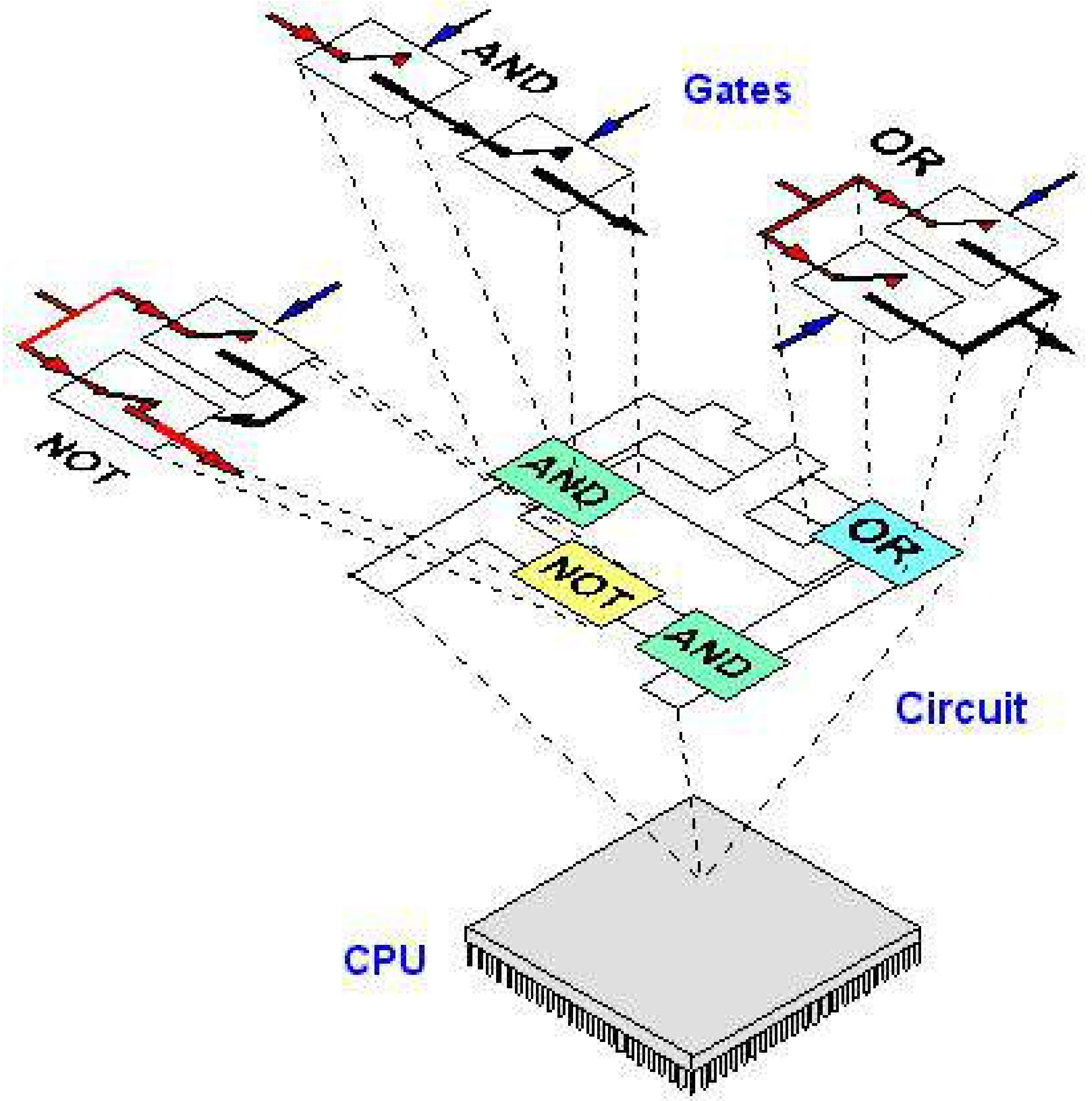
Chapter 3
Logic Gates

Objectives:

- **Learning differnt types of gates**
- **Designing the logical circuits**



LOGIC GATES

3.1 Introduction

After Shannon applied Boolean algebra in telephone switching circuits, engineers realized that Boolean algebra could be applied to computer electronics as well.

In the computers, these Boolean operations are performed by logic gates.

Elementary logic gates

Gates are digital (two-state) circuits because the input and output signals are either low voltage (denotes 0) or high voltage (denotes 1). Gates are often called logic circuits, because they can be analyzed with Boolean algebra.

> A gate is simply an electronic circuit which operates on one or more signals and always produces an output signal.

Gates are classified into two types.

(a) Basic gates

(b) Derived gates

(a) Basic gates

There are three basic logic gates:

1. NOT gate (inverter)
2. OR gate
3. AND gate

(b) Derived gates

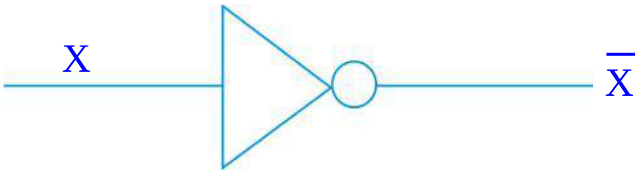
There are four derived gates:

1. NOR gate
2. NAND gate
3. XOR gate (Exclusive OR gate)
4. XNOR gate (Exclusive NOR gate)

3.1.1 Inverter (NOT gate)

An inverter is also called a NOT gate, because the output is not the same as the input. The output is sometimes called the complement (opposite) of the input.

An Inverter is a gate with only one input signal and one output signal; the output state is always the opposite of the input state.



X	$\overline{\mathbf{X}}$
Low	High
High	Low

X	$\overline{\mathbf{X}}$
0	1
1	0

A low input i.e., 0 produces high output i.e., 1 and vice versa. NOT operation is symbolized as ‾ or i.e., NOT X is written as X^1 or $\overline{X}$.

3.1.2 OR Gate

The OR gate has two or more input signals but only one output signal. If one or more input signals are 1 (high), the output signal is 1 (high).

An OR gate can have as many inputs as desired. No matter how many inputs are there, the action of OR gate is the same.

The OR gate has two or more input signals but only one output signal. If any of the input signals is 1 (high), the output signal is 1 (high).

Following tables show OR action:

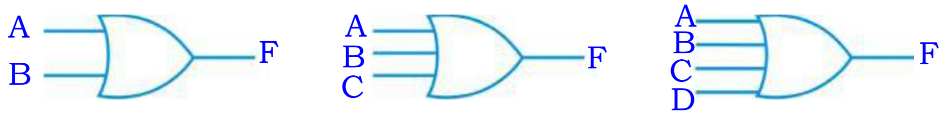
X	**Y**	**F=X+Y**
0	0	0
0	1	1
1	0	1
1	1	1

Table 3.3 Truth table for 2-input OR gatea

X	**Y**	**Z**	**F=X+Y+Z**
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Table 3.4 Truth Table for three input OR gate

The symbol for OR gate is given below:



OR operation is symbolized as + i.e., X or Y is written as X+Y.

3.1.3 AND gate

This gate also can have two or more inputs and always gives single output. If any input is 0, the output is 0. To obtain output as 1, all inputs must be 1. Thus, the AND represents the logical multiplication.

The AND Gate can have two or more input signals and produce one output signal. When all the inputs are high then the output is high. Otherwise, the output is low.

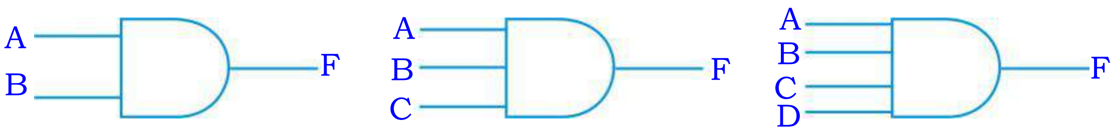
X	**Y**	**F=X.Y**
0	0	0
0	1	0
1	0	0
1	1	1

Table 3.5: 2- input AND gate

X	**Y**	**Z**	**F=X.Y.Z**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Table 3.6: 3-input AND gate

The symbol for AND is



AND operation is symbolized as X AND Y is written as X.Y

3.2 Derived gates

3.2.1 NOR Gate

NOR gate is nothing but NOT OR gate or inverted OR gate. This means, an OR gate is always followed by a NOT gate to give NOR gate.

This gate also accepts two or more than two inputs and always produces single output. If either of the two inputs is 1 (high), the output will be 0 (low). Also, if all the inputs are low, then the output is high.

The NOR gate has two or more input signals but only one output signal. If all the inputs are 0 (low), then the output signal is 1 (high).

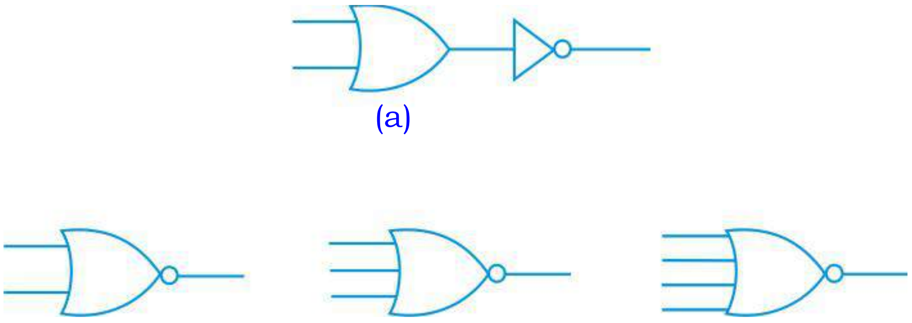
X	**Y**	**$F = \overline{X+Y}$**
0	0	1
0	1	0
1	0	0
1	1	0

Table 3.7: 2-input NOR gate

X	**Y**	**Z**	**$F=\overline{X+Y+Z}$**
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Table 3.8: 3-input NOR gate

NOR operation is symbolized as i.e., X NOR Y is written as $\overline{X + Y}$.



(a)

(b) (c) (d)

3.2.2 NAND Gate

NAND gate is NOT AND gate or inverted AND gate. This means, an AND gate is always followed by a NOT gate to give NAND gate.

NAND gate can also have two or more inputs. This gate produces 0 (low) for all 1 (high) inputs and produces 1 (high) for other input combinations.

The NAND Gate has two or more input signals but only one output signal. If all of the inputs are 1 (high), then the output produced is 0 (low).

NAND action is illustrated in following Truth Tables (2.9 and 2.10)

X	**Y**	**F = $\overline{XY}$**
0	0	1
0	1	1
1	0	1
1	1	0

Table 3.9: Truth table of 2-input NAND gate

X	**Y**	**Z**	**F = $\overline{XYZ}$**
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Table 3.10 Truth table of 3-input NAND gate

The logical meaning of NAND gate can be shown as follows:
NAND operation is symbolized as i.e., X NAND Y is written as $\overline{X.Y}$.

3.2.3 XOR Gate or Exclusive-OR Gate

The XOR Gate can also have two or more inputs, but produces one output signal. Exclusive–OR gate is different from OR gate. OR gate produces output 1 for any input combination having one or more 1's, but XOR gate produces output 1 for only those input combinations that have odd number of 1's.

- ➢ Accepts two or more inputs and produces single output.
- ➢ The output is 0 if there are even number of 1’s in the inputs.
- ➢ The output is 1 if there are odd number of 1’s in the inputs.

In Boolean algebra, ⊕ sign stands for XOR operation. Thus, A XOR B can be written as A⊕B.

Following Truth Tables (2.11 and 2.12) illustrates XOR operation.

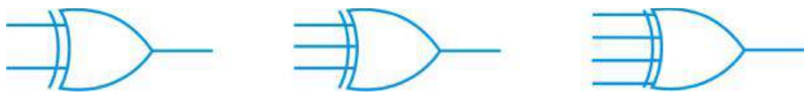
No. of 1’s Even/odd	X	Y	Z = A⊕B
Even	0	0	0
Odd	0	1	1
Odd	1	0	1
Even	1	1	0

Table 3.11 Truth table of 2-input XOR gate

No. of 1’s	**X**	**Y**	**Z**	**F**
Even	0	0	0	0
Odd	0	0	1	1
Odd	0	1	0	1
Even	0	1	1	0
Odd	1	0	0	1
Even	1	0	1	0
Even	1	1	0	0
Odd	1	1	1	1

Table 3.12 3 input XOR gate

The symbols of XOR gates are given below:



XOR addition can be summarized as following:

0⊕ 0 = 0; 0 ⊕ 1 = 1; 1⊕ 0 = 1; 1⊕ 1 = 0

The operation representing XOR may be written as $F = x \oplus y = \overline{x}\, y + x\overline{y}$

3.2.4 XNOR Gate or Exclusive NOR Gate

An XOR gate is followed by a NOT gate (inventor) becomes XNOR gate. Thus, The XNOR Gate is logically equivalent to an inverted XOR gate. Thus XNOR produces 1 (high) as output when the input combination has even number of 1's or when all the inputs are 0's.

Following truth tables 2.13 and 2.14 illustrate XNOR action.

No. of 1's Even/odd	**X**	**Y**	**F**
Even	0	0	1
Odd	0	1	0
Odd	1	0	0
Even	1	1	1

Table 3.13 2 input XNOR gate

No. of 1's	**X**	**Y**	**Z**	**F**
Even	0	0	0	1
Odd	0	0	1	0
Odd	0	1	0	0
Even	0	1	1	1
Odd	1	0	0	0
Even	1	0	1	1
Even	1	1	0	1
Odd	1	1	1	0

Table 3.14 3 input XNOR gate

Following are the XNOR gate symbols:

The bubble (small circle), on the output of NAND, NOR, XNOR gates represents complementation.

The expression representing XNOR may be written as

$F = (\overline{X}+Y) \,.\, (X+\overline{Y})$

The XNOR operation is symbolized as $\odot$ i.e. X NOR Y is written as $X\odot Y$.

Now that you are familiar with logic gates, you can use them in designing logic circuits.

3.2.5 Circuit diagrams

Boolean algebra is useless unless it can be translated into hardware, in the form of gates. This translation of Boolean algebra in the gates' form is known as logic circuits. A logic circuit can be represented diagrammatically using the traditional symbols of gates. Let us see how this is done, in the following examples.

Example 2.1: Design a circuit to realize the following:

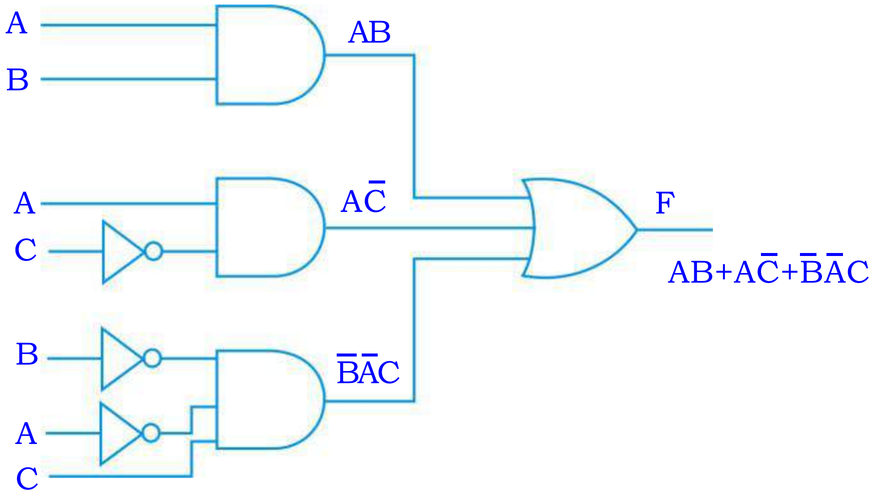
$F(A, B, C) = AB + A\overline{C} + \overline{B}\overline{A}C$

Solution: The given Boolean expression can also be written as follows

$F(A, B, C) = A . B + A . C + B . A . C$

or F(A, B, C) = (A AND B) OR (A AND (NOT C)) OR ((NOT B) AND (NOT A) AND C)

Now these logical operators can easily be implemented in form of logic gates. Thus circuit diagram for above expression will be as follows:


A
B
AB
A
C
$A\overline{C}$
F
$AB+A\overline{C}+\overline{B}\overline{A}C$
B
A
C
$\overline{B}\overline{A}C$

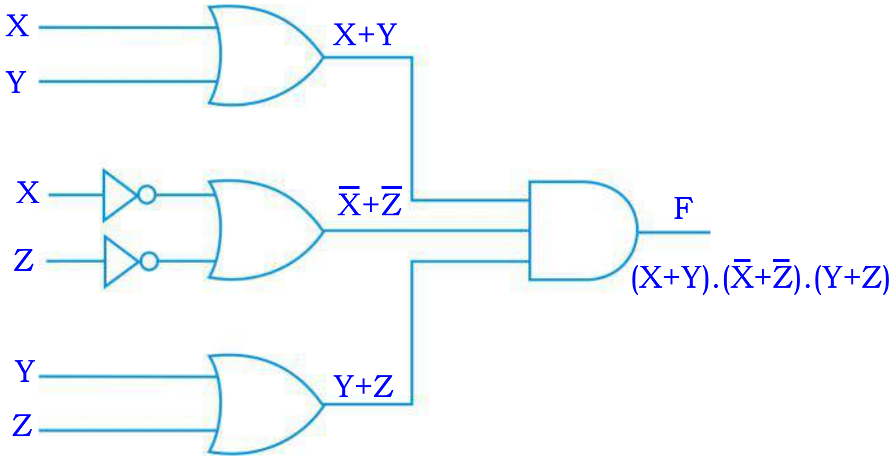
Example 2.2 Draw the diagram of digital circuit for the function:

$F(X, Y, Z) = (X + Y) . (\overline{X} + \overline{Z}) . (Y + Z)$

Solution: Above expression can also be written as

F(X, Y, Z) = (X OR Y) AND ((NOT X) OR (NOT Z)) AND (Y OR Z)

Thus circuit diagram will be


X
Y
X+Y
X
Z
$\overline{X}+\overline{Z}$
Y
Z
Y+Z
F
$(X+Y).(\overline{X}+\overline{Z}).(Y+Z)$

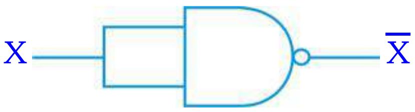
3.2.6 NAND, NOR as Universal Gates

We can design circuits using AND, OR and NOT gates as we have done so far. But NAND and NOR gates are more popular as these are less expensive and easier to design. Also, The basic gates AND, OR and NOT can easily be implemented using NAND/NOR gates. Thus NAND, NOR gates are also referred to as Universal Gates.

> Universal gate is a gate using which all the basic gates can be designed. NAND and NOR gates are called as the universal gates.

NAND–to-NOT logic

Not Operation



NOT X = X NAND X

$= \overline{X.X}$ {De Morgan's Second Theorem}

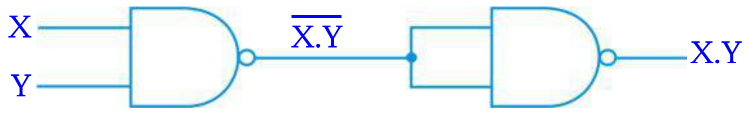
$= \overline{X} + \overline{X}$ {De Morgan's Second Theorem}

$= \overline{X}$

NAND–to-AND logic

AND and OR operations from NAND gates are shown below:

AND operation



AND operation using NAND is

X . Y = (X NAND Y) NAND (X NAND Y)

Proof: X NAND Y = $\overline{X.Y}$

= $\overline{X} + \overline{Y}$ (De Morgan's Second Theorem)

(X NAND Y) NAND (X NAND Y)

= $(\overline{X} + \overline{Y})$ NAND $(\overline{X} + \overline{Y})$

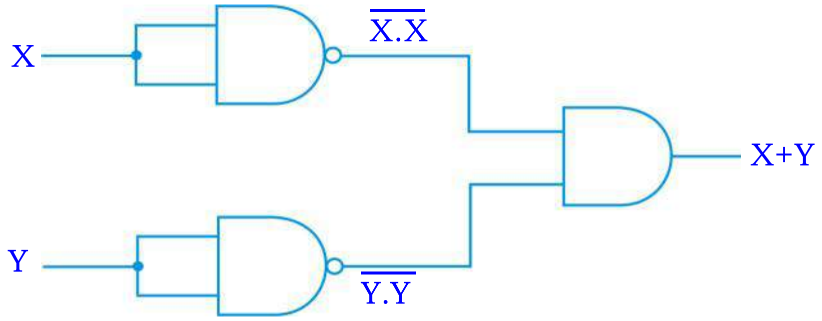
= $\overline{(\overline{X} + \overline{Y}) . (\overline{X} + \overline{Y})}$ (De Morgan's Second Theorem)

= $\overline{\overline{X}}.\overline{\overline{Y}} + \overline{\overline{X}}.\overline{\overline{Y}}$ (De Morgan's First Theorem)

= X . Y + X . Y ($\overline{\overline{X}}$ = X)

= XY (X + X = X)

OR Operation



X + Y = (X NAND X) NAND (Y NAND Y)

Proof : X NAND X = $\overline{X.X}$ {De Morgan's Second Theorem}

= $\overline{X} + \overline{X}$ {De Morgan's Second Theorem}

= $\overline{X}$ { X + X = X }

Similarly, Y NAND Y = $\overline{Y}$

Therefore, (X NAND X) NAND (Y NAND Y) = $\overline{X}$ NAND $\overline{Y}$

= $\overline{(\overline{X} . \overline{Y})}$

= $\overline{\overline{X}} + \overline{\overline{Y}}$ {$\overline{\overline{X}}$ = X, $\overline{\overline{Y}}$ = Y}

= X + Y

NAND-to-NAND logic is best suited for Boolean expression in Sum-of-Products form.

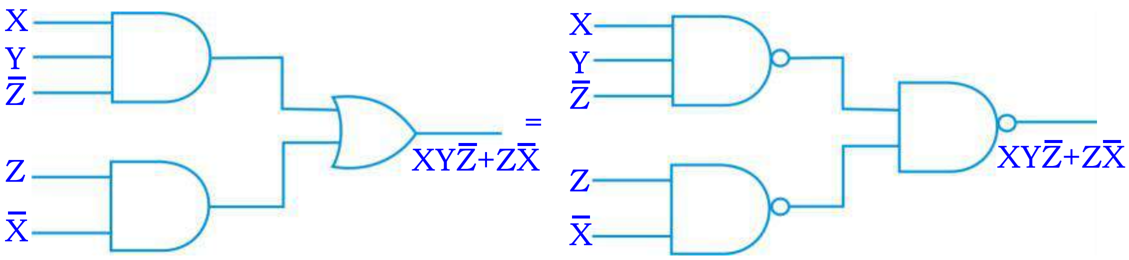
Design rule for NAND-TO-NAND logic Network (only for two-level-circuits)

1 Derive simplified sum-of products expression.

2 Draw a circuit diagram using AND, OR and NOT gates.

3 Just replace AND, OR and NOT gates with NAND gates

For example, $XY\overline{Z}$ + $\overline{X}Z$ can be drawn as follows, assuming that inputs and their complements are available:



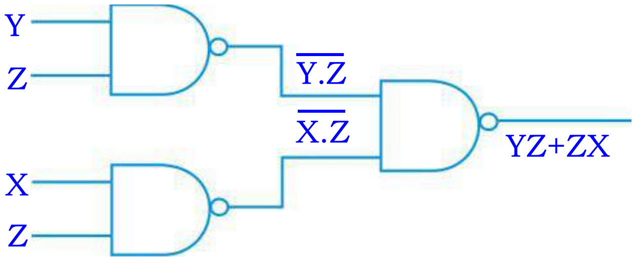
Example 2.3: Draw the diagram of a digital circuit for the function

F(X, Y, Z) = YZ + XZ using NAND gates only.

Solution : F(X, Y, Z) = YZ + XZ can be written as

= (Y NAND Z) NAND (X NAND Z)

Thus logic circuit diagram is



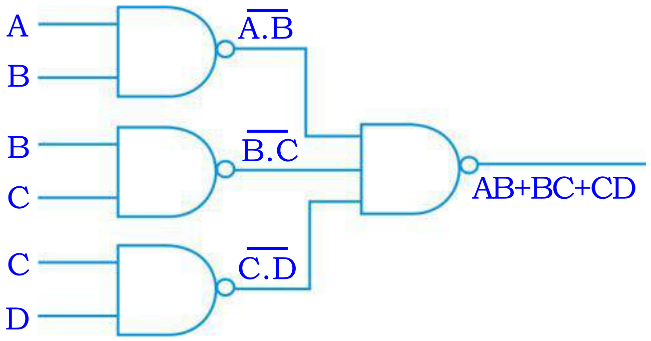
Example 2.4: Draw the diagram of digital circuit:

F (A, B, C) = AB + BC + CD using NAND-to-NAND logic.

Solution: F(A, B, C) = AB + BC + CD

= (A NAND B) NAND (B NAND C) NAND (C NAND D)

Thus logic circuit is



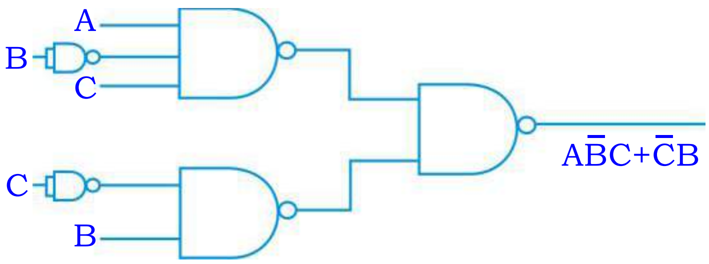
Example 2.5: Draw the circuit diagram for

$F = A\overline{B}C + \overline{C}B$ using NAND-to-NAND LOGIC ONLY.

Solution: $F = A\overline{B}C + \overline{C}B$

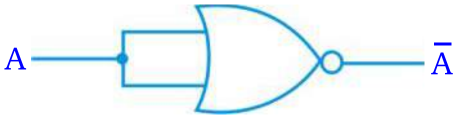
= ((A NAND (NOT B) NAND C) NAND ((NOT C) NAND B))

Circuit Diagram is



NOR-to-NOT logic

NOT, AND and OR operations can be implemented in NOR-to-NOR form as shown on below.



NOT X = X NOR X

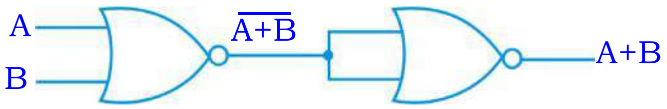
$= \overline{X+X}$

$= \overline{X}.\overline{X}$

$= \overline{X}$

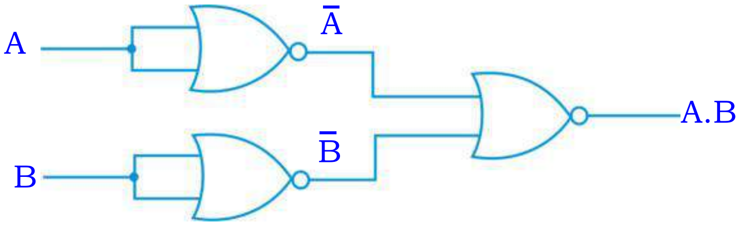
NOR to OR Operation

A + B = (A NOR B) NOR (A NOR B)



NOR to AND Operation

A . B = (A NOR A) NOR (B NOR B)



NOR-to-NOR logic is best suited for Boolean expression in Product-of-sum form.

Design rule for NOR-to-NOR logic network (only for two-level-circuits):

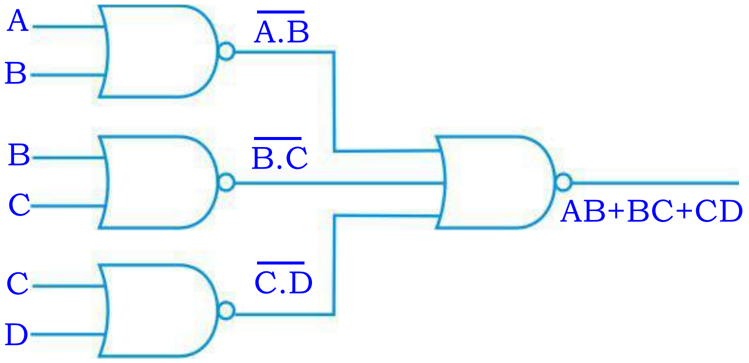
1 Derive a simplified product-of-sums form of the expression.

2 Draw a circuit diagram using NOT, OR and AND gates.

3 Finally substitute NOR gates tor NOT, OR and AND gates.

Example 2.6: Represent (X + Y) (Y + Z) (Z + X) in NOR-to-NOR form.

Solution: (X + Y) (Y + Z) (Z + X) = (X NOR Y) NOR (Y NOR Z) NOR (Z NOR X)



One mark questions:

1. What is a logic gate?
2. Mention the three basic logic gates?
3. Which basic gate is named as Inverter?
4. Which are the three logic operations?
5. Write the standard symbol for AND gate.
6. Write the truth table for AND gate.
7. Write the logic circuit for AND gate.
8. Write the standard symbol for OR gate.
9. Write the truth table for OR gate.
10. Write the logic circuit for OR gate.
11. Write the standard symbol for NOT gate.
12. Write the truth table for NOT gate.
13. Write the logic circuit for NOT gate.
14. What is a truth table?
15. What is meant by universal gates?
16. Mention different universal gates.
17. What is the output of the two input NAND gate for the inputs: A=0, B=1?
18. What are the values of the inputs to a three input NAND gate, if its output is 1?
19. What are the values of the inputs to a three input NAND gate, if its output is 0?
20. What is the output of the two input OR gate for the inputs: A=0, B=0?
21. What are the values of the inputs to a three input OR gate, if its output is 0?
22. What are the values of the inputs to a three input OR gate, if its output is 1?
23. For the truth table given below, what type of logic gate does the output X represent?

A	B	C	X
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

24. For the truth table given below, what type of logic gate does the output X represent?

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

25.State any one of the principle from duality of theorems.

Three Marks Questions

1. What is meant by proof by perfect induction? Give an example.

2. Write the truth table and standard symbol of NAND gate.

3. Explain the working of NAND gate .(write the output conditions)

4.Write the truth table and standard symbol of NOR gate.

5.Explain the working of NOR gate .(write the output conditions)

6.Draw the logic gate diagram to implement AND and OR gates using NAND gates only. (any two gates)

7.Draw the logic gate diagram to implement AND and OR gates using NOR gates only. (any two gates)

8.Draw the logic gate diagram to implement NOT using
(a) only NOR gates (b) only NAND gates.

9.State De Morgan's theorems.

10.What is principle of duality? Give an example.

11.Give the dual form of (any two) :
(a) 0.X + X.Y + 1.X
(b) X.(Y+Z) = X.Y + X.Z
(c) X + $\bar{X}$. Y = X + Y
(d) 1 + X = 1

12.Simplify the following logical expression using De Morgan's theorems.
(a) (A+B).C
(b) (A+BC).(D+EF)

13. Prove the following rules using the proof by perfect induction.

(a) $X\overline{Y} + XY = X$

(b) $X + Y = X + Y$

14. Draw logic circuit diagram for the following expressions.

(a) $Y = AB + \overline{B}C + \overline{C}\overline{A}$

(b) $Y = \overline{X}\overline{Y} + Z\overline{X} + \overline{Y}Z$

15. Simplify the following Boolean expressions.

(a) $A\overline{B}\overline{C} + \overline{A}\overline{B}\overline{C} + \overline{A}B\overline{C} + \overline{A}\overline{B}C$

(b) $AB + A\overline{B} + \overline{A}C + \overline{A}\overline{C}$

****