ECE 385

Spring 2023

Lab 1

Experiment #1

Aleena Majeed
17:50 - 17:55
Wei Ren

Introduction

In Experiment 1, we created a 2-to-1 mux by using only NAND gates located in our ECE 385 kit. Our main objective was to eliminate the static 1 hazard that occurred in our original design of the circuit, and then rebuild mux using our solutions to fix the propagation delay in the circuit, which causes the static hazard in our original circuit.

Description of Circuit

For this lab, we were given 2 circuits to draw. Circuit A is a 2-to-1 multiplexer consisting of inputs A, B (which is a select input), and C and an output Z. We are able to design this circuit using 4 NAND gates, however we know that this circuit will cause a static-1 hazard. Firstly, we can create circuit A by finding boolean expressions for a 2-to-1 multiplexer.

A	B	C	Z
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Z

A/BC	0	1	11	10
0	0	1	0	0
1	0	1	1	1

With the help of a truth table and using De Morgan's law, we can write the boolean expression as:

$$z = B'C + AB$$

This allows us to use NAND gates to implement our 2-to-1 mux, as per our lab kit we only have access to a 7400 chip, which consists of 4 NAND gates.

However, circuit A has a glitch. This glitch is caused by the nonzero propagation delay from the inverter that makes B a select input. The propagation delay of this inverter will cause a glitch from the fact that the input B and the output have different propagation delays due to each input going through a different number of gates, so we cannot fully determine what the output will be until both the delays (of the inverter and the output) are accounted for.

Thus, we were asked to create circuit B, which is a redesign of circuit A, but without the static hazard. This is done by adding an extra overlapping term to our boolean expressions, and we obtain this term through the K-map that we made for the original 2-to-1 mux design. We originally did not add this term, as it would not allow for the minimal SOP of our K-map.

A	B	C	Z
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Truth table for a 2-to-1 mux

Z

A/BC	0	1	11	10
0	0	1	0	0
1	0	1	1	1

K-map for 2-to-1 mux (fixed the static hazard)

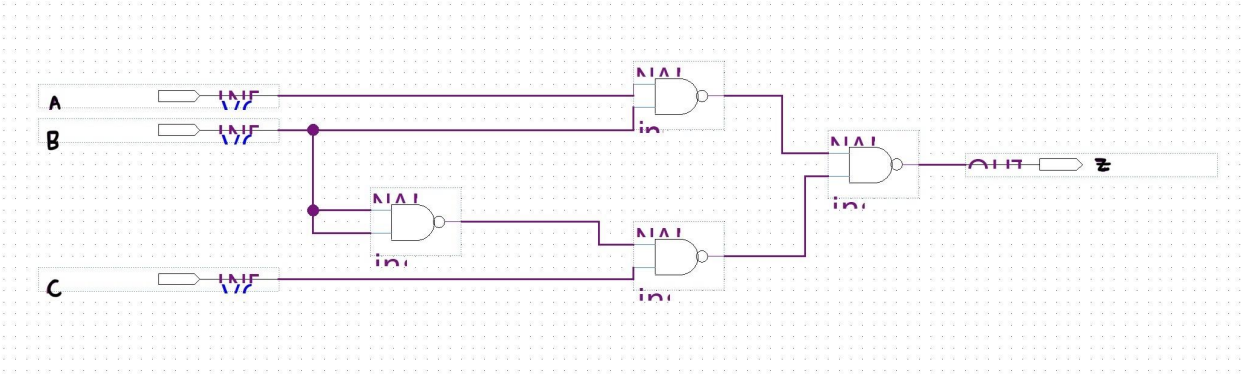
With the addition of the extra prime implicant, our new Boolean expression looks like this:

$$z = B'C + AB + AC$$

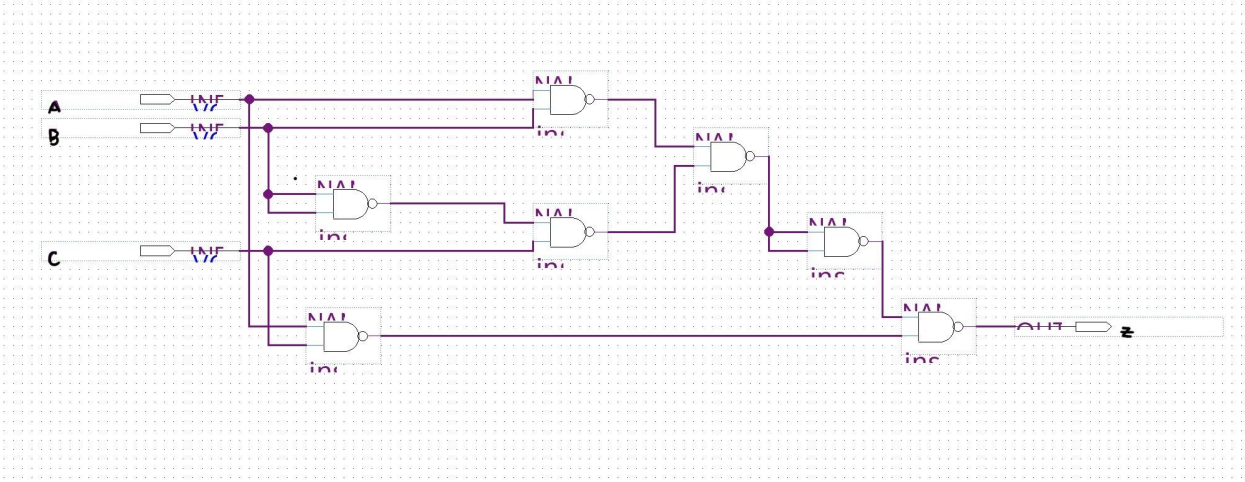
Now, if we now treat this term as one in our circuit B, we can remove the glitch because with this extra term, inputs A, B and C will all be traveling through the same number of gates (each initially entering 2 different NAND gates).

Logic Diagram - Quartus

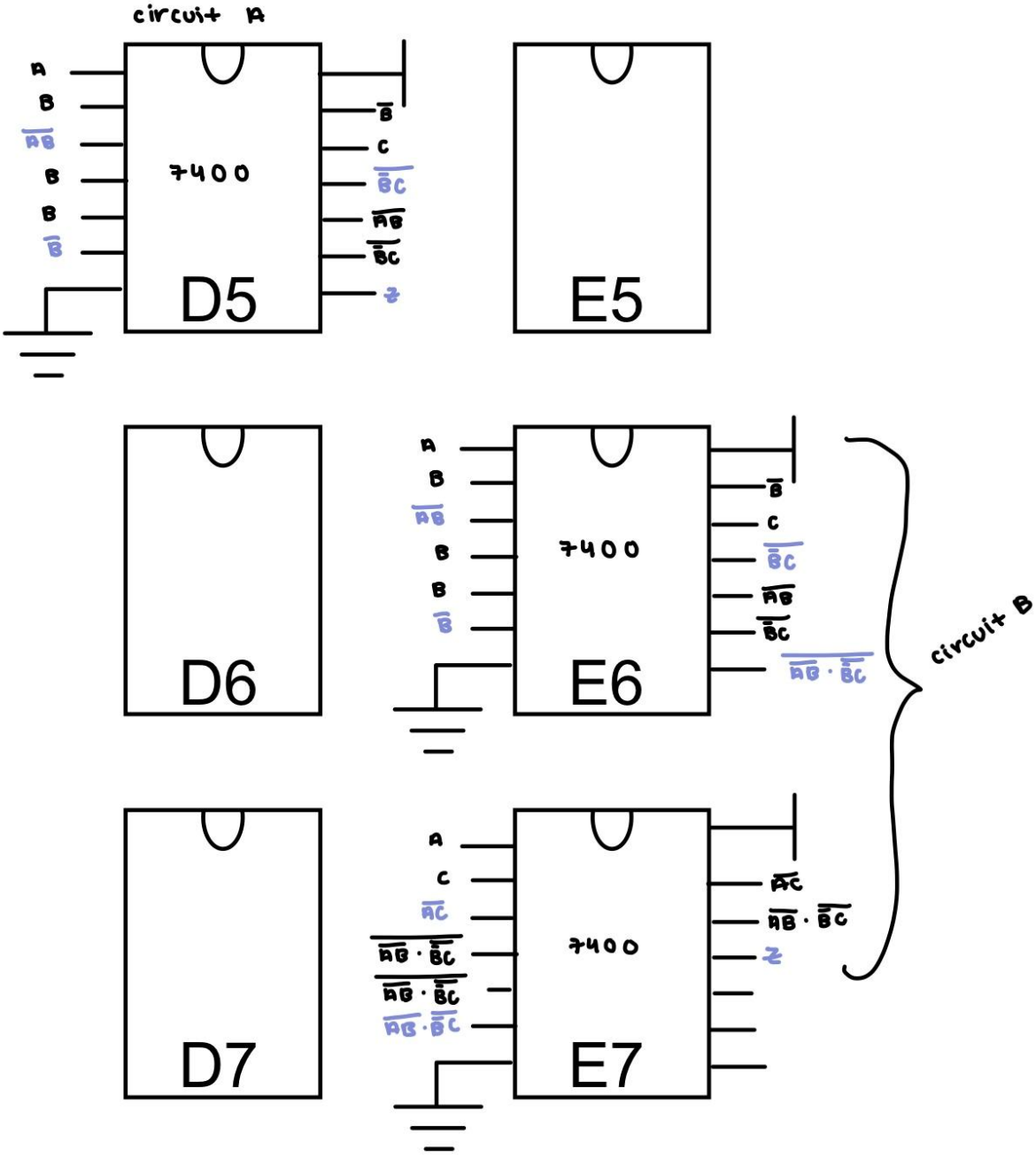
Circuit A:



Circuit B:



Component Layout



Answers to Pre-Lab

Why might not all groups observe static hazards?

When building circuit A, it is possible that not all groups will see a static hazard despite all having the same circuit. This can occur because the propagation delay from the inverter for the input B may not always be visible in the output of an oscilloscope. With our circuit A, if we have all inputs, A, B, and C, set to 1, our

output Z should be 1 as well. However, the propagation delay occurs when we change the input B because it's a select input and must pass through an extra gate. So, if we keep A and C at 1, and change B to a 0, there will be a moment in time where the output displays a 0, instead of a 1 like it's supposed to. This is because A and B are being NANDed together, and the previous value of B' and C will be NANDed together before the new value of B is able to pass through the entire circuit and output the correct value.

But, when B transitions from low to high, the glitch is not noticeable because the output of the circuit, Z, does not have a momentarily incorrect output value: the output stays the same even though there is a propagation delay that could cause the output to temporarily be an unexpected value. Therefore, on input B's rising edge, we cannot see the static hazard in the output of our circuit for the reason that the output value before and after the delay is 0.

Why does the hazard appear when you chain an odd number of inverters together in place of a single inverter or when you add a small valued capacitor to the output of the inverter?

This can occur because it is possible that the glitch is somewhat buried inside the noise that is shown on the oscilloscope. Despite this setback, there are solutions to make the glitch visible in the oscilloscope. One way is to add a small value capacitor to the output of the inverter of the circuit. This will cause a delay in the glitch of the circuit due to the increase in the RC time constant of the overall circuit, where our time constant, , is directly proportional to the capacitance. This time constant tells us how long it will take for the capacitor to charge and discharge, and this time is what will cause a delay in the glitch for it to become visible.

Another solution to this problem is to add an odd number of inverters in place of the single inverter. This is a solution because adding only an odd number allows for the circuit to still work correctly, still allows the circuit to invert the input, whereas an even number would just give the input right back. Also, adding more inverters will cause an even larger propagation delay, which will cause a larger delay because each inverter itself has a propagation delay.

Answers to Lab

Does the circuit from part B respond like the circuit from part A?

The circuit for part B did not respond like the circuit from part A because we were able to get rid of the static hazard that came from the propagation delay. This means that the output of the circuit for part A was always what we expected it to be, whereas the circuit for part B had a temporarily unexpected output.

Describe and save the output and explain any differences between it and the results obtained in part 2.

As we can see below, circuit A has a static hazard occurring at the output, which we can see from the large dip in the purple graph, displaying how the output of the circuit is outputting a '0' at a time when it should be showing a '1'. This hazard only occurs for a few nanoseconds (approximately 20) due to the propagation delay of the gate from the select input B.

We can also see below that circuit B gives the response that we're expecting, and we can see this by looking at the purple graph of the oscilloscope for circuit B. We can now see that there's no longer an unexpected output of 0 shown in the response,

CIRCUIT A:

CIRCUIT B:

For the circuit of part A of the pre-lab, at which edge (rising/falling) of the input B are we more likely to observe a glitch at the output?

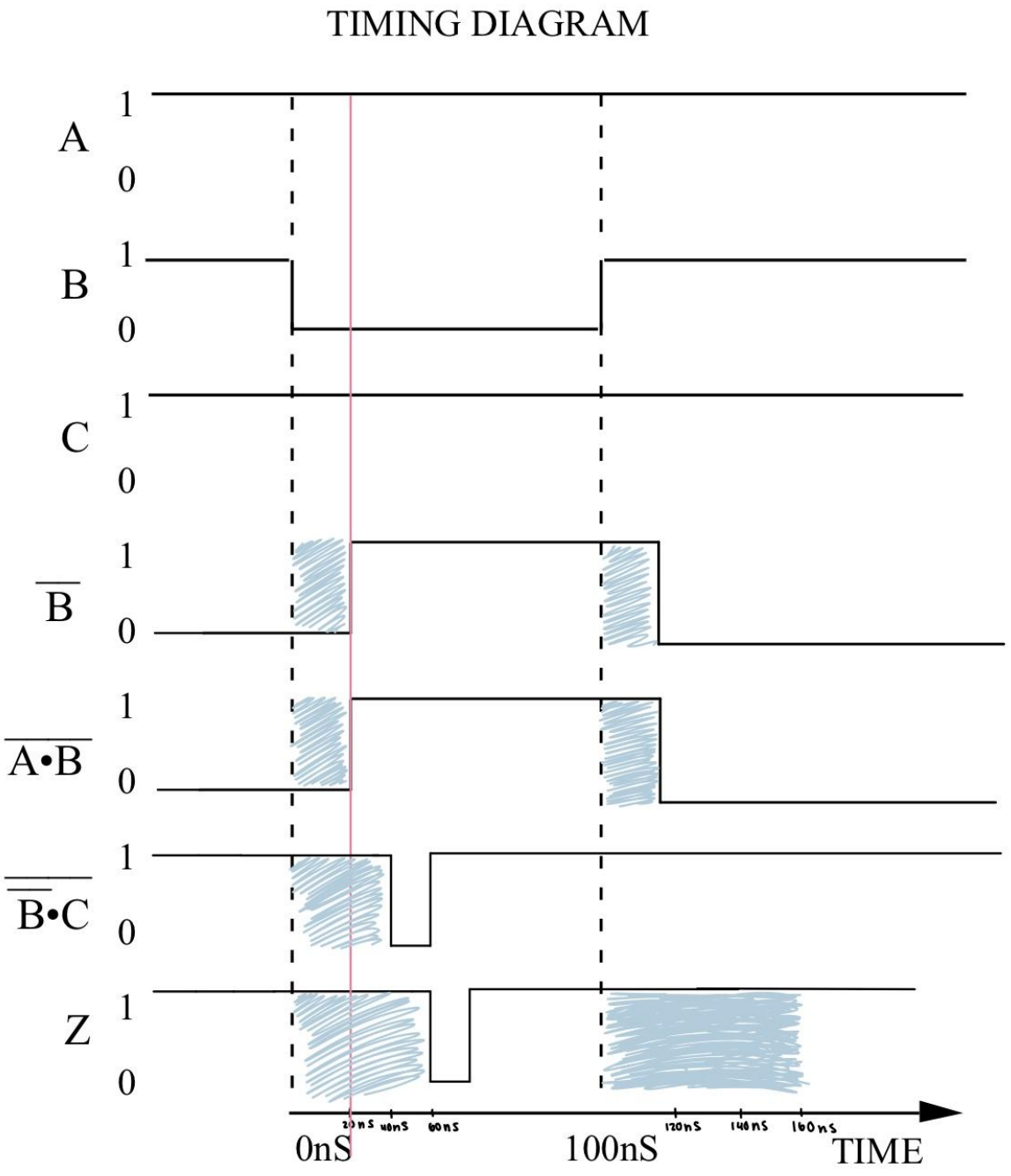
For circuit A, we are most likely to observe a glitch at the falling edge of input B because of the propagation delay. This is because when the input B is 0, the output momentarily has a delay where the output is also 0, even if it's supposed to be 1. This occurs because there is a moment when the output Z is using the B' value from the previous input, because the new input of B has a delay through the inverter so B' has not been updated yet. It is not a rising edge however because when the input changes from 0 to 1, the delay does not cause the output to temporarily be low, and thus the glitch is not noticeable.

Answers to Post-Lab

How long does it take the output Z to stabilize on the falling edge of B (in ns)? How long does it take on the rising edge (in ns)? Are there any potential glitches in the output, Z? If so, explain what makes these glitches occur.

When the input B changes from a high to low input, the output Z takes 20 nS to stabilize on this falling edge once again due to the propagation delay. This is the same for the rising edge of the input B, where it takes the output Z 20 nS to come to a stable value. A glitch will be visibly present in the output of Z during the falling edge of B, due to the fact that the B input has a propagation delay from an inverter, and this delay will cause the Z output to temporarily be an incorrect

value based on the actual input of B. However, with the rising edge, there is a delay as well; however, it is not visibility noticeable because as B transitions from low to high, the output Z is not affected by the propagation delay: the output stays the same.



Explain how and why the debouncer circuit given in the General Guide Figure 17 (GG.32) works. Specifically, what makes it behave like a switch and how the ill effect of the mechanical contact bounces is eliminated?

The debouncing circuit works by not allowing either of the inputs to fluctuate at the same time. The switch will have one input flipping back and forth at a time, and this is to ensure that the output is not changing due to both inputs at the same time. The debouncing circuit below behaves like a switch and eliminates the ill effect of mechanical contact bounces because it allows for a smooth transition between logic high and logic lows, which means that there is no bouncing between the two states. The use of the pull up resistors also ensures that each input always has a value and that none are left floating. The circuit below works like a switch because it is an SR latch. This means that when the S states are enabled we are setting the circuit, and when the R states are enabled then the circuit is being reset and therefore, this circuit mimics a switch.

General Guide #6

What's the advantage of larger noise immunity?

Noise immunity is having a large threshold between '0' and '1' input levels: when the input level is low, we have a large and positive threshold, whereas if the level is high then we have a large negative threshold. This is desirable because if there is a large threshold at the input levels, then small amounts of noise aren't able to have an effect on the output. Therefore, large amounts of noise would be required in order for the input to be disturbed and thus, disturbing the output as well.

Why is the last inverter observed rather than simply the first?

The last inverter is observed rather than the first because going through multiple inverters will create more noise. Therefore, by observing the output of the last inverter, we can more easily tell if there is noise immunity or not because by seeing if any of the noise that we expect to see in the output is actually negated by the immunity.

Given a graph of output voltage (VOUT) vs. input voltage (VIN) for an inverter, how would you calculate the noise immunity for the inverter?

Given a graph of Vout and Vin, the way we could calculate the threshold is by

finding the difference between the maximum and minimum values of Vin at both the high and low values, and doing the same process for Vout. This will provide the threshold range. In order to find the noise immunity, we want to take the lesser of the 2 values to find the immunity for each Vin and Vout.

General Guide #31

If we have two or more LEDs to monitor several signals, why is it bad practice to share resistors?

It is poor practice for 2 LEDs to share the same resistor when monitoring signals because it is possible that we could burn out an LED with too much current. Since LEDs have different turn on voltages, the two LEDs require different amounts of voltage in order to turn on, and according to Ohm's law, V = IR, the LEDs will also have differing values of current running through them as well. If LED 1 has a high turn on voltage and LED 2 has a low turn on voltage in a parallel configuration where voltages across both LEDs should be equivalent, LED 2 will have too large of a voltage and thus will have too much current going through it due to the higher turn on voltage of LED 1. This disparity in turn on voltages across the LEDs can cause one to ultimately burn out.

Conclusions

In the end of the experiment, we were able to conclude circuit A contained a static hazard due to propagation delays. However, we were also able to find a solution to the static hazard by rewriting the boolean expressions for circuit A, by including all adjacent 1 terms as prime implicants. In doing this, we were able to abolish the static hazard because this method ultimately allowed for each input to go through the same number of gates before reaching the output gate and thus, eliminating the issue of a propagation delay.