ECE 385

Spring 2020

Experiment # 1

Introductory TTL Experiment

Megh Shah
Lab Section: ABC, Tuesday 11:30 AM
TA: David Zhang, Gene Shiue

Introduction:
Despite theoretical computations and predictions, not all chips and circuits designed will act ideal. It is important to learn the many errors and issues that appear in real world designs as it could conflict with results and outcomes. By designing a 2-to-1 multiplexer in certain ways, we can demonstrate these errors. One such error that occurs is a Static-1 Hazard which is when the output is a false low for a split second before switching back to a high. By analyzing different ways to build a circuit and the glitches that are produced as a result, we can design the most optimal and efficient circuit that produces little to no errors for the best functionality.

Written Description of the Operation of Circuits:
In the prelab, we were tasked to create two different circuits. The purpose of the first circuit was to produce the Static-1 Hazard. To produce this glitch, we created a K-map with the signals A, B, and C.

		BC			
		00	01	10	11
A	0	0	1	0	0
	1	0	1	1	1

As you can see, from the K-map above we can create a minimal SOP expression with the signals given:

$$Z = BA + B'C$$

By creating this minimal SOP expression, we are simplifying the circuit to use only the essential and least amount of chips. With the expression shown above, we are using only 4 NAND gates which can be implemented with only one 7400 chip. Furthermore, since the expression states that A and C will always be a high signal, we directly wired those signals to the positive voltage line. Due to this circuit not being symmetrical, as one of the expressions requires more NAND gates than the other, the circuit produces a Static-1 Hazard as current in the expression with less gates travels faster and changes the output before the other expression has time to revert the output back.

The significance of creating the second circuit was to showcase that despite non-ideal chips and transistors, we can still eliminate or minimize the glitches and hazards which can occur. To design this glitch-less circuit, we must account for all adjacent terms in the K-map. The following K-map shows which term we must account for in this scenario.

		BC			
		00	01	10	11
A	0	0	1	0	0
	1	0	1	1	1

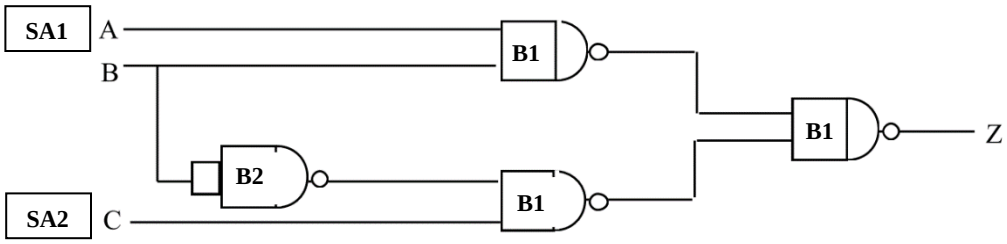
By accounting for this extra term, we are changing the expression of this design and making it not as simplified.

$$Z = BA + B'C + AC$$

Even though the circuit is not as efficient, as we are adding 2 more NAND gates to get the same output, this circuit eliminates the error as the extra term that we are adding ensures that the output stays at the correct signal and does not suffer from a Static-1 Hazard.

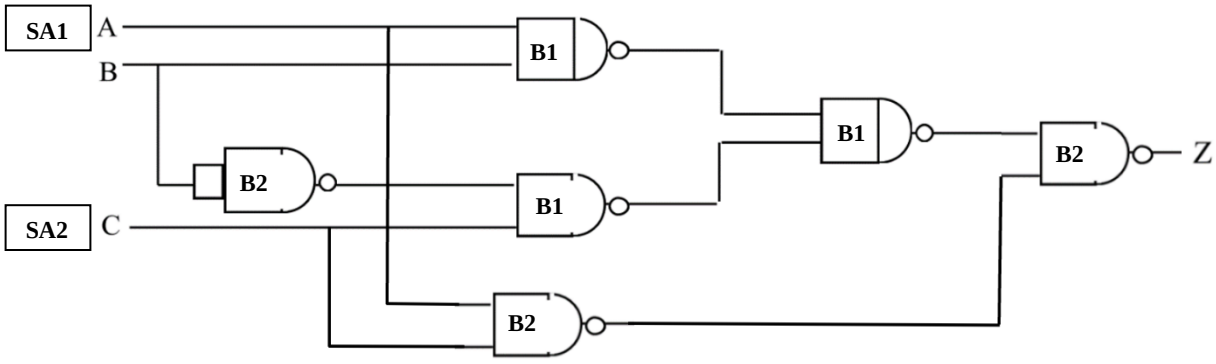
Circuit Design:

The following circuit designs showcase the circuits that we built. These circuits were designed by taking the expressions that we derived in the previous part, applying DeMorgan's Law, and converting all the gates to NAND gates.



$$Z = BA + B'C$$

This circuit diagram represents the circuit which produces the glitch. The Static-1 Hazard occurs due to the bottom half of the circuit having an extra NAND gate. Due to this, the result of the top half of the circuit will reach the NAND gate that results in output Z nanoseconds earlier, and will be NANDed with the previous value of the bottom half of the circuit rather than the new one. This will create a false low signal for a split second until the correct input from the bottom half of the circuit reaches the rightmost NAND gate and reverts the output back to a high signal.



$$Z = BA + B'C + AC$$

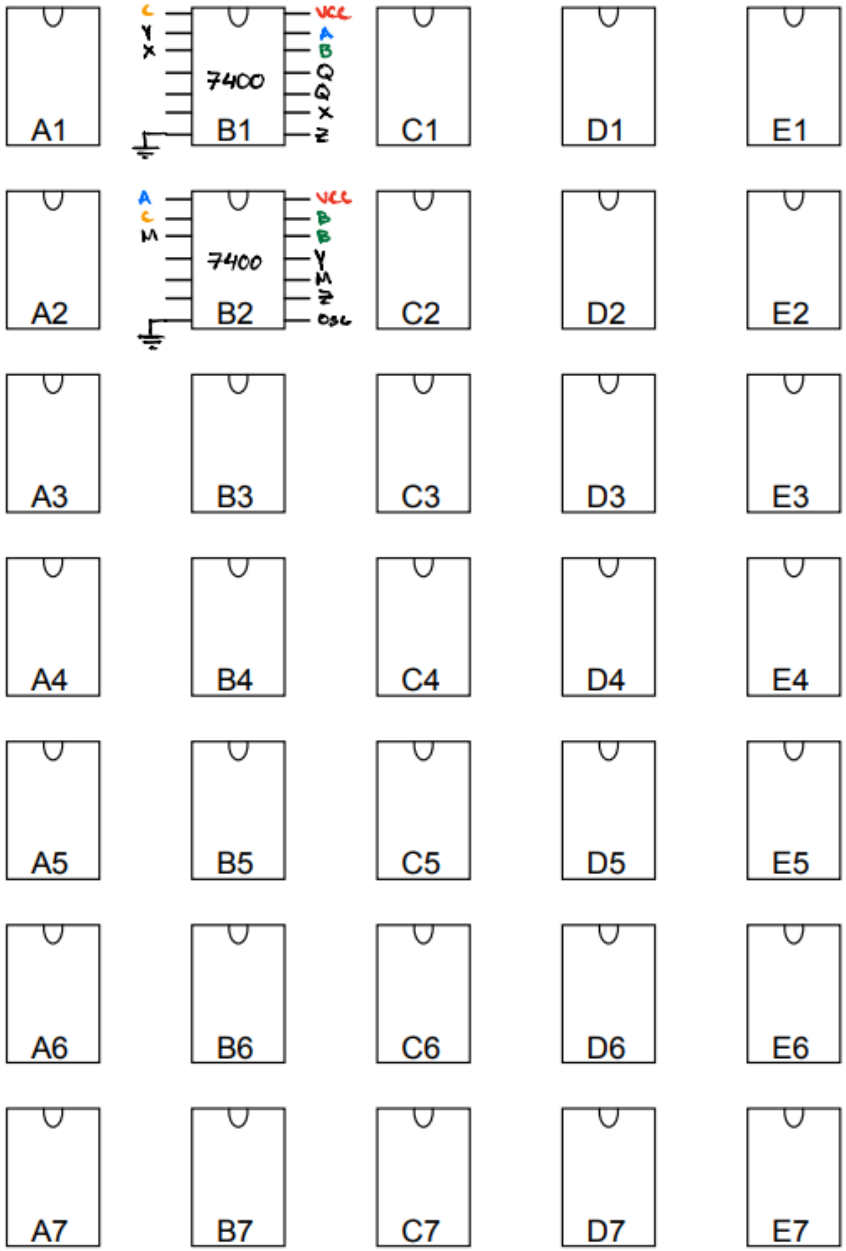
This circuit represents the glitch-less circuit. It is essentially the same circuit as the first one, however, the addition of the extra NAND gate at the bottom with signals A and C allows the NAND gate which results in output Z to remain at a high signal when signal B is oscillating. This ensures that the glitch does not occur as the extra part added travels at the same speed as the top part and keeps the signal at a high before the middle part of the circuit is able to reach the rightmost NAND gate.

Answers to Pre-Lab Questions:

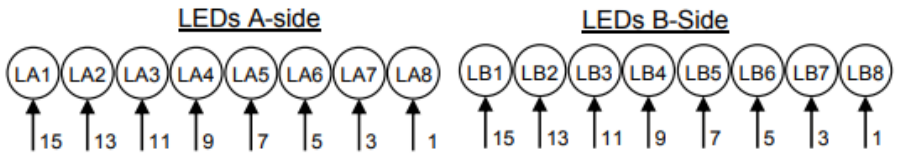
Regarding static hazards, not all groups or users will experience these glitches. This is because some chips function very well and have little to no propagation delays. This allows the input to the gates and the outputs of the gates to travel and be computed almost instantly. If the chip performs well and there is no delay, one can still recreate the glitch by chaining together multiple inverters. By adding extra gates, the pulse has to travel to each and every gate and the logic on the inputs has to be performed. This increases the propagation delay time and will make it easier for users to identify the glitch as the glitch will last for a few more nanoseconds.

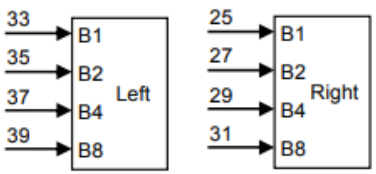
COMPONENT LAYOUT AND I/O ASSIGNMENT

PROTOBOARD

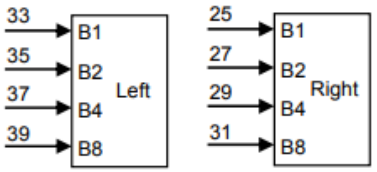


16-bit I/O BOARD

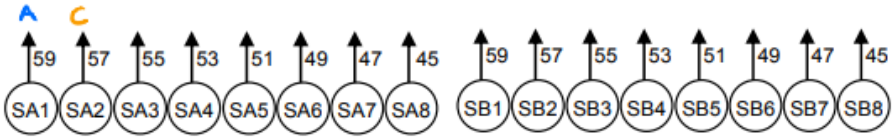




HEX A-side



HEX B-side

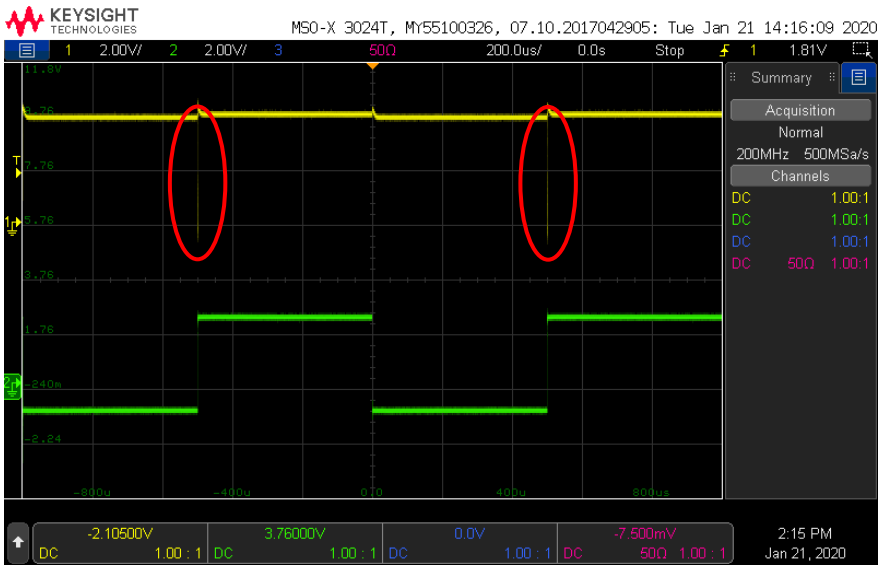


SWITCHES A-side

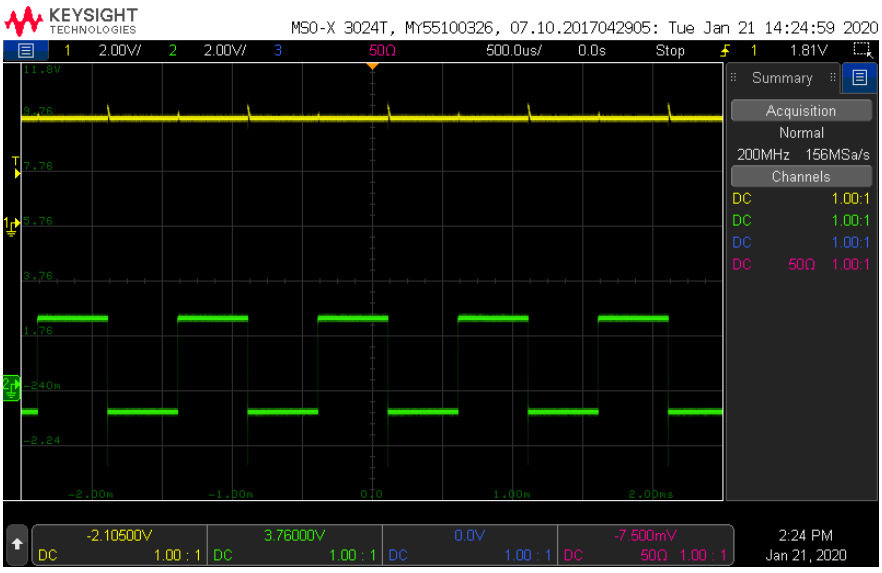
SWITCHES B-side

Lab Procedure and Documentation:

To complete the lab and remove the glitch, we first created the glitch circuit using the method and concepts as described above. Afterwards, we set signals A and C to a high signal and drove signal B with a 1 Mhz, 5 volts peak-to-peak, square wave. When we measured output Z with the oscilloscope, we noticed that whenever a rising edge occurred on signal B, the output would momentarily drop from a high signal to a low signal. This should not occur as the correct output should remain a high signal no matter the value of signal B is when A and C are both set to high.



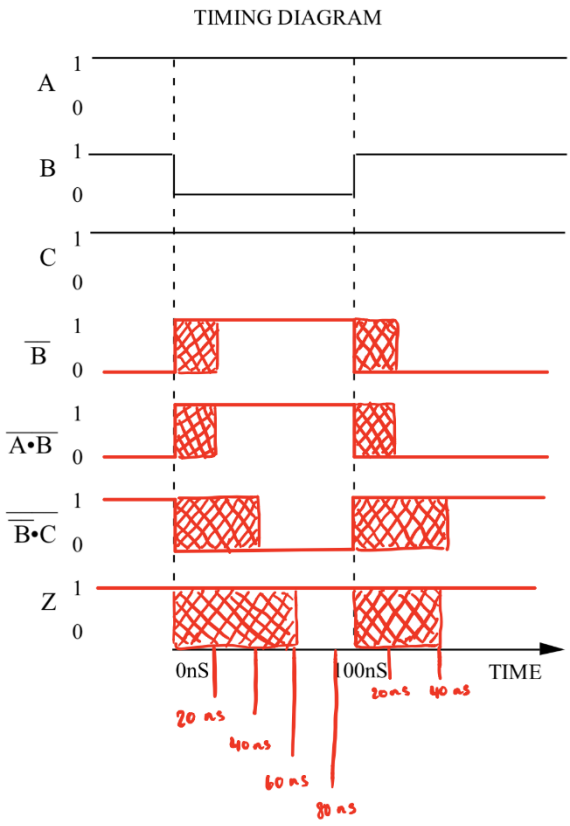
The oscilloscope reading above demonstrates the Static-1 Hazard as you can see the false low signal when signal B rises from a low signal to a high signal. To fix this error, we added the adjacent minterms on the K-map in the form of NAND gates. After implementing this, we noticed that our Static-1 Hazard disappeared. This is due to the fact that the extra gates and expression ensures that the output remains at a high while the other parts of the circuit are still passing electricity through the logic gates and have yet to reach the output.



Answers to Lab Questions:

Upon completing the two circuits that were required, we compared the truth tables of both circuits which can be found above. These tables show that the two circuits essentially operate the same as we are looking at the ideal output. However, when hooking up input B to the function generator instead of the switch and observing the outputs on the oscilloscope rather than creating a truth table, we can see that the results differ a little. Essentially the output after a 'long' time, in our case a long time is roughly a second, for both circuits is the same. However, due to the rising and falling edges of the pulse generator, there is a Static-1 Hazard that occurs for a split-second. Regarding this glitch, we are more likely to observe this glitch at the rising edge of input B. Since the bottom half of the circuit has an extra gate to pass through, when signal B rises from a low to high, it will take more time for that high signal to reach the output compared to the top portion of the circuit which will reach the output, with its low signal, first.

Answers to Post-Lab Questions:



After the falling edge of input B occurs, it takes approximately 60ns for output Z to stabilize. This is because the maximum propagation delay for one 7400 chip is 20ns, and since we have three, the maximum delay that we must account for will be 60ns. Regarding the rising edge of input B, it takes approximately 40ns in order for output Z to stabilize because after 40ns the value of AB' will be a 0 and any signal NANDed with a 0 will always be 1. There are potential glitches in output Z regarding this timing diagram. This is mostly due to the fact that the propagation delay of the chips is not always 20ns. Sometimes the delay could be more and sometimes it could be virtually nonexistent. The areas that are shaded represent areas where the signals could be indeterminate as they are still switching and could this cause a glitch in the circuit.

Regarding switches, many switches suffer from mechanical contact bounces. This means that when a switch or button is flipped or pressed, the two metal contact points connect and disconnect several times before the connection is stable. A debouncer circuit eliminates this issue by having the signal bounce between two gates until an input signal is received which causes the latch to sustain that input until another signal is received.

Answers to General Guide Questions:

GG.6 Question:

Noise immunity of a gate is recognized as the maximum amplitude of a pulse in the opposite direction than the voltage signal. The advantage of a larger noise immunity is that the incoming pulse, or noise, can have a greater amplitude and not disrupt this signal, so the chip can register the signal as a 1 or a 0. This is useful as it allows the signals and chips to behave as normal despite non-ideal scenarios. When measuring noise immunity, the last inverter is observed rather than the first as each time the signal passes through an inverter, the signal gains more noise. Thus, by observing the last inverter, we can deduce what the noise immunity of the pulse is as it will represent the greatest amount of noise which can be added without affecting the output. Given a graph of VOUT vs VIN for an inverter, we can determine the noise immunity of the inverter by specifying the minimum value of the ranges of VIN and VOUT. These ranges represent the nominal logic "0" or "1" and can be calculated by taking the voltage line and splitting it in half and determining the smallest value of that line.

GG.29 Question:

LEDs can be used to display the output of various signals. In order to use an LED, we need to ensure that we wire it up correctly, and limit the current that goes into it. If too high of a current goes into the LED circuit, it could potentially burn the circuit. Thus, it is important we take precautions to prevent this from happening. One such way to ensure this does not happen is to use individual resistors for each LED when monitoring signals. It is bad practice to share resistors because resistors do not act ideally. If one were to connect several LEDs to a single resistor, the resistor would unequally divide the current among the LEDs. Due to this, some LEDs could receive a current which is greater than what it can handle and could cause it to burn out while other LEDs may not receive the minimum current required to power it up or its brightness will be too low to notice. Generally, it is important that each LED has its own resistor to monitor the current intake.

Conclusion:

Upon completing this lab, I believe that I have learnt many valuable lessons. One lesson is that mathematical computations and theoretical concepts may not apply to full extent in the real world. This is due to the fact that not all hardware act as ideal. There will most likely be some discrepancies regarding how the product is manufactured and the environment in which it is used. Due to circumstances like these, we were able to produce the Static-1 Hazard glitch when building our circuit. However, initially when we attempted to produce this glitch, we could not create it due to the propagation delay of the chips being low. This led to use adding extra inverters to extend the time it takes for the signal to reach from the input to the output. After adding several inverters, we were able to see the glitch occur in the oscilloscope. In the future, it is important that we add more inverters as it increases the chances of producing the glitch. If we are attempting to fix the circuit, then we should account for all adjacent terms in the K-map as it will minimize the effect of the glitch. Overall, this lab provided us with the understanding that hardware is not always ideal, and that when creating circuits in the real world we must account for these irregularities.