

|  |  |
| --- | --- |
| Student Name | Afzal Ali Ahmed Shah |
| Roll no | **2024-SE-04** |
| Semester | **2nd** |
| Instructor Name | **MAM. Sidra** |
| Subject | **CA&LD** |
| Course Code | **CS-1206** |
| Credit Hours | **3+1** |
| Date | **06-June-2025** |
| Department of Software Engineering | |

**Lab Report: NAND Gate Implementation Using AND and NOT Gates**

**Objective:**

To **implement a NAND gate** using **basic logic gates** (AND + NOT) and verify its functionality using **Electronics Workbench / Multisim**.

**Theory:**

A **NAND gate** is the **inverse of an AND gate**.

**Standard NAND Expression:**

Y=(A⋅B)’

So, a NAND gate can be **constructed by combining**:

* One **AND gate**
* One **NOT gate** (to invert the output of the AND gate)

**Boolean Expression:**

Y=(A⋅B)’

Where:

* **A, B** = Inputs
* **Y** = Output (NAND result)

**Components Required:**

* **Electronics Workbench / Multisim**
* Logic Gates: AND, NOT
* 2 Input Switches (A and B)
* 1 LED (to display NAND output)
* Connecting Wires

**🔌 Circuit Diagram**

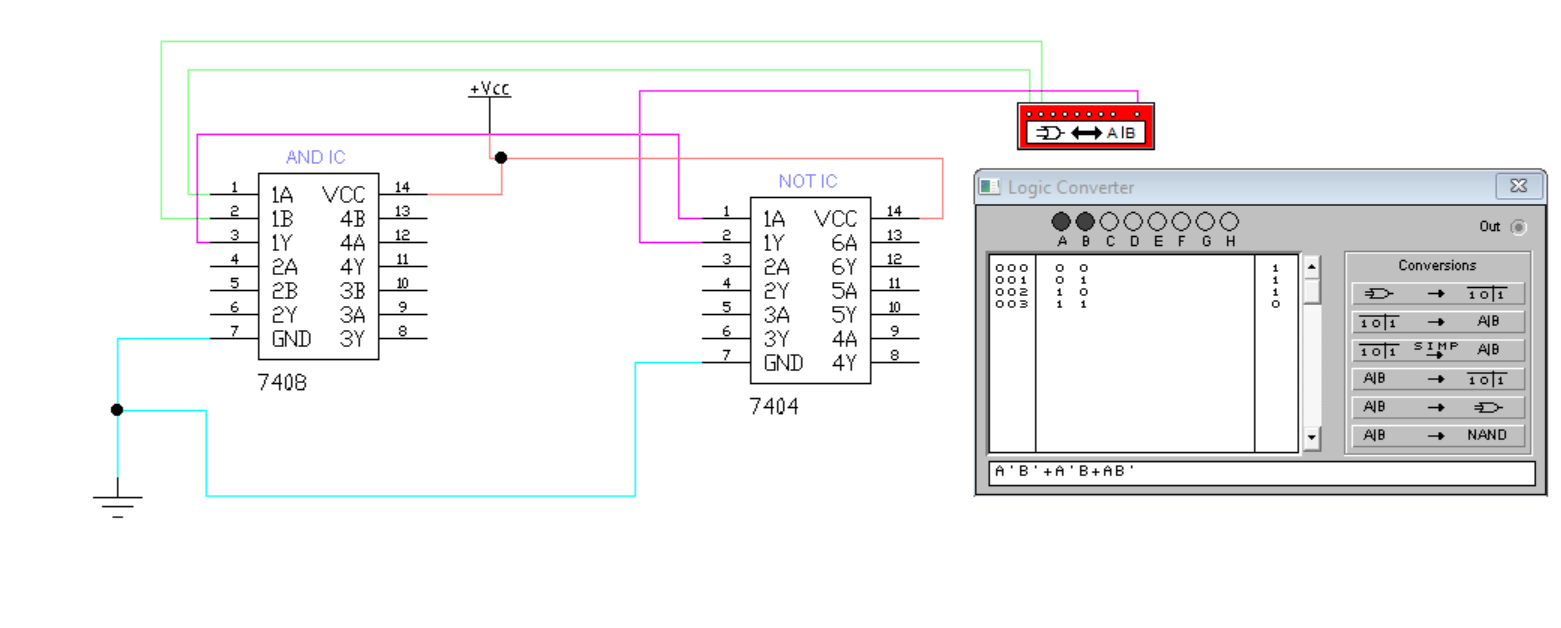
**Connections:**

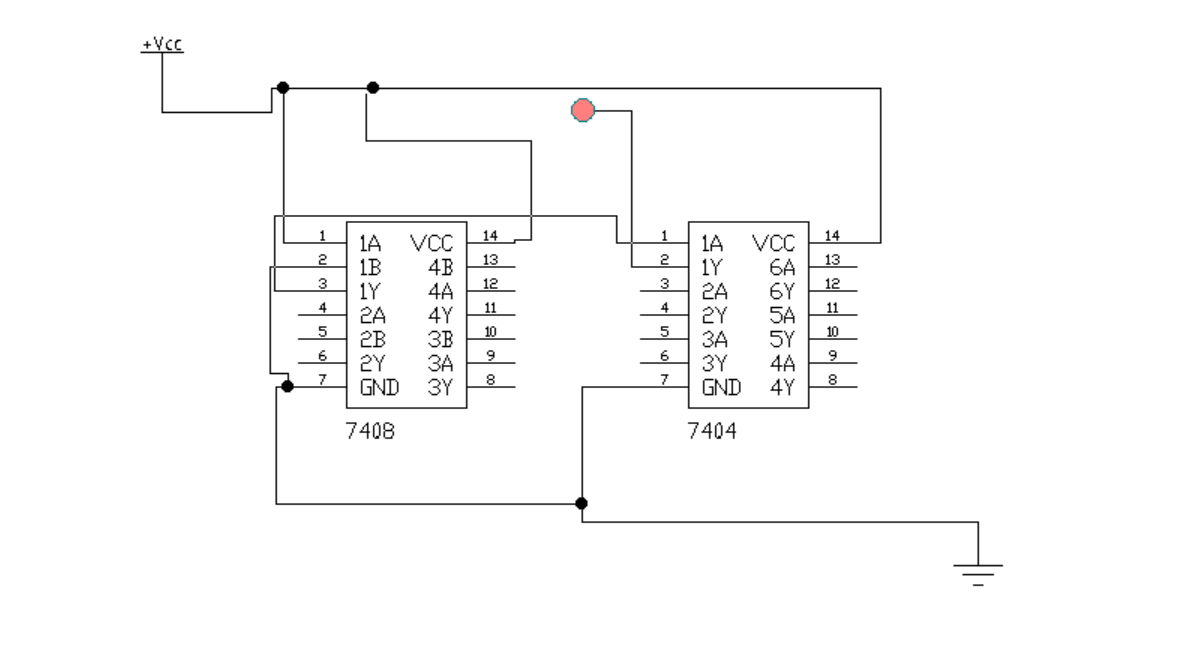
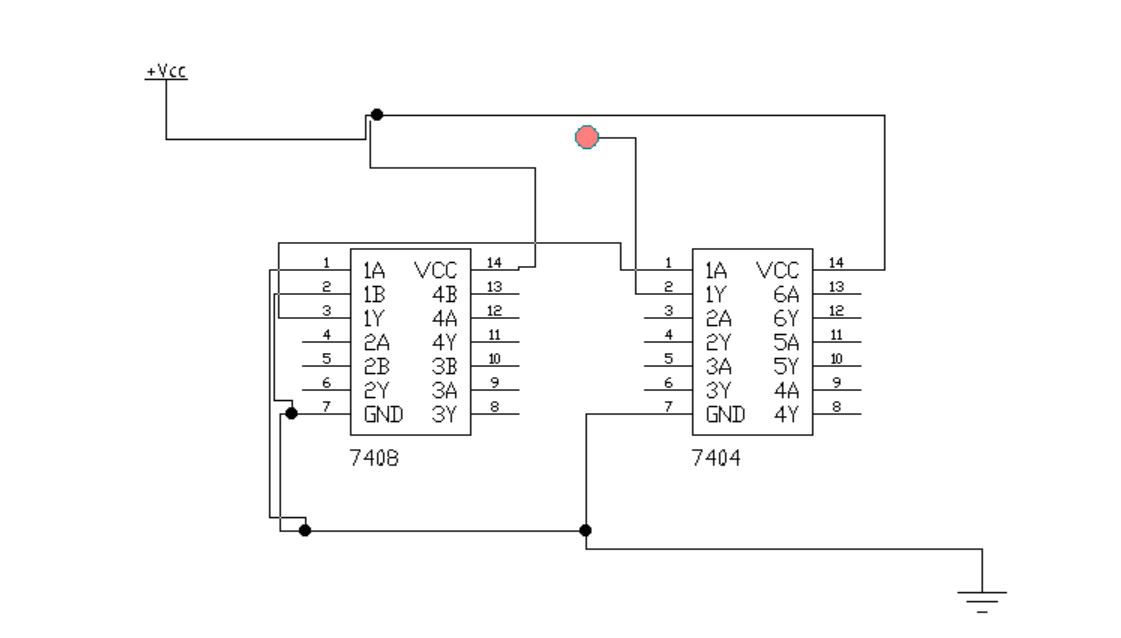
* Inputs A and B → AND gate
* Output of AND gate → NOT gate
* Output of NOT gate → LED (shows the NAND result)

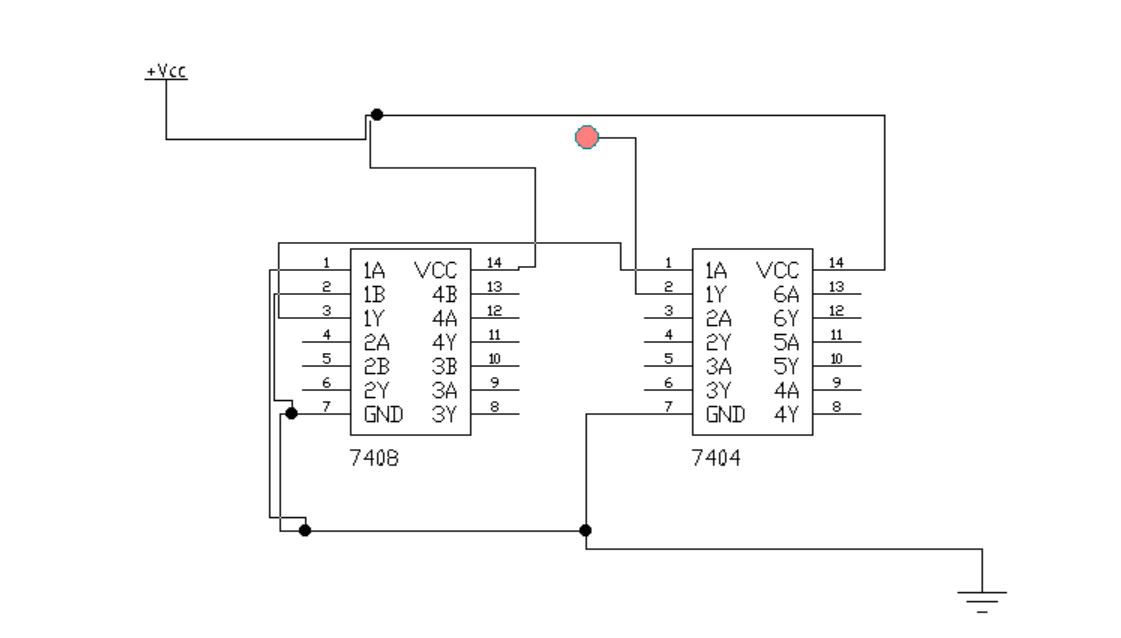
**Truth Table for NAND Implementation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | A·B (AND Output) | NAND = ¬(A·B) | LED Status (ON = 1, OFF = 0) |
| 0 | 0 | 0 | 1 | ON |
| 0 | 1 | 0 | 1 | ON |
| 1 | 0 | 0 | 1 | ON |
| 1 | 1 | 1 | 0 | OFF |

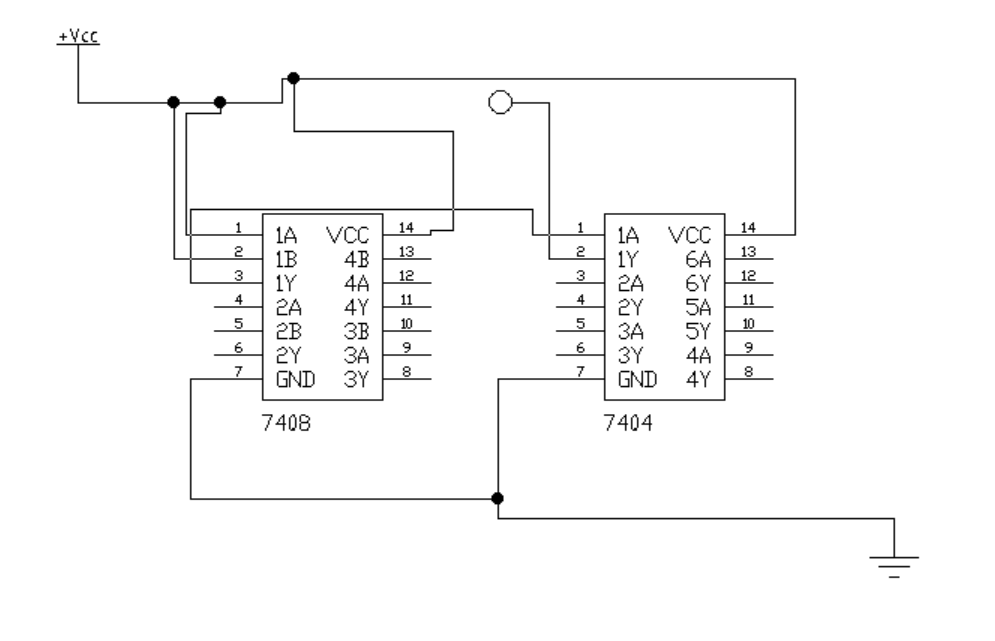
**Explanation of Output (LED States):**

* The **AND gate** gives output **1 only when both inputs are 1**.
* The **NOT gate** inverts that output.





* So, the **LED is ON** for all cases **except when both A and B are 1**.



**Conclusion:**

* The NAND gate was successfully implemented using **AND and NOT gates**.
* The **LED outputs** matched the **expected truth table** for a NAND gate.
* This confirms the **correct logical behavior** and the **validity of gate-level construction**.



***CLICK on it and all pics are here***

**Goal: Implement NOR using OR and AND gates**

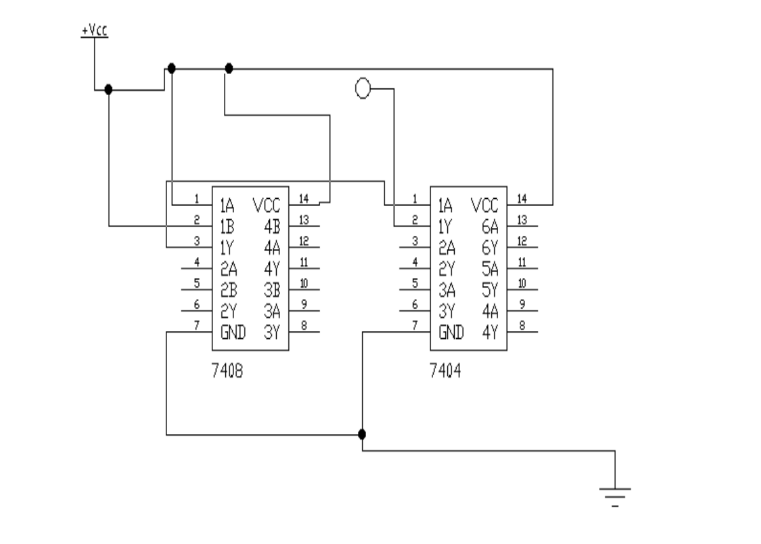
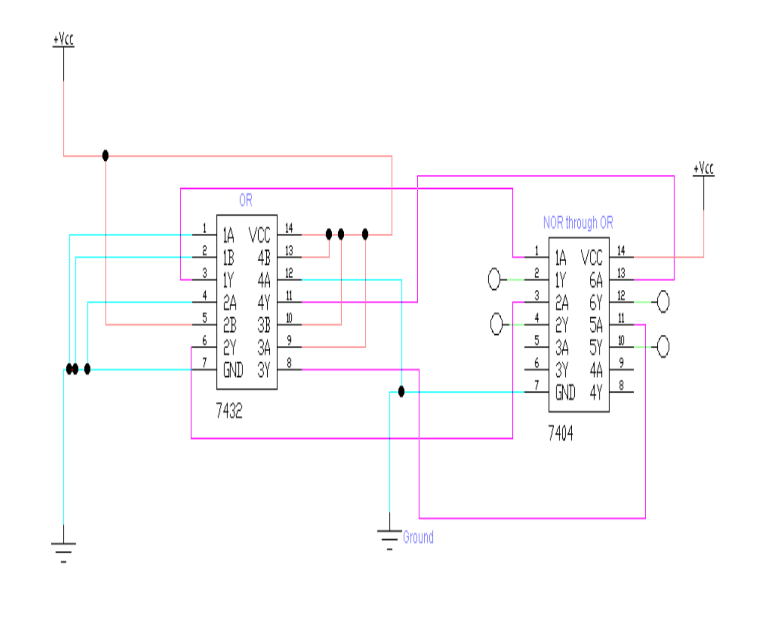
**🧠 Target Logic:**

The NOR gate=(A+B​ )’

We want to implement this using:

* OR gate
* NOT gate (which can be made using AND + extra wiring, if needed)

**✅ 1. Logic Expression:**

The NOR function is:

Note: If NOT gate is not available, you can create it using AND gate and connecting both inputs to the same inverted signal (using transistor level or custom logic). But normally, you can use a NOT gate directly.

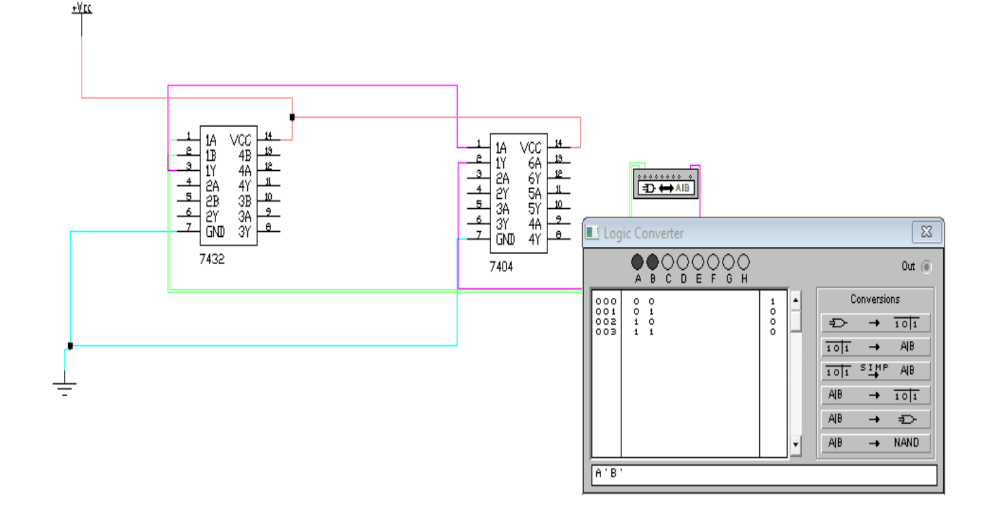
**✅ Circuit Design Steps (For Electronics Workbench):**

1. **Place two logic switches** → these are inputs A and B
2. **Connect A and B to a 2-input OR gate**
3. **Take the output of OR gate and connect it to a NOT gate**
4. **Connect the NOT gate output to an LED or logic indicator**
5. **Label wires clearly: A, B, A+B, ¬(A+B)**

**✅ Truth Table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | A + B | ¬(A + B) | LED State |
| 0 | 0 | 0 | 1 | ON |
| 0 | 1 | 1 | 0 | OFF |
| 1 | 0 | 1 | 0 | OFF |
| 1 | 1 | 1 | 0 | OFF |

**✅ Screenshot Instructions for Lab Report:**



* Capture the circuit showing:
  + **OR gate connected to NOT gate**
  + **Inputs A and B** using switches
  + **LED** showing output
* Show different states:
  + (A=0, B=0) → Output = 1 (LED ON)
  + (A=1, B=1) → Output = 0 (LED OFF)
* Label logic levels clearly in the screenshot.

**✅ Conclusion (For Report):**

We successfully implemented a **NOR gate** using an **OR gate followed by a NOT gate**. The output matches the expected NOR logic behavior, verifying that: