ECE 385

Spring 22

Experiment #1

Introductory Experiment

Siddarth Iyer
TA: Ruihao Yao (RY)

Purpose of Circuit

In Experiment #1, we design two 2-to-1 Multiplexer circuits using NAND Gates. The objective of this lab is to demonstrate and understand a glitch occurring in the first implementation of the circuit, and modify the circuit by adding additional terms in order to remove the static-1 Hazard and prevent the glitch from occurring.

Written Description of Circuit

Part A: Naive Implementation (Includes Glitches)

We initially create a naive 2-to-1 MUX. The 2-to-1 MUX takes in A and C as inputs, and uses B as a select bit to decide which input is outputted to output Z. Therefore if B is 0, the output is C, and if B is 1, the output is A. Using this, we create a truth table, draw out the K-Map and derive the minimal SOP expression.

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

	B'C'	B'C	BC	BC'
A'	0	1	0	0
A	0	1	1	1

Minimal SOP Expression: Z = AB + B'C

Fig 1: Truth Table, K-Map, SOP for Naive MUX implementation

Based on the expression, we then create the NAND implementation of the expression in order to optimize the circuit. We use NAND since it is a universal gate, and cuts down on the chips required from 3 to 1 since one 7400 chip can be used to implement it.

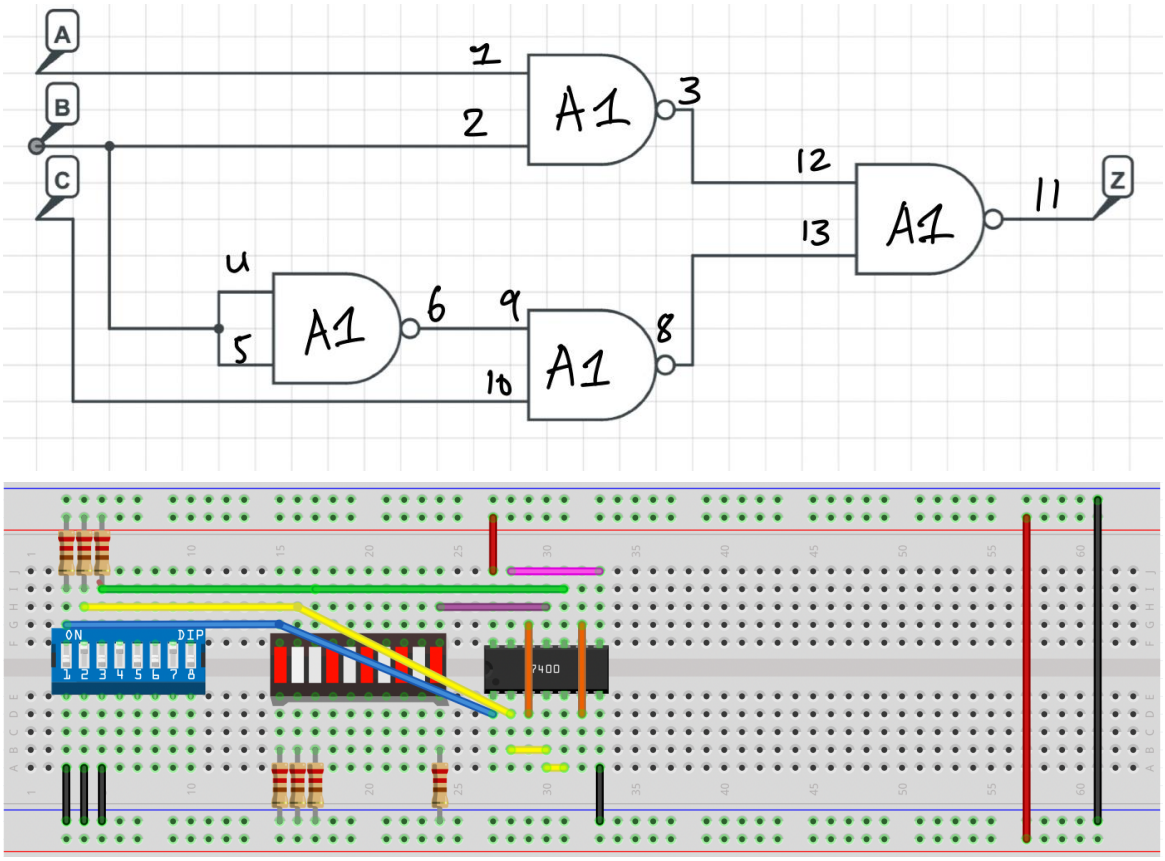


Fig 2: Circuit Diagrams for Naive implementation

Part B: Redundant Implementation(Glitch Free)

While testing the circuit, it is observed that when certain inputs are changed, a glitch occurs. This is due to gate delays, which are not accounted for while deriving the expression. For example, while transitioning from ABC=111 to 101, though the output Z should be 1 in both cases, it momentarily goes to 0 due to gate delays before switching back to 1. In order to avoid this, we re-analyze the k-map and add an additional term AC to remove the glitch.

	B'C'	B'C	BC	BC'
A'	0	1	0	0
A	0	1	1	1

Revised Expression for Z: $Z = AB + B'C + AC$

Fig 3: Revised K-Map and SOP Expression

Based on this new expression, we modify the circuit from part A to account for the 'AC' term.

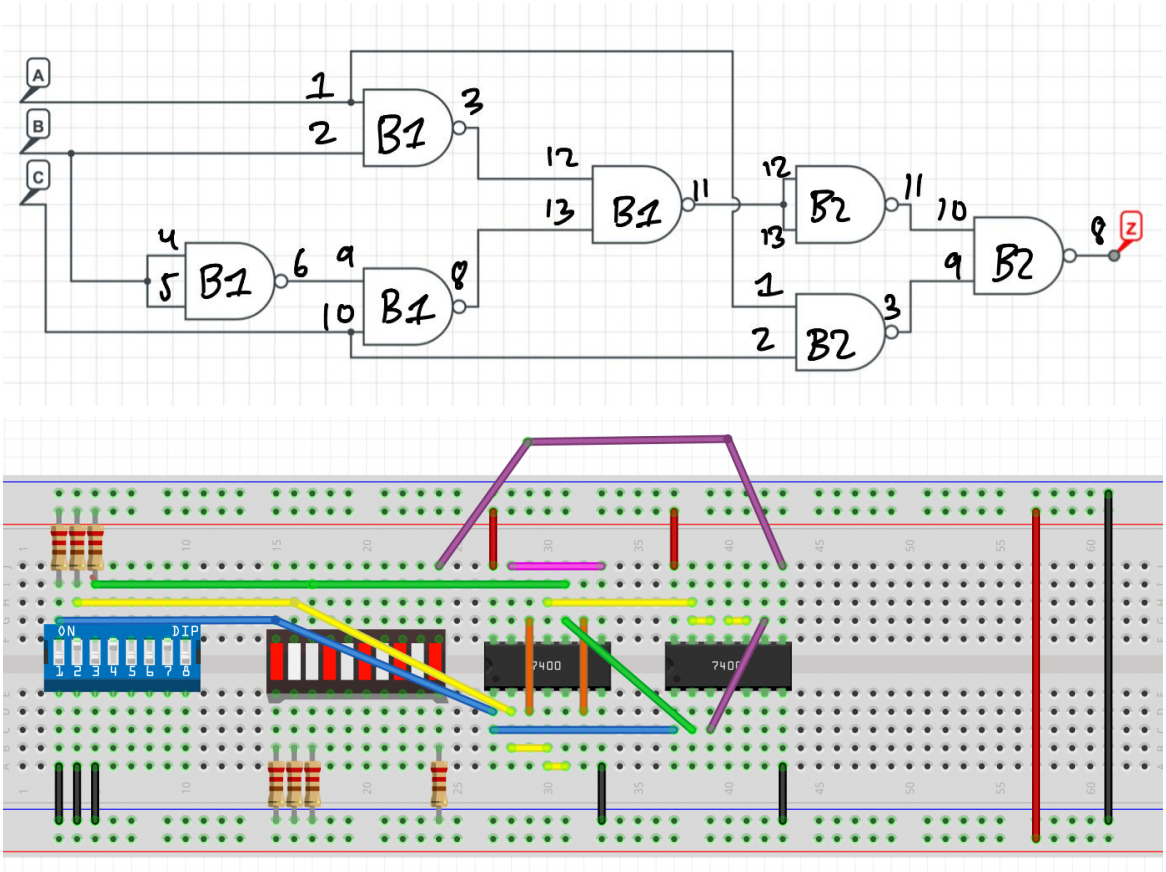


Fig 4: Circuit Diagrams for redundant implementation

Documentation:

Pre-Lab Questions

a) Not all groups may observe static hazards (why?) If you do not observe a static hazard, chain an odd number of inverters together in place of the single inverter from Figure 16 or add a small capacitor to the output of the inverter until you observe a glitch. Why does the hazard appear when you do this?

Ans: Static hazards typically occur as a result of gate delays. These delays are in the range of 0 to 20 ns. Since the circuit is relatively simple and uses few gates, the gate delay is negligible and the static hazard may not be observed. By adding an odd number of

inverters, or a small capacitor, this delay is amplified, and the glitch becomes apparent due to the larger delay.

Lab Questions

a) Complete a truth table of the output. Does it respond like the circuit of part A?

Ans: The truth tables for circuits A and B are the same. However, the expression for the output of B has an additional term to remove glitches. The functionality is the same except for circuit A has glitches.

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

Fig 5: Truth table for Redundant Implementation

b) *Describe and save the output and explain any differences between it and the results obtained in part 2*

Ans: From the first oscilloscope reading corresponding to part A, as B (yellow) transitions from 1 to 0, the output (purple) momentarily falls to 0 before rising again to 1. In the second oscilloscope reading, as B falls to 0, the output remains fairly steady and holds the constant value 1. In the first case, a glitch has occurred, while in the second case no glitch occurs. (AC=11 therefore output should remain 1)

Fig 6: Oscilloscope readings for Circuit 1 and 2 respectively

c) *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?*

Ans: A glitch is more likely to be observed during the falling edge of input B. For AC=11 when B switches from 1 to 0, due to the gate delay in the inverter, the inputs of the final NAND are momentarily both 1 making the output 0 for a short duration when it should've been 1. On the rising edge, however, at least one input to the final NAND gate is 0, the output remains 1, and no glitch occurs.

Post-lab Questions

a) Given that the guaranteed minimum propagation delay of a 7400 is 0ns and that its guaranteed maximum delay time is 20ns, complete the timing diagram below for the circuit of part A

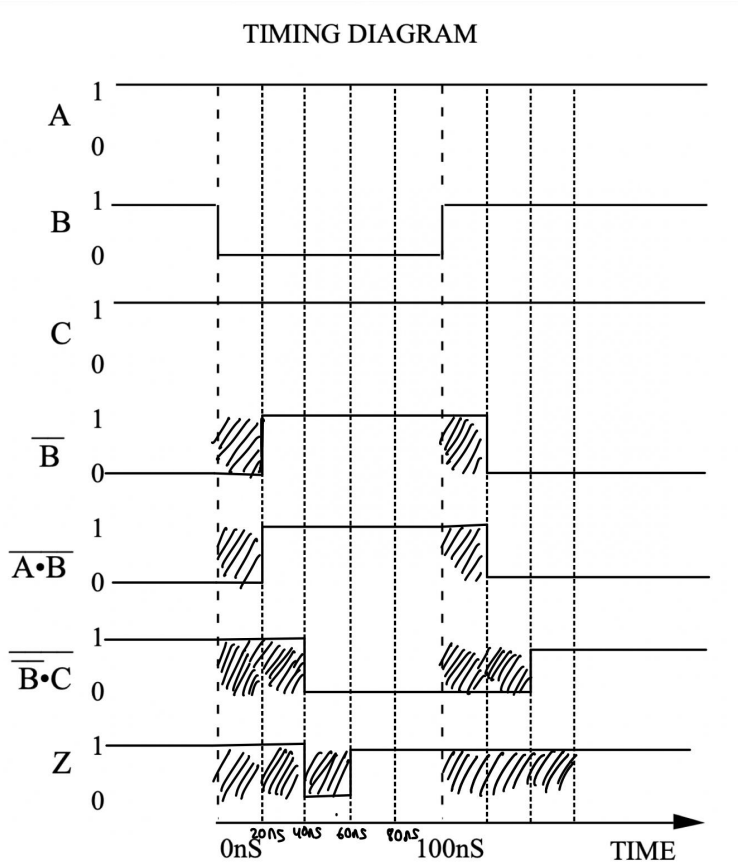


Fig 7: Timing Diagram

b) How long does it take the output Z to stabilize on the falling edge of B (in ns)?

Ans: 60 ns (0-60 ns)

c) How long does it take on the rising edge (in ns)?

Ans: 60 ns (100-160 ns)

d) Are there any potential glitches in the output, Z? If so, explain what makes these glitches occur.

Ans: A glitch occurs in Z between 40 and 60 ns. This is due to the time delay while inverting input B.

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

Ans: The debouncer circuit is used to avoid contact bounce and allow a smooth transition from 0 to 1. It consists of an SPDT switch, with either toggle connected to pull up resistors, and either input of S'-R' latch. Either input of the latch will have logical 1 or 0 as its input. When the switch is in position B, QN will always be 1, and Q will be 0. Now, when the switch toggles from B to A, during the short duration when the switch is being toggled, the latch will hold the value and Q will still remain 0. Once the switch is at position A, Q will become 1, and QN will become 0, and the latch stores and holds these values. In this way, the circuit acts as a switch outputting either 0 or 1, and since the latch holds the value, the ill effect of contact bounce has been eliminated.

General Guide Questions

a) What is the advantage of a larger noise immunity?

Ans: By having a larger noise immunity, the circuit becomes more stable, since logic levels will not fluctuate as a result of external noise.

b) Why is the last inverter observed rather than simply the first? Given a graph of output voltage (VOUT) vs. input voltage (VIN) for an inverter, how would you calculate the noise immunity for the inverter?

Ans: After each inversion, the voltage level of the inverter keeps decreasing and therefore the last inverter will have the lowest voltage level. To calculate the noise immunity, we

check the range for Vout which stays consistent for a range of Vin.

For logical 0: the range of Vin for which output of 1 is stable is 0.35 to 1.15.

Noise immunity of logic 0 = 1.15-0.35 = 0.8 V

For logical 1: the range of Vin for which output of 0 is stable is 1.35 to 3.5.

Noise immunity of logic 1 = 3.5-1.35 = 2.15V

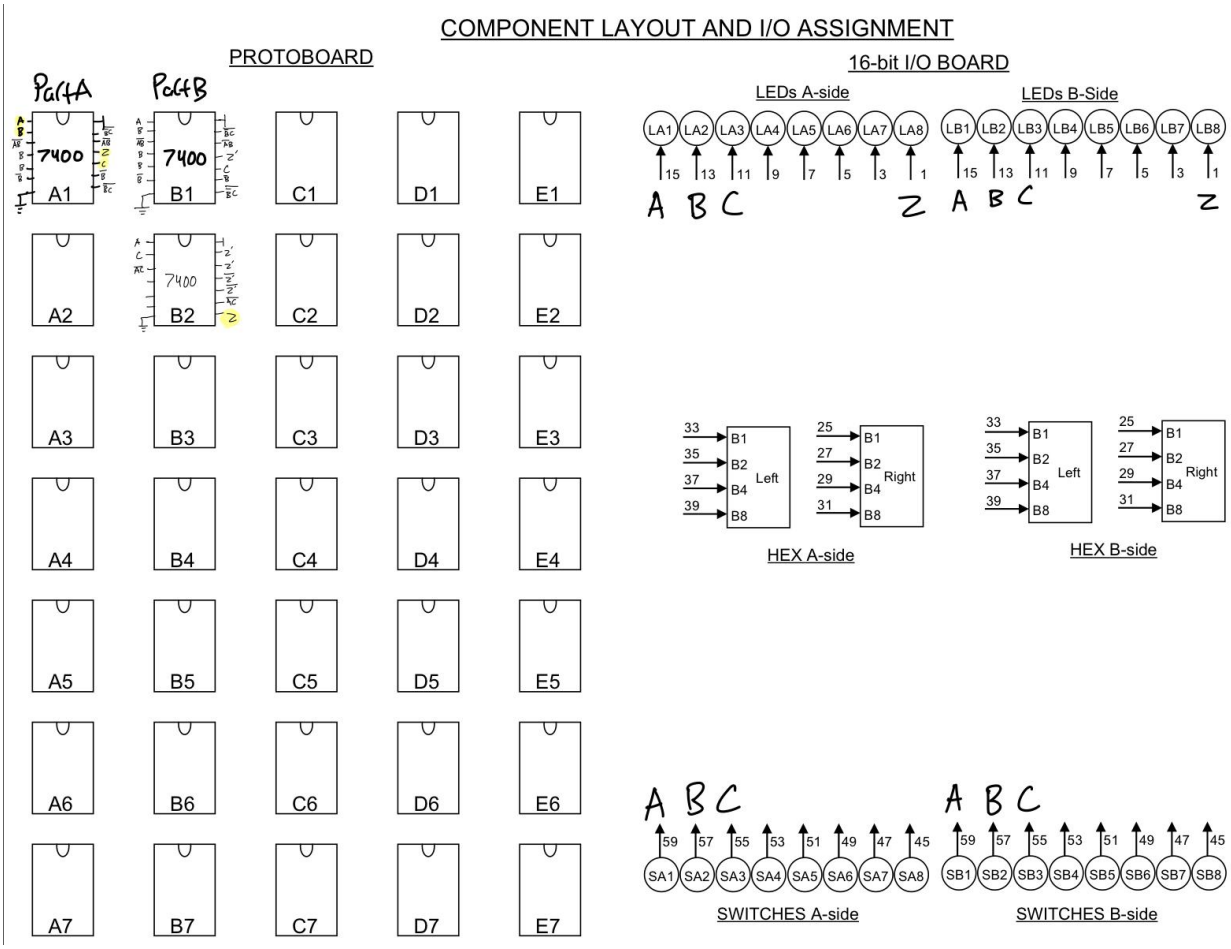
The noise immunity of the chip is the lower of these values, so the answer is 0.8 V.

c) Why is a capacitor necessary close to each chip?

Ans: The capacitor acts as a decoupling capacitor, and provides power to the CMOS chip, preventing it from temporarily short-circuiting when the CMOS chip transitions from low to high or high to low, and hence reduces the overall noise in the circuit

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

Ans: If LEDs share resistors, they are connected in parallel. Since each LED may have different characteristics and properties, the current flowing through each would be irregular, and the LED may exceed its power rating causing it to burn out.



Conclusion

This lab introduces and reinforces the concept of static-1 hazards and glitches when building circuits. It builds on the concepts learned from ECE 120 and magnifies issues such as glitches which were overlooked but may have real-world consequences if not dealt with and fixed. When creating a simple 2-to-1 mux we faced glitches, which were then fixed by adding additional minterms to eliminate hazards and glitches.