**The University of Azad Jammu and Kashmir, Muzaffarabad**



**Course:** CAand Logic Design

**Course Code:** CS-1205

**Roll No:** 2024-SE-18

**Lab Report:** 1

**Submitted To:** Engr. Sidra Rafique

**Submitted By:** Syed Muhammad Abdulrehman Naseem

**Date:** 24 April, 2025

***Department Of Software Engineering***

**Lab Report: Verification of Basic Logic Gates**

**Objective:**

To study and understand the basic functioning and truth tables of various logic gates including **AND, OR, NOT, NAND, NOR,** and **XOR.**

**Apparatus:**

Power Supply (Virtual – EWB)

Breadboard (Virtual)

Connecting Wires (Virtual)

Logic Gate ICs (**7408, 7432, 7404, 7400, 7402, 7486**)

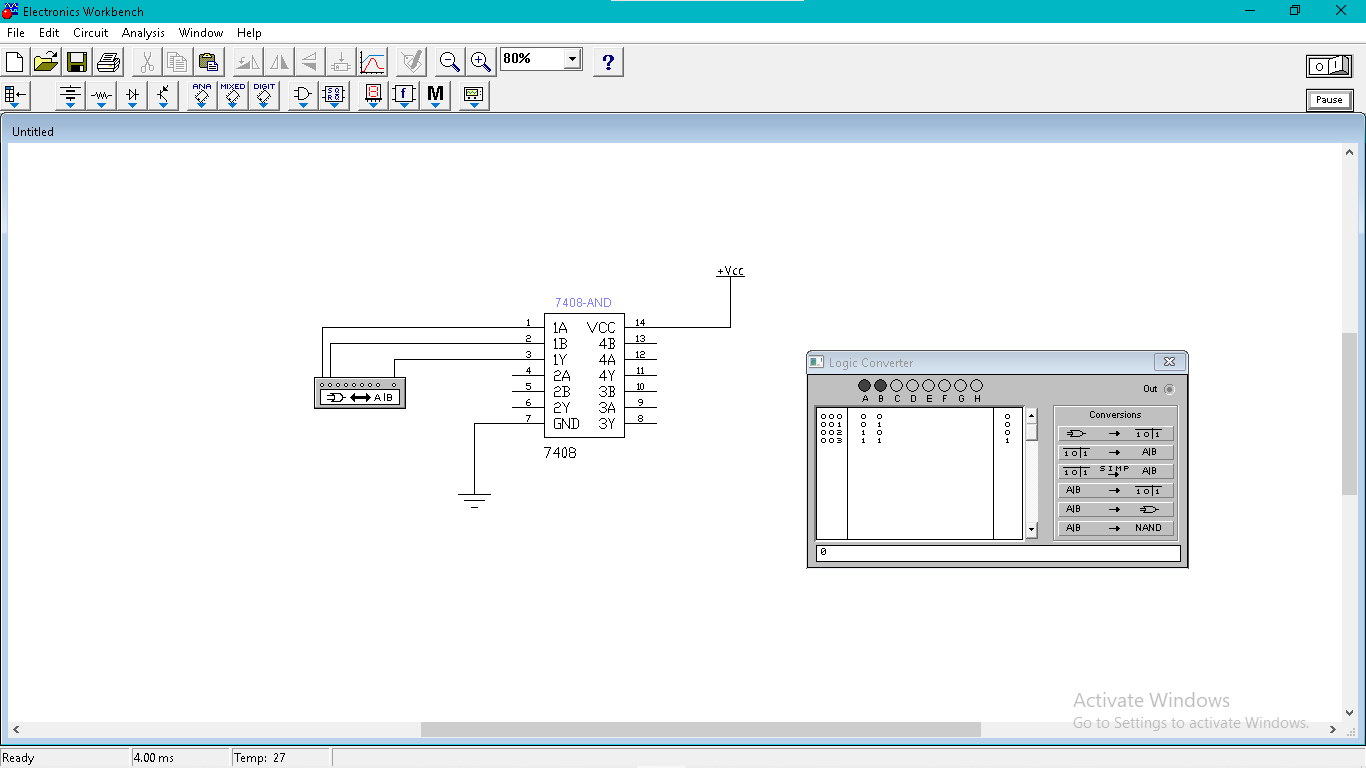
Logic Probes / LEDs for output detection

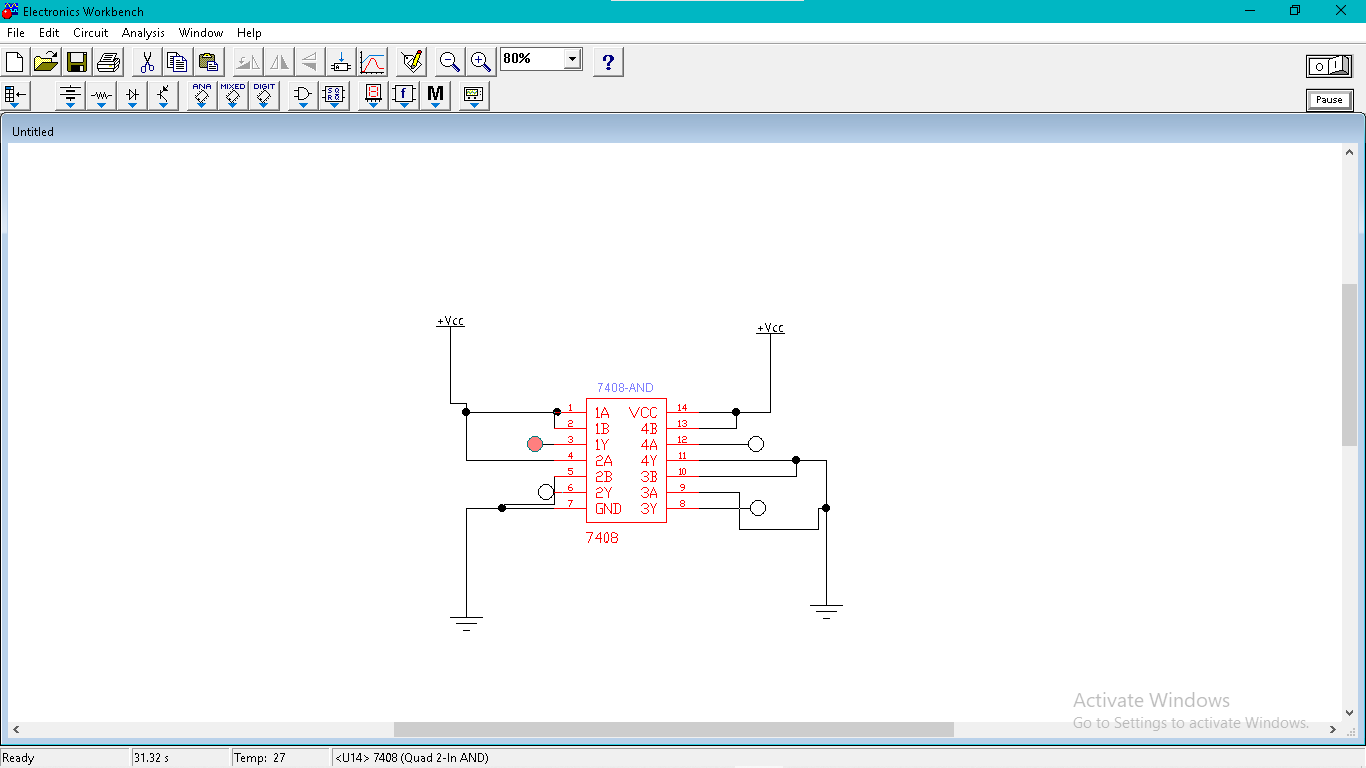
**Theory:**

Logic gates are the basic building blocks of digital electronics. Each gate performs a specific logical function based on binary inputs (0 or 1). The common gates are:

**IC Pinouts & Truth Tables:**

AND Gate (7408): Output is 1 only when both inputs are 1. (Y = A·B)

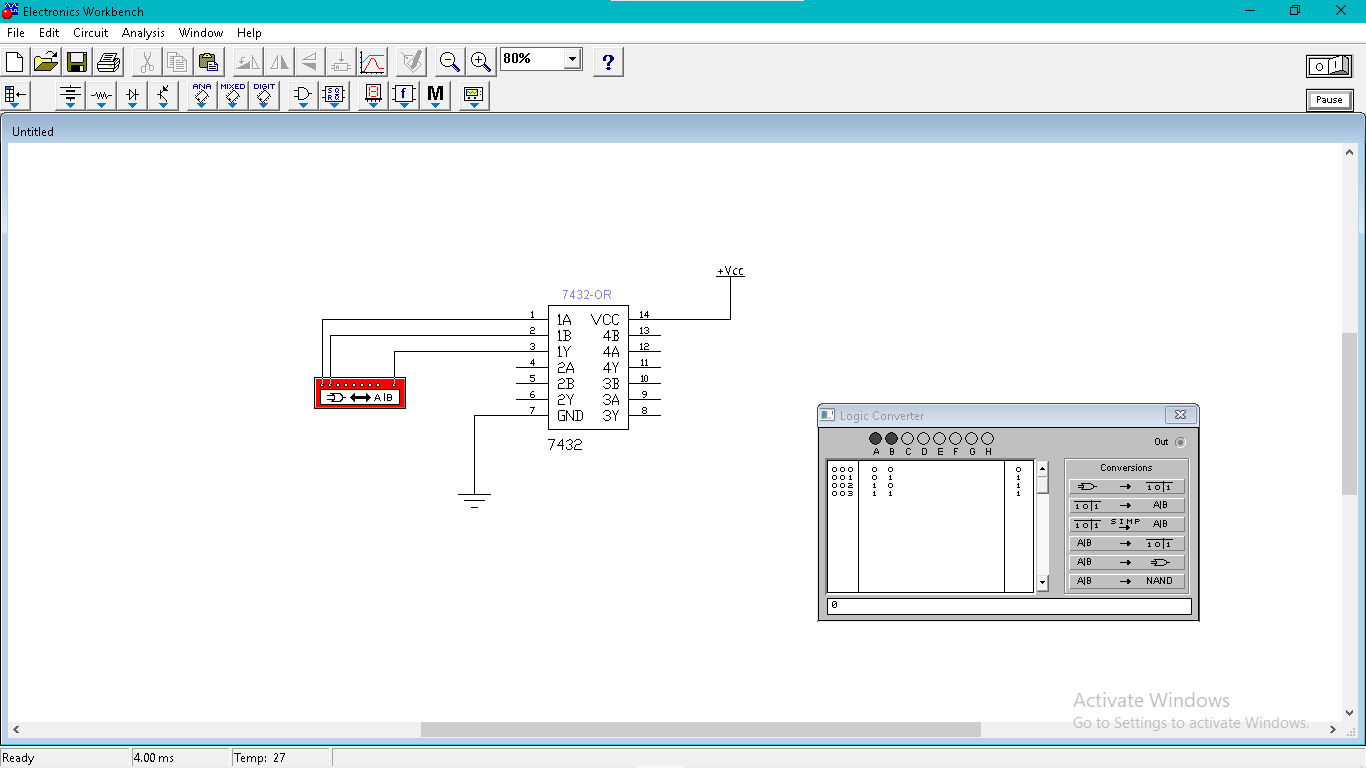


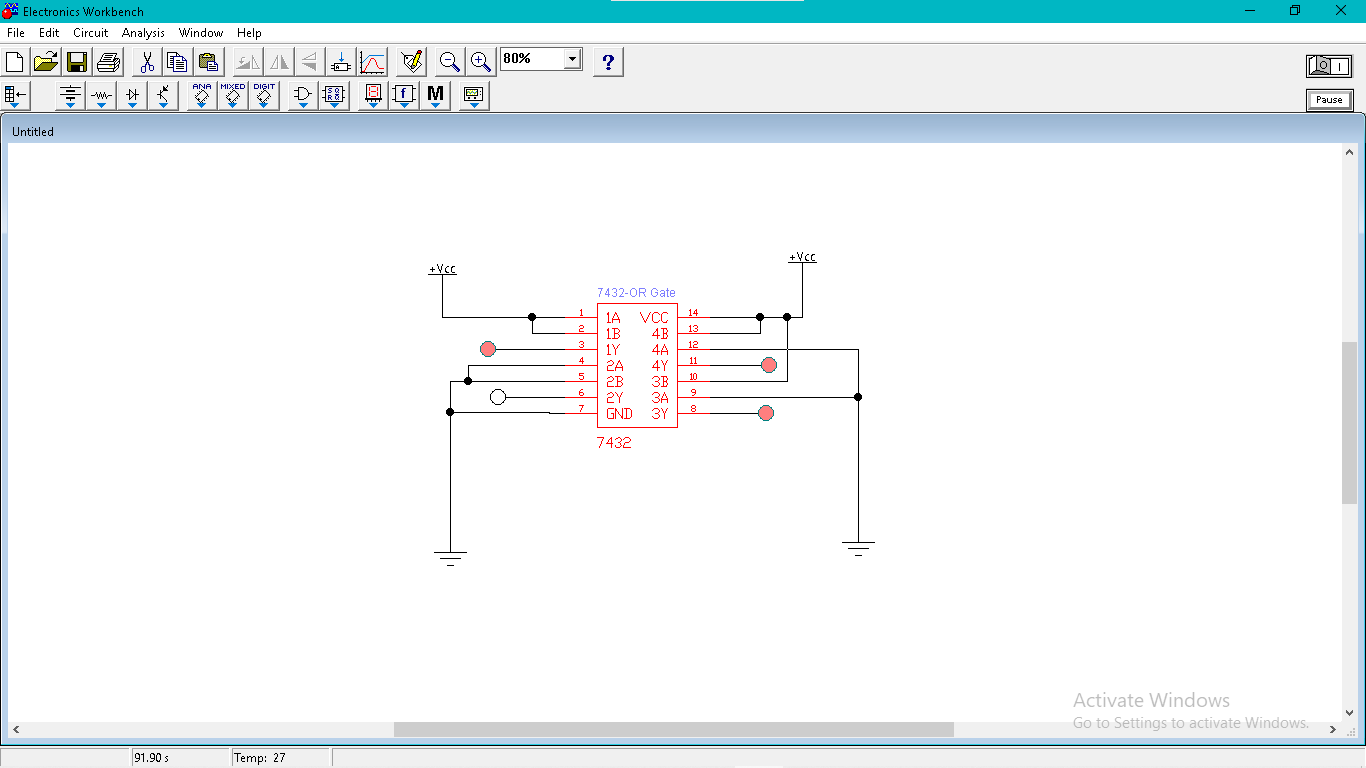


**Truth Table:**

|  |  |  |
| --- | --- | --- |
| **A** | **B** | **X= A.B** |
| **0** | **0** | **0** |
| **0** | **1** | **0** |
| **1** | **0** | **0** |
| **1** | **1** | **1** |

OR Gate (7432): Output is 1 if at least one input is 1. (Y = A + B)



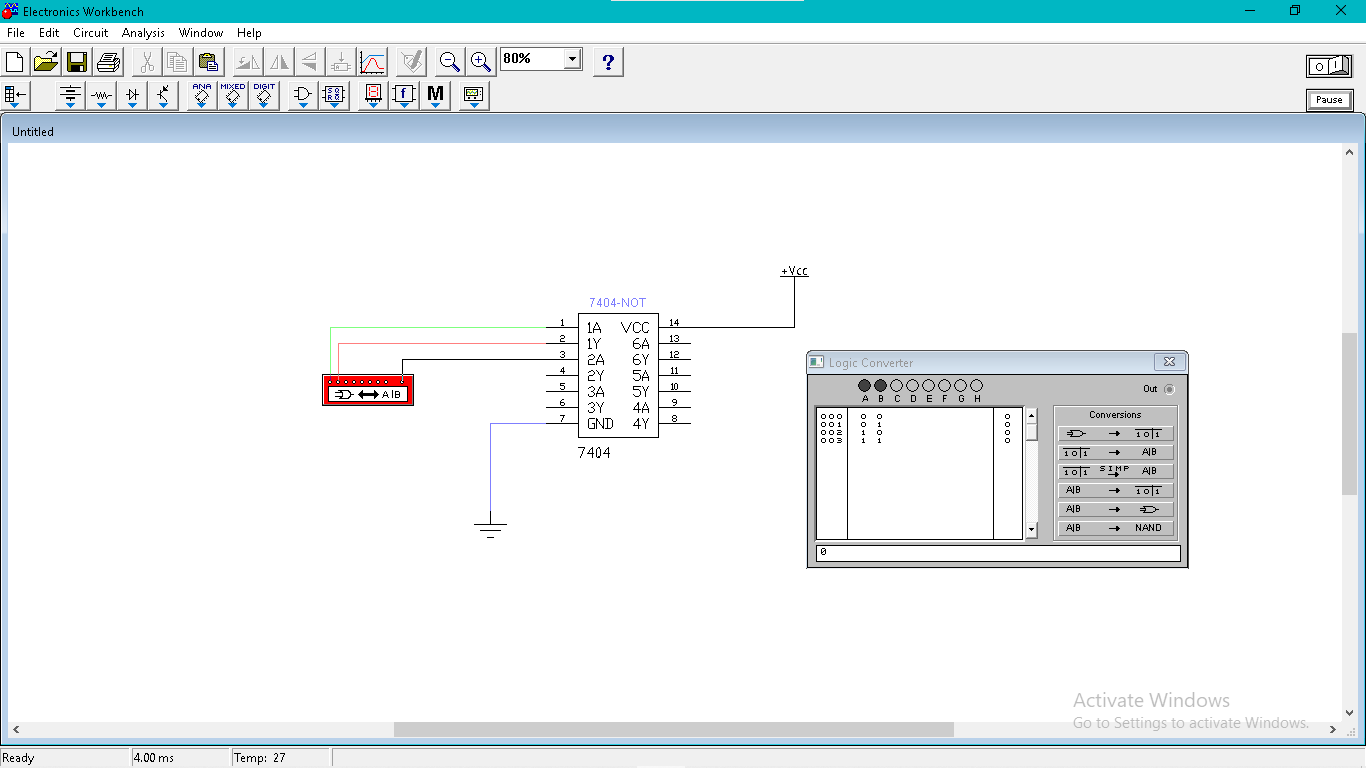
****

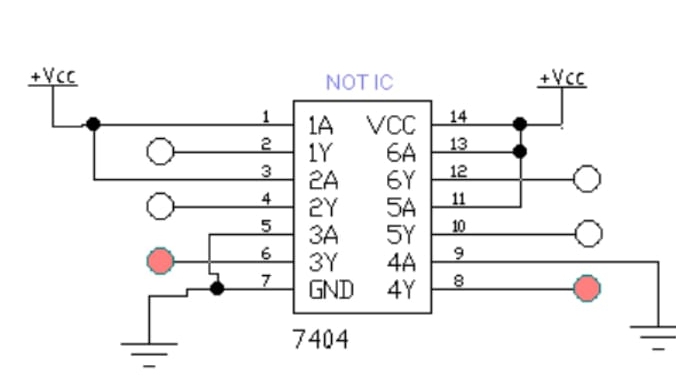
**Output of Or gate is 1, when any of its input is 1.**

**Truth Table:**

|  |  |  |
| --- | --- | --- |
| **A** | **B** | **X= A+B** |
| **0** | **0** | **0** |
| **0** | **1** | **1** |
| **1** | **0** | **1** |
| **1** | **1** | **1** |

NOT Gate (7404): Output is the inverse of the input. (Y = Ā)



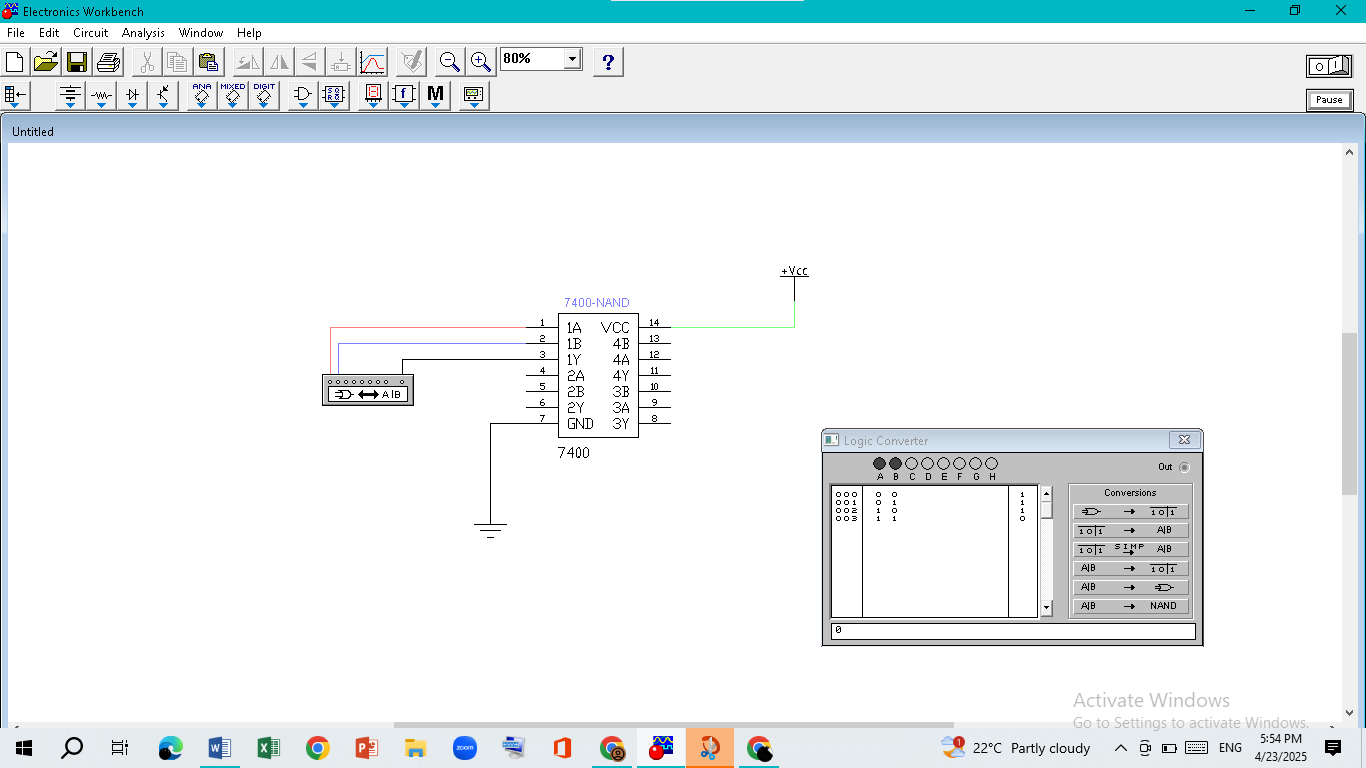
****

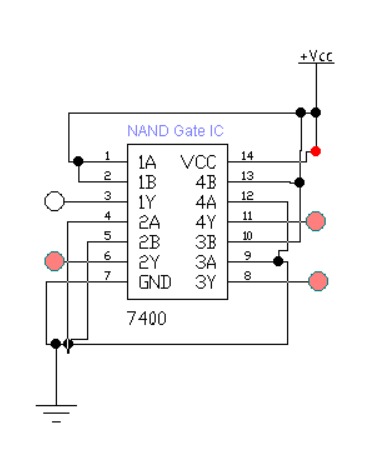
In case of NOT gate, output is 1 when input is zero.

**Truth Table:**

|  |  |  |
| --- | --- | --- |
| **A** | **B** | **X= ~A** |
| **0** | **0** | **0** |
| **0** | **1** | **0** |

NAND Gate (7400): Output is 0 only when both inputs are 1. (Y = ~(A·B))

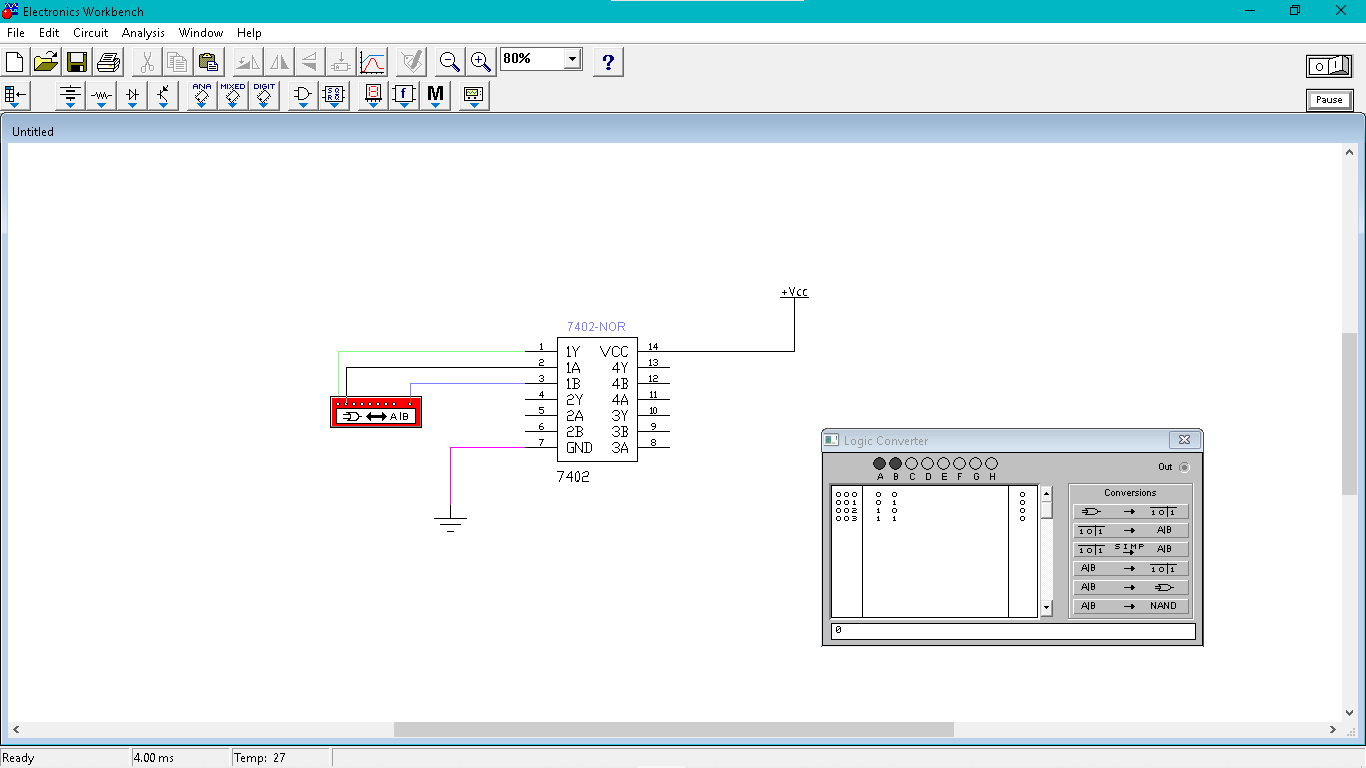


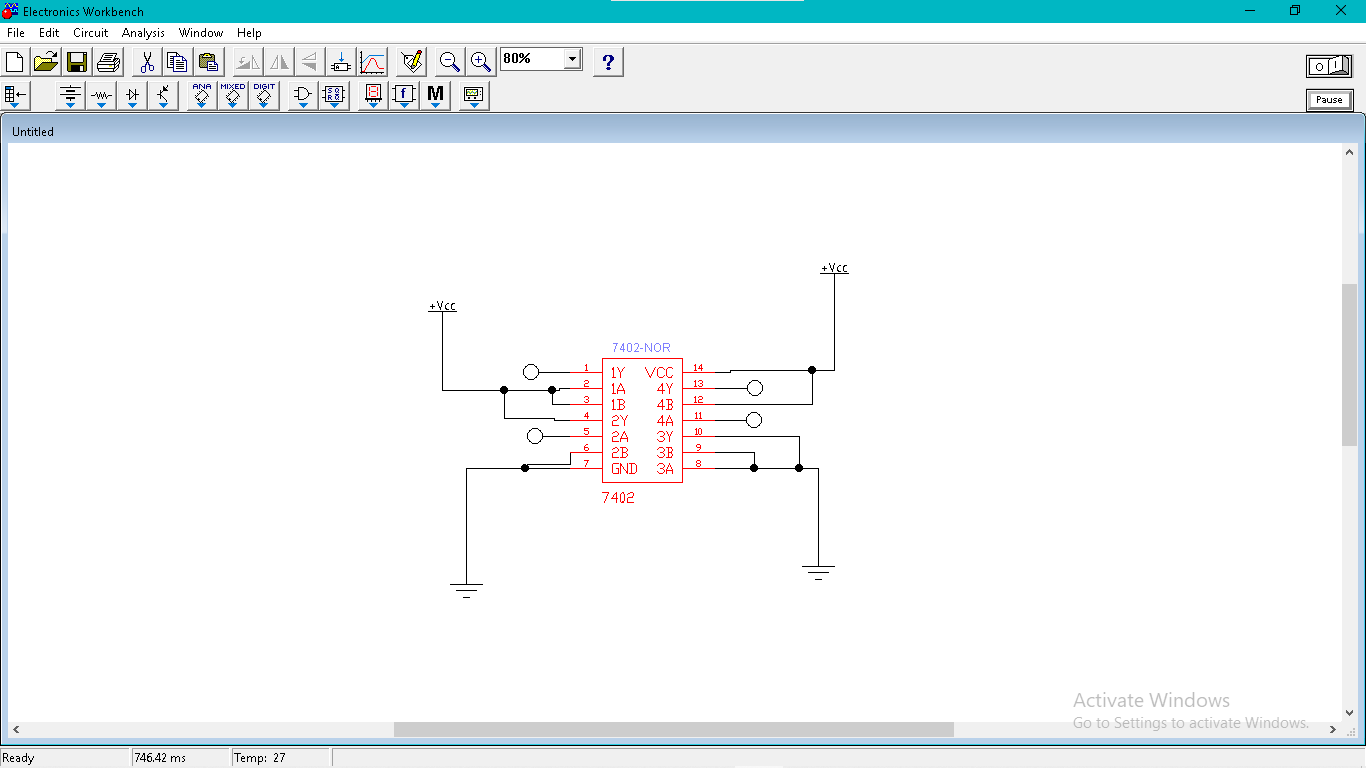


**Truth Table:**

|  |  |  |
| --- | --- | --- |
| **A** | **B** | **X= (~(A·B))** |
| **0** | **0** | **1** |
| **0** | **1** | **1** |
| **1** | **0** | **1** |
| **1** | **1** | **0** |

NOR Gate (7402): Output is 1 only when both inputs are 0. (Y = ~(A + B))

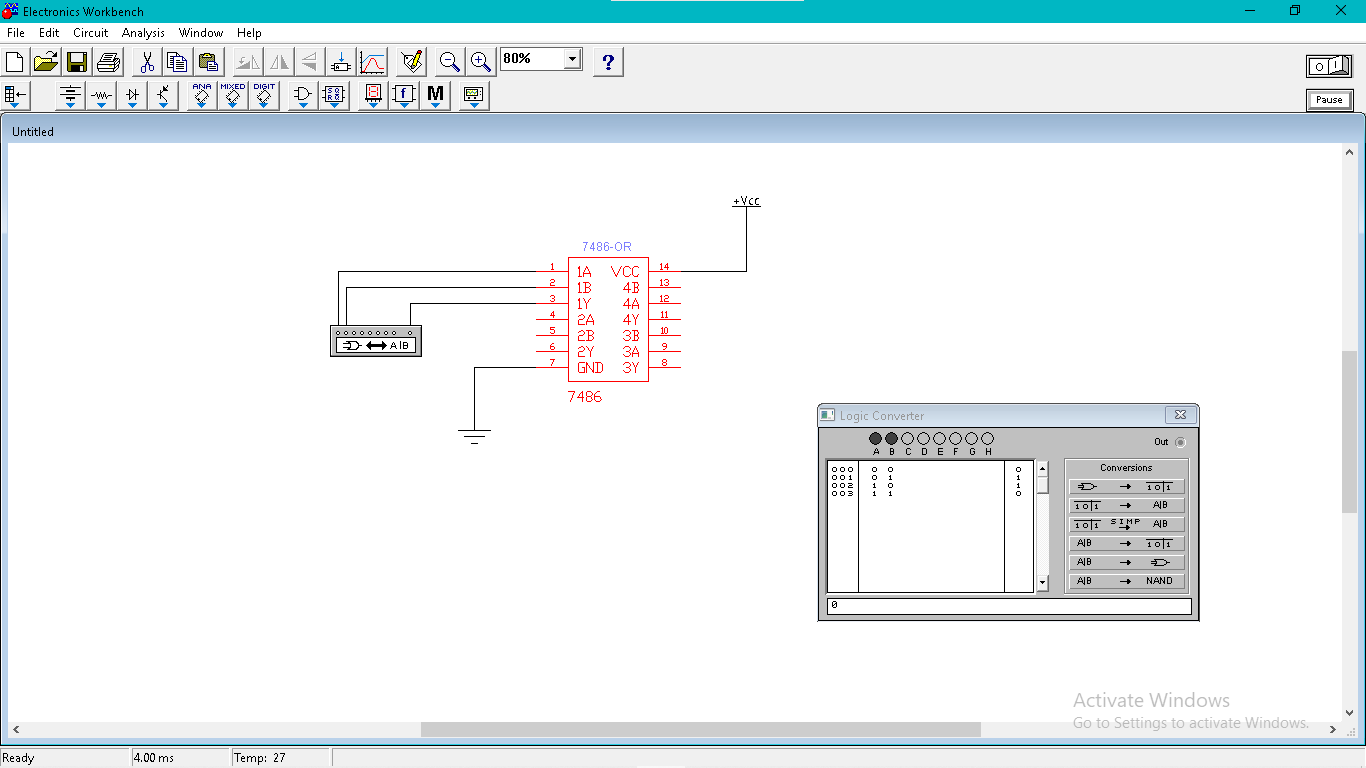


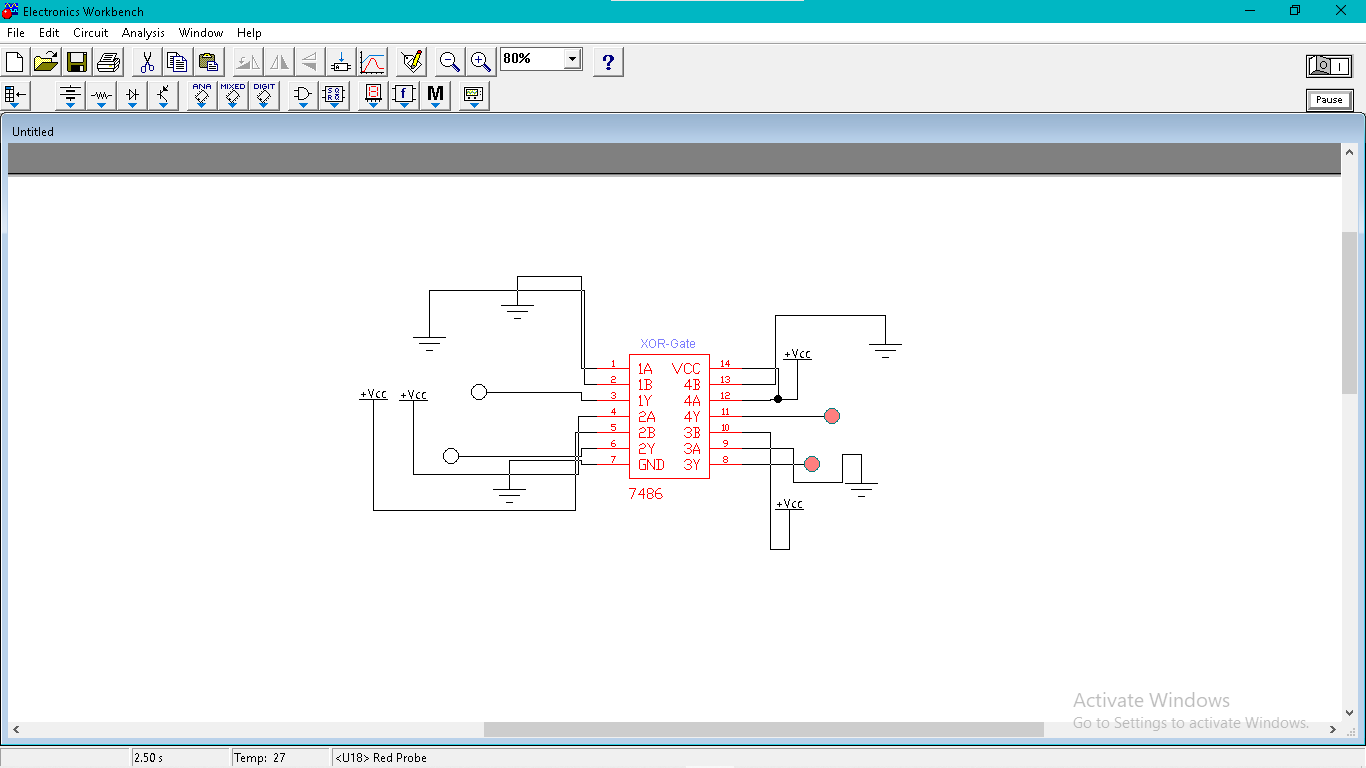


**Truth Table:**

|  |  |  |
| --- | --- | --- |
| **A** | **B** | **X= (~(A+B))** |
| **0** | **0** | **1** |
| **0** | **1** | **0** |
| **1** | **0** | **0** |
| **1** | **1** | **0** |

XOR Gate (7486): Output is 1 when inputs are different. (Y = AB' + A'B)





**Truth Table:**

|  |  |  |
| --- | --- | --- |
| **A** | **B** | **X= (AB' + A'B)** |
| **0** | **0** | **0** |
| **0** | **1** | **1** |
| **1** | **0** | **1** |
| **1** | **1** | **0** |

**Procedure:**

1. First, I opened the EWB (Electronics Workbench) software to design the circuits virtually.

2. Then, I selected he required ICs from the component library:

7408 for AND gate

7432 for OR gate

7404 for NOT gate

7400 for NAND gate

7402 for NOR gate

7486 for XOR gate.

3. After that, I placed a breadboard on the working area to make connections neat and organized.

4. I connected the input pins (A and B) of the gates using logic switches available in the software.

5. I connected the output pin (Y) to a logic indicator or LED to observe the output clearly.

6. I attached the VCC to the power pins of ICs and connected Ground (GND) properly.

7. Then, I checked each gate one by one by giving different combinations of inputs (00, 01, 10, 11) and noted the output.

8. For each combination, I filled the corresponding truth table for:

AND

OR

NOT

NAND

NOR

XOR

9. I verified that the outputs matched the theoretical truth tables of each logic gate.

**Conclusion:**

By completing this lab, we successfully verified the logical operations of basic gates using virtual simulation. We observed how each gate behaves with different input combinations and filled in the truth tables accordingly. This enhances our understanding of digital electronics and logic gate functionality.