**Experiment / Assignment / Tutorial No. 1**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |  |
| --- | --- |
| **Batch: B-1 Roll No.: 16010122104** | **Experiment / assignment / tutorial No.: 1** |
|  |  |
| **Title:** Basic Gates & Universal Gates |  |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Objective:** To study the basic gates: AND, OR, NOT and universal gates: NAND, NOR,

XOR, XNOR

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Expected Outcome of Experiment:**

**CO1:** Recall basic gates and binary, octal & hexadecimal calculations and conversions. **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_****\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** **Books/ Journals/ Websites referred:**

* Vlab Link: http://vlabs.iitkgp.ernet.in/dec/#
* R. P. Jain, “Modern Digital Electronics”, Tata McGraw Hill
* http://www.ee.surrey.ac.uk/Projects/Labview/gatesfunc/
* http://www.electronics-tutorials.ws/boolean/bool\_6.html

**Pre Lab/ Prior Concepts:**

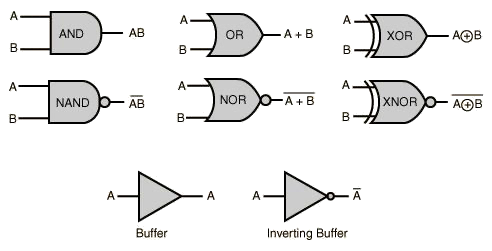
Gate is a logic circuit with one or more inputs but only one output. Gates are digital (two state) circuit because the input & output are either low or high. Gates provide high output for certain combinations of input & for other combinations the output is low. Total number of combinations for a gate is 2^n; where n is number of input.

**Classification:** The two types of gate are:

1. **Basic or Fundamental Gates:**

1. **Derived Gates:**

# Symbols of gates



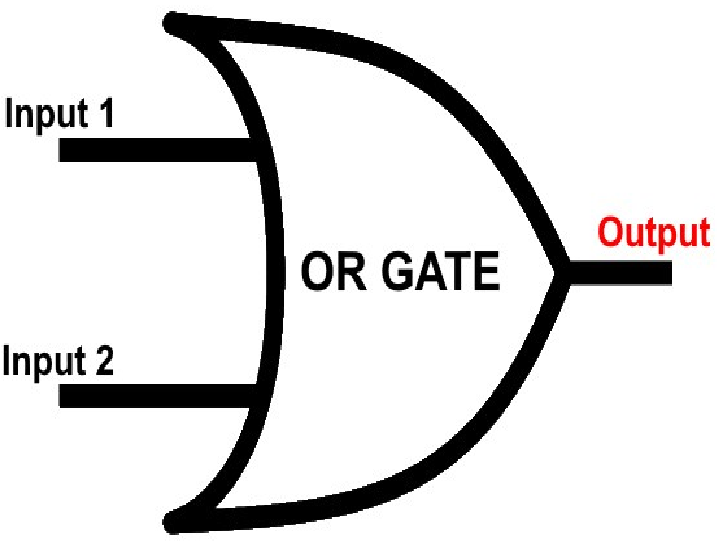
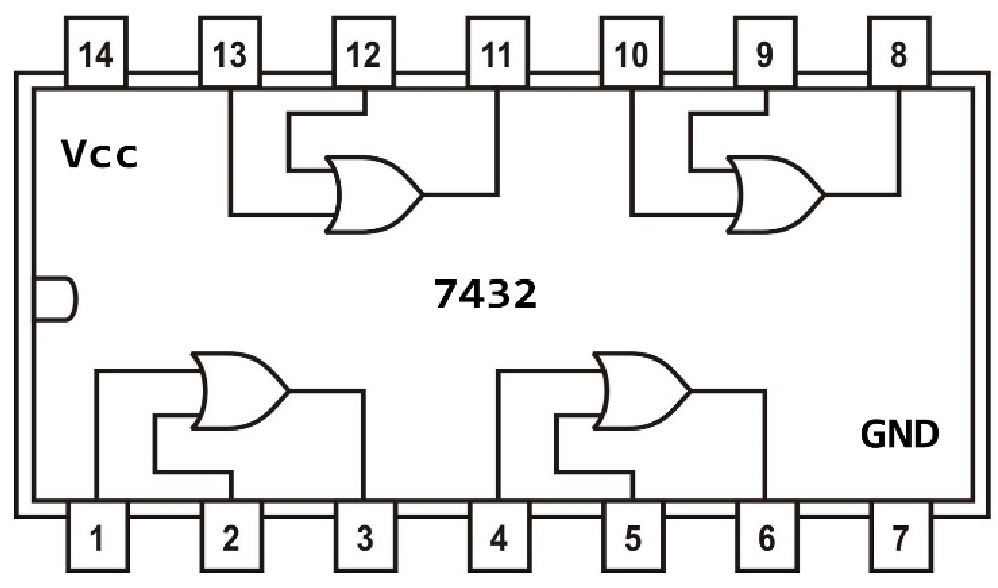
**Implementation Details:**

# Basic Gates

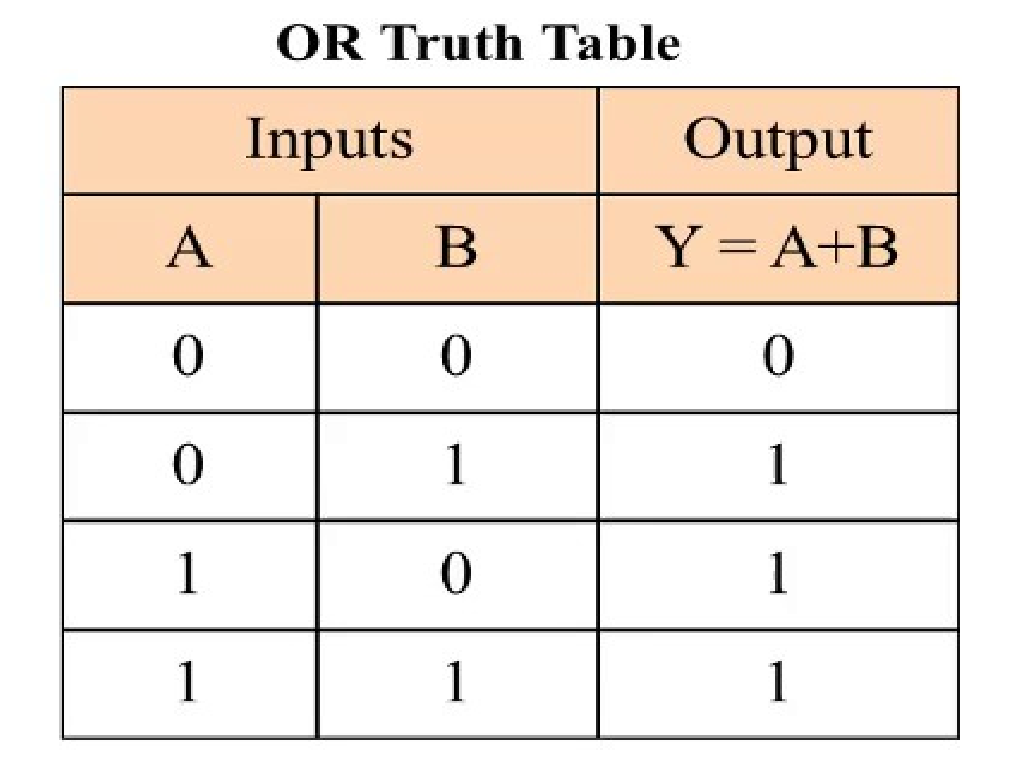
1. **OR gate:** The OR gate has two or more inputs but only 1 output. If any or all the inputs are high, the output is high. If all the inputs are low, the output is low.

# Y= A + B

**Symbol for OR gate** **Pin Diagram For IC 7432**

The truth table for OR operations are:

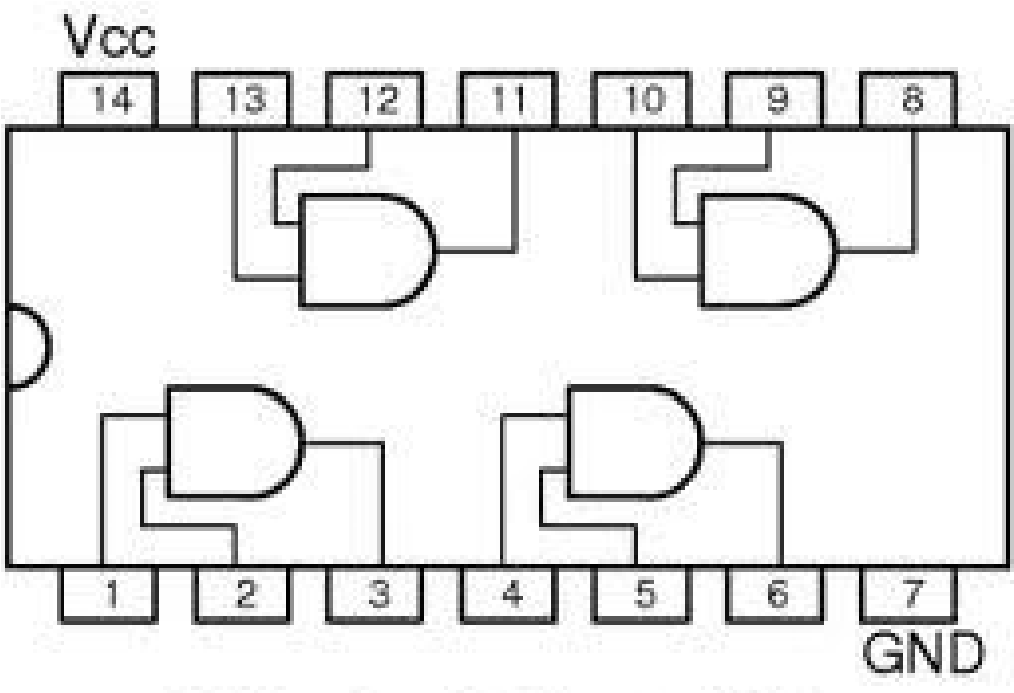


2. **AND gate:** The AND gate has two or more inputs but only one output. If any or all inputs are high then output is also high

Y = A.B

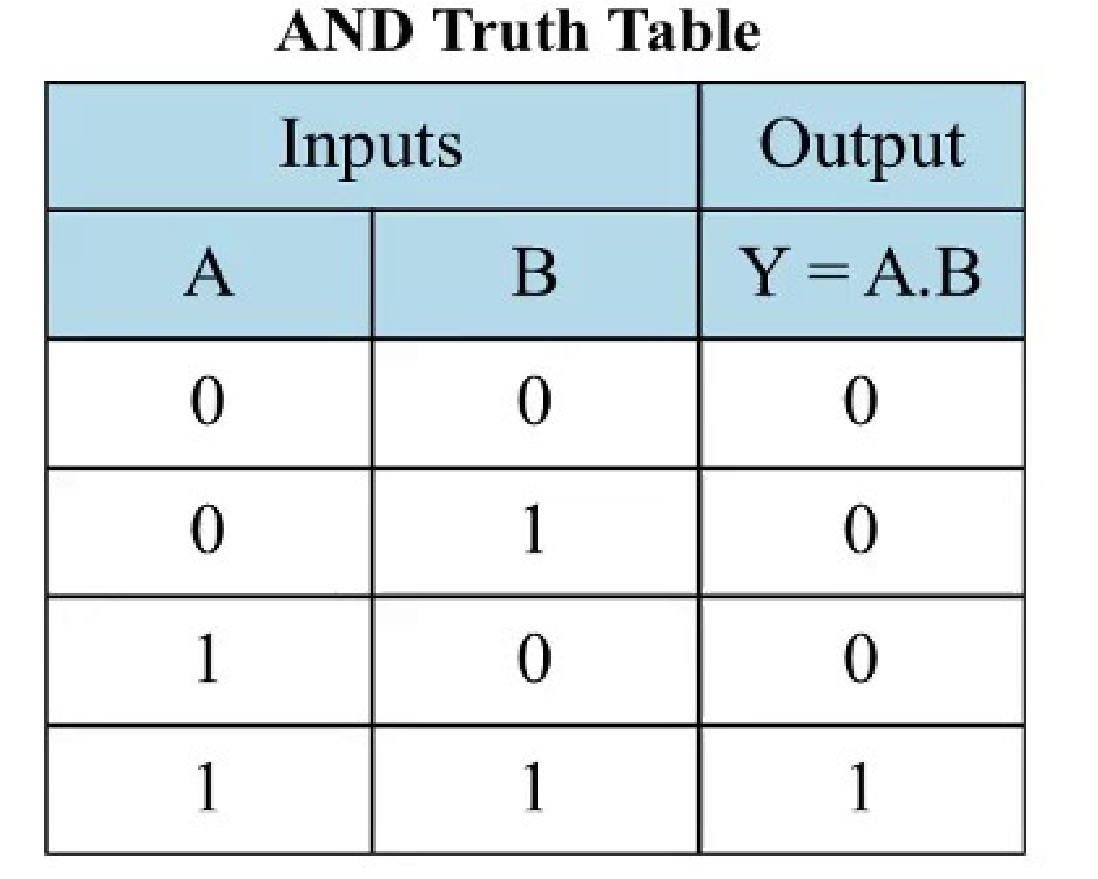
## Symbol for AND gate Pin Diagram For IC 7408





**7408**

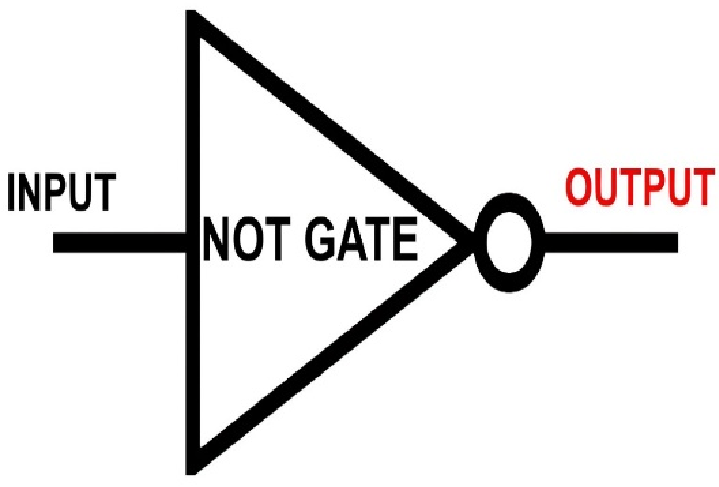
The truth table for AND operations are:

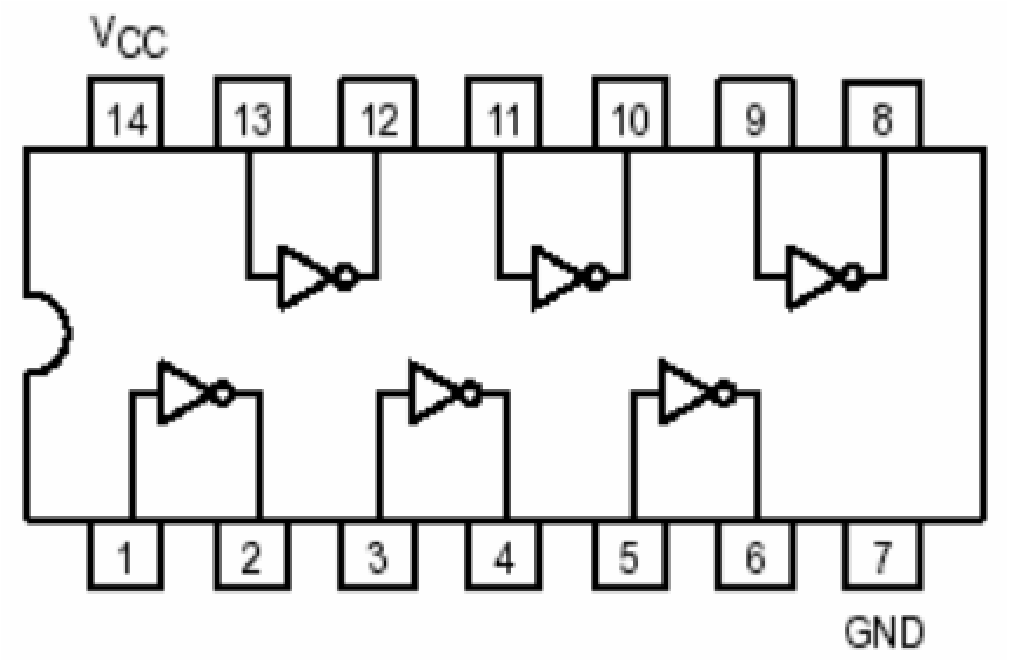


3. **NOT gate:** The Not gate is a gate with only one input and one output. The output is always in opposite state of an input. A NOT gate is also called as Inverter because it performs inversion.

Y= 𝐴

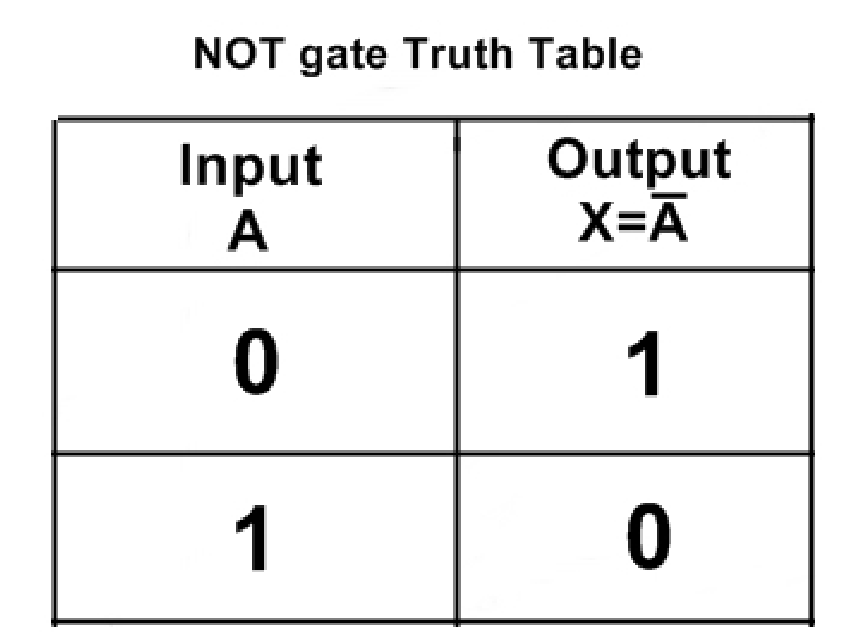
## Symbol for NOT gate Pin Diagram For IC 7404





**7404**

The truth table for NOT operations is:



## Derived Gates/Universal Gates

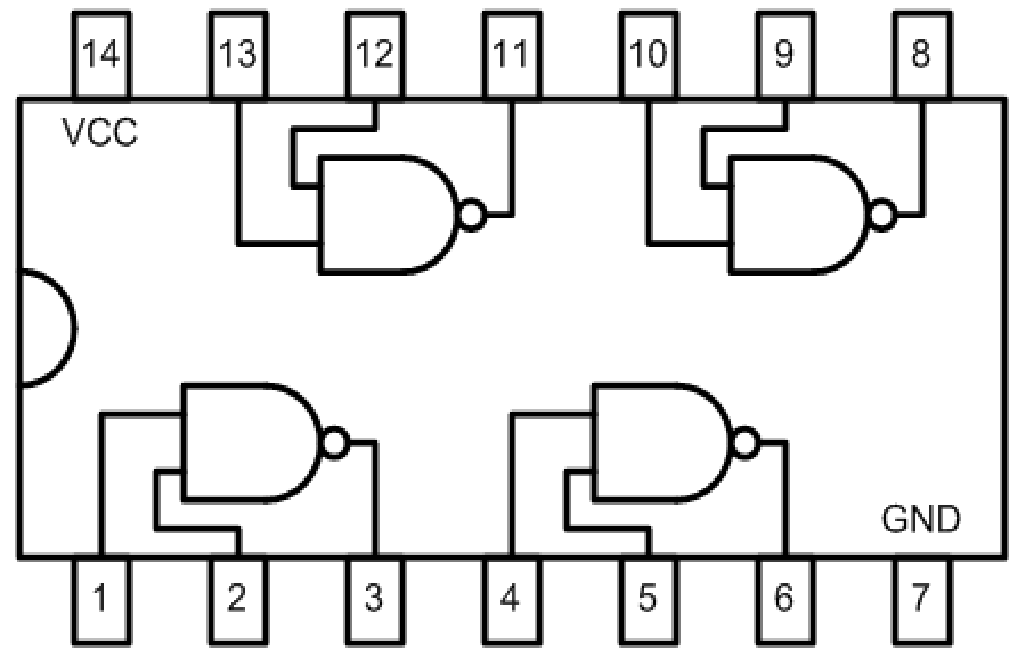
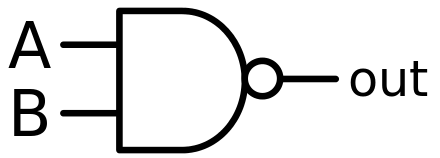
* NAND gate
* NOR gate
* EX-OR gate
* EX-NOR gate

**1. NAND gate:** This is a NOT-AND gate which is equal to an AND gate followed by a NOT gate. The outputs of all NAND gates are high if any of the inputs are low. The symbol is an AND gate with a small circle on the output. The small circle represents inversion.

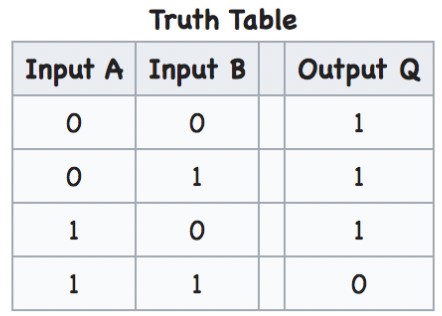
# Y= A.B

## 

## Symbol Pin Diagram for IC 7400



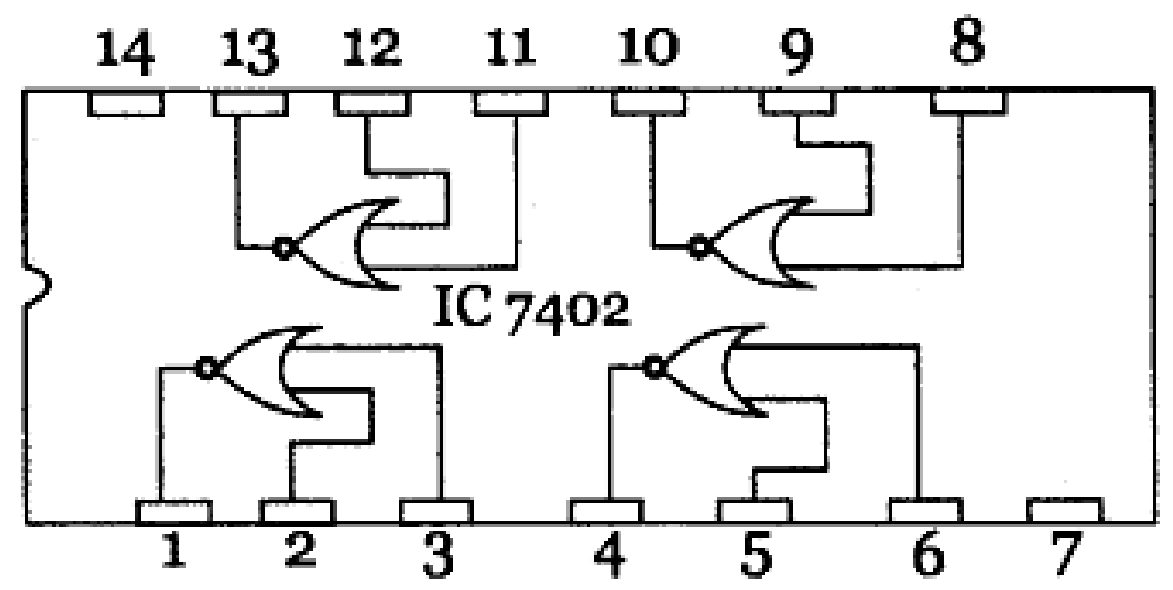
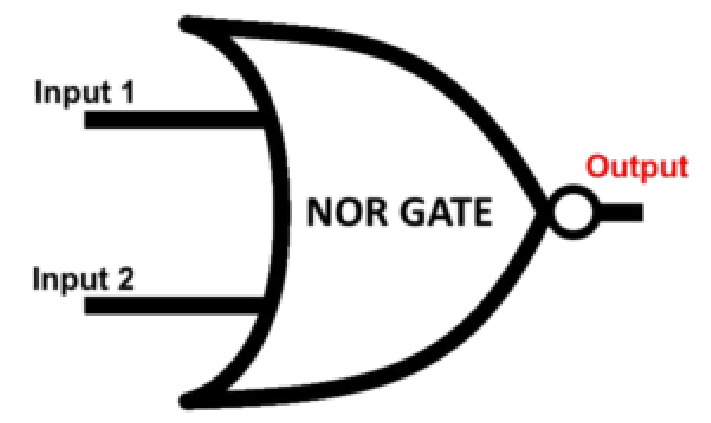
The truth table for NAND operations is:



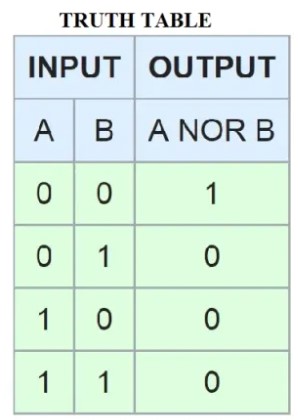
**2. NOR gate:** This is a NOT-OR gate which is equal to an OR gate followed by a NOT gate. The outputs of all NOR gates are low if any of the inputs are high. The symbol is an OR gate with a small circle on the output. The small circle represents inversion.

# Y= 𝐴 + 𝐵

## Symbol for NOR gate Pin Diagram For IC 7402



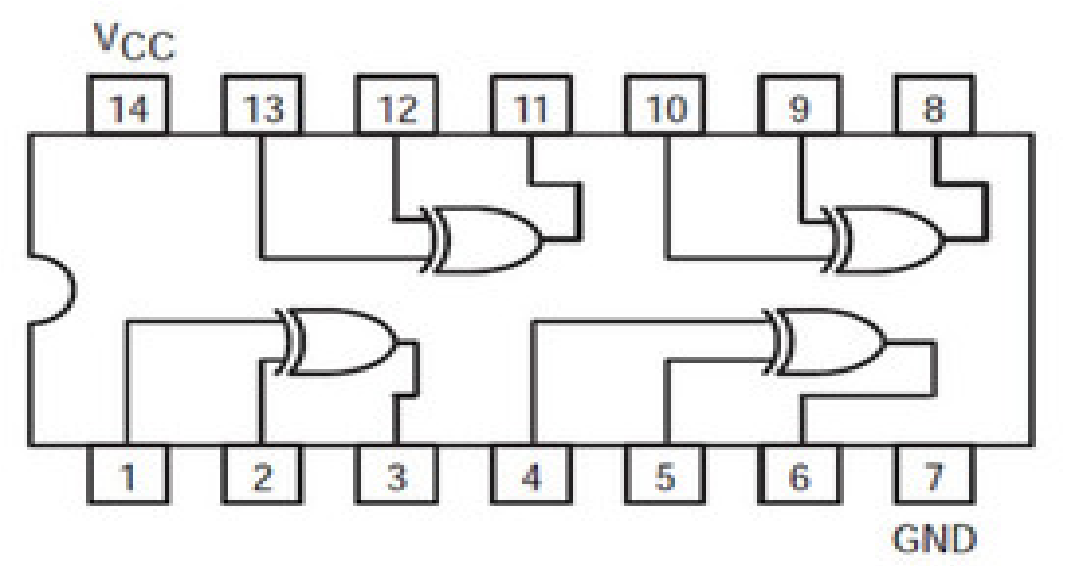
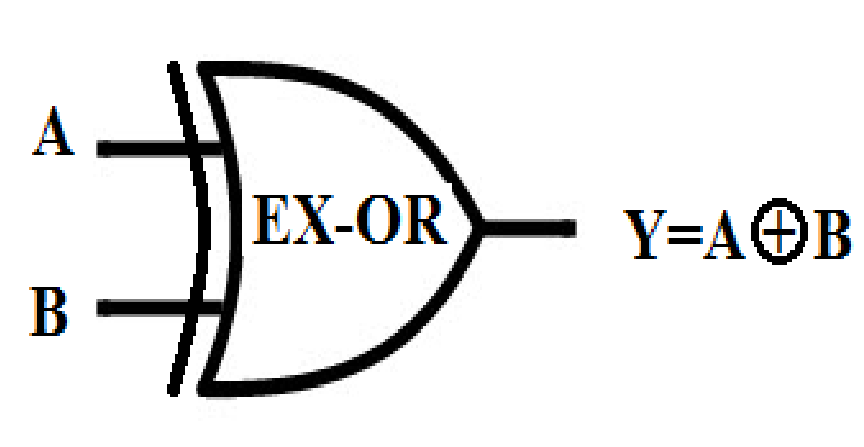
The truth table for NOR operations are:



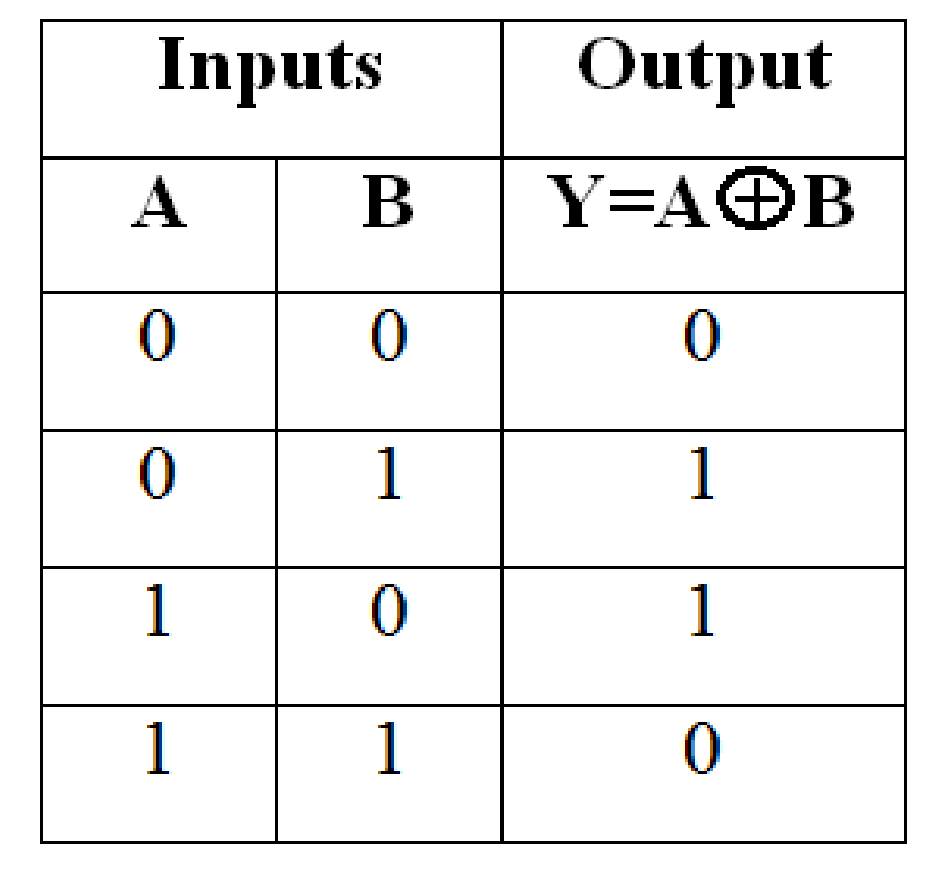
**3. EX-OR gate**: The 'Exclusive-OR' gate is a circuit which will give a high output if either, but not both, of its two inputs are high. An encircled plus sign (  ) is used to show the EX-OR operation

# Y= 𝐴⨁𝐵

## Symbol for Ex-OR gate Pin Diagram For IC 7486



The truth table for XOR operations is:

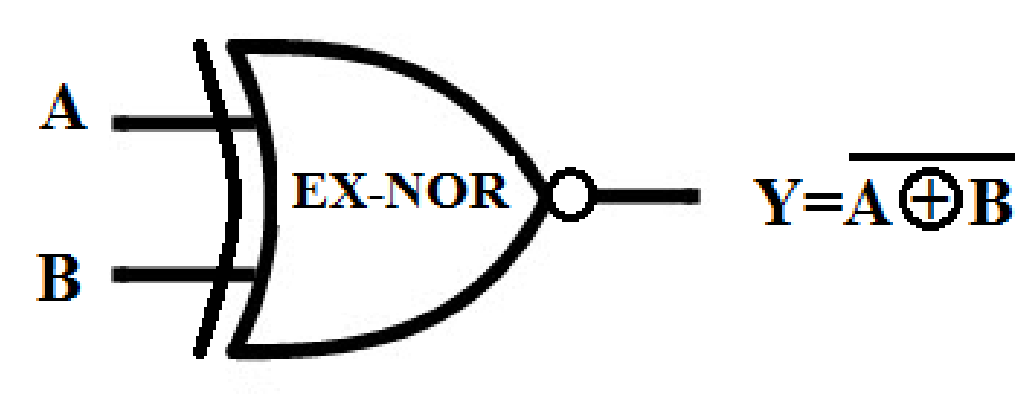


1. **EX-NOR gate**: The 'Exclusive-NOR' gate circuit does the opposite to the EOR gate. It

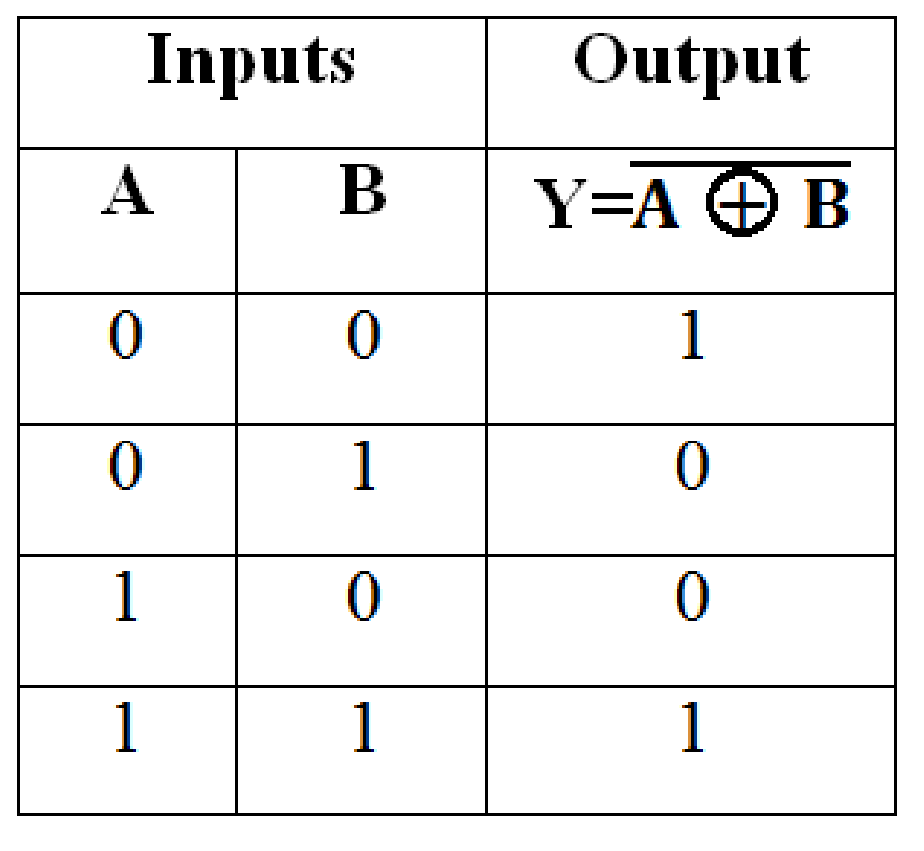
will give a low output if either, but not both, of its two inputs are high. The symbol is an EXOR gate with a small circle on the output. The small circle represents inversion

Y= 𝐴⨁𝐵

## Symbol for Ex-NOR gate



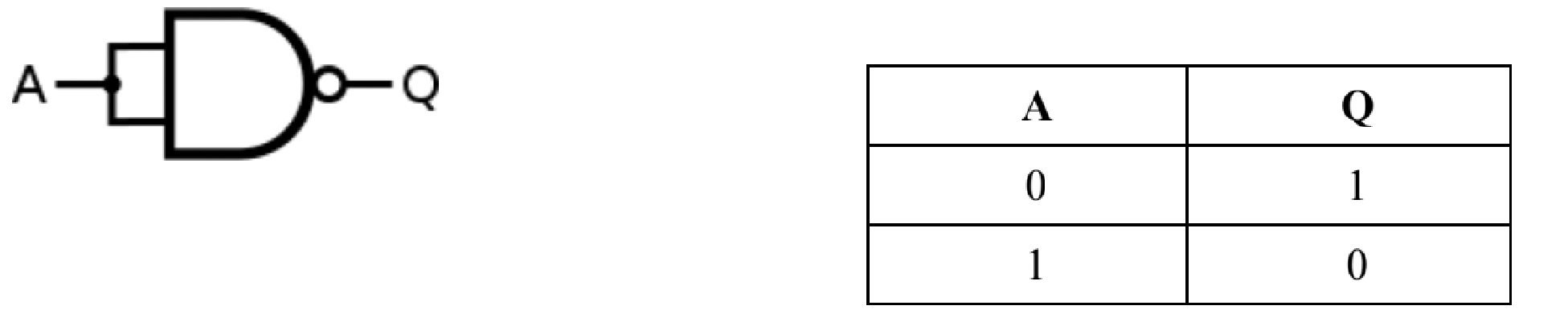
The truth table for XNOR operations is:



**Implementation Using NAND Gate**

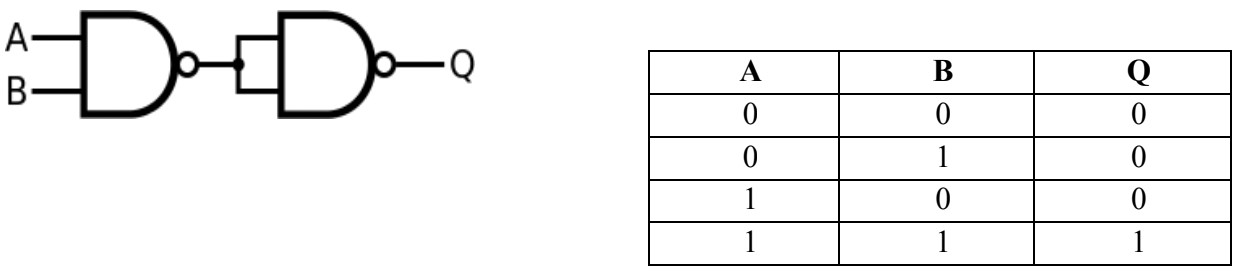
# NOT GATE

A NOT gate is made by joining the inputs of a NAND gate together. Since a NAND gate is equivalent to an AND gate followed by a NOT gate, joining the inputs of a NAND gate leaves only the NOT gate.



# AND GATE

An AND gate is made by inverting the output of a NAND gate as shown below.



# OR GATE

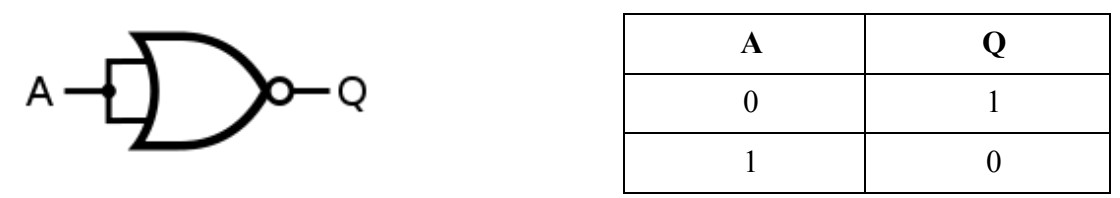
If the truth table for a NAND gate is examined or by applying De Morgan's Laws, it can be seen that if any of the inputs are 0, then the output will be 1. To be an OR gate, however, the output must be 1 if any input is 1. Therefore, if the inputs are inverted, any high input will trigger a high output.



**IMPLEMENTATION USING NOR GATE**

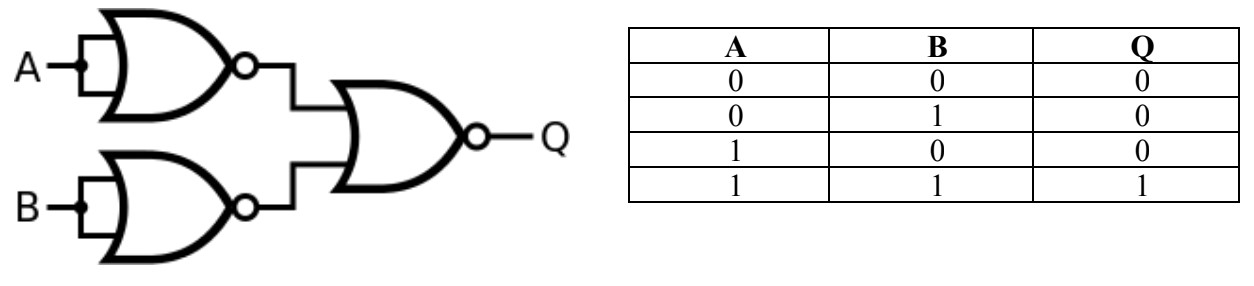
# NOT GATE

This is made by joining the inputs of a NOR gate. As a NOR gate is equivalent to an OR gate leading to NOT gate, this automatically sees to the "OR" part of the NOR gate, eliminating it from consideration and leaving only the NOT part.



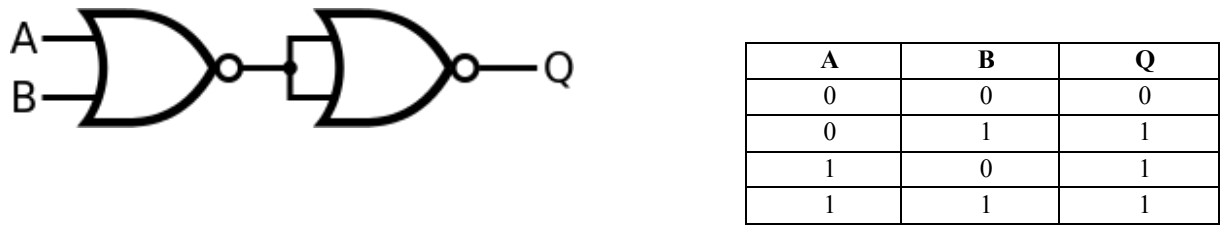
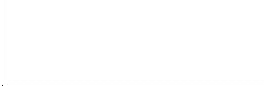
# AND GATE

An AND gate gives a 1 output when both inputs are 1. Therefore, an AND gate is made by inverting the inputs of a NOR gate. Again, note that a NOT gate is equivalent to a NOR with its inputs joined.



# OR GATE

An OR gate is made by inverting the output of a NOR gate. Note that we already know that a NOT gate isequivalent to a NOR gate with its inputs joined.



**Conclusion:**

Studied about logic gates and how they work in combinations. The basic gates: AND, OR, NOT and universal gates: NAND, NOR, XOR, XNOR

## Post Lab Descriptive Questions

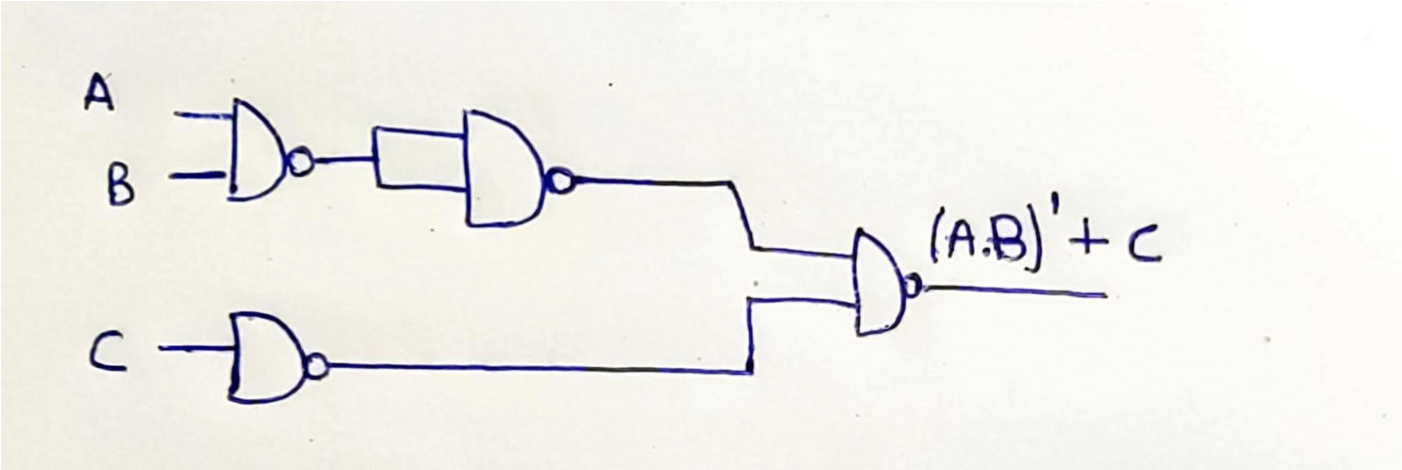
1. Verify the expression (A∙B)' + C by:

1. Using NAND Gate directly.
2. Using AND & NOT gate consecutively.

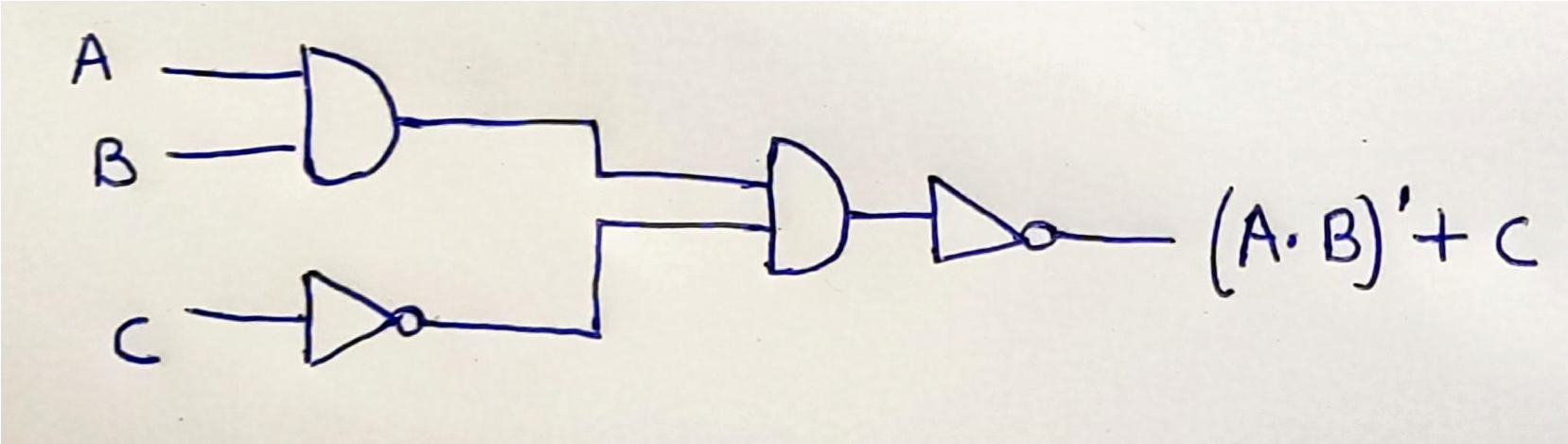
## Truth Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A | B | C | A.B | (A.B)’ | (A.B)’ + C |
| 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 1 |

1. Using NAND Gate directly



1. Using AND & NOT gate consecutively.



2. Implement the following expressions using combination of gates:

a) (A'+B)∙B

## Truth Table

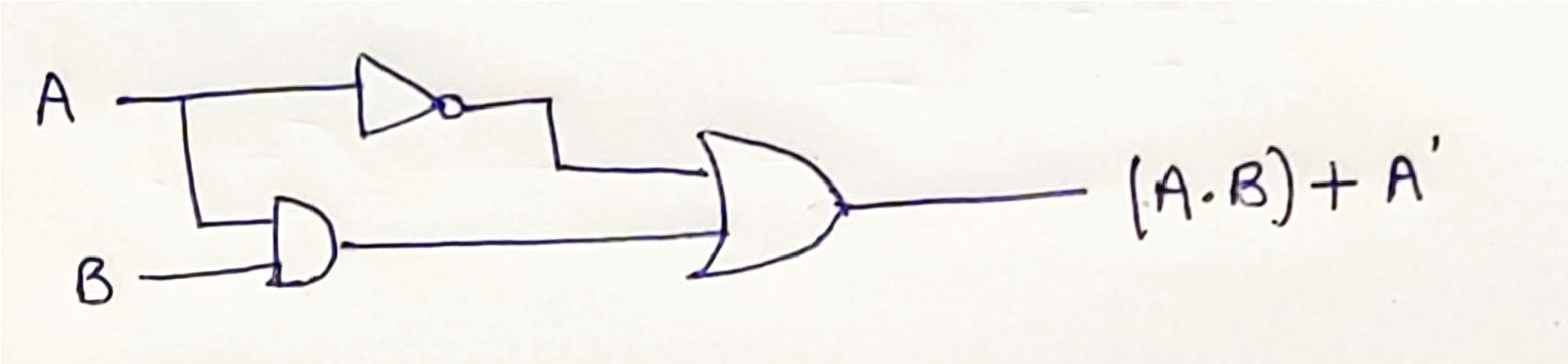
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | A’ | A’+B | (A’+B).B |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 |



b) (A∙B)+A'

## Truth Table

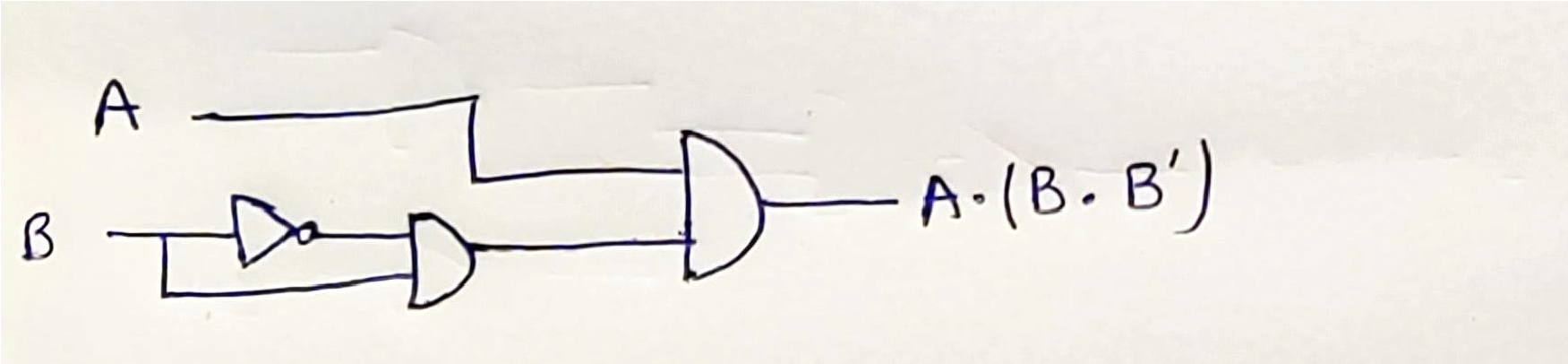
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | A’ | A.B | (A.B) + A’ |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 |



c) A∙ (B∙B')

## Truth Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | B’ | B.B’ | A.(B.B’) |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 |



d) (A'⊕B)∙A

## Truth Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | A’ | A’ ⨁ B | (A’ ⨁ B).A |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 |

