BASIC ELECTRONICS

DIGITAL CIRCUITS — UNIT-4

BASIC LOGIC GATE

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

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

A Gate is a basic electronic circuit which operates on one or more signals to produce an output signal.

Gates are digital (two-state 0 or 1) 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.

There are *three* types of logic gates:

- Inverter (NOT gate)
- OR gate
- AND gate

Gate A Gate is a basic electronic circuit which operates on one or more signals to produce an output signal.

INVERTER (NOT GATE)

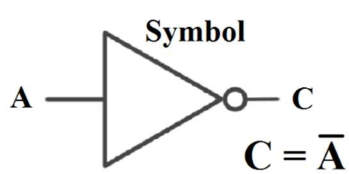
An **Inverter (NOT Gate)** is a gate with only one input signal and one output signal, the output state is always the opposite of the input state.

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.

Following tables summarize the operation:

X	$\overline{X}$	X	$\overline{X}$
Low	High	0	1
High	Low	1	0

A low input *i.e.*, 0 produces high output *i.e.*, 1, and vice versa. The symbol for inverter is given below:



Truth Table

INPUT	OUTPUT
A	NOT A
0	1
1	0

Inverter (Not Gate) An Inverter (Not Gate) is a gate with only one input signal and one output signal; the output state is always the opposite of the input state.

OR GATE

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 is 1 (high).

If all inputs are 0 then output is also 0. If one or more inputs are 1, then output is 1.

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: one or more 1 (high) inputs produce output as 1.

OR Gate 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

Table *Two input OR gate*

X	Y	F
0	0	0
0	1	1
1	0	1
1	1	1

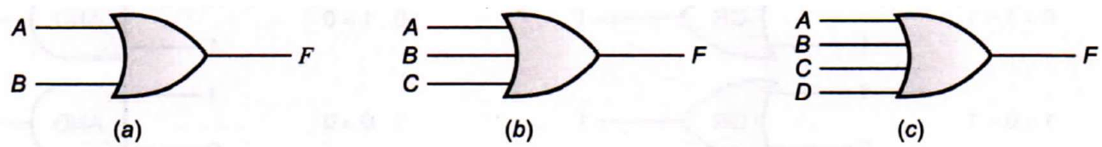
$F = X + Y$

Table *Three input OR gate*

X	Y	Z	F
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

$F = X + Y + Z$

The symbol for OR gate is given below :



(*a*) Two input OR gate (*b*) Three input OR gate (*c*) Four input OR gate.

AND GATE

The **AND Gate** can have two or more than two input signals and produce an output signal. When all the inputs are 1 *i.e.,* high then the output is 1 otherwise output is 0 only.

If any of the inputs is 0, the output is 0. To obtain output as 1, all inputs must be 1.

An AND gate can have as many inputs as desired. Following tables illustrate AND action.

Table *Two input AND gate*

X	Y	F
0	0	0
0	1	0
1	0	0
1	1	1

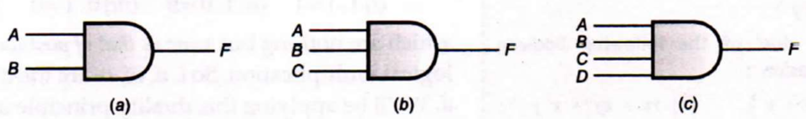
Here, $F = X . Y$

Table *Three input AND gate*

X	Y	Z	F
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

Here, $F = X . Y . Z$

The symbol for AND is



(*a*) 2-input AND gate (*b*) 3-input AND gate (*c*) 4-input AND gate

AND Gate The **AND Gate** can have two or more than two input signals and produce an output signal. When all the inputs are 1 *i.e.,* high then the output is 1 otherwise output is 0 only.

MORE ABOUT LOGIC GATES

We have covered three basic logic gates NOT, OR, AND so far, but there are some more logic gates also which are derived from three basic gates (*i.e.,* AND, OR and NOT). These gates are more popular than NOT, OR and AND and are widely used in industry. This section introduces NOR, NAND, XOR, XNOR gates.

NOR GATE

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

If either of the two inputs is 1 *(high)*, the output will be 0 *(low)*. NOR gate is *inverted* OR *gate.*

The NOR gate can have as many inputs as desired. No matter how many inputs are there, the action of NOR gate is the dame *i.e.*, All 0 (*low*) inputs produce output as 1.

Following truth tables illustrate NOR action:

Table *2-input NOR gate*

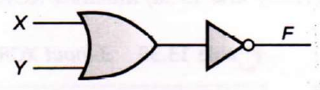
X	Y	F
0	0	1
0	1	0
1	0	0
1	1	0

$F = \overline{X+Y}$

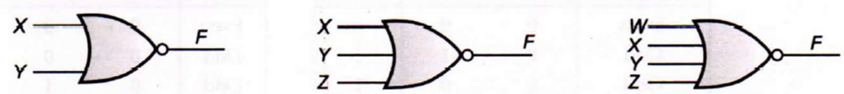
Table *3-input NOR gate*

X	Y	Z	F
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

$F = \overline{X+Y+Z}$



(*a*)



(*b*) (*c*) (*d*)

(*a*) Logical meaning of NOR gate (*b*) 2 input NOT gate
(*c*) 3 input NOR gate (*d*) 4 input NOR gate

Nor Gate The Nor Gate has two or more input signals but only one output signal. If all the inputs are 0 (*i.e.*, low), then the output signal is 1 (high).

NAND GATE

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 *gate is inverted* AND *gate.* Thus, for all 1 (*high*) inputs, it produces 0 (*low*) output, otherwise for any other input combination, it produces a 1 (*high*) output. NAND gate can also have as many inputs as desired.

NAND action is illustrated in following Truth Tables:

Table *2-input NAND gate*

X	Y	F
0	0	1
0	1	1
1	0	1
1	1	0

$F = \overline{XY}$

The logical meaning of NAND gate can be shown as follows :

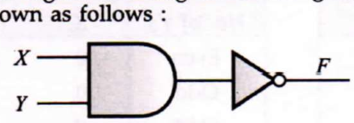
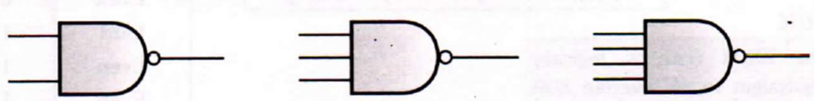


Table *3-input NAND gate*

X	Y	Z	F
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

$F = \overline{XYZ}$

The symbols of 2, 3, 4 input NAND gates are given below :



(*a*) (*b*) (*c*)

(*a*) 2-input NAND gate (*b*) 3-input NAND gate (*c*) 4-input NAND gate.

NAND Gate 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).

XOR GATE (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.

In Boolean algebra (+) sign stands for XOR operation. Thus, *A XOR B* can be written as *A* (+) B.

Following Truth Tables illustrate XOR operation:

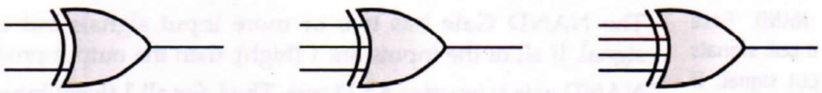
Table *2-input XOR gate*

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

Table *3-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

The symbols of XOR gates are given below :



(a) (b) (c)

(*a*) 2-input XOR gate (*b*) 3-input XOR gate (*c*) 4-input XOR gate.

XOR addition can be summarised as follows :

$0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$, $1 \oplus 1 = 0$

XOR gate XOR gate produces output 1 for only those input combinations that have odd number of 1's.

NOTE

Remember odd number of 1's produce output 1.

XNOR GATE (EXCLUSIVE NOR GATE)

The **XNOR Gate** is logically equivalent to an inverted XOR *i.e.*, XOR gate followed by a NOT gate (inventor). Thus, XNOR produces 1 (high) output when the input combination has even number of 1's.

XNOR gate XNOR gate produces output 1 for only those input combinations that have even number of 1's.

Following tables illustrate XNOR action:

Table *2-input XNOR gate*

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

Table *3-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

NOTE

The XNOR Gate is logically equivalent to an inverted XOR *i.e.*, XOR gate followed by a NOT gate (inventor).