
01	0	X	X	0	01	X	X	X	1
11	X	X	X	X	11	X	X	X	X
10	X	X	X	X	10	X	X	X	X

$$\boldsymbol{\sim K2 = q5}$$

Optimization:

This can not be optimized any further.

For J3									
	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	0	1	X	X	00	1	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	1	X	X	10	X	X	X	X

$$J3 = q4 + q5$$

Optimization:

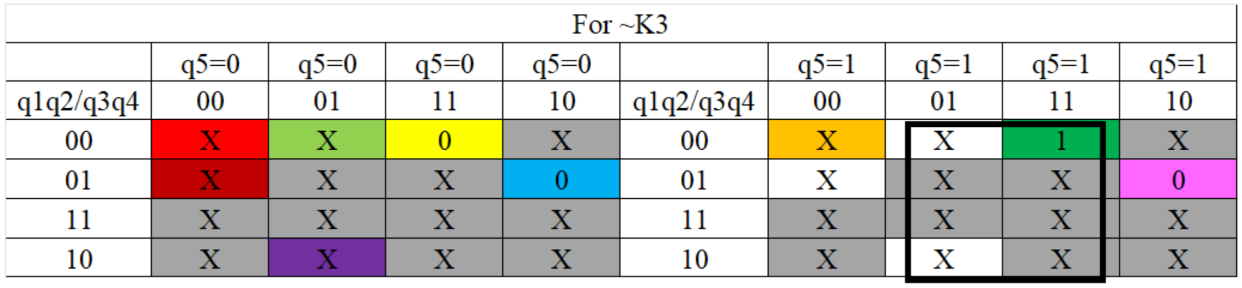
Even though adding an OR gate would involve the usage of fewer gates, it would however not account for the extra chip to be used in the physical build. Thus, NAND implementation was done. The product of sums method was not adopted because it involved surrounding only four boxes which gave us three variables.

Syed Ali Zaidi
zaidis33

$$J3 = q4 + q5$$

$$J3 = \overline{\overline{q4 + q5}}$$

$$\boldsymbol{J3 = \overline{\overline{q4} * \overline{q5}}}$$



For ~K3									
	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	X	X	0	X	00	X	X	1	X
01	X	X	X	0	01	X	X	X	0
11	X	X	X	X	11	X	X	X	X
10	X	X	X	X	10	X	X	X	X

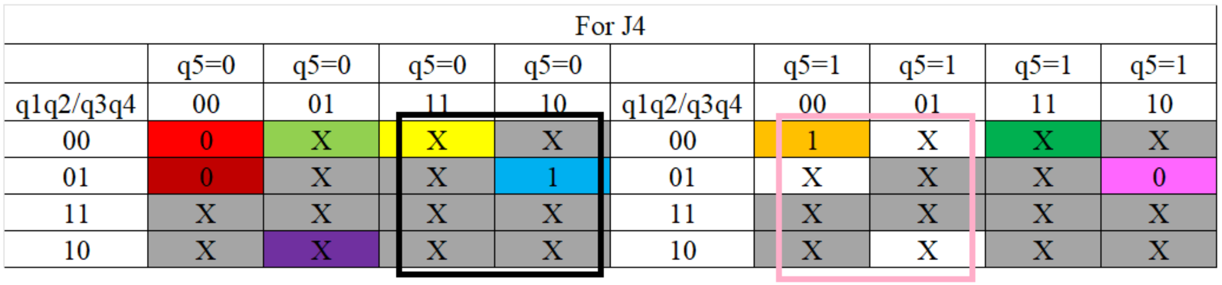
$$\sim K3 = q5 * q4$$

Optimization:

When determining the number of chips to be used for the physical build, I encountered an obstacle where I had to add another chip just to facilitate one AND gate. Thus, I converted the expression to NAND to reduce the number of chips.

$$\sim K3 = q5 * q4$$

$$\boldsymbol{\sim K3 = \overline{\overline{q5 * q4}}}$$



For J4									
	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	0	X	X	X	00	1	X	X	X
01	0	X	X	1	01	X	X	X	0
11	X	X	X	X	11	X	X	X	X
10	X	X	X	X	10	X	X	X	X

$$J4 = \overline{q5} * q3 + q5 * \overline{q3}$$

Optimization:

I converted the expression above to a simple NAND expression which does not allow any OR gates. I also implemented the product of sums method but that gave me a very similar answer to the sums of products method.

Syed Ali Zaidi
zaidis33

$$J4 = \overline{q5} * q3 + q5 * \overline{q3}$$

$$J4 = \overline{\overline{\overline{q5} * q3 + q5 * \overline{q3}}}$$

$$\boldsymbol{J4 = \overline{\overline{\overline{q5} * q3} * \overline{q5 * \overline{q3}}}}$$

For ~K4									
	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	X	1	1	X	00	X	X	0	X
01	X	X	X	X	01	X	X	X	X
11	X	X	X	X	11	X	X	X	X
10	X	0	X	X	10	X	X	X	X

$$\boldsymbol{\sim K4 = \overline{q5} * \overline{q1}}$$

Optimization:

This can not be optimized any further.

For J5									
	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	1	1	0	X	00	X	X	X	X
01	0	X	X	0	01	X	X	X	X
11	X	X	X	X	11	X	X	X	X
10	X	1	X	X	10	X	X	X	X

$$J5 = \overline{q1} * \overline{q2} * \overline{q3} + \overline{q3} * q4$$

Optimization:

The sums of products gave a very complicated expression thus I used the product of sums to get a more optimized solution as seen below.

For J5									
	q5=0	q5=0	q5=0	q5=0		q5=1	q5=1	q5=1	q5=1
q1q2/q3q4	00	01	11	10	q1q2/q3q4	00	01	11	10
00	1	1	0	X	00	X	X	X	X
01	0	X	X	0	01	X	X	X	X
11	X	X	X	X	11	X	X	X	X
10	X	1	X	X	10	X	X	X	X

$$\boldsymbol{J5 \ = \ \overline{q2} * \overline{q3}}$$

$$\boldsymbol{\sim K5 = 0}$$

Optimization:

This can not be optimized any further.

Analytical Solution Summary

Below is a table showing all of the expressions and how many gates were used for each input.

Input	**Boolean Expression**	**Logic Gates**
J1	$q3 * \overline{q4} * \overline{q5}$	2 AND
~K1	0	-
J2	$\overline{\overline{q1} * \overline{q5 * q3}}$	2 NAND
~K2	$q5$	-
J3	$\overline{\overline{q4} * \overline{q5}}$	1 NAND
~K3	$\overline{\overline{q5 * q4}}$	2 NAND
J4	$\overline{\overline{\overline{q5} * q3} * \overline{q5 * \overline{q3}}}$	3 NAND
~K4	$\overline{q5} * \overline{q1}$	1 AND
J5	$\overline{q2} * \overline{q3}$	1 AND
~K5	0	-

I have also made a table comparing my preliminary design to my optimized design.

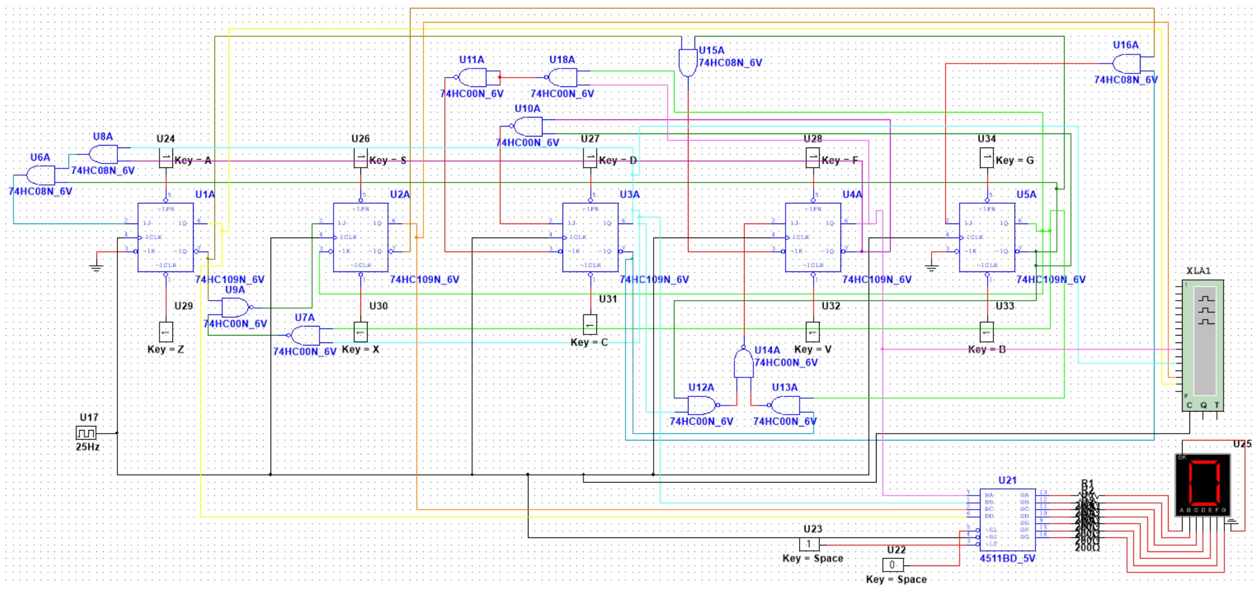
Syed Ali Zaidi
zaidis33

Input	**Logic Gates (Preliminary Design)**	**Logic Gates (Optimized Design)**
J1	2 AND	2 AND
~K1	-	-
J2	1 OR, 1 AND	2 NAND
~K2	-	-
J3	1 OR	1 NAND
~K3	1 AND	2 NAND
J4	1 OR, 2 AND	3 NAND
~K4	1 AND	1 AND
J5	-	1 AND
~K5	1 AND	-

As seen above, the optimized design has lesser types of gates as compared to the preliminary design. The total number of gates are around the same which was because I wanted my physical build to include as few chips as possible. I could have made some expressions include one AND gate instead of two NAND gates (for ~K3), but the problem was that I would have to include one more chip for just one AND gate (as one chip has only four gates) thus I converted into NAND. Moreover, the NAND gates are also made up of lesser transistors which is very optimized on its own.

Syed Ali Zaidi
zaidis33

Digital Solution

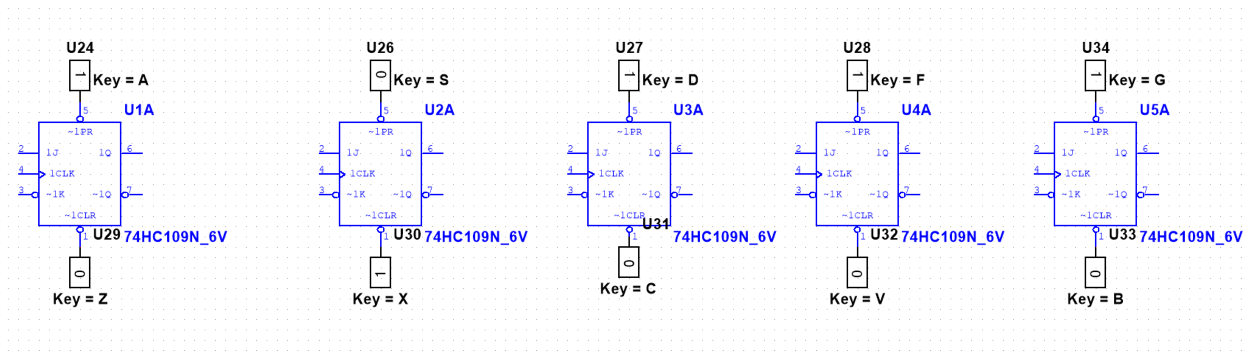


The diagram above shows the circuit that I created in Multisim. I used five JK flip-flops, four to represent the student number bits and one to represent the counter bit. The first flip-flop (left-most) carries the value of the most significant bit and the last one (right-most) carries the value of the counter.

For the clock signal, I used a clock signal source at 20 Hz and also connected it to the decoder chip. From the decoder chip, I placed 200 Ω resistors to prevent the LEDs in the 7-segment display to be damaged.

Pre-set and Clear Inputs

I used interactive digital sources as the power supply for the pre-set and clear pins in order to easily start the simulation from the number 4 through simple keystrokes. After the display showed 4, I just set all of them to 1 and the cycle starts. This was done by setting the pre-set and clear pins as shown below:



This is based on the fact that the pins represent the not, thus in order to stop the effect of the pre-set and clear pins, we have to give them a high voltage. In my case, I was forcing the second bit to be equal to 1 (not pre-set is 0 which means pre-set is 1) and the rest of them to be 0 (not pre-set is 1 which means pre-set is 0). The reverse case applies for clear.

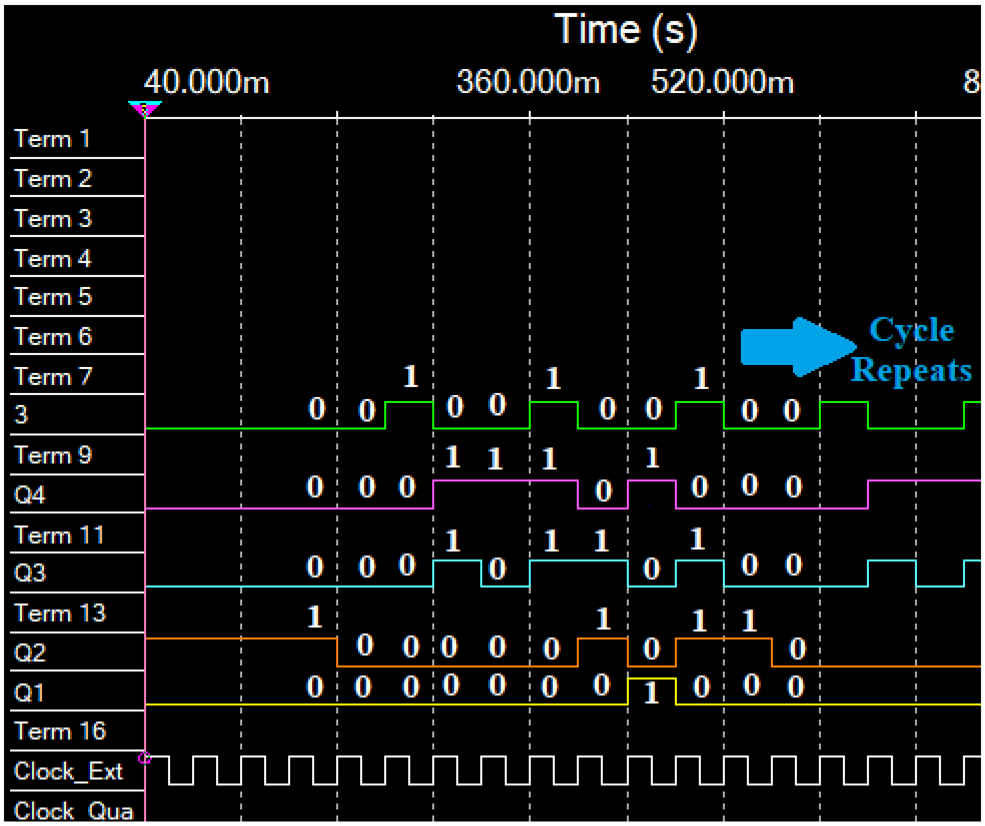
Colour Coding

In addition, I also colour-coded my wires to be easily able to figure out where the wire is connected instead of going back to its source. I adopted a general rule of making the outputs (q) to have a light-colour and the not of the outputs ($\bar{q}$) to have the dark-colour of the same colour. The colours are shown in the table below:

Output	**Colour**
q1	Yellow
~q1	Dark Yellow (Golden)
q2	Orange
~q2	Brown
q3	Light Blue
~q3	Dark Blue
q4	Pink
~q4	Purple
q5	Light Green
~q5	Dark Green
Clock Signal	Black

Timing Diagram and Cycle

I also attached logic analyzer which produces the timing diagrams for each output. I attached the C input of the analyzer to the clock signal and the rest to each of the five outputs. I set the readings to be external and then the following diagram was produced:



I have labelled each state which gives us the following cycle:

01000 → 00000 → 00001 → 00110 → 00010 → 00111 → 01100 → 10010 → 01101 → 01000 → CYCLE REPEATS

This is identical to the bits in my student number (excluding the last bit as it is the counter). The results are shown in the table below:

Digital Solution					
#	**q1**	**q2**	**q3**	**q4**	**q5**
4	0	1	0	0	0
0	0	0	0	0	0
0	0	0	0	0	1
3	0	0	1	1	0
1	0	0	0	1	0
3	0	0	1	1	1
6	0	1	1	0	0
9	1	0	0	1	0
6	0	1	1	0	1

Multisim Demonstration

I have also made a video going through the loop simulation in Multisim which can be found here: https://youtu.be/I9w9Y0zNfEs

Comparison between Analytical and Digital Solution

As seen below, my results from the digital solution are identical to the binary numbers defined in the analytical solution. The timing diagram and the Multisim demonstration also prove that the circuit does indeed produces the desired result. Thus, we can now construct the physical circuit.

#	Analytical Solution	Digital Solution
4	01000	01000
0	00000	00000
0	00001	00001
3	00110	00110
1	00010	00010
3	00111	00111
6	01100	01100
9	10010	10010
6	01101	01101

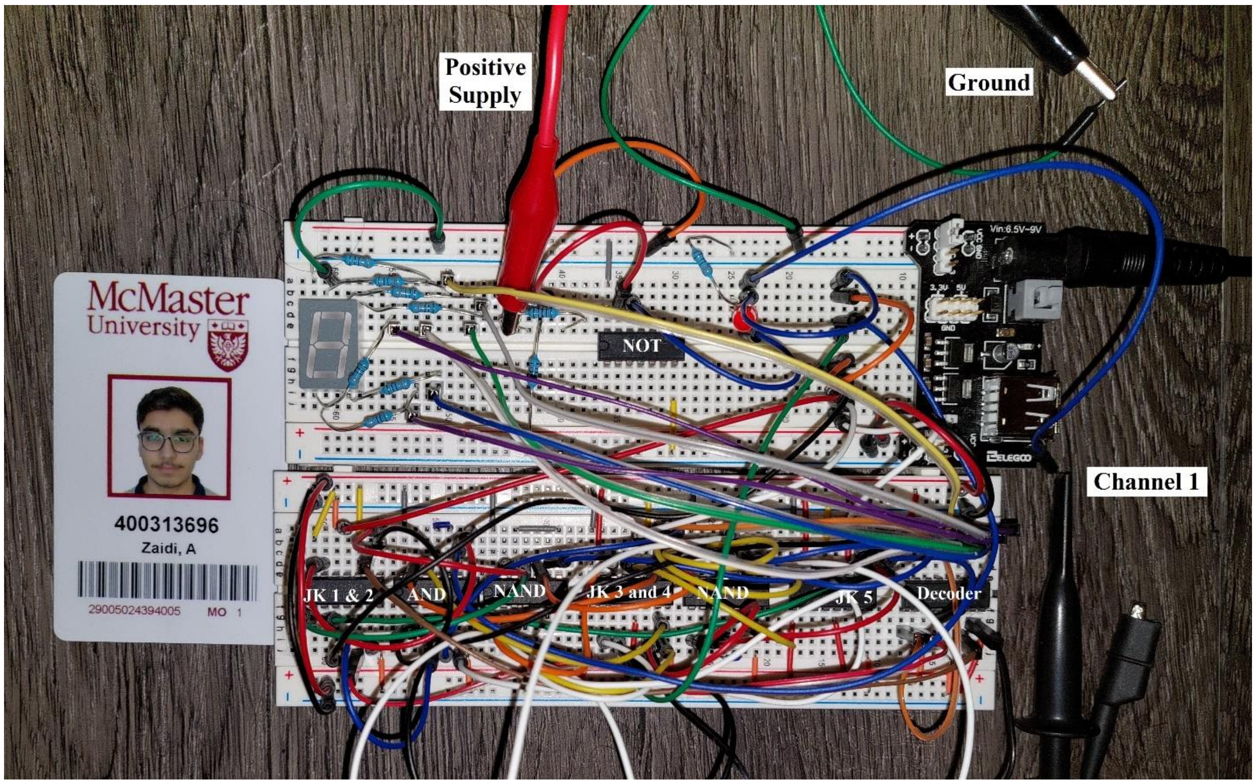
Syed Ali Zaidi
zaidis33

Physical Solution

For my physical solution, I replicated the Mulisim circiut on a breadboard. The circuit involves three parts. The top breadboard's right part is the clock signal circuit which produces a clock signal through a transistor and a NOT gate that is inputted to each flip-flop's CLK input. The bottom cirucit involves the JK flip-flops, the AND and NAND gates and the decoder chip. This is the circiut that produces the outputs which are the binary numbers of the student number. The third part is the top left which is the 7 segment display given an input through the decoder chips and connected via resistors to prevent damaging the LEDs.

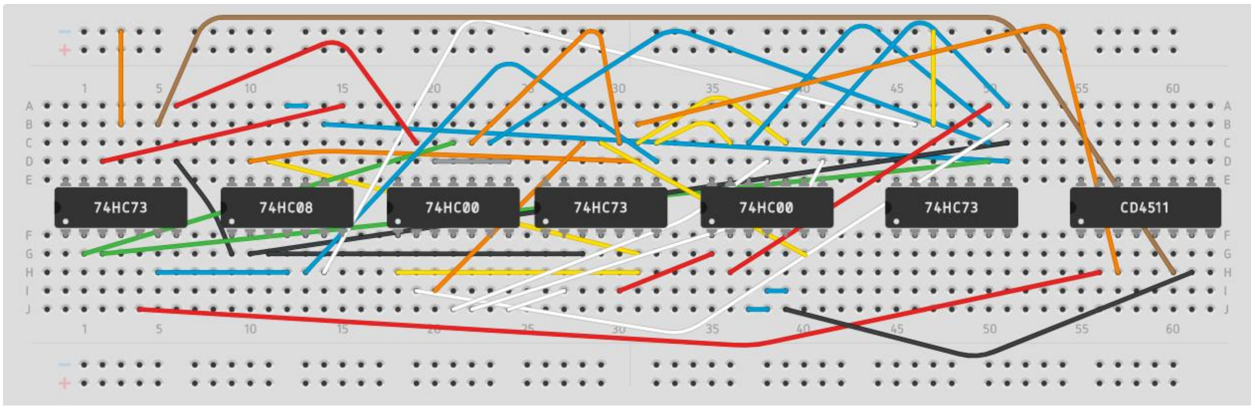
The bottom breadboard is where we used the expressions from the analytical portion and connected the wires in line with the boolean expressions. The positive and negative rails of the breadboard are connected by two wires shown on the extreme left. The pre-sets and clears were set to 1. This was because I had a 0 in my student number and it looped infinitely thus it did not matter where it started. To save space, I made those wires fixed too.

I tried to colour code at first but then due to the varying sizes of wires and the limited number of colours, I gave up on that (note the red and green colour for first slip-flop -- q1 and q2 -- and the orange and yellow colour for the second flip-flop -- q3 and q4). I, however, tried to still maintain a colour coding scheme by making wires which each go to a specific input the same colour. I also gave the positive supply the red and the negative supply the black colour.



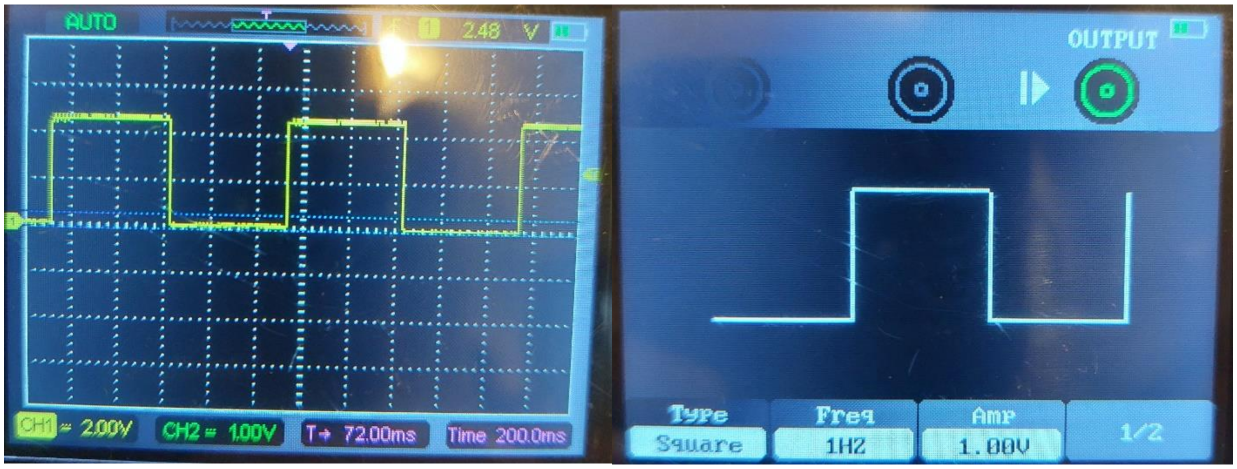
Syed Ali Zaidi
zaidis33

To better visualize what is going on in the bottom breadboard, I made a Tinker-Cad model which shows each wire clearly. Note that I have left out the voltage supply, ground, clock, pre-set and clear inputs as it is clear where each input goes, and I wanted to make it less clustered.



Clock Signal Waveform

Below is the clock signal measured by the Hantek Oscilloscope:

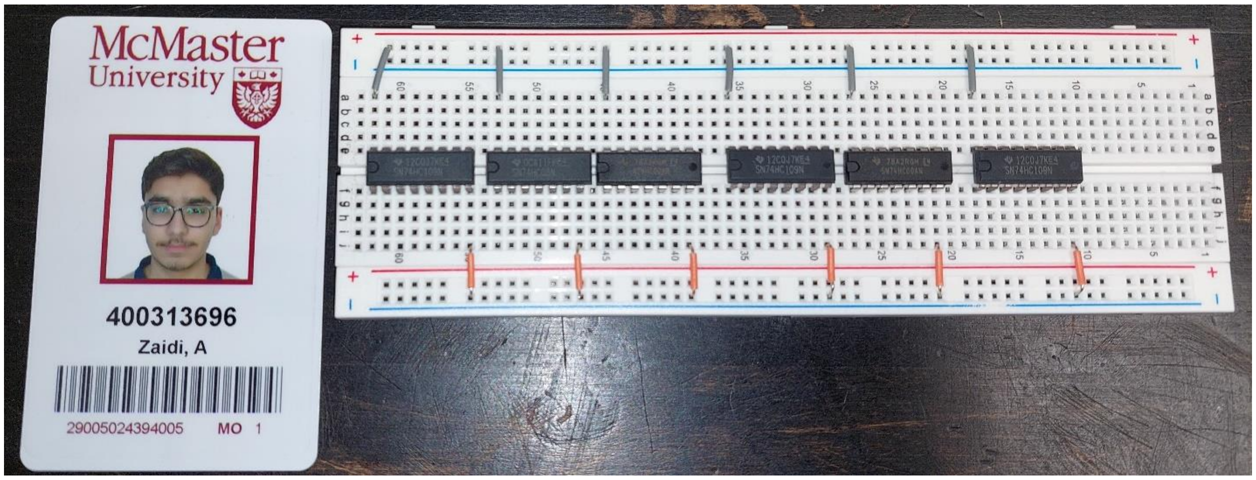


Physical Circuit Progress

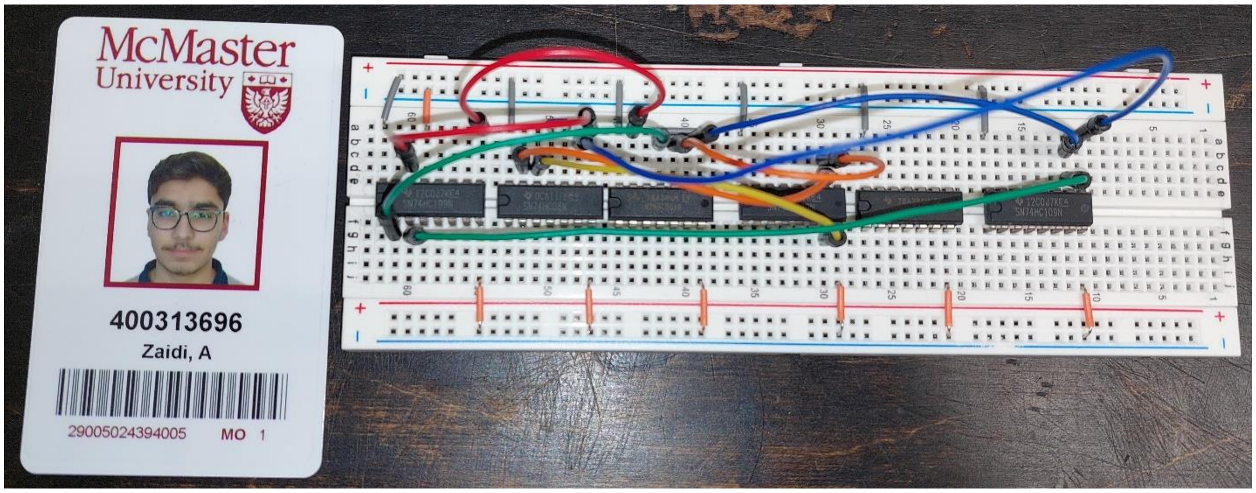
I also took some pictures while building the circuit for a better understanding of each flip-flop.

I first connected the vcc and ground to each chip. At this point I should have also connected the pre-sets and clears but I did that later.

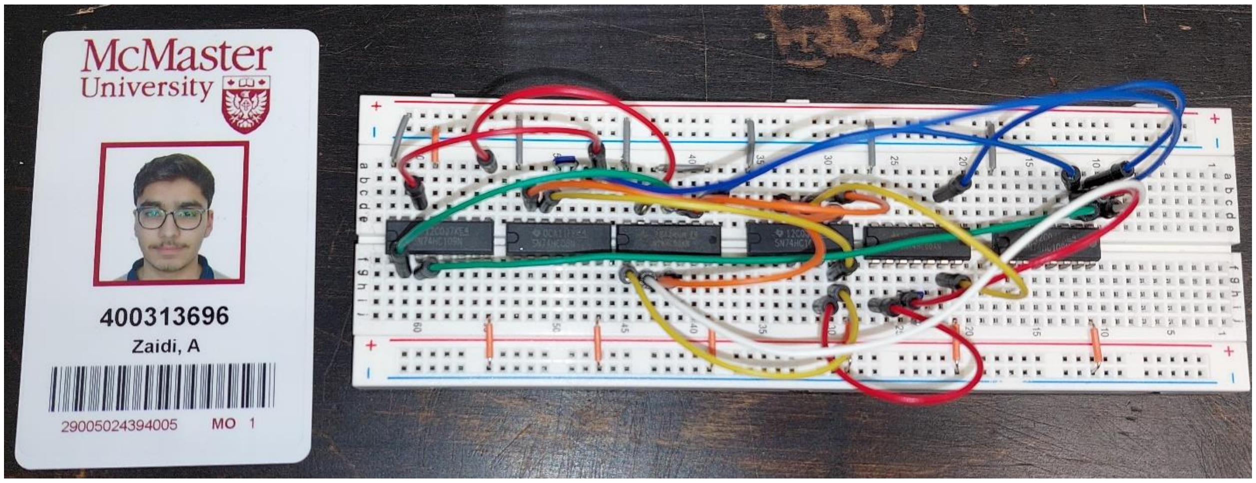
Syed Ali Zaidi
zaidis33



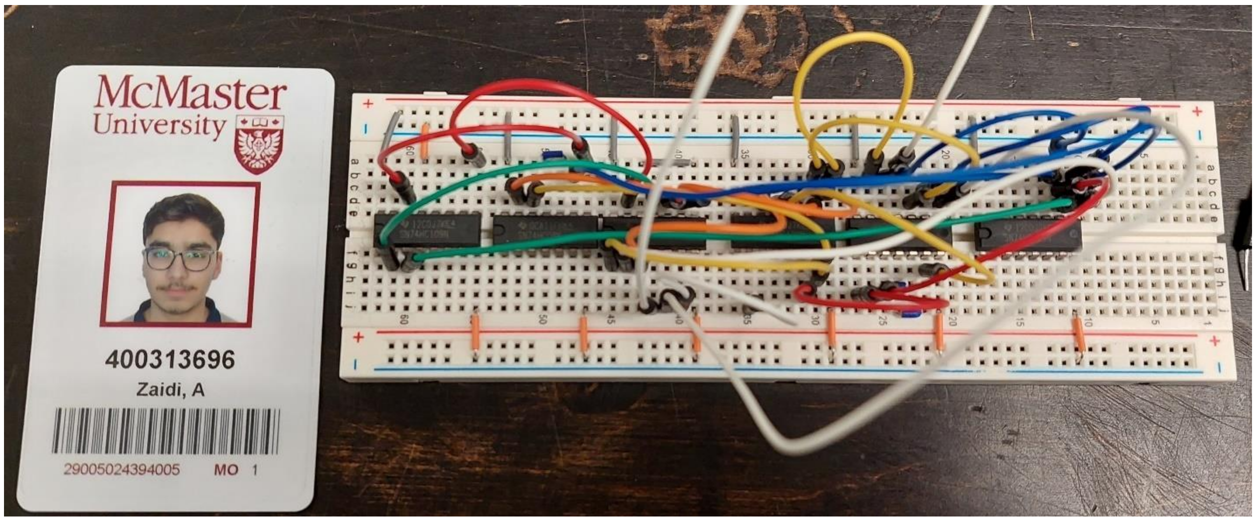
Next, I connected my first and second flip flop's inputs in accordance with the boolean expressions. Note that on the first flip-flop chip the first flip-flop is at the top and the second flip-flop is at the bottom.



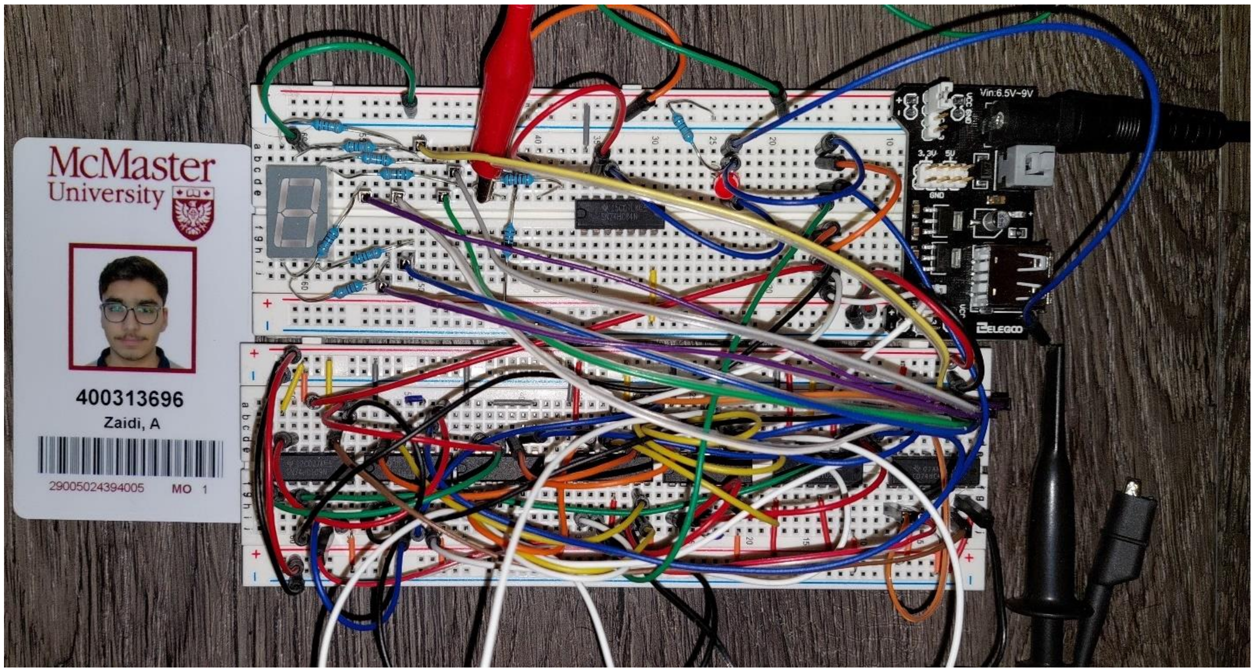
After that I connected the inputs of the third flip-flop which is (as shown) the top one on the fourth chip (from the left).



Then, I started the process of connecting the inputs of the fourth flip-flop (bottom one of the fourth chip).



Then after a simple connection of the fifth flip-flop, I connected my clock signal to each of the clock signal inputs of the flip-flops. I also gave the decoder chip the inputs of each flip-flop and carried that output to the 7-segment display which displayed the numbers.



Physical Demonstration of Circuit

I made a video showing my circuit completing the cycle of my student number which can be found here: https://youtu.be/1PvB5kO4Nj8

Physical Solution summary

As shown in the video above, my circuit completes the loop of numbers which is:

4 → 0 → 0 → 3 → 1 → 3 → 6 → 9 → 6 → 4 → CYCLE REPEATS

This concludes that the physical solution yielded the expected result meaning that my implementation was correct.

Comparison of Solutions

As seen above, all of my solutions produce the desired output. The analytical solution gives the Boolean expressions which were confirmed correct from the Multisim circuit when the simulation showed the correct cycle. The physical solution was confirmed correct as it produced my student number in the desired order with the correct numbers. Thus, not only were my solutions agreeing (or complementing) each other but they also produced the correct result.

Syed Ali Zaidi
zaidis33

Discussion

Overall, this design project was extremely useful for me as it allowed me to use all of my knowledge from logic gates and JK flip-flops to create a circuit that actually performs a practical task. The previous labs were more focused on visualizing the effects of the logic gates and flip-flops. Moreover, we also had to design our own circuits which had no objective. However, in this project, we were given the task of looping through our student numbers.

I spent a significant amount of time debugging my solutions. For my analytical solution, I did not consider looping through after the last value, so I changed that. This, however, caused a problem with the J and K inputs of my previous number (as the inputs depend on the next state as well). So, after some time, I figured that out and corrected it.

For my digital solution, I had lesser issues but mainly it was just me getting used to the decoder chip and the 7-segment display which made me a lot comfortable when making the physical build.

For my physical build, however, I was getting an unexpected output when I first powered it. I was getting a sequence of 4→0→2→4. This was very unexpected. I consulted the lab technician and he said that I should start with debugging my first JK flip-flop as the most significant bit was not producing a result (which produced 9). Thus, with an oscilloscope probe (from the lab), I checked all of the inputs of the first flip-flop which led me to seeing that there was no output being produced by ~q4 (which was a part of q1's input). I checked the inputs and outputs of the fourth flip-flop and that was when I caught the error. The flip-flop had inputs which produced a signal but there was no output. I suspected that the chip might have an issue and sure enough when I replaced the chip, the circuit gave me the correct output.

For digital circuits, the most amazing thing that I found was that it does not involve any sort of error. For my analogue circuits, I had to look at many factors (voltage leak, internal impedance of inductor, etc.) but for these types of circuits there can be only two states which makes it very difficult to have an error unless there is something wrong with the analytical solution.

Reflection

This design project exposed me to digital systems which are going to be essential for not only my future courses but also for my profession. I now have a wide understanding of digital logic and analysis. Focusing on minute details like understanding logic gates at the transistor level was what made this course special as it not only provided us with practical skills but also theoretical knowledge. Learning about how these logic gates and flip-flops work has now laid a foundation for studying their use in computers and embedded systems.

Video Presentation

https://youtu.be/MlfdaK6JAzc