Lecture 2

DIGITAL LOGIC GATES AND BOOLEAN ALGEBRA

- The operation of the digital computer is based on the storage and processing of binary data.
- storage elements that can exist in one of two stable states,1 and 0 of circuits than can operate on : binary data under the control of control signals to implement the various computer functions.

- The digital circuitry in digital computers and other digital systems is designed, and
- its behavior is analyzed, with the use of a mathematical discipline known as Boolean algebra.
- The techniques were subsequently used in the analysis and design of electronic digital circuits.
- Boolean algebra turns out to be a convenient(مناسبه) tool in two areas:

■■ Analysis: It is an economical way of describing the function of digital circuitry.

■■ Design: Given a desired function, Boolean algebra can be applied to develop a simplified implementation of that function.

- operations are AND, OR, and NOT, which are symbolically represented by dot, plus sign, and over bar:

$$A \text{ AND } B = A . B$$

$$A \text{ OR } B = A + B$$

$$\text{NOT } A = A$$

- The operation AND yields true (binary value 1) if and only if both of its operands are true.
- The operation OR yields true if either or both of its operands are true.
- The unary operation NOT inverts the value of its operand. For example, consider the equation

$$D = A + (B . C)$$

- D is equal to 1 if A is 1 or if both B = 0 and C = 1.
- Otherwise D is equal to 0.

- lists three other useful operators: XOR, NAND, and NOR. The exclusive- or (XOR) of two logical operands is 1 if and only if exactly
- one of the operands has the value 1. The NAND function is the complement (NOT) of the AND function, and the NOR is the complement of OR:

$$A \text{ NAND } B = \text{NOT } (A \text{ AND } B) = AB$$

$$A \text{ NOR } B = \text{NOT } (A \text{ OR } B) = A + B$$

- As we shall see, these three new operations can be useful in implementing certain digital circuits.

(a) Boolean Operators of Two Input Variables

P	Q	NOT P ($\overline{P}$)	P AND Q ($P \cdot Q$)	P OR Q ($P + Q$)	P NAND Q ($\overline{P \cdot Q}$)	P NOR Q ($\overline{P + Q}$)	P XOR Q ($P \oplus Q$)
0	0	1	0	0	1	1	0
0	1	1	0	1	1	0	1
1	0	0	0	1	1	0	1
1	1	0	1	1	0	0	0

(b) Boolean Operators Extended to More than Two Inputs (A, B, . . .)

Operation	Expression	Output = 1 if
AND	$A \cdot B \cdot \ldots$	All of the set {A, B, ...} are 1.
OR	$A + B + \ldots$	Any of the set {A, B, ...} are 1.
NAND	$\overline{A \cdot B \cdot \ldots}$	Any of the set {A, B, ...} are 0.
NOR	$\overline{A + B + \ldots}$	All of the set {A, B, ...} are 0.
XOR	$A \oplus B \oplus \ldots$	The set {A, B, ...} contains an odd number of ones.

Logic gates

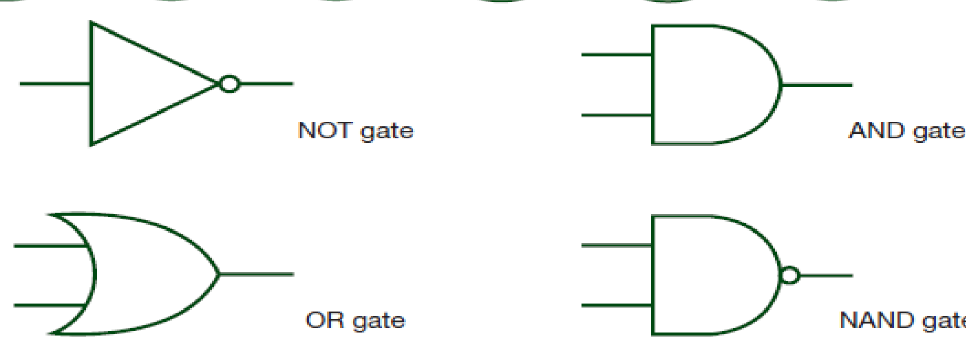
A large number of electronic circuits (in computers, control units, and so on) are made up of logic gates. These process signals which represent true or false.

Logical functions are implemented by the interconnection of gates.

A gate is an electronic circuit that produces an output signal that is a simple Boolean operation on its input signals. The basic gates used in digital logic are

NOT AND OR NAND NOR XOR

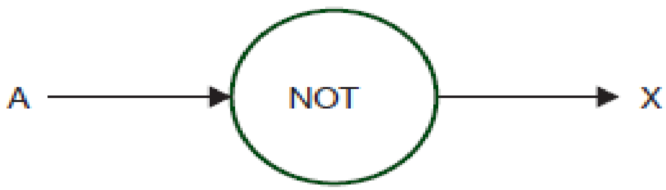
There are many different logic gates but we will concentrate on these.

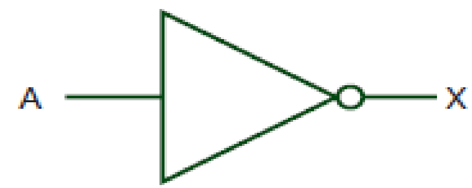


- Each gate is defined in three ways: graphic symbol, algebraic notation, and truth table.

Description of the six logic gates

NOT gate





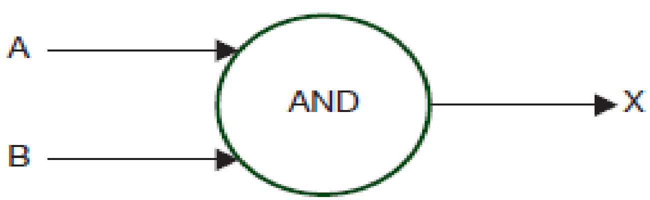
The output (X) is true (i.e. 1 or ON) if:

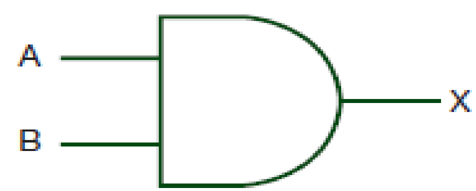
INPUT A is **NOT TRUE** (i.e. 0 or OFF)

Truth table for: ***X = NOT A***

INPUT A	OUTPUT X
0	1
1	0

AND gate





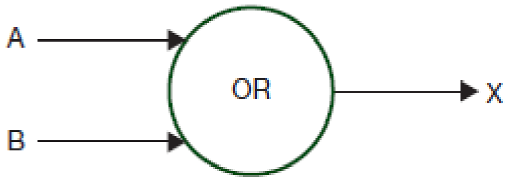
The output (X) is true (i.e. 1 or ON) if:

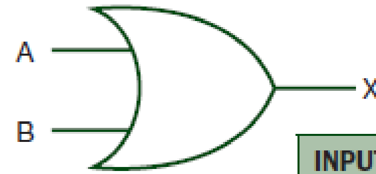
INPUT A AND INPUT B are **BOTH TRUE** (i.e. 1 or ON)

Truth table for: ***X = A AND B***

INPUT A	INPUT B	OUTPUT X
0	0	0
0	1	0
1	0	0
1	1	1

OR gate





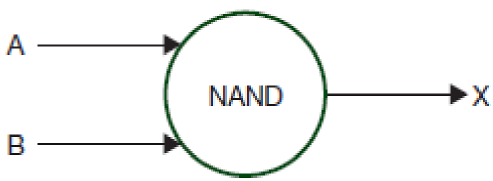
The output (X) is **true** (i.e. 1 or ON) if:

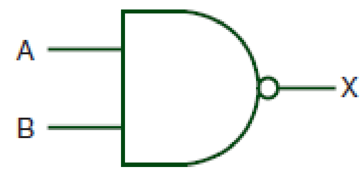
INPUT A OR INPUT B is **TRUE** (i.e. 1 or ON)

Truth table for: ***X = A OR B***

INPUT A	INPUT B	OUTPUT X
0	0	0
0	1	1
1	0	1
1	1	1

NAND gate





The output (X) is **true** (i.e. 1 or ON) if:

INPUT A AND INPUT B are **NOT BOTH TRUE** (i.e. 1 or ON)

Truth table for: ***X = NOT A AND B***

INPUT A	INPUT B	OUTPUT X
0	0	1
0	1	1
1	0	1
1	1	0

The output (X) is **true** (i.e. 1 or ON) if:

INPUT A OR (NOT INPUT B) OR (NOT INPUT A) OR INPUT B

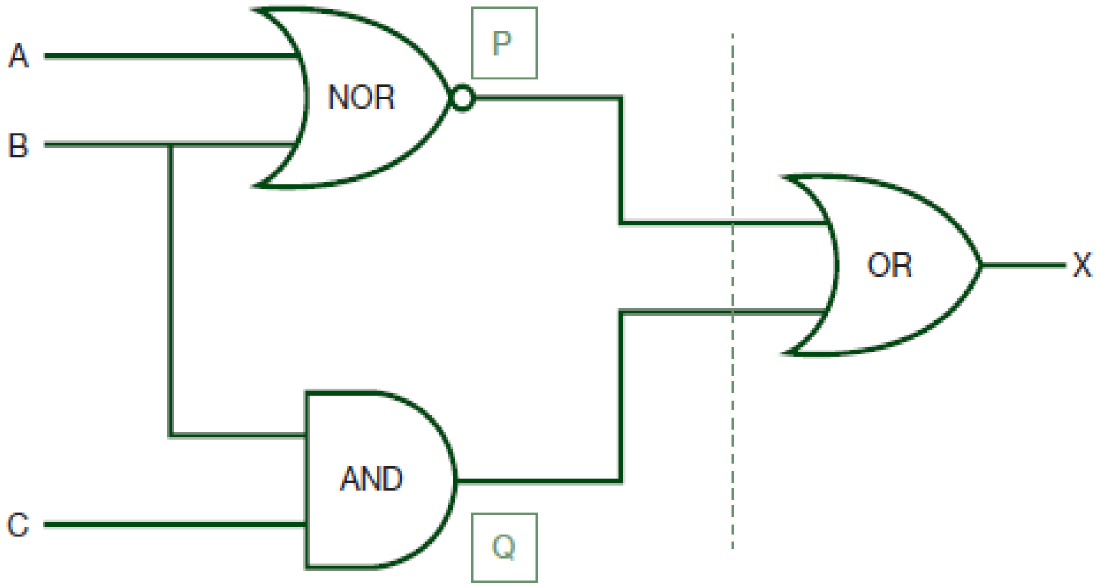
is TRUE (i.e. 1 or ON)

Truth table for: ***X = A OR (NOT B) OR (NOT A) OR B***

INPUT A	INPUT B	OUTPUT X
0	0	0
0	1	1
1	0	1
1	1	0

Example 1

Produce a truth table from the following logic circuit (network).



To show how this works, we will split the logic circuit into two parts (shown by the dotted line).

First part

There are 3 inputs; thus we must have 2^3 (i.e. 8) possible combinations of 1s and 0s.

To find the values (outputs) at points P and Q, it is necessary to consider the truth tables for the NOR gate (output P) and the AND gate (output Q) i.e.

P = A NOR B
Q = B AND C

INPUT A	INPUT B	INPUT C	OUTPUT P	OUTPUT Q
0	0	0	1	0
0	0	1	1	0
0	1	0	0	0
0	1	1	0	1
1	0	0	0	0
1	0	1	0	0
1	1	0	0	0
1	1	1	0	1

Second part

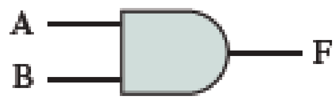
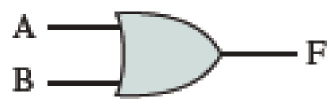
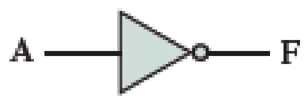
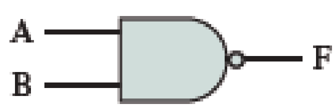
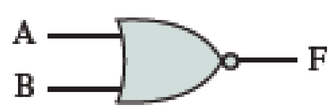
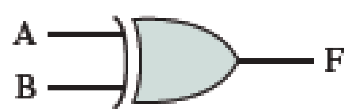
There are 8 values from P and Q which form the inputs to the last OR gate.

Hence we get X = P OR Q which gives the following truth table:

INPUT P	INPUT Q	OUTPUT X
1	0	1
1	0	1
0	0	0
0	1	1
0	0	0
0	0	0
0	0	0
0	1	1

INPUT A	INPUT B	INPUT C	OUTPUT X
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

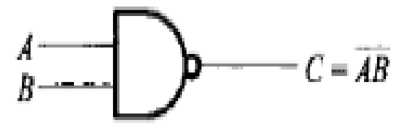
Which now gives us the final truth table for the logic circuit given at the start of the example:

Name	Graphical Symbol	Algebraic Function	Truth Table
AND		$F = A \bullet B$ or $F = AB$	A B | F 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1
OR		$F = A + B$	A B | F 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1
NOT		$F = \overline{A}$ or $F = A'$	A | F 0 | 1 1 | 0
NAND		$F = \overline{AB}$	A B | F 0 0 | 1 0 1 | 1 1 0 | 1 1 1 | 0
NOR		$F = \overline{A + B}$	A B | F 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 0
XOR		$F = A \oplus B$	A B | F 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 0

Basic Logic gats

- **Advanced Logic gates:**

1. NAND Gate:



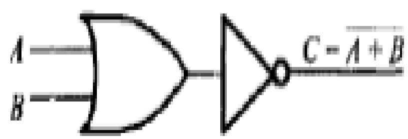
The NAND output is generated by inverting the output of an AND operation.

The truth table for the NAND gate is:

NAND gate Truth Table

A	B	$C = \overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

2. NOR Gate:



The NOR output is produced by inverting the output of an OR operation.

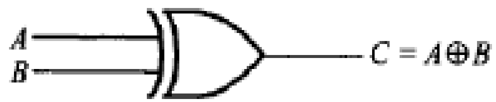
The truth table for the NOR gate is:

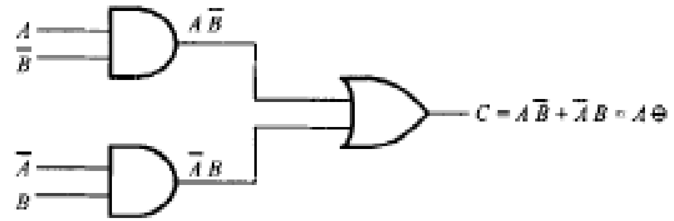
NOR gate Truth Table

A	B	$C = \overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

- **Advanced Logic gates:**

3. Exclusive-OR Gate (XOR) :





$$C = A \oplus B = A\bar{B} + \bar{A}B$$

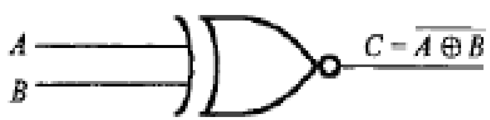
The Exclusive-OR operation (XOR) generates an output of 1 if the inputs are different and 0 if the inputs are the same.

The $\oplus$ or $\forall$ symbol is used to represent the XOR operation.

The truth table for Exclusive-OR operation is :

Inputs		Output
A	B	$C = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

4. Exclusive-NOR Gate (XNOR) :



The one's complement of the Exclusive-OR operation is known as the Exclusive-NOR operation..

The XNOR operation is represented by the symbol $\odot$.

The truth table for XNOR operation is :

XNOR gate Truth Table

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