Lab Report IV: Digital Logic

Enrique Rivera
Lab Partner: Marcus

November 15, 2024

Lab 10: Digital Circuits I

Objective

The objective of Lab 10 was to familiarize students with basic logic gates and combinational circuits. The main focus was on building simple logic circuits using AND, OR, and NAND gates and verifying their truth tables through practical implementation.

Procedure - Question 1

In this part of the lab, we used both TTL (7400N or 74LS00N) and CMOS (74HCT00N) ICs to implement individual logic gates. Each IC contains multiple NAND gates that can be configured into other basic gates (AND and OR) by connecting the gates in specific ways.

1. Power Supply Setup: - We set the power supply (Vcc) to 5.04V, which is within the recommended operating range for both TTL and CMOS ICs.
- Ground (GND) was connected to the ground terminal on the breadboard, and Vcc was applied to the power pins on the ICs.

2. Connecting the ICs: - For each IC (TTL and CMOS), we connected Vcc (5.04V) to the designated power pin and ground to the ground pin of the ICs.
- All unused inputs were grounded to prevent floating states, as floating inputs can lead to unpredictable behavior in CMOS and TTL logic gates.

3. Setting Up Inputs with SPDT Switches: - We used SPDT (Single Pole Double Throw) switches on the breadboard to control the inputs for each gate. These switches allowed us to easily switch between 0 (ground) and 1 (Vcc) for each input, enabling us to test all possible input combinations.

4. Constructing NAND Gate Circuit: - To test the NAND gate, we used one of the gates in the 7400N and 74HCT00N ICs. The input pins were connected to the SPDT switches for A and B, allowing us to manually control

the input combinations (0 or 1).
- LEDs were connected to the output pin through a current-limiting resistor to visualize the output. A high output (1) would turn the LED on, while a low output (0) would turn the LED off.

5. Testing Procedure: - We applied all possible combinations of inputs (A and B) using the SPDT switches to observe the output.
- For each combination, we recorded the LED state and documented whether the output matched the expected truth table values.

6. Verification with Multiple Gates: - We repeated the procedure for additional gates on each IC to verify consistency in behavior. This was done by connecting different gate sections of the same IC, setting up inputs in a similar fashion, and observing the output with the same method.
- We compared results between the TTL (7400N) and CMOS (74HCT00N) gates to observe any differences in behavior or performance.

Results - Question 1

The following table and graph illustrate the expected truth table for the NAND gate tested in this question:

Table 1: Truth Table for NAND Gate

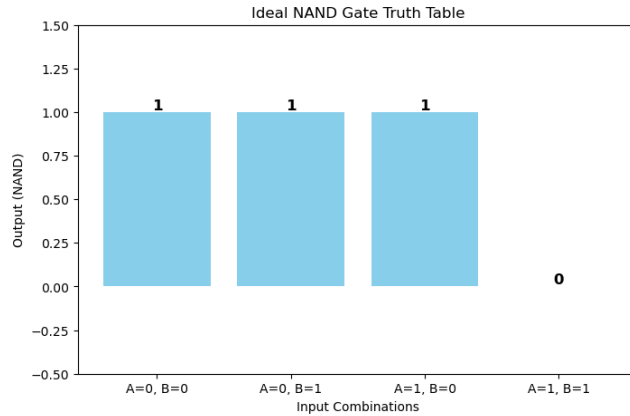| Input A	Input B	NAND Output
0	0	1
0	1	1
1	0	1
1	1	0

Figure 1: Ideal NAND Gate Truth Table Visualization

Question 1 - Discussion

The results confirmed that the NAND gate produces an output of 1 for all input combinations except when both inputs are 1, where it outputs 0. This behavior was consistent across both the TTL (7400N) and CMOS (74HCT00N) ICs, demonstrating their reliability in performing NAND operations. The truth table and visualization in Figure 1 match the theoretical expectations, validating the functionality of the NAND gates in our circuit. Additionally, grounding all unused inputs ensured stable outputs, particularly for the CMOS IC, which is more susceptible to interference from floating inputs.

Procedure - Question 2

In Question 2, we expanded the NAND gate circuit to implement a clocked flip-flop using two additional NAND gates. This configuration allowed us to create a clocked RS (Reset-Set) flip-flop, where the output changes state only when the clock is active.

1. Circuit Setup: - We used the TTL (7400N) and CMOS (74HCT00N) ICs to build the flip-flop circuit.
- Vcc was set to 5.04V, and all unused inputs were grounded to prevent floating states.
- The circuit was connected so that two additional NAND gates were used to introduce a clock input, along with the S (Set) and R (Reset) inputs.

Procedure - Question 2

In Question 2, we used multiple NAND gates to construct an **AND gate**. By configuring NAND gates in a specific arrangement, we replicated the behavior of a standard AND gate.

1. Circuit Setup: - We used the TTL (7400N) and CMOS (74HCT00N) ICs, each containing multiple NAND gates.
- Vcc was set to 5.04V, and all unused inputs were grounded to prevent floating states.
- To create an AND gate using NAND gates, we connected two NAND gates in series:
- The inputs (A and B) were connected to the first NAND gate.
- The output of the first NAND gate was connected to both inputs of the second NAND gate (essentially inverting the output of the first NAND gate).

2. Input Control with SPDT Switches: - SPDT switches on the breadboard were used to control the A and B inputs, allowing us to toggle each input between 0 (ground) and 1 (Vcc).
- LEDs were used to indicate the output state.

3. Testing Procedure: - We applied all possible combinations of A and B inputs using the SPDT switches and observed the output.
- The output state was recorded for each combination to confirm that the NAND gate arrangement produced the expected AND gate behavior.

Results - Question 2

The table below summarizes the expected truth table for the constructed AND gate:

Table 2: Truth Table for AND Gate Constructed with NAND Gates

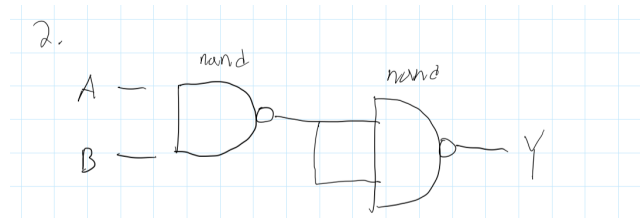| Input A	Input B	AND Output
0	0	0
0	1	0
1	0	0
1	1	1

Figure 2: NANDS Gates to Construct AND Gate

Question 2 - Discussion

The results confirmed that the arrangement of NAND gates successfully produced an AND gate. The output matched the expected truth table for an AND gate, outputting 1 only when both inputs were 1. This demonstrates that NAND gates can be used to create other basic gates, such as AND.

Procedure - Question 3

In Question 3, we used multiple NAND gates to construct an **OR gate**. This required additional configuration, as NAND gates inherently perform a different operation.

    1. Circuit Setup: - We used the TTL (7400N) and CMOS (74HCT00N) ICs to build the OR gate circuit with NAND gates.
- Vcc was set to 5.04V, and all unused inputs were grounded.
- To create an OR gate using NAND gates, we used three NAND gates as follows:
- The inputs A and B were each inverted by connecting them to separate NAND gates (where each input was connected to both pins of its NAND gate).
- The outputs of these two NAND gates were then used as inputs to a third NAND gate.

    2. Input Control with SPDT Switches: - SPDT switches were used to control the inputs A and B, allowing us to switch each input between 0 and 1.
- An LED was connected to the final output to visualize the OR gate's behavior.

    3. Testing Procedure: - We applied all possible combinations of A and B inputs using the SPDT switches and observed the output.
- The output was recorded for each combination to confirm that the circuit configuration produced the expected OR gate behavior.

Results - Question 3

The table below summarizes the expected truth table for the constructed OR gate:

Table 3: Truth Table for OR Gate Constructed with NAND Gates

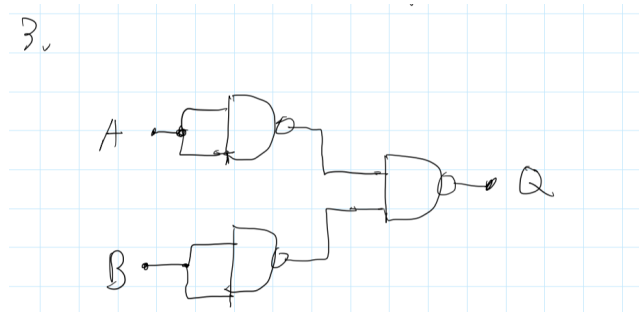| Input A	Input B	OR Output
0	0	0
0	1	1
1	0	1
1	1	1

Figure 3: NANDS Gates to Construct OR Gate

Question 3 - Discussion

The results confirmed that the configuration of NAND gates successfully produced an OR gate. The output followed the expected truth table for an OR gate, outputting 1 when either input was 1. This demonstrates that NAND gates can be used to create OR gates, providing flexibility in logic circuit design by combining basic components.

Procedure - Question 4

For Question 4, we set up the CD4007 MOS transistor array, which contains six complementary MOS transistors (three N-channel and three P-channel FETs). This IC allowed us to configure the transistors to perform basic logic operations.

   1. Component Identification: - We identified the pins on the CD4007 corresponding to the N-channel and P-channel FETs, referring to Figure 2 in the lab manual for pin layout.
- The N-channel FETs are used for low-side switching, and the P-channel FETs are used for high-side switching.

   2. Setup: - We configured the breadboard to prepare for the following exercises, ensuring that each transistor could be connected appropriately according to each circuit diagram.

Procedure - Question 5: Passive Pullup Inverter

In Question 5, we built an inverter with a passive pullup resistor as shown in Figure 3.
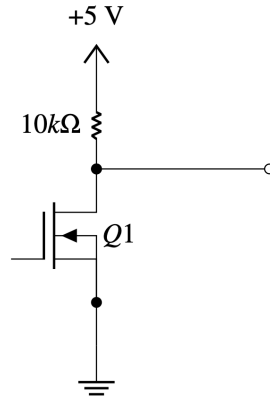
Figure 3: Inverter with passive pullup.

Figure 4: Passive Pullup Inverter Circuit

1. Circuit Construction: - We connected the TTL output of the pulse generator to the input of the inverter.
- A 1k resistor was used as a pullup, connecting the output to +5V.

2. Testing and Observations: - We confirmed that the circuit inverted the input signal, outputting a low signal when the input was high and vice versa.
- Next, we gradually increased the frequency of the input signal to determine the inverter's behavior at higher frequencies.

Results - Question 5

As we increased the frequency, we observed that the output started to degrade at high frequencies. The passive pullup could not fully charge or discharge the capacitance quickly enough, resulting in slower transitions and a less defined output.

Question 5 - Discussion

The results indicate that while the passive pullup inverter performs adequately at lower frequencies, it suffers from significant degradation at higher frequencies. This is due to the time constant associated with the resistor and the inherent capacitance, which limits the circuit's ability to respond to rapid changes in input.

Procedure - Question 6: Active Pullup Inverter

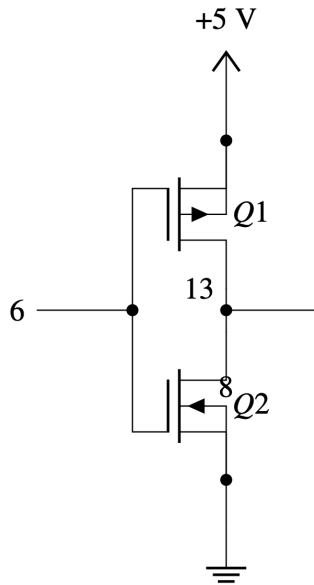In Question 6, we constructed an inverter using an active pullup configuration, as shown in Figure 4.

Figure 4: Inverter with active pullup.

Figure 5: Passive Pullup Inverter Circuit

1. Circuit Construction: - We used both an N-channel and a P-channel FET from the CD4007 to create an active pullup inverter.
- The input signal was connected to both transistors, with the P-channel FET connected to +5V and the N-channel FET connected to ground.

2. Testing and Observations: - We applied a TTL pulse from the generator to the input and observed the output on the oscilloscope.
- As in the previous question, we increased the frequency of the input signal to observe any changes in behavior.

Results - Question 6

The active pullup inverter demonstrated a more robust performance at higher frequencies compared to the passive pullup inverter. The output transitions

were sharper and more consistent, with less signal degradation observed even as the input frequency increased significantly.

Question 6 - Discussion

The active pullup inverter is more effective at handling high-frequency signals than the passive pullup configuration. The complementary configuration of N-channel and P-channel FETs allows for faster transitions, as each transistor actively pulls the output to the desired level, reducing the impact of capacitance.

Procedure - Question 7: NAND Gate

In Question 7, we used the FETs within the CD4007 to construct a NAND gate.
    1. Circuit Construction: - Two N-channel and two P-channel FETs were arranged according to the NAND gate configuration.
- The input signals were applied to the gates of the transistors, and the output was taken from the connection point between the P-channel and N-channel transistors.

    2. Testing and Observations: - We tested all input combinations (A and B) and recorded the output for each case.

Results - Question 7

The table below summarizes the expected output for the constructed NAND gate:

Table 4: Truth Table for NAND Gate

Input A	Input B	Output
0	0	1
0	1	1
1	0	1
1	1	0

Question 7 - Discussion

The constructed NAND gate behaved as expected, producing an output of 1 unless both inputs were 1, in which case the output was 0. This demonstrates that the complementary MOS transistors in the CD4007 can be used to build functional logic gates.

Lab 11: Digital Circuits II