Logic Gates

Logic gate is a kind of electronic circuit. The mathematical operations of Boolean Algebra are represented by these logic gates. ***The electric circuits which are used for the practical usage of Boolean Algebra are called Logic Gates***. Or, ***the electric circuits which controls the flow of logical signals are also known as Logic Gates***. We can find single output through Logic Gates by numerous inputs.

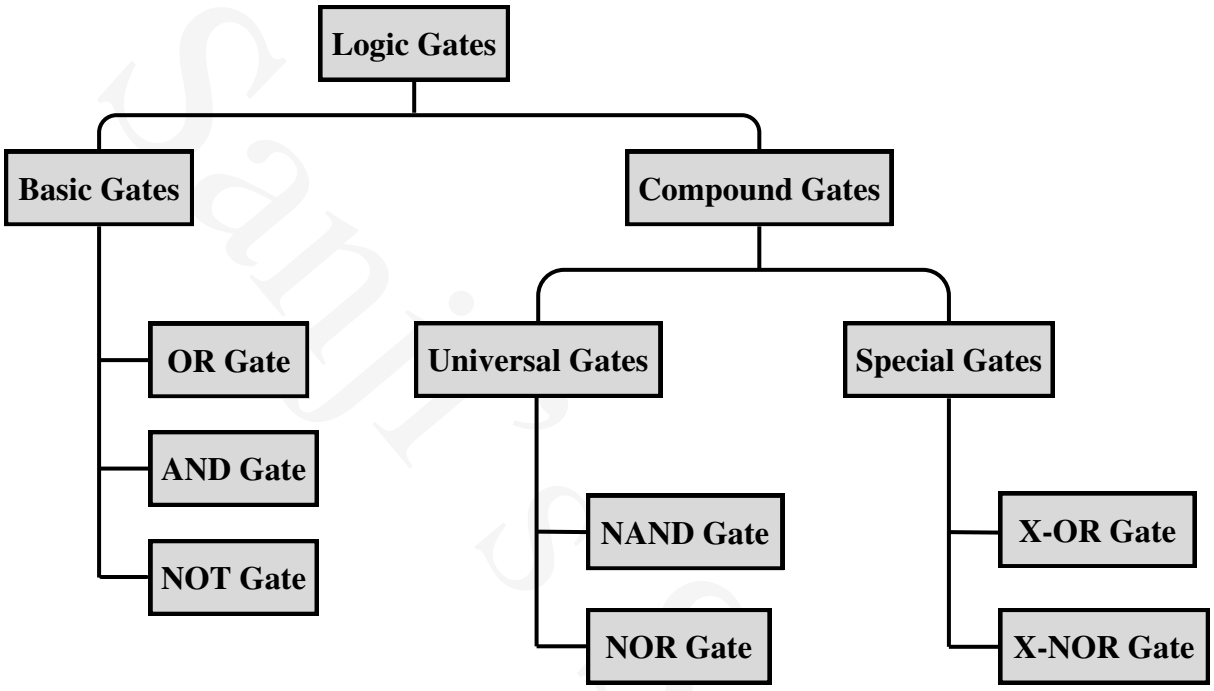


Fig. : Classification of Logic Gates

- **Types of Logic Gates:**

The mathematical operations of Boolean Algebra are usually expressed by three mathematical operations i.e. addition, multiplication and complementary and other operations are done by combining those basic operations. Generally, Logic Gates can divided into two types: 1) Basic Gates and 2) Compound Gates.

1. Basic Logic Gates: ***The gates which can perform single mathematical operations are called Basic Gates***. Those gates are:

i) OR Gate : for compound addition
ii) AND Gate : for compound multiplication
iii) NOT Gate : for compound complementary

2. Compound Logic Gates: ***The gates which are formed by combining two or more basic gates are called Compound Gates***. Those gates are:

i) NAND Gate : formed by the combination of AND gate and NOT gate
ii) NOR Gate : formed by the combination of OR gate and NOT gate
iii) X-OR Gate : formed by using OR, AND or NOT gate
iv) X-NOR Gate : formed by the combination of X-OR gate and NOT gate

A. Basic Gates:

i. OR Gate: The logic gate which is used to operate the addition operation of Boolean Algebra is called OR Gate. This gate can have two or more than two input and output is only one. Two or more switches in this gate remains in "parallel" way (given in figure 2i). The algebraic function of OR gate is $X = A + B$ where A and B are its input. Here + (plus) sign indicates OR operation.

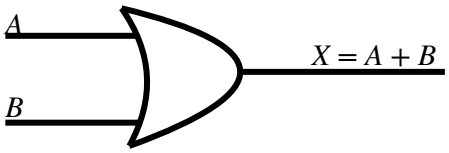


Fig. 1i: Symbol of OR Gate

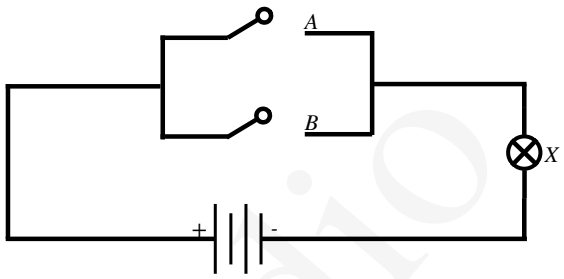


Fig. 2i: Switching circuit of OR Gate

Input		Output
A	B	$X = A+B$
0	0	0
0	1	1
1	0	1
1	1	1

Fig. 3i: Truth Table of OR Gate

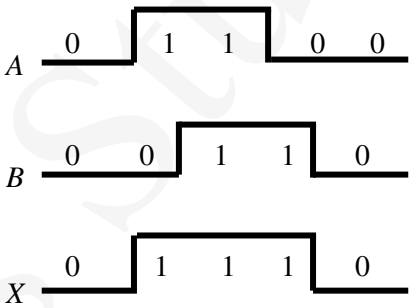


Fig. 4i: Digital Symbol of OR Gate

ii. AND Gate: The logic gate which is used to operate the multiplication operation of Boolean Algebra is called AND Gate. This gate can have two or more than two input and output is only one. Two or more switches in this gate remains in "series" way (given in figure 2ii). The algebraic function of AND gate is $X = A.B$ where A and B are its input. Here . (dot) sign indicates AND operation.

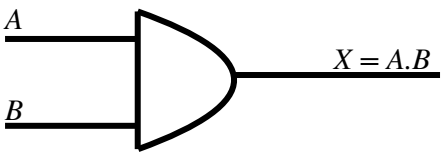


Fig. 1ii: Symbol of AND Gate

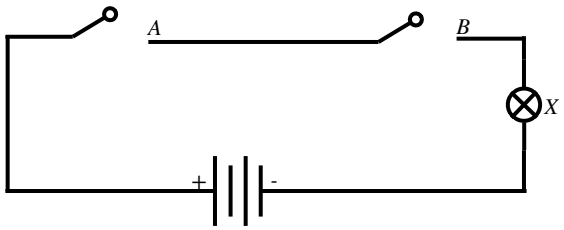


Fig. 2ii: Switching circuit of AND Gate

Input		Output
A	B	$X = AB$
0	0	0
0	1	0
1	0	0
1	1	1

Fig. 3ii: Truth Table of AND Gate

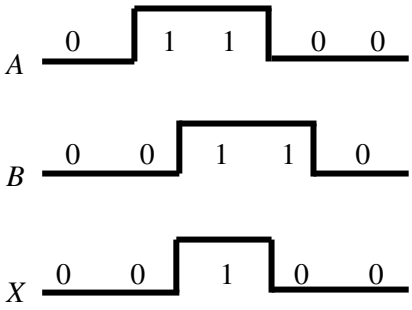


Fig. 4ii: Digital Symbol of AND Gate

iii. NOT Gate: The logic gate which is used to operate the inverse operation of Boolean Algebra is called NOT Gate. This gate have only one input and output. Since the output is inverse to the input, thus this gate is also known as *Inverter*. If the input is A then output will be $X = A$ or A' (inverse of $\overline{A}$).

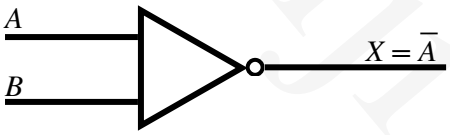


Fig. 1iii: Symbol of NOT Gate

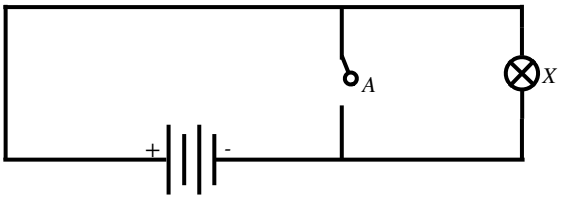


Fig. 2iii: Switching circuit of NOT Gate

Input	Output
A	$X = \overline{A}$
0	1
1	0

Fig. 3iii: Truth Table of NOT Gate

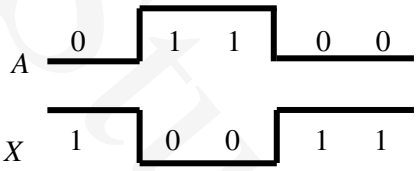


Fig. 4iii: Digital Symbol of NOT Gate

Know More:

Buffer Gate: A special type of gate where we will get the same value as an output what we will give to it as input. Buffer gate is made by using two NOT gates successively. This gate has only one input and output. Generally, this gate is used to amplify a weak signal i.e. to increase the wave in circuit.

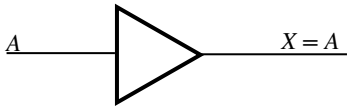


Fig. 1' : Symbol of Buffer Gate

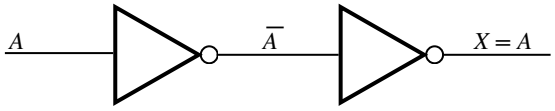


Fig. 3' : Buffer gate by 2 successive NOT gates.

Input	Output
A	$X = A$
0	0
1	1

Fig. 2': Truth Table of Buffer Gate

B. Universal Gates:

i. NAND Gate: The gate which implies the output of AND gate through NOT gate is called NAND gate i.e. AND gate + NOT gate = NAND gate. If A and B are two inputs, then NAND gate is $X = \overline{A.B}$ i.e. opposite to AND gate.

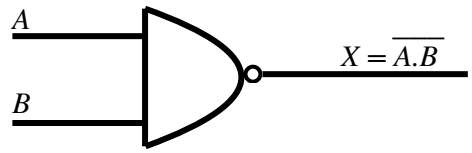


Fig. 1iv: Symbol of NAND Gate

Input			Output
A	*B*	*AB*	*X*
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Fig. 2iv: Truth Table of NAND Gate

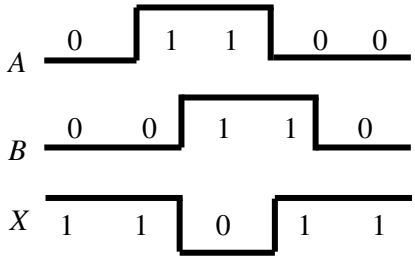


Fig. 3iv: Digital Symbol of NAND Gate

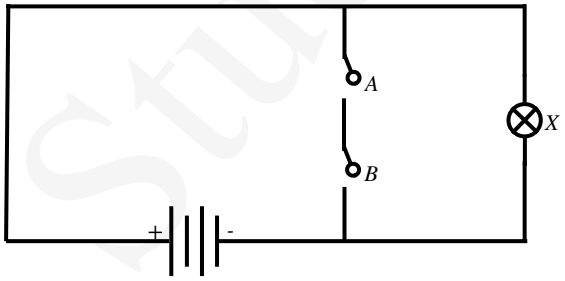


Fig. 4iv: Switching circuit of NAND Gate

WARNING!

Don't write this: ~~NOT gate + AND gate = NAND gate~~ *[Follow the sequence of the basic gates*

Write like this: AND gate + NOT gate = NAND gate *forming the required compound gates.]*

ii. NOR Gate: The gate which implies the output of OR gate through NOT gate is called NOR gate i.e. OR gate + NOT gate = NOR gate. If A and B are two inputs, then OR gate is $X = \overline{A+B}$ i.e. opposite to OR gate.

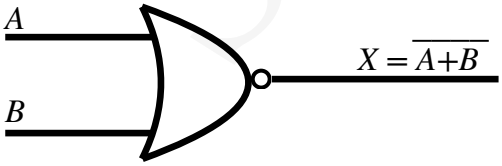


Fig. 1v: Symbol of NOR Gate

Input			Output
A	*B*	*A+B*	*X*
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Fig. 2v: Truth Table of NOR Gate

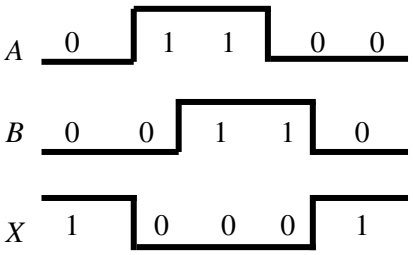


Fig. 3v: Digital Symbol of NOR Gate

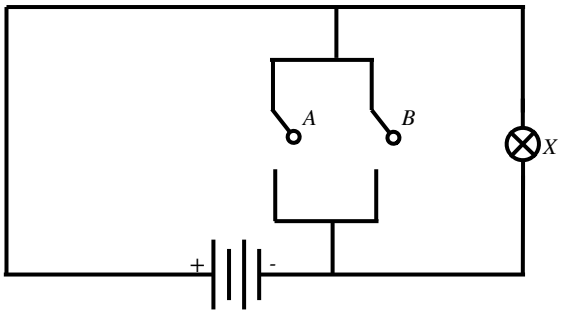


Fig. 4v: Switching circuit of NOR Gate

> *Note:*
> Always write the name of the respective gates in block (uppercase form) letters. For example - it'll be wrong to write the AND gate as *And* or *and*.

C. Special Gates:

i. X-OR Gate: Exclusive OR gate can be said as X-OR gate in short. If input is A and output is B, the Boolean rule will be $X = A \oplus B = \bar{A}B + A\bar{B}$ by which the symbol '$\oplus$' indicates the X-OR operation. Since this gate is used much, it is given the same priority like the basic gates. This gate can be made by OR, AND or NOT gates; again it can be found in Integrated Circuit (IC) form.

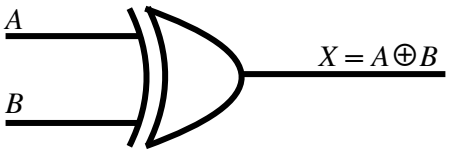


Fig. 1vi: Symbol of X-OR Gate

Input		Output
A	*B*	*X*
0	0	0
0	1	1
1	0	1
1	1	0

Fig. 2vi: Truth Table of X-OR Gate

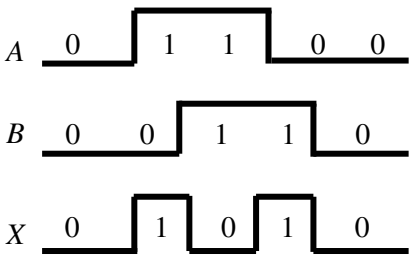


Fig. 3vi: Digital Symbol of X-OR Gate

Input				Output
A	*B*	$\bar{A}B$	$A\bar{B}$	*X*
0	0	1	1	0
0	1	1	0	1
1	0	0	1	1
1	1	1	1	0

Extended Truth Table of X-OR gate for further understanding

ii. X-NOR Gate: The gate formed by combining X-OR gate and NOT gate is X-NOR gate. It implies the output of X-OR gate through NOT gate. If input is A and output is B, the value of X for X-NOR gate according to Boolean rule, $X = \overline{A \oplus B} = \overline{\bar{A}B + A\bar{B}} = \bar{A}\bar{B} + AB$.

**X-NOR gate + NOT gate = X-NOR gate*

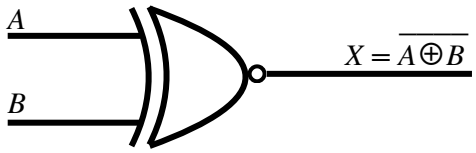


Fig. 1vii: Symbol of X-NOR Gate

Input		Output
A	*B*	*X*
0	0	1
0	1	0
1	0	0
1	1	1

Fig. 2vii: Truth Table of X-NOR Gate

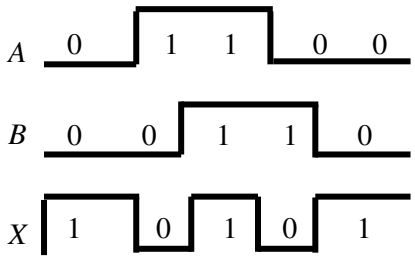


Fig. 3vii: Digital Symbol of X-NOR Gate

Input			Output
A	*B*	$A \oplus B$	*X*
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

Extended Truth Table of X-NOR gate for further understanding

- **<u>Universality of NAND and NOR Gates:</u>**

The gates by which operations of all basic gates can be implemented or work as like basic gates are called universal gates. Generally, we can create or implement any gate or logic circuit by using basic gates (OR, AND & NOT) i.e. if any gate works like those basic gates, it is possible to create any gate or logic circuit with them. Then we can say that, ***the gates by which basic gates (OR, AND & NOT) can be implemented are called universal gates***. NAND and NOR gates are universal as we can implement basic gates (OR, AND & NOT) and thus they are known as *universality of NAND & NOT gates*.

Universality of NAND gate/ Implementation of basic gates by using NAND gate

i) From NAND gate to NOT gate:

The connection shown below will make NAND gate work like NOT gate as in this case, two inputs will become the same (A).

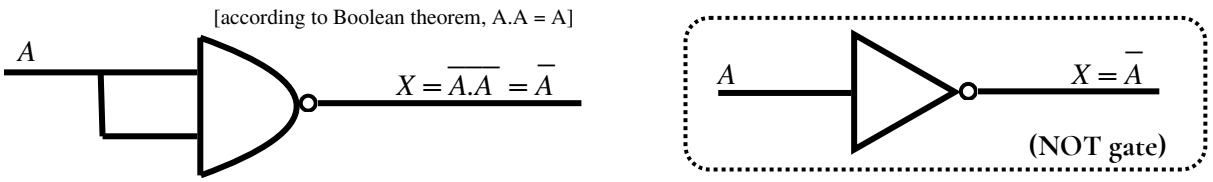


Fig. : Implementation of NOT gate by NAND gate

ii) From NAND gate to OR gate:

The connection shown below can implement the OR gate by NAND gate.

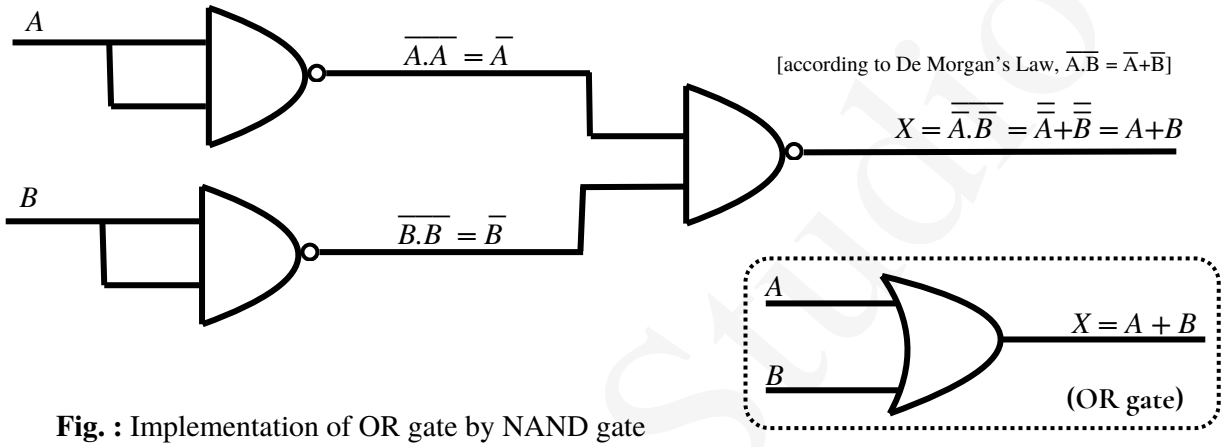


Fig. : Implementation of OR gate by NAND gate

iii) From NAND gate to AND gate:

The connection shown below can implement the AND gate by NAND gate.

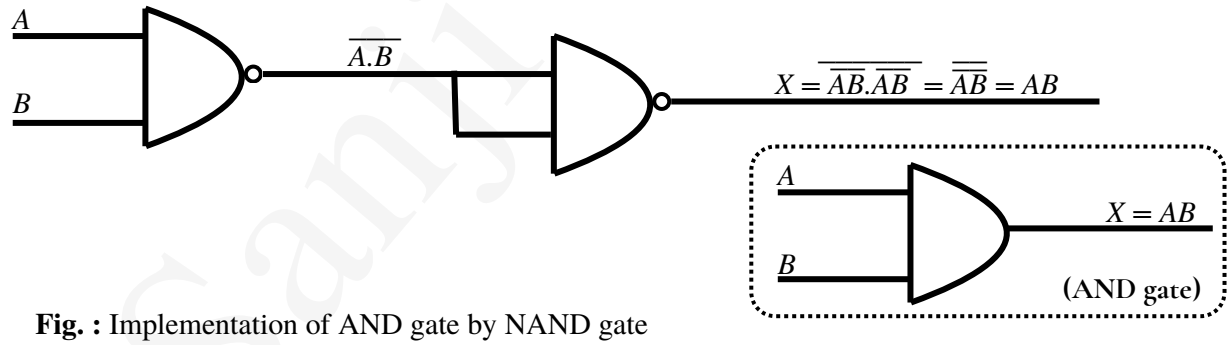


Fig. : Implementation of AND gate by NAND gate

Since, basic gates (OR, AND & NOT) can be implemented by NAND gate, thus NAND gate is an universal gate.

Universality of NOR gate/ Implementation of basic gates by using NOR gate

i) From NOR gate to NOT gate:

The connection shown below will make NOR gate work like NOT gate.

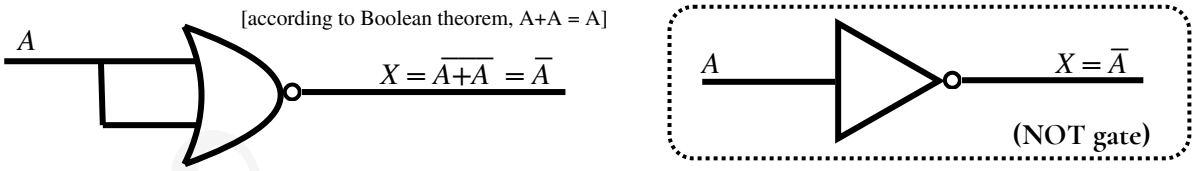


Fig. : Implementation of NOT gate by NOR gate

ii) From NOR gate to OR gate:

The connection shown below can implement the OR gate by NOR gate.

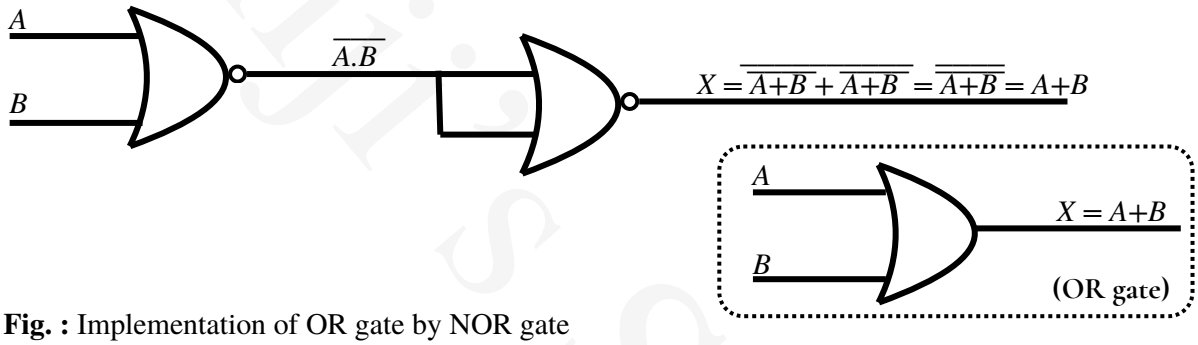


Fig. : Implementation of OR gate by NOR gate

iii) From NOR gate to AND gate:

The connection shown below can implement the AND gate by NOR gate.

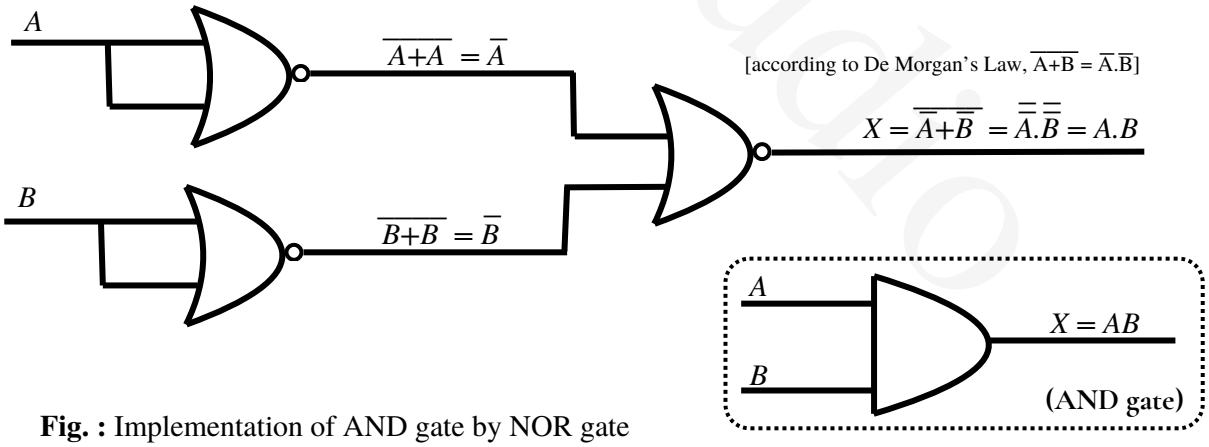


Fig. : Implementation of AND gate by NOR gate

Since, basic gates (OR, AND & NOT) can be implemented by NOR gate, thus NOR gate is an universal gate.

- **Implementation of Special Gates through Basic and Universal Gates:**

Special gates can be implemented through either basic gates (OR, AND & NOT) or universal gates (NAND & NOR). Those implementation are shown below:

Implementing Special Gates through Basic Gates

1(A). X-OR gate (2-input):-

Required equation, $X = A \oplus B = \bar{A}B + A\bar{B}$

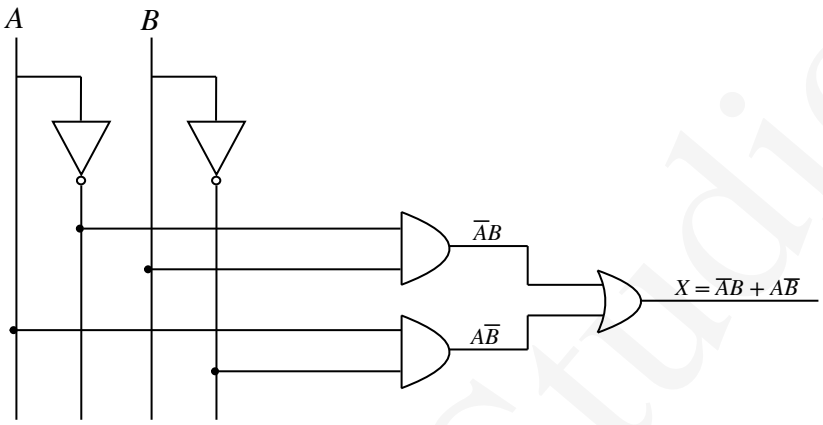


Fig. 1(A): Implementation of X-OR gate (2-input) by basic gates (NOT, AND & OR)

Tips & Tricks:

First, observe how much variable you're given and then draw that amount of straight lines for the amount of variables. Then insert necessary gates (basic & universal) to implement the assigned special gates. You must know and write down the equation for specified special gates before implementing.

(B). X-OR gate (3-input):-

Required equation, $X = A \oplus B \oplus C = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC$

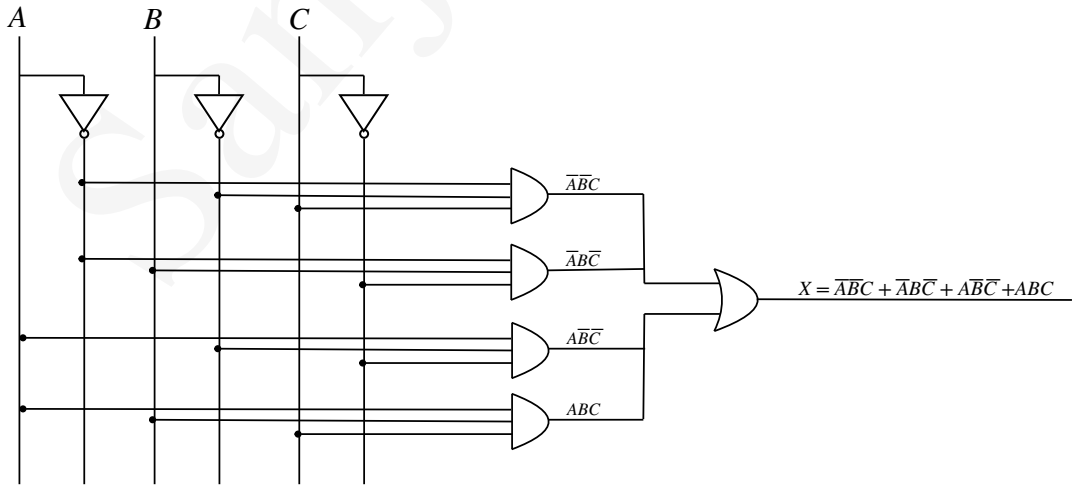


Fig. 1(B): Implementation of X-OR gate (3-input) by basic gates (NOT, AND & OR)

2(A). X-NOR gate (2-input):-

Required equation, $X = \overline{A \oplus B} = \bar{A}\bar{B} + AB$

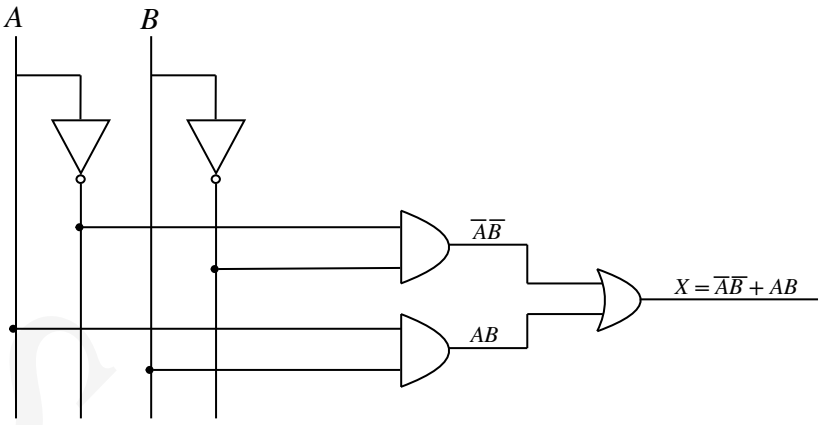


Fig. 2(A): Implementation of X-NOR gate (2-input) by basic gates (NOT, AND & OR)

(B). X-NOR gate (3-input):-

Required equation, $X = \overline{A \oplus B \oplus C} = \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$

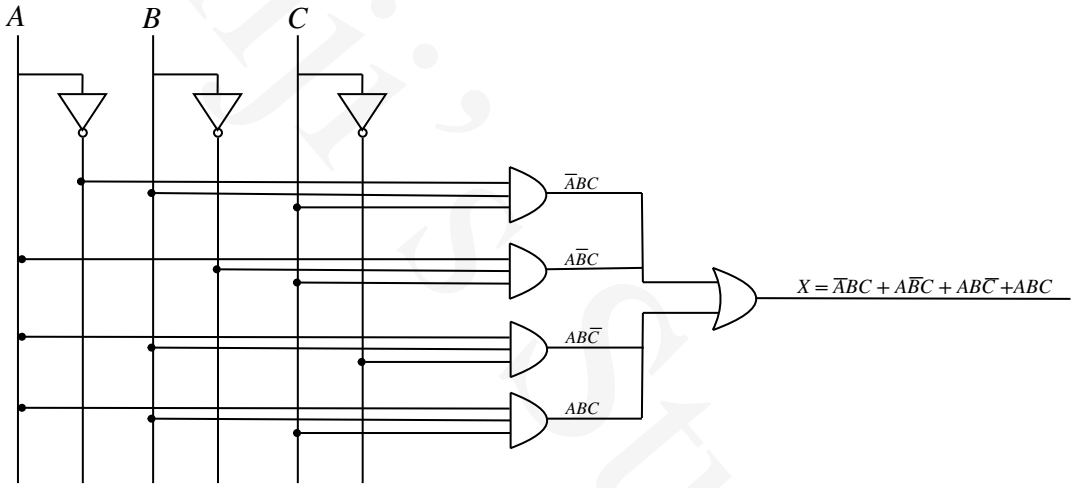


Fig. 2(B): Implementation of X-NOR gate (3-input) by basic gates (NOT, AND & OR)

Implementing Special Gates through Universal Gates

1(A). X-OR gate (2-input) only by *NAND gate*:-

Required equation, $X = A \oplus B = \bar{A}B + A\bar{B}$

$= \overline{\overline{\bar{A}B + A\bar{B}}}$

$= \overline{(\overline{\bar{A}B})} \cdot \overline{(\overline{A\bar{B}})}$ *[De Morgan's Theorem]*

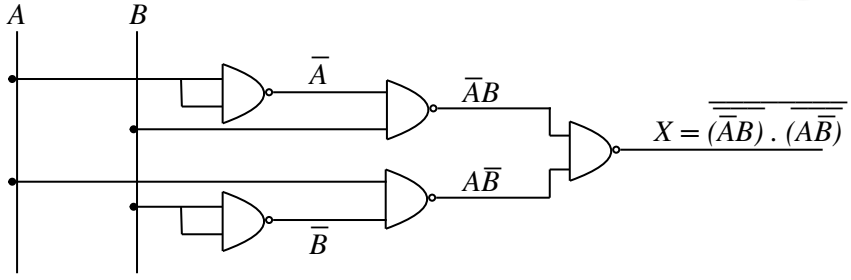


Fig. 1(A): Implementation of X-OR gate (2-input) by only NAND gate

(B). X-NOR gate (2-input) only by *NAND gate*:-

Required equation, $X = \overline{A \oplus B} = \bar{A}\bar{B} + AB$

$= \overline{\overline{\bar{A}\bar{B}} + AB}$

$= \overline{(\bar{A}\bar{B})} \cdot \overline{(AB)}$ *[De Morgan's Theorem]*

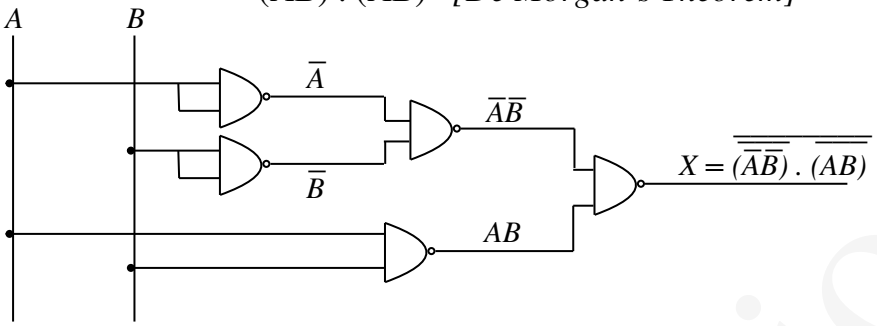


Fig. 1(B): Implementation of X-NOR gate (2-input) by only NAND gate

2(A). X-OR gate (2-input) only by *NOR gate*:-

Required equation, $X = A \oplus B = \bar{A}B + A\bar{B}$

$= \overline{\overline{\bar{A}B}} + \overline{\overline{A\bar{B}}}$

$= \overline{(\bar{A}+B)} + \overline{(A+\bar{B})}$ *[De Morgan's Theorem]*

$= \overline{(A+\bar{B})} + \overline{(\bar{A}+B)}$

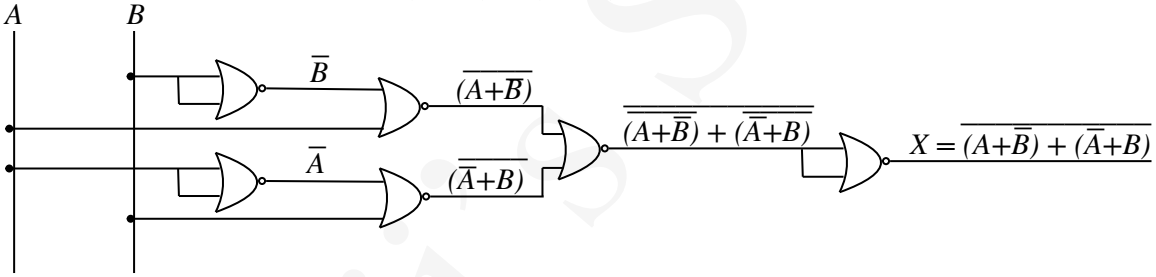


Fig. 2(A): Implementation of X-OR gate (2-input) by only NOR gate

(B). X-NOR gate (2-input) only by *NOR gate*:-

Required equation, $X = \overline{A \oplus B} = AB + \bar{A}\bar{B}$

$= \overline{\overline{AB}} + \overline{\overline{\bar{A}\bar{B}}}$

$= \overline{(\bar{A}+\bar{B})} + \overline{(\bar{A}+\bar{B})}$ *[De Morgan's Theorem]*

$= \overline{(\bar{A}+\bar{B})} + \overline{(A+B)}$

$= \overline{\overline{(\bar{A}+\bar{B})} + (A+B)}$

$= \overline{\overline{(\bar{A}+\bar{B})} + \overline{(A+B)}}$

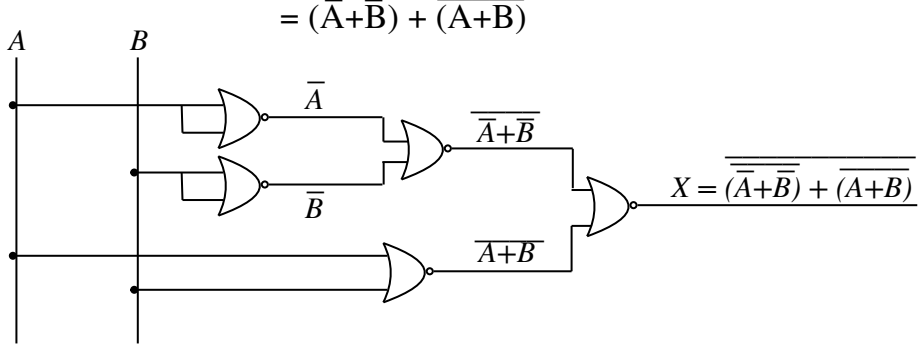
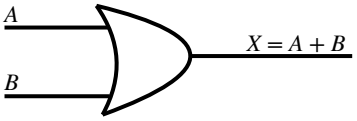
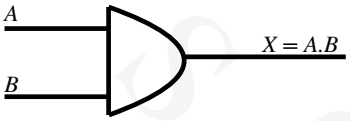
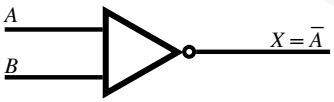
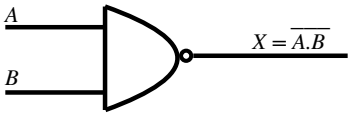
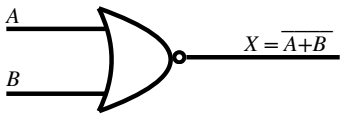
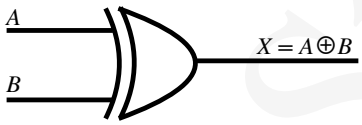
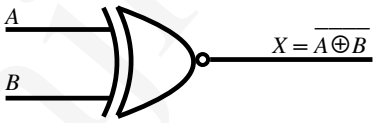


Fig. 2(B): Implementation of X-NOR gate (2-input) by only NOR gate

At a glance of Logic Gates

<table>
<tr><th>Name of the gates</th><th>Logic Function</th><th>Logic Circuit</th><th>Truth Table</th></tr>
<tr><td>OR</td><td>X = A+B</td><td> 
</td><td>
<table>
<tr><th colspan="2">Input</th><th>Output</th></tr>
<tr><td>A</td><td>B</td><td>X = A+B</td></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>AND</td><td>X = A.B</td><td> 
</td><td>
<table>
<tr><th colspan="2">Input</th><th>Output</th></tr>
<tr><td>A</td><td>B</td><td>X = AB</td></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>NOT</td><td>X = $\overline{A}$</td><td> 
</td><td>
<table>
<tr><th>Input</th><th>Output</th></tr>
<tr><td>A</td><td>X = $\overline{A}$</td></tr>
<tr><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td></tr>
</table>
</td></tr>
<tr><td>NAND</td><td>X = $\overline{A.B}$</td><td> 
</td><td>
<table>
<tr><th colspan="3">Input</th><th>Output</th></tr>
<tr><td>A</td><td>B</td><td>AB</td><td>X</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Name of the gates</th><th>Logic Function</th><th>Logic Circuit</th><th>Truth Table</th></tr>
<tr><td>NOR</td><td>$X = \overline{A+B}$</td><td> 
</td><td>
<table>
<tr><th colspan="3">Input</th><th>Output</th></tr>
<tr><td>A</td><td>B</td><td>A+B</td><td>X</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td></tr>
</table>
</td></tr>
<tr><td>X-OR</td><td>$X = A \oplus B$</td><td> 
</td><td>
<table>
<tr><th colspan="2">Input</th><th>Output</th></tr>
<tr><td>A</td><td>B</td><td>X</td></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
</table>
</td></tr>
<tr><td>X-NOR</td><td>$X = \overline{A \oplus B}$</td><td> 
</td><td>
<table>
<tr><th colspan="2">Input</th><th>Output</th></tr>
<tr><td>A</td><td>B</td><td>X</td></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table>
</td></tr>
</table>

The End of Part 2: Logic Gate