The University of Azad Jammu and Kashmir Muzaffarabad, AJK

Name	Feeza Kulsoom
Roll No	2024-SE-03
Course	CALD
Semester	SE_2nd Semester
Lab No	03 , 04
Title	Verification of De_Morgan's Law
Submitted to	Engr. Sidra Rafique
Submission Date	3rd June ,2025

Department of Software Engineering

- **NAND Gate:**

- **NAND Gate simple circuit:**

Procedure of NAND gate simple circuit:

I opened Electronics Workbench and started a new project.

1. I selected the 7400 IC, which has NAND gates in it.
2. I connected Vcc pin 14to power and GND pin 7to ground.
3. I used the first NAND gate by connecting two inputs to pin 1 and pin 2.
4. I took the output from pin 3, which gave me the NAND result.
5. Then I added switches to test the inputs and ran the circuit.
6. The output was correct for all input combinations, just like a NAND gate truth table.
7. The circuit was simple and worked perfectly.

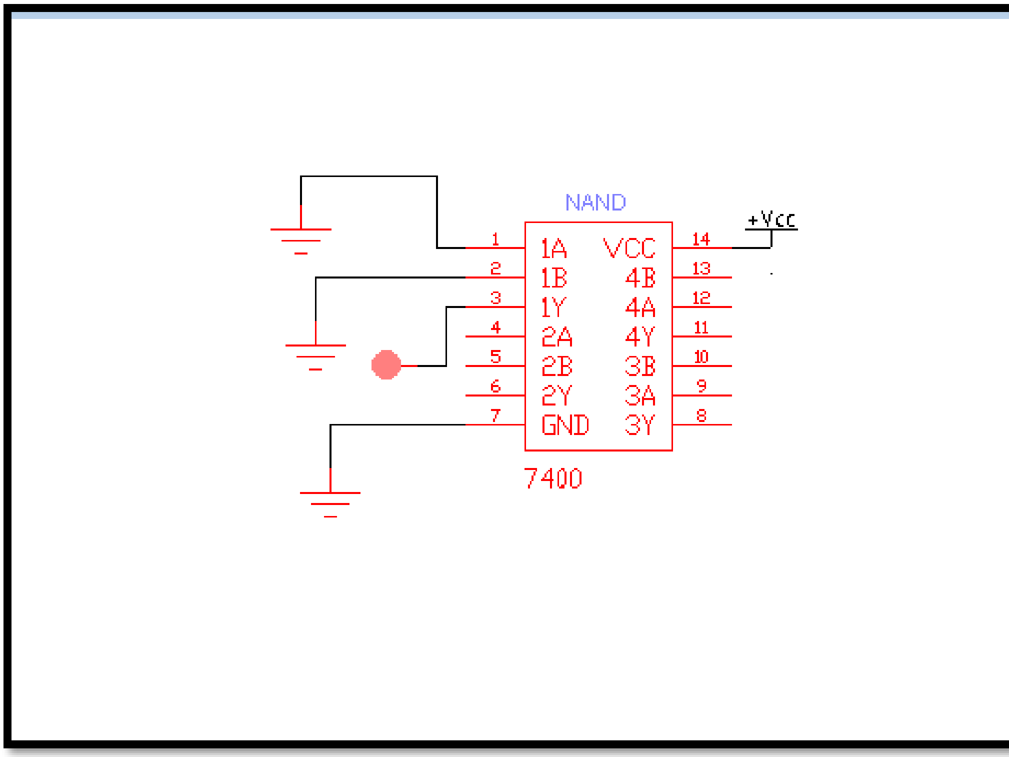


Fig 1: EWB Implementation of simple NAND gate

- ## NAND Gate complicated circuit:

Procedure for NAND Gate complicated circuit:

First, I opened Electronics Workbench and started a new file.

1. I picked the 7408 IC for the AND gate and the 7404 IC for the NOT gate from the component library.
2. Then, I connected Vcc and GND to both ICs (pin 14 to power and pin 7 to ground).
3. I connected two inputs to the AND gate at pins 1A and 1B of the 7408.
4. The output from the AND gate pin 3was connected to the input of the NOT gate pin 1 .
5. I took the final output from pin 2 . This gave me the NAND output.
6. After wiring, I tested the circuit by giving different inputs and checked if the output was correct.

7. It worked like a NAND gate, so the circuit was successful.

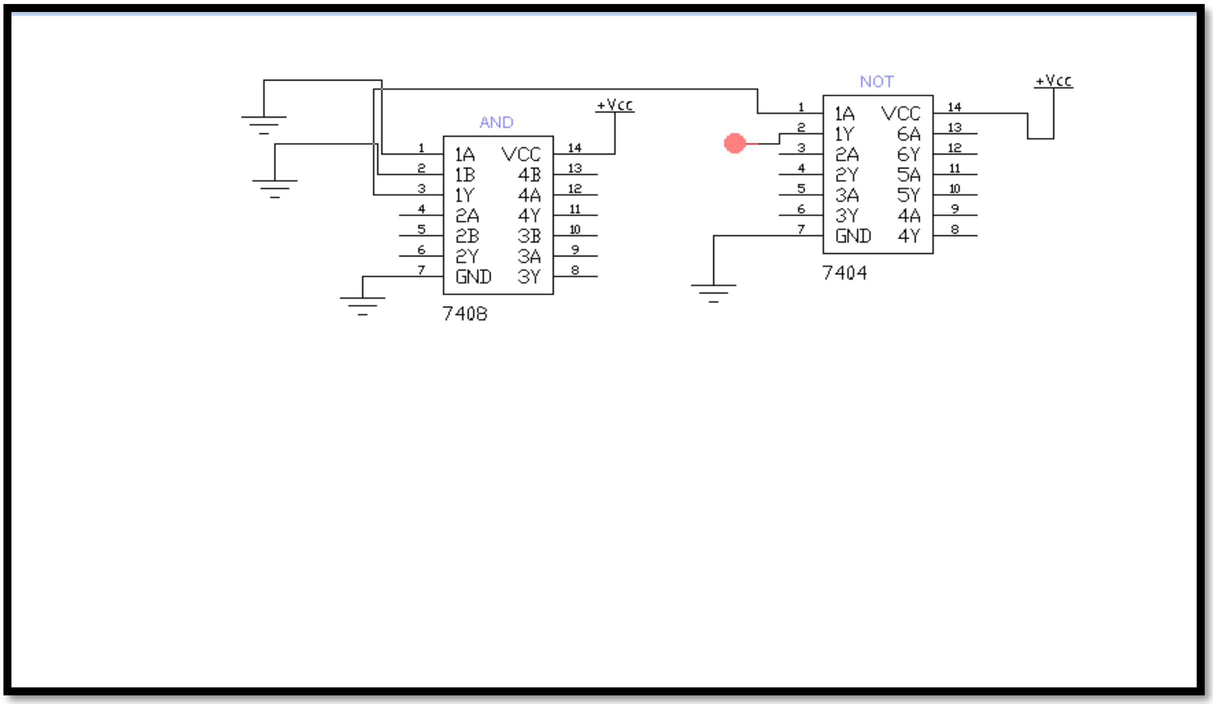
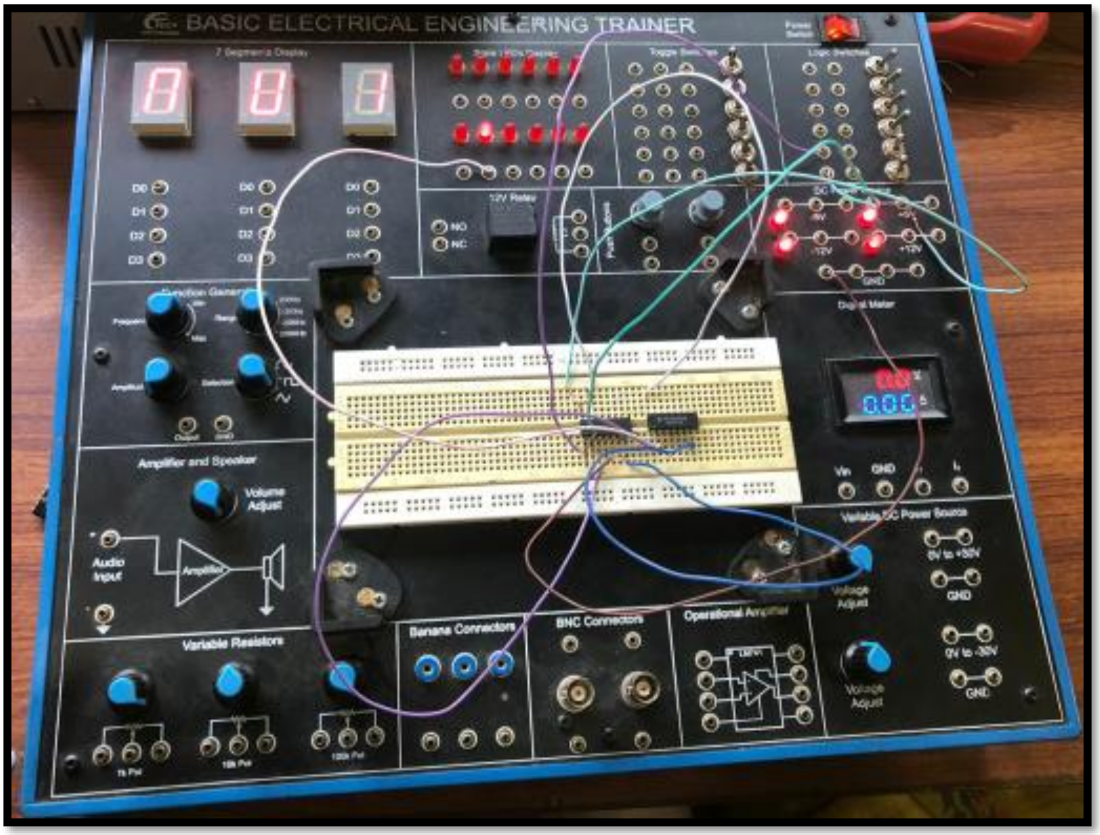


Fig 2: EWB Implementation of complicated NAND gate

- **<u>Hardware circuit</u>**



- **<u>Table:</u>**

A	B	A.B	(A . B)’
1	1	1	0
1	0	0	1
00	1	0	1
0	0	0	1

- <u>**NOR Gate:**</u>

  - <u>**NOR Gate simple circuit:**</u>

Procedure

1. I opened Electronics Workbench and started a new project.
2. I selected the 7402 IC, which has NOR gates in it.
3. I connected Vcc pin 14 to power and GND pin 7 to ground.
4. I used the first NOR gate by connecting two inputs to pin 1 and pin 2.
5. I took the output from pin 3, which gave me the NOR result.
6. Then I added switches to test the inputs and ran the circuit.
7. The output was correct for all input combinations, just like a NOR gate truth table.
8. The circuit was simple and worked perfectly.

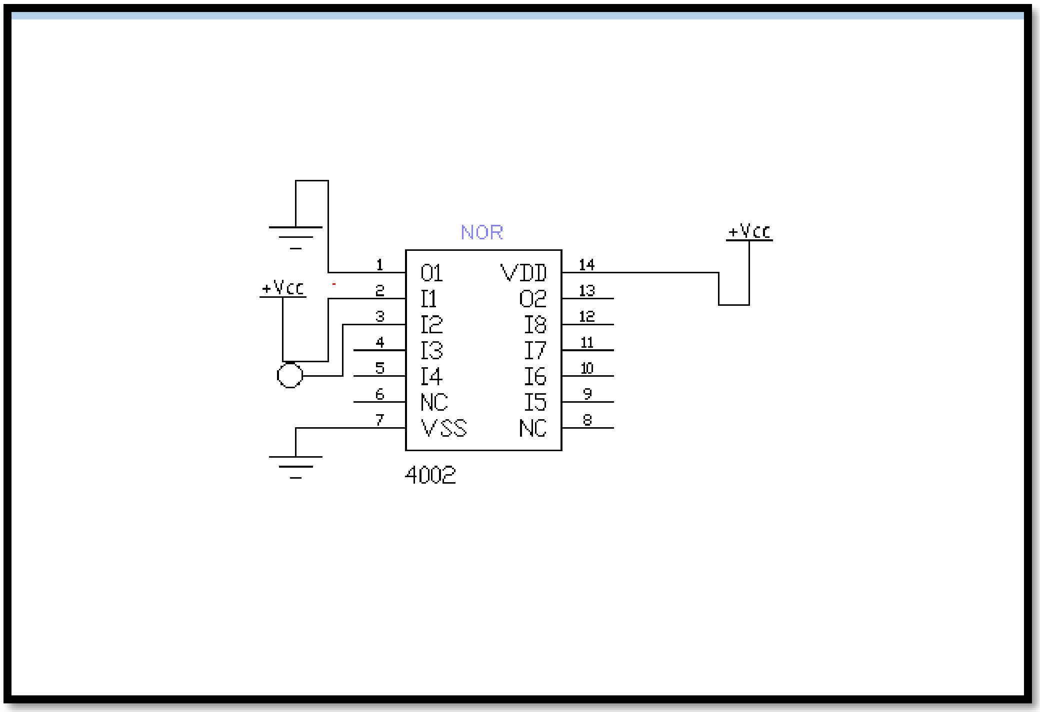


Fig 3: EWB Implementation of simple NOR gate

- **<u>NOR Gate complicated circuit:</u>**

Procedure

I opened Electronics Workbench and created a new project.

1. I selected a basic OR gate and a NOT gate from the components list.
2. I connected two input switches to the inputs of the OR gate.
3. I connected the output of the OR gate directly to the input of the NOT gate.
4. The output of the NOT gate became the final NOR output.
5. I added an LED to the output of the NOT gate to see the result.
6. Then I connected power and ground to all components as needed.

8.The output matched the NOR gate truth table exactly.

9.The circuit worked perfectly and showed how to build a NOR gate from simpler gates.

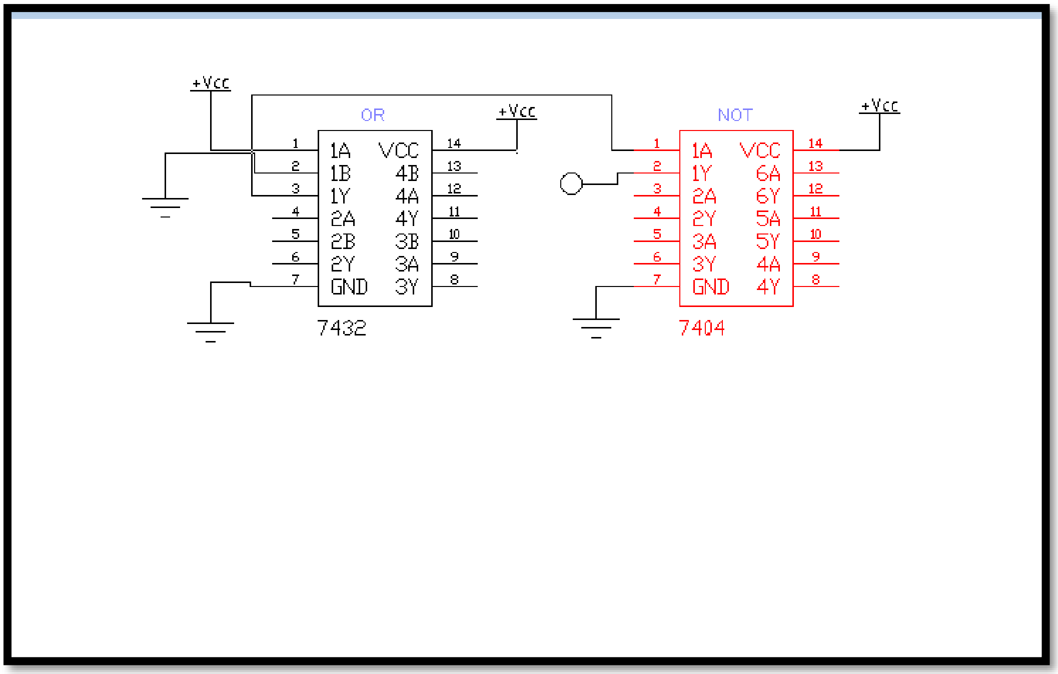


Fig 4: EWB Implementation of complicated NOR gate

- **Table:**

A	B	A+B	(A+B)’
1	1	1	0
1	0	1	0
0	1	1	0
0	0	0	1

- **<u>Demorgan's Law for NAND Gate:</u>**

Procedure

I opened Electronics Workbench and started a new project.

1. I added the 7400 IC (NAND gate), 7404 IC (NOT gate), and 7432 IC (OR gate) to the workspace.
2. I connected Vcc (pin 14) and GND (pin 7) on all three ICs.
3. For the left side of the circuit, I used a single NAND gate from the 7400:
    - I connected two switches to the inputs (pins 1 and 2).
    - The output was taken from pin 3, which showed the direct NAND result of A and B.
4. For the right side, I built the equivalent circuit using DeMorgan's Law:
    - I took the same two inputs (A and B) and connected each to a NOT gate .
    - Their outputs were then connected to an OR gate .
    - This way I created: A' + B' which, according to DeMorgan's Law, is equal to (A · B)'.
5. I connected LEDs to both outputs (NAND and OR-NOT combo) to compare them.
6. I tested all input combinations using the two switches:
    - (0,0), (0,1), (1,0), (1,1)

which is DeMorgan's Theorem.

7. The LEDs lit up the same on both sides every time, so the proof worked perfectly.
8. This was a clean and clear way to verify DeMorgan's Law using logic gates and ICs.

- **Formula:**

(A.B)’=A’+B’

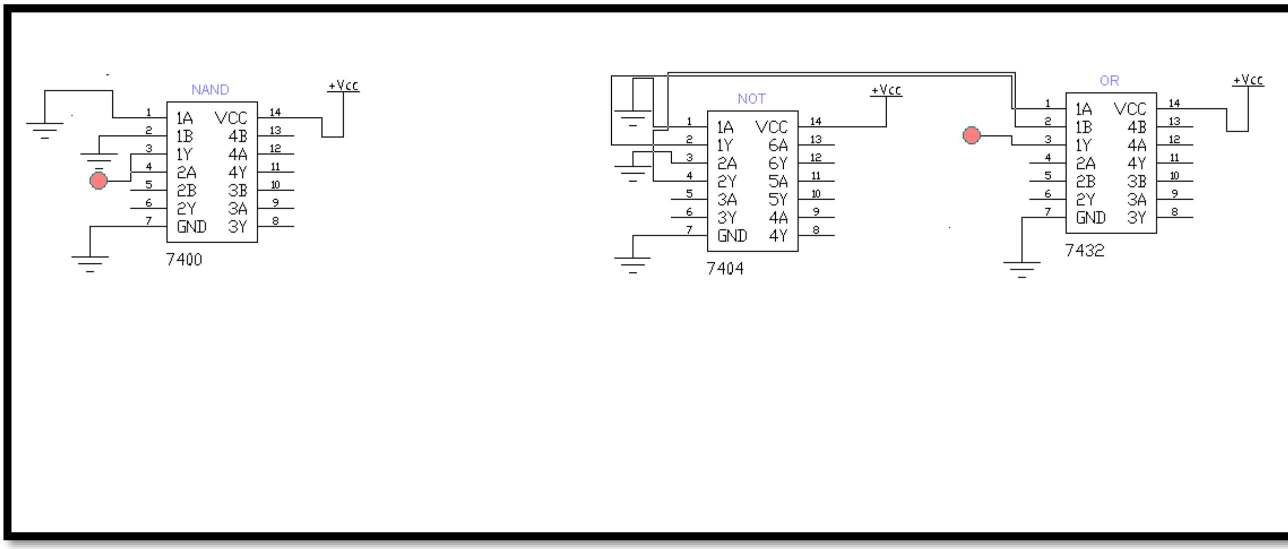
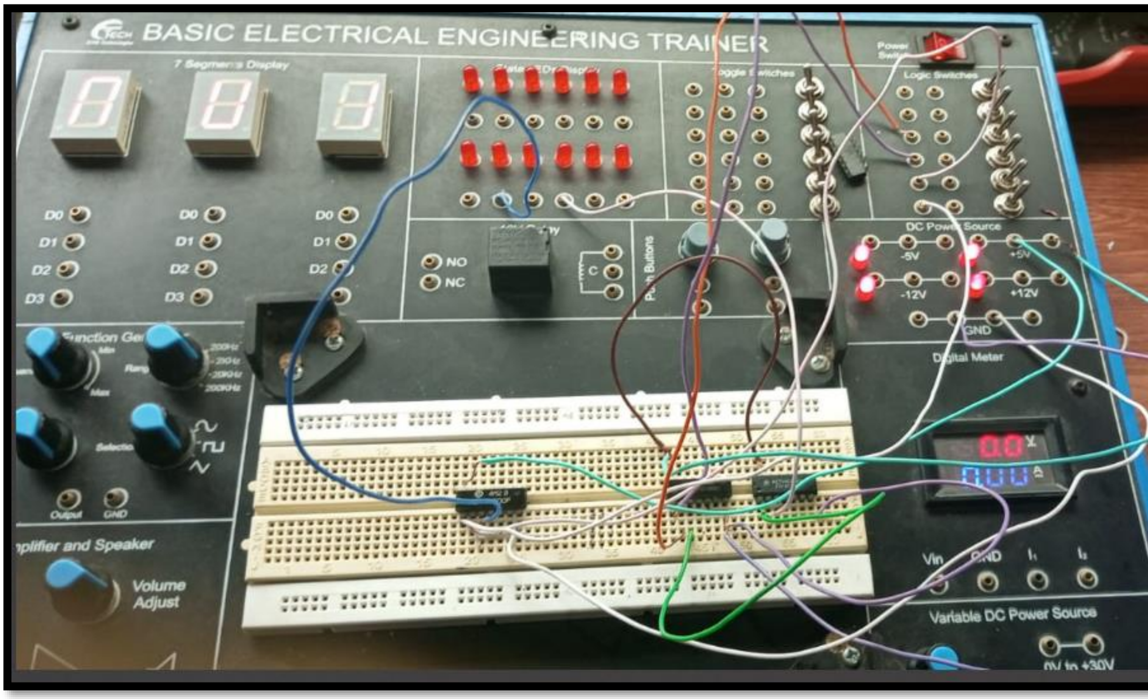


Fig 5: EWB Implementation of Demorgan’s law for NAND gate

- **Hardware circuit**



- **Table:**

A	B	A’	B’	A.B	(A.B)’	A’+B’
1	1	0	0	1	0	0
1	0	0	1	0	1	1
0	1	1	0	0	1	1
0	0	1	1	0	1	1

- **<u>Demorgan’s law for NOR Gate:</u>**

Procedure

I opened Electronics Workbench and started a new project.

1. I added the 7402 IC (NOR gate), 7408 IC (AND gate), and 7404 IC (NOT gate) to the workspace.
2. I connected Vcc (pin 14) and GND (pin 7) on all three ICs.
3. On the left side of the circuit, I used a single NOR gate :
   - I connected two switches as inputs to pins 1 and 2 of the IC.
   - The output from pin 3 gave me NOR(A, B) directly.
4. On the right side, I built the DeMorgan equivalent using NOT and AND gates:
   - I passed the same two inputs through two NOT gates from the 7404.
   - Then I connected those inverted outputs to an AND gate from the 7408.
   - This built: $A' \cdot B'$ which is the same as $(A + B)'$ by DeMorgan’s Law.
5. I connected LEDs to both outputs to visually compare the NOR and the AND-NOT combination.
6. I tested all input combinations using the switches:
   - (0,0), (0,1), (1,0), (1,1)
7. For every test, both outputs matched — showing the same logic level on both LEDs.
8. The circuit worked exactly how it should — both sides gave the same output for all input cases.

- **Formula:**

(A+B)’=A’.B’

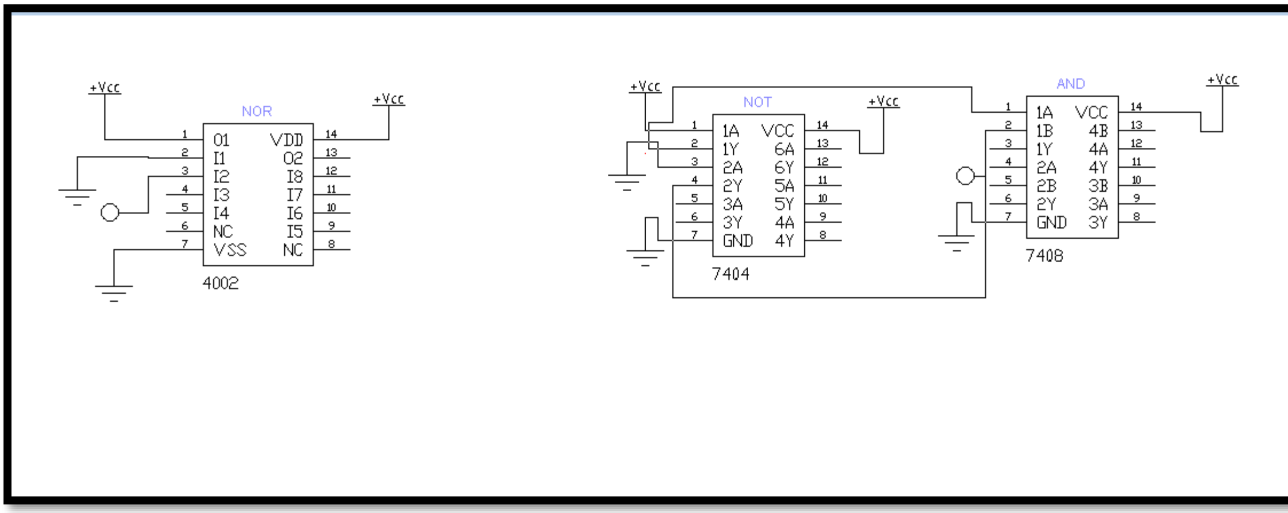


Fig 6: EWB Implementation of Demorgan’s law for NOR gate

- **Table:**

A	B	A’	B’	A+B	(A+B)’	A’.B’
1	1	0	0	1	0	0
1	0	0	1	1	0	0
0	1	1	0	1	0	0
0	0	1	1	0	1	1

- **Half Adder**
- **Half Adder simple circuit**

Procedure: How I Created a Simple Half Adder Circuit

1. **Objective:**

   I wanted to make a half adder that adds two 1-bit inputs (A and B) and gives me the sum and carry outputs using a simple circuit.

2. **Understanding the Half Adder:**

   I know the half adder takes inputs A and B and produces:

- Sum = A XOR B
- Carry = A AND B

Since this is a simple circuit, I directly used XOR and AND gates for these outputs.

3. **Building the Circuit**:

- I connected inputs A and B directly to an XOR gate to get the Sum output.
- Then I connected inputs A and B to an AND gate to get the Carry output.

4. **Testing the Circuit:**

   I tested all possible inputs:

- A=0, B=0 → Sum=0, Carry=0
- A=0, B=1 → Sum=1, Carry=0
- A=1, B=0 → Sum=1, Carry=0
- A=1, B=1 → Sum=0, Carry=1

The outputs were exactly what I expected.

5. Conclusion:

By using the XOR and AND gates directly, I built a simple and efficient half adder. This helped me understand how these basic gates work together to perform addition.

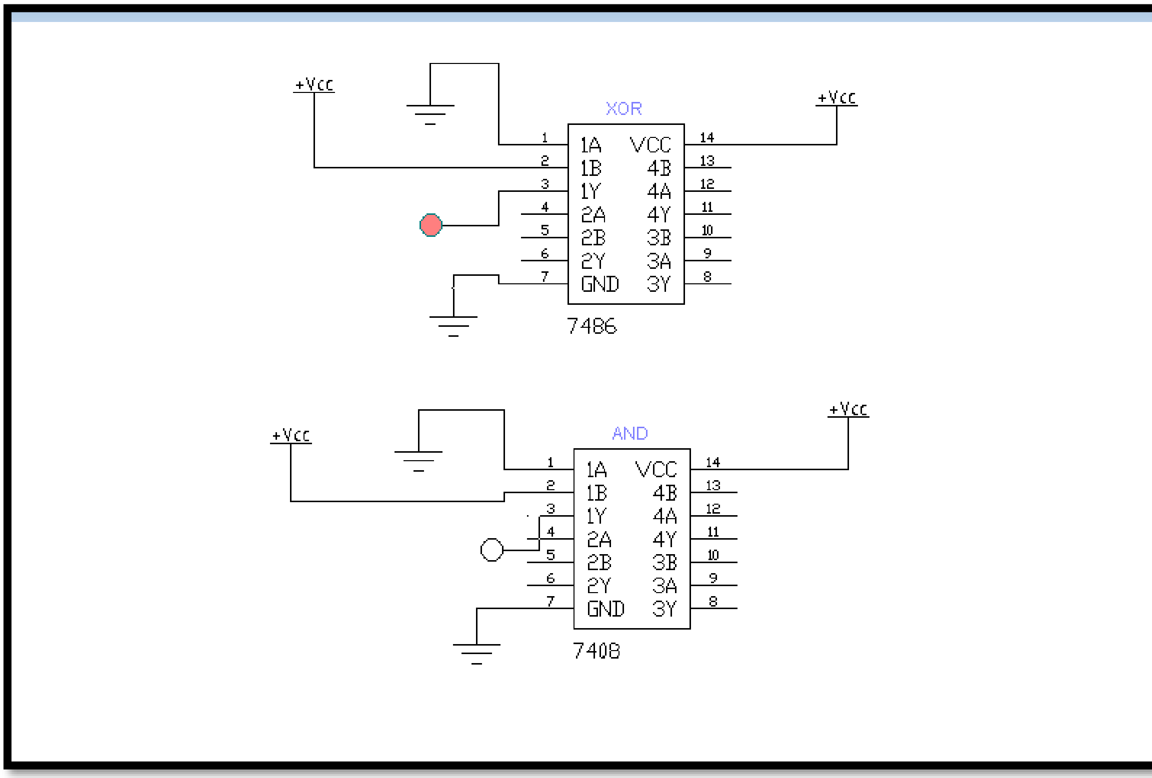
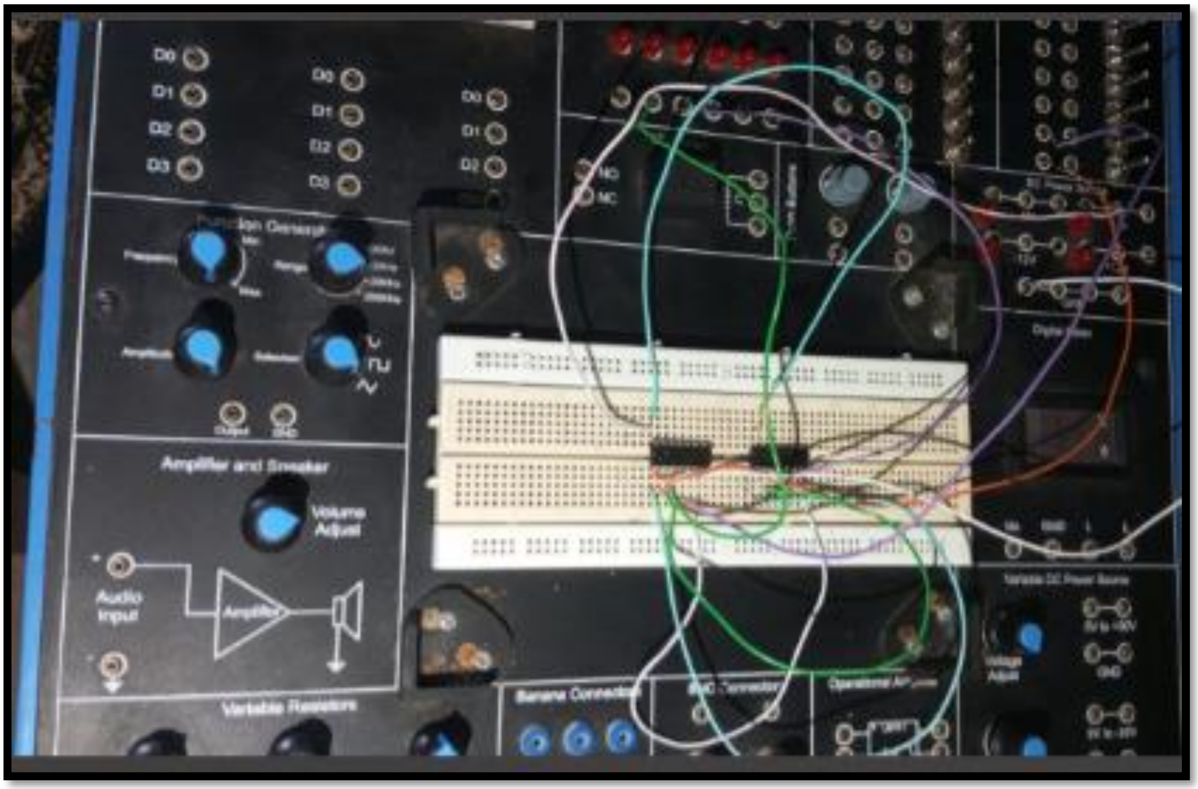


Fig 7: EWB Implementation of simple Half Adder

- **<u>Hardware circuit:</u>**



- **<u>Half Adder Complicated circuit</u>**

Procedure: How I Created a Half Adder Using NOT, AND, and OR Gates

1. **Objective:**

I wanted to build a half adder that adds two inputs and outputs sum and carry, but only using NOT, AND, and OR gates because XOR gate wasn't allowed.

2. **Understanding the Logic:**

I know a half adder has two inputs (A and B) and two outputs:

- Sum = A XOR B
- Carry = A AND B

Since I can't use XOR directly, I needed to express XOR using only NOT, AND, and OR gates.

3.Building the Circuit:

- I connected input A to a NOT gate to get A', and input B to AND gate.
- Then I connected input A with AND gate and input B with NOT gate.
- Then I ANDed A' with B, and A with B'.
- I ORed the results of those two AND gates to produce the sum output.
- For carry, I simply ANDed A and B directly.

4. Testing the Circuit:

I tested the circuit for all possible inputs:

- For A=0, B=0 → Sum=0, Carry=0
- For A=0, B=1 → Sum=1, Carry=0
- For A=1, B=0 → Sum=1, Carry=0
- For A=1, B=1 → Sum=0, Carry=1

Everything worked exactly as expected.

5.Conclusion:

By building the half adder using NOT, AND, and OR gates, I learned how to create XOR functionality from basic gates. This helped me understand how complex logic circuits are made from simple parts.

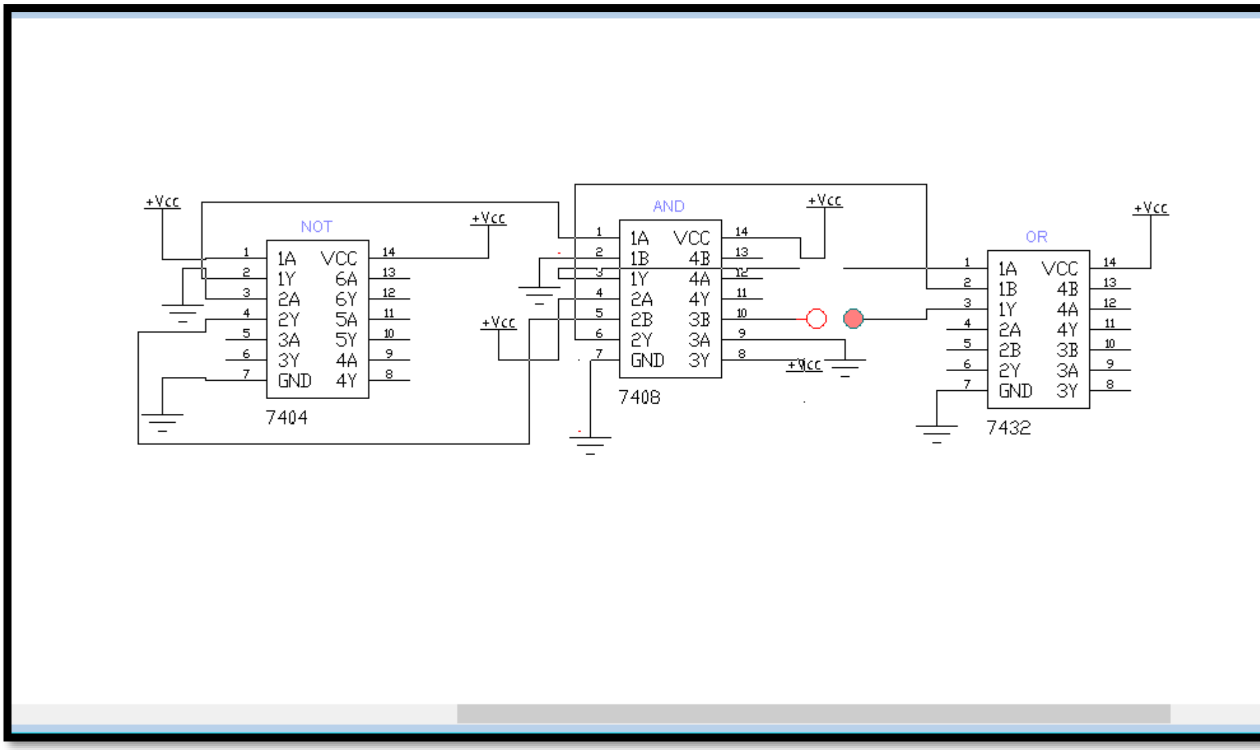
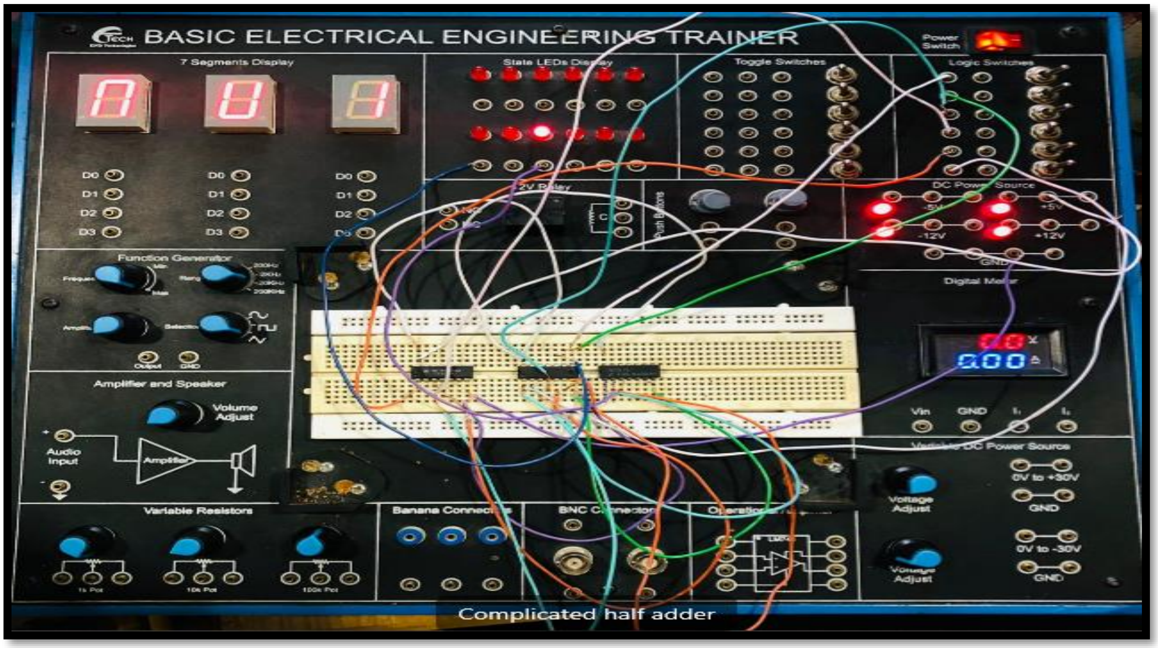


Fig 8: EWB Implementation of complex Half Hadder

- **Hardware circuit:**



- **Table:**

A	B	A’	B’	A’.B	A.B’	A’.B+A.B’(Sum)	A.B(carry)
1	1	0	0	0	0	0	1
1	0	0	1	0	1	1	0
0	1	1	0	1	0	1	0
0	0	1	1	0	0	0	0
A	B	A’	B’	A’.B	A.B’	A’.B+A.B’(Sum)	A.B(carry)
1	1	0	0	0	0	0	1
1	0	0	1	0	1	1	0
0	1	1	0	1	0	1	0
0	0	1	1	0	0	0	0