# ExperimentNo.\_3\_

**AIM:** 1) To study & verify logic circuit using AOI Logic,

1. To study & verify De Morgan’s theorem.

**APPARATUS:** Connecting wires, power supply, power project board, LED, ICs.

* 1. **AOI LOGIC:**

**THEORY:** AND-OR-Invert (AOI) logic and AOI gates are two-level compound (or complex) logic functions constructed from the combination of one or more AND gates followed by a NOR gate. So, basically here to solve the circuit we must need AND, OR and NAND gate.

Input Ex-NOR gate using AOI logic:

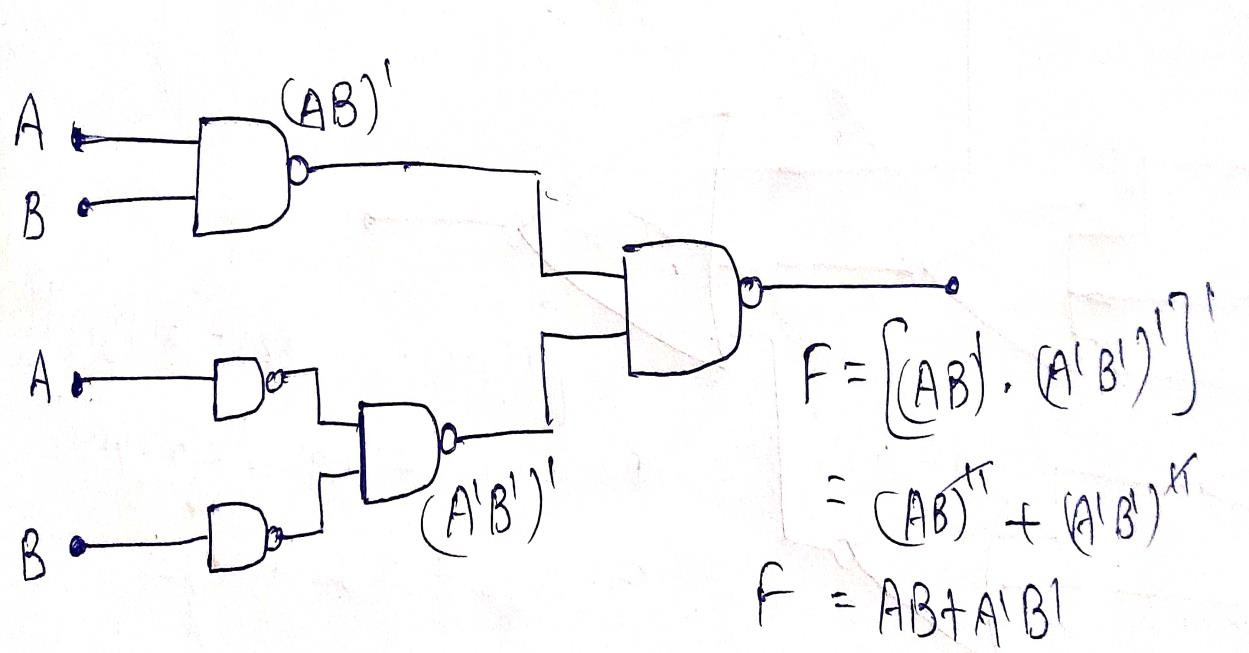
The **Algebraic Expression** for The Ex-NOR gate is **AB’+A’B.**

The Ex-NOR gate is a two input, one output logic circuit. The output is assumed to be1 when both the inputs assume logic 1 state / 0 states.

The output assumes a logic 0 state when one of the inputs assumes a 0 state and the other assumes 1 state.

1. NOR gate is also known as a coincidence gate or equality detector.

### Fig1.Block Diagram = Logic Diagram of AB`+A`B.



**Truth Table:**

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

**Boolean Expression:**

From the truth table following Boolean expression can be derived.

Boolean Expression for X-NOR gate (X) = A X-NOR B = (A ⊕ B)’ = AB’+A’B.

### PROCEDURE:

* 1. Do the connection as per Logic diagram. Connect all IC’s Vcc (Pin 14) and Ground (Pin7) with Power supply properly.
  2. Apply Logical inputs as per truth table with considering Positive Logic (0V for 0 inputand5V for 1 input).
  3. Observe and record the output voltage using DMM or test whether LED is on or off using LED tester in observation table.

4. Verify the working of AOI law by using circuits by comparing truth-table and observation table.



### TinkerCadSimulation (Link & Image):

### Image:

**Fig.2 AOI Logic Circuit**

**Link:<https://www.tinkercad.com/things/Bwfhckifkon-logic>**

**Observation Table:**

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

* 1. **De Morgan’s Theorem:**

**THEORY: (**A+B)’ = A’B’ and (AB)’ = A’+B’

The most important logic theorem for digital electronics, this theorem says that any logical binary expression remains unchanged if we:

* Change all variables to their complements.
* Change all AND operations to ORs.
* Change all OR operations to ANDs.
* Take the complement of the entire expression.

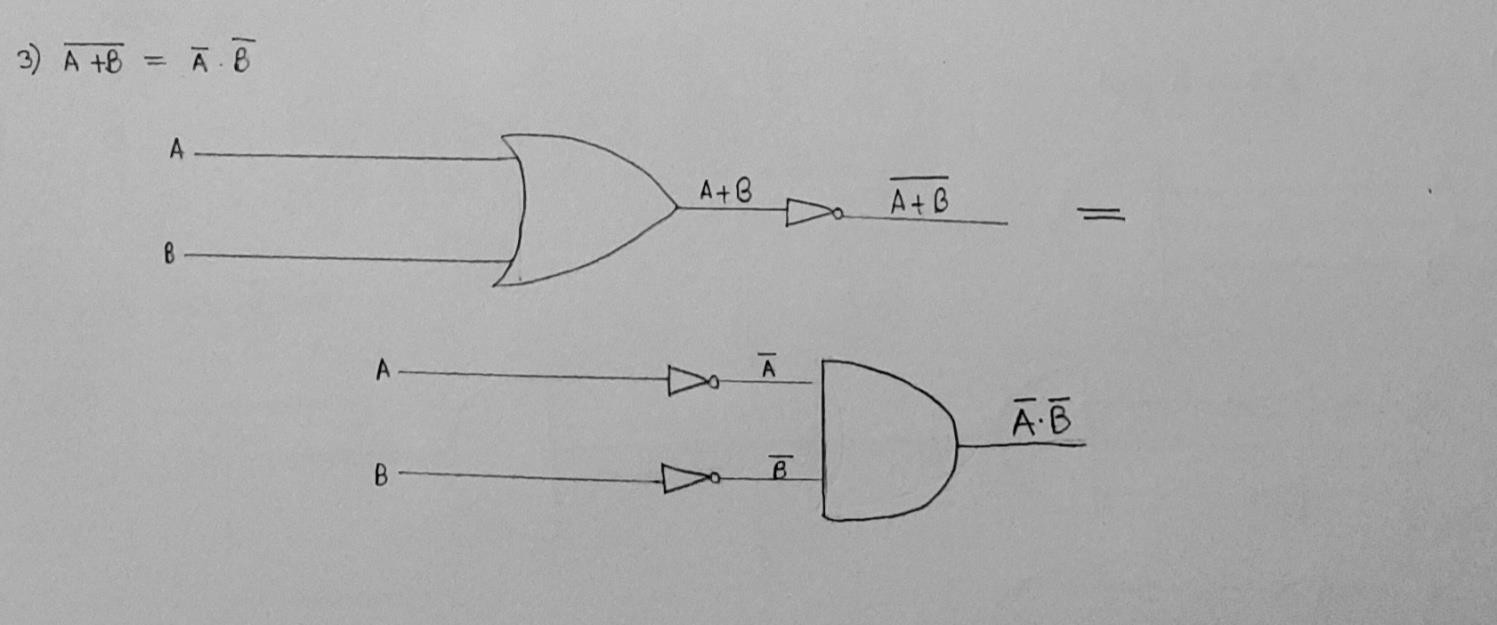
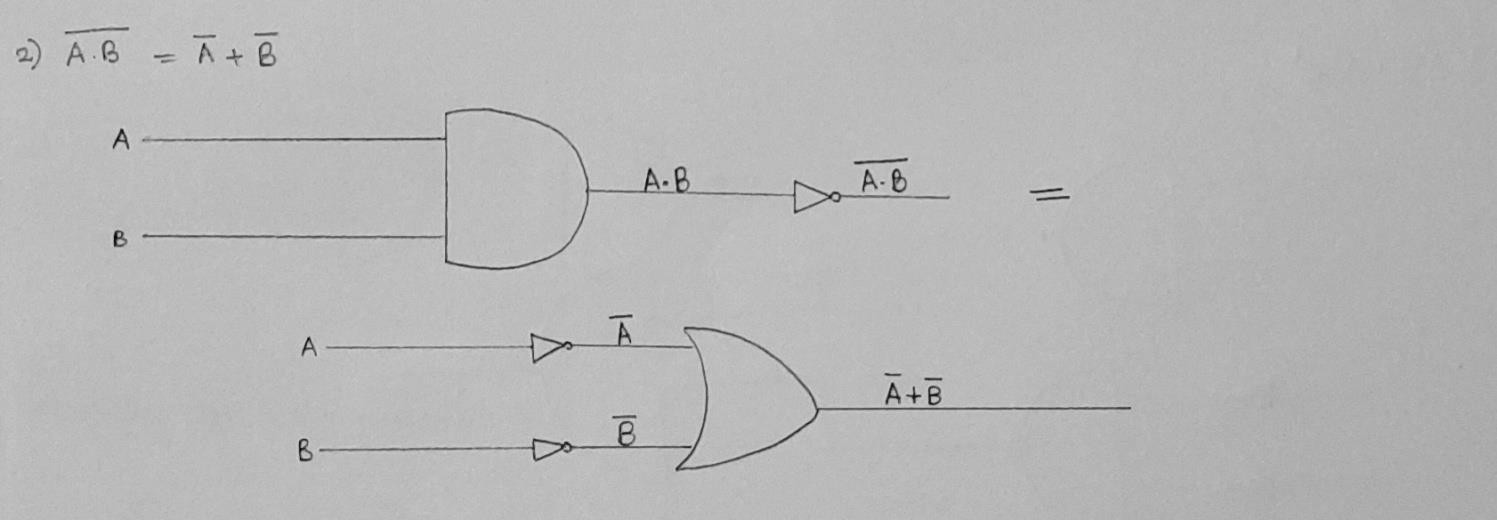
A practical operational way to look at De Morgan's Theorem is that the inversion bar of an expression may be broken at any point and the operation at that point replaced by its opposite

(i.e., AND replaced by OR or vice versa).

Two forms of De Morgan's Theorem implemented with basic gates.

**De Morgan Applications:** De Morgan's Theorem is useful in the implementation of the basic gate operations with alternative gates, particularly with NAND and NOR gates which are readily available in IC form.

### Fig1.Block Diagram = Logic Diagram of (A+B)’ = A’B’ and (AB)’ = A’+B’.



**Truth Table:**



**Boolean Expression:**

) (A+B)’ = A’B’,

) (AB)’ = A’+B’.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **A** | **B** | **A`** | **B`** | **A+B** | **A.B** | **(A+B)`** | **(A.B)`** | **A`+B`** | **A`.B`** |
| **1** | **1** | **0** | **0** | **1** | **1** | **0** | **0** | **0** | **0** |
| **1** | **0** | **0** | **1** | **1** | **0** | **0** | **1** | **1** | **0** |
| **0** | **1** | **1** | **0** | **1** | **0** | **0** | **1** | **1** | **0** |
| **0** | **0** | **1** | **1** | **0** | **0** | **1** | **1** | **1** | **1** |

1

2

**PROCEDURE:**

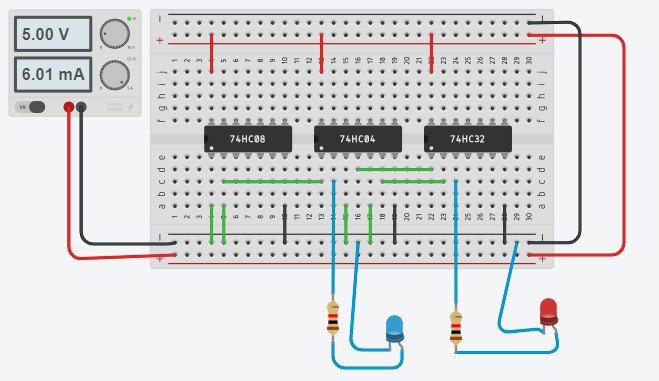
## Do the connection as per Logic diagram. Connect all IC’s Vcc (Pin 14) and Ground (Pin 7) with Power supply properly.

1. Apply Logical inputs as per truth table with considering Positive Logic (0V for 0 input and 5V for 1 input).
2. Observe and record the output voltage using DMM or test whether LED is on or off using LED tester in observation table.
3. Verify the working of De Morgan’s theorem by comparing truth table and observation table.

### TinkerCadSimulation (Link & Image):

(AB)’ = A’+B’.

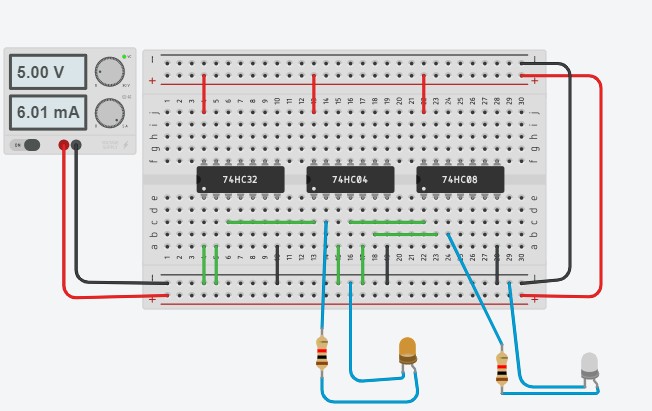
### Image:



**Link:** [**https://www.tinkercad.com/things/bgjvlHMikAYrK-demorganexp/editel**](https://www.tinkercad.com/things/bgjvlHMikAYrK-demorganexp/editel)

(A+B)’ = A’B’,

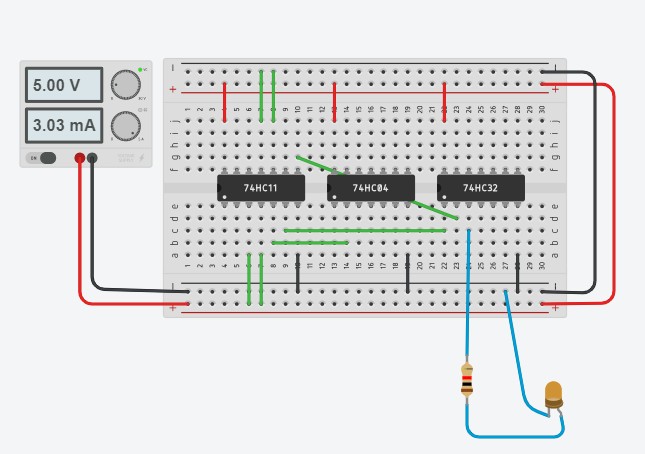
### Image:



**Link:** [**https://www.tinkercad.com/things/LdkcsOH1GY4p-demorgan2exp/editel**](https://www.tinkercad.com/things/LdkcsOH1GY4p-demorgan2exp/editel)

**R=XYZ’+YZ**

**Image:**



**Link:** [**https://www.tinkercad.com/things/Jdkcsdkc8UV-QUE1/editel**](https://www.tinkercad.com/things/Jdkcsdkc8UV-QUE1/editel)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **A** | **B** | **A`** | **B`** | **A+B** | **A\*B** | **(A+B)`** | **(A\*B)`** | **A`+B`** | **A`\*B`** |
| **1** | **1** | **0** | **0** | **1** | **1** | **0** | **0** | **0** | **0** |
| **1** | **0** | **0** | **1** | **1** | **0** | **0** | **1** | **1** | **0** |
| **0** | **1** | **1** | **0** | **1** | **0** | **0** | **1** | **1** | **0** |
| **0** | **0** | **1** | **1** | **0** | **0** | **1** | **1** | **1** | **1** |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **X** | **Y** | **Z** | **Z’** | **XY** | **XYZ’** | **YZ** | **R=XYZ’+YZ** |
| **1** | **1** | **1** | **0** | **1** | **0** | **1** | **1** |
| **1** | **1** | **0** | **1** | **1** | **1** | **0** | **1** |
| **1** | **0** | **1** | **0** | **0** | **0** | **0** | **0** |
| **1** | **0** | **0** | **1** | **0** | **0** | **0** | **0** |
| **0** | **1** | **1** | **0** | **0** | **0** | **0** | **0** |
| **0** | **1** | **0** | **1** | **0** | **0** | **1** | **1** |
| **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** |
| **0** | **0** | **0** | **1** | **0** | **0** | **0** | **0** |

**CONCLUSION:**

I have learnt and made circuit of XOR Gate using (NOT, AND, OR Gates) which follows AOI Logic and De Morgan’s theorem using (NAND, NOT, OR Gates).

# ExperimentNo.\_4\_

**AIM:** To Verify Operation Of NAND & NOR Gate As Universal Gates.

**APPARATUS:** Logic Gate, ICs, Connecting-wires, Bread Board, Power supply, LED, Resistor.

**THEORY:** AND, NOT and OR gates are the basic gates. We can create any logic gate or any Boolean expression by combining them. Now NOR and NAND gates have the particular property that any one of them can create any logical Boolean expression if designed in a proper way.

This is the circuit diagram of a NAND gate used to make work like a NOT gate, the original logic gate diagram of NOT gate is given beside.

NOR gate as universal gate:

The above diagram is of an OR gate made by only using NOR gates. The output of this gate is exactly similar to that of a single OR gate. As we can see the circuit arrangement of OR gate using NOR gates is similar to that of AND gate using NAND gates.

So, from the above discussion it is clear that all the three basic gates (AND, OR, NOT) can be made by only using NOR gate or only using NAND gate. And thus it can be aptly termed as Universal Gate.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Truth Table:**  **1.Obtaining AND using NAND.** | | | | | |
|  | **X** | **Y** | **F=F’** | **ANS=(F’)’** |  |
| **1** | **1** | **0** | **1** |

|  |  |  |  |
| --- | --- | --- | --- |
| **1** | **0** | **1** | **0** |
| **0** | **1** | **1** | **0** |
| **0** | **0** | **1** | **0** |

### Obtaining OR using NAND.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **X** | **X’** | **Y** | **Y’** | **F** | **ANS=F’** |
| **1** | **0** | **1** | **0** | **0** | **1** |
| **1** | **0** | **0** | **1** | **0** | **1** |
| **0** | **1** | **1** | **0** | **0** | **1** |
| **0** | **1** | **0** | **1** | **1** | **0** |

1. **Obtaining AND using NOR.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **X** | **X’** | **Y** | **Y’** | **F** | **ANS=F’** |
| **1** | **0** | **1** | **0** | **0** | **0** |
| **1** | **0** | **0** | **1** | **1** | **1** |
| **0** | **1** | **1** | **0** | **1** | **1** |
| **0** | **1** | **0** | **1** | **1** | **1** |

1. **Obtaining OR using NOR.**

|  |  |  |  |
| --- | --- | --- | --- |
| **X** | **Y** | **F=F’** | **ANS=(F’)’** |
| **1** | **1** | **0** | **1** |
| **1** | **0** | **0** | **1** |
| **0** | **1** | **0** | **1** |
| **0** | **0** | **1** | **0** |

1. **AB+A’B’**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **A** | **B** | **A’** | **B’** | **A\*B** | **A’\*B’** | **F** | **F’** |
| **1** | **1** | **0** | **0** | **1** | **0** | **0** | **1** |
| **1** | **0** | **0** | **1** | **0** | **0** | **0** | **1** |
| **0** | **1** | **1** | **0** | **0** | **0** | **0** | **1** |
| **0** | **0** | **1** | **1** | **0** | **1** | **0** | **1** |

1. **XYZ+X’Y’Z+X’YZ’**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **X** | **Y** | **Z** | **X’** | **Y’** | **Z’** | **XYZ** | **XY’Z** | **X’YZ’** | **F** |
| **1** | **1** | **1** | **0** | **0** | **0** | **1** | **0** | **0** | **1** |
| **1** | **1** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | **0** |
| **1** | **0** | **1** | **0** | **1** | **0** | **0** | **1** | **0** | **1** |
| **1** | **0** | **0** | **0** | **1** | **1** | **0** | **0** | **0** | **0** |
| **0** | **1** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** |
| **0** | **1** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** |
| **0** | **0** | **1** | **1** | **1** | **0** | **0** | **0** | **0** | **0** |
| **0** | **0** | **0** | **1** | **1** | **1** | **0** | **0** | **0** | **0** |



**BOOLEAN EXPRESSION:**

From The Truth Table, Following Boolean Expression Can Be Derived:

1. ((A.B)’)’ = A\* B

2. (A’+B’)’=A\*B

3. (A’\*B’)’=A+B

4. ((A+B)’)’=A+B

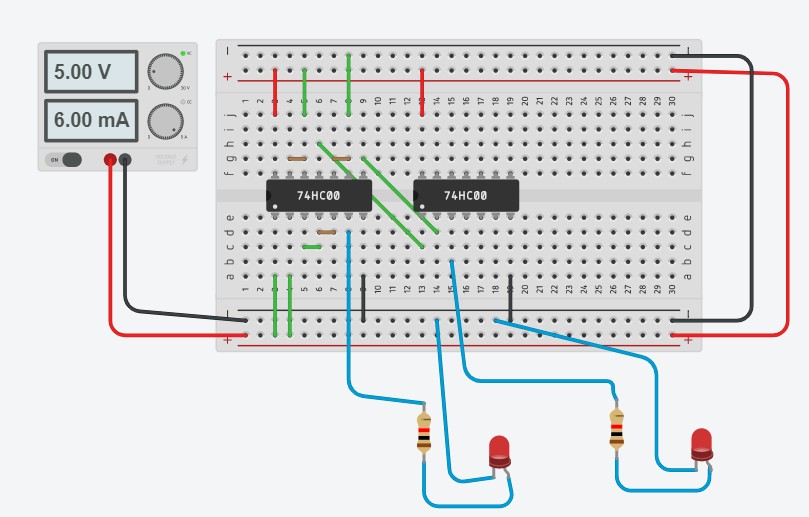
1. AB+A’B’
2. XYZ+X’Y’Z+X’YZ’

### PROCEDURE:

1. Mount ICs 7400, 7402 on the power project board.
2. Connect pin number 7 and 14 of all ICs to ground and +5V supply respectively.
3. Make the connection as shown in the logic diagram.
4. Observe and record the outputs.
5. Verify the truth table of all the data.

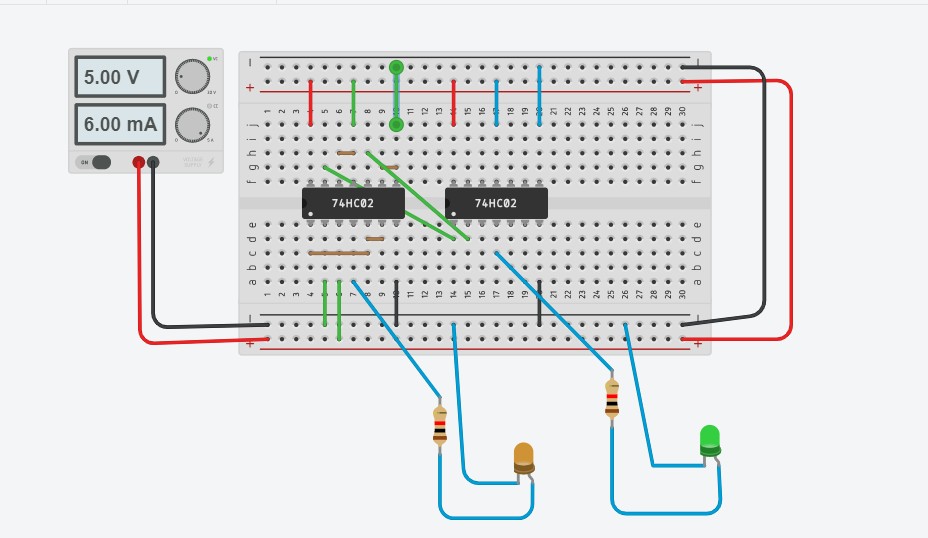
### Tinker CAD and images:

1 and 2. Giving AND and OR using NOR Gate: Image:



Link: <https://www.tinkercad.com/things/5cjnkhksHxnsN-NandtoAnd/editel>

3 and 4. Giving AND and OR using NOR Gate: Image:

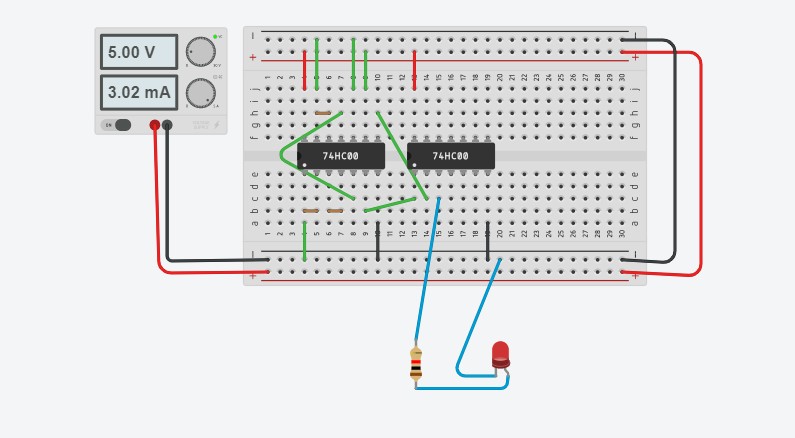


Link:

<https://www.tinkercad.com/things/SDCSMxcn5x8ccT-andor/editel>.

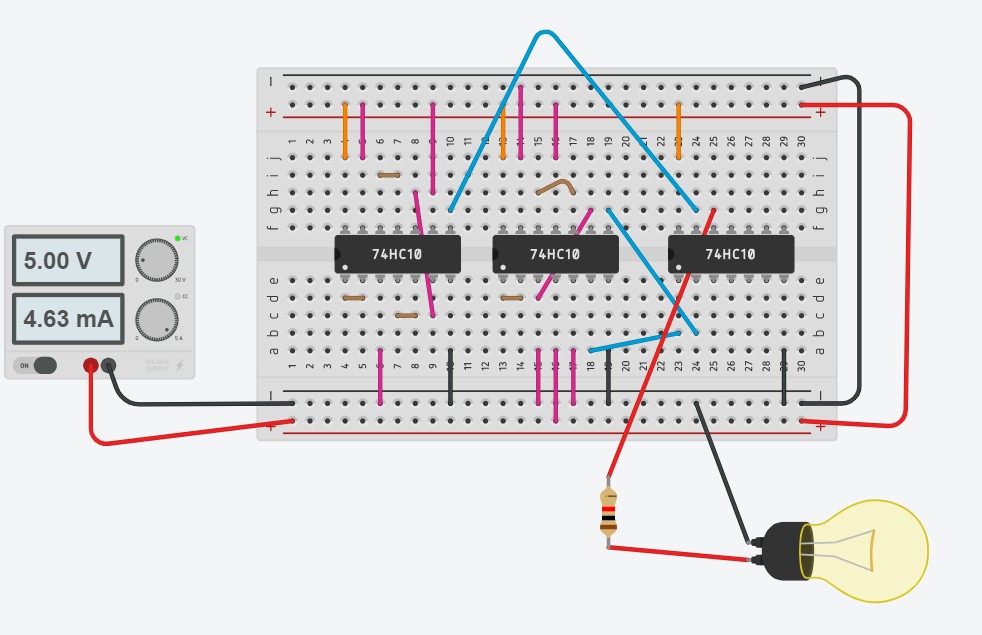
5.AB+A’B’

Image:



Link: https://www.tinkercad.com/things/0AMNpmVcohN-newexp /editel.

1. XYZ+X’Y’Z+X’YZ Image:



Link: https://www.tinkercad.com/things/jkitlHmf2oYM-exp/editel

**CONCLUSION:** We have study that how to implement AND, NOT, OR gate using NAND & NOR gate and perform Tinker CAD simulation and verify our circuit from truth table of AND, NOT, OR gate.