BENHA UNIVERSITY
FACULTY OF ENGINEERING AT SHOUBRA

ELC301
Electronic Engineering

Lecture #5
Logic Gates

Instructor:
Dr. Moataz Elsherbini

Logic Gates

Agenda

- Basic Concepts
- Logic Gates
- Examples of logic Gates

- **Digital electronic circuits**, are electronics that represent signals by **discrete** bands of analog **levels**, rather than by continuous ranges (as used in analogue electronics).

- In most cases the number of **states** is **two**.
- They are represented by **two voltage bands**: one near a reference value (typically termed as "***ground***" or zero volts), and the other a value near the ***supply*** voltage.
- These correspond to the "false" ("**0**") and "true" ("**1**") values of the **Boolean** domain, respectively, yielding **binary** code.

Boolean Algebra	Boolean Logic	Voltage State
Logic "1"	True (T)	High (H)
Logic "0"	False (F)	Low (L)

- A digital circuit is often constructed from small electronic circuits called **logic gates** that can be used to create combinational logic.
- Each logic gate represents a **function of boolean logic**.
- A logic gate is an arrangement of electrically controlled switches, better known as **transistors**.
- Logic gates often use the **fewest number of transistors** in order to reduce their size, power consumption and cost, and increase their reliability.
- **Integrated circuits** are the **least expensive** way to make logic gates in large volumes. Integrated circuits are usually designed by engineers using electronic design automation software

- Binary variables take on one of two values.
- Logical operators operate on binary values and binary variables.
- Basic logical operators are the logic functions AND, OR and NOT.
- Logic gates implement logic functions.
- Boolean Algebra: a useful mathematical system for specifying and transforming logic functions.
- We study Boolean algebra as a foundation for designing and analyzing digital systems!

Binary Variables

- Recall that the two binary values have different names:
  - True/False
  - On/Off
  - Yes/No
  - 1/0
- We use 1 and 0 to denote the two values.
- Variable identifier examples:
  - A, B, y, z, or X_1 for now
  - RESET, START_IT, or ADD1 later

Logical Operations

- The three basic logical operations are:
  - AND
  - OR
  - NOT
- AND is denoted by a dot (·).
- OR is denoted by a plus (+).
- NOT is denoted by an over bar (¯), a single quote mark (') after.

Notation Examples

- Examples:
  - Y=A.B is read “Y is equal to A AND B.”
  - z = x+y is read “z is equal to x OR y.”
  - $X = \bar{A}$ is read “X is equal to NOT A.”
- Note: The statement:
  - 1 + 1 = 2 (read “one plus one equals two”)

  is not the same as

  - 1 + 1 = 1 (read “1 or 1 equals 1”).

Operator Definitions

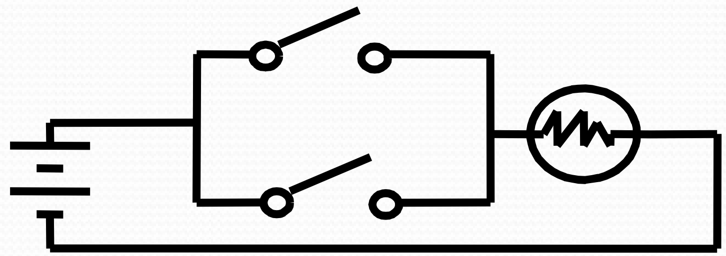
- Operations are defined on the values "0" and "1" for each operator:

AND	OR	NOT	Buffer
$0 \cdot 0 = 0$	$0 + 0 = 0$	$\overline{0} = 1$	$0 = 0$
$0 \cdot 1 = 0$	$0 + 1 = 1$	$\overline{1} = 0$	$1 = 1$
$1 \cdot 0 = 0$	$1 + 0 = 1$		
$1 \cdot 1 = 1$	$1 + 1 = 1$		

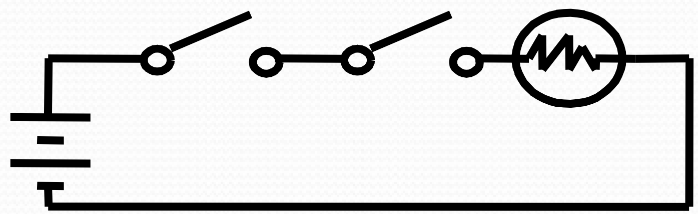
Logic Function Implementation

- **Using Switches**
  - Inputs:
    - logic 1 is switch closed
    - logic 0 is switch open
  - Outputs:
    - logic 1 is light on
    - logic 0 is light off.
  - NOT input:
    - logic 1 is switch open
    - logic 0 is switch closed

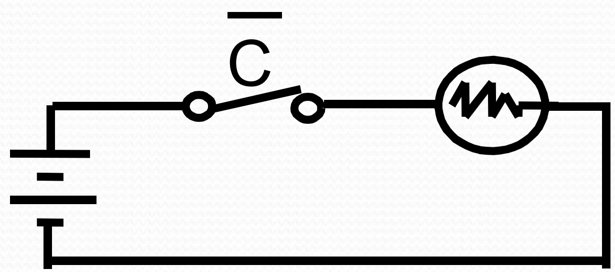
Switches in parallel => OR



Switches in series => AND

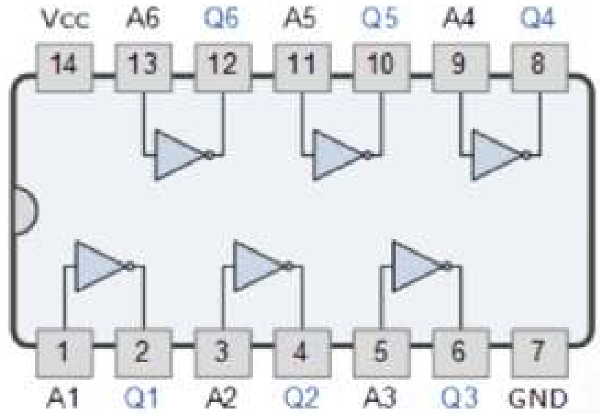


Normally-closed switch => NOT

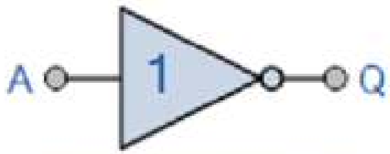


NOT Gate -- Inverter

If A is NOT true, then Q is true



- The Logic NOT Gate Truth Table

Symbol	Truth Table	
 Inverter or NOT Gate	A	Q
	0	1
	1	0
Boolean Expression Q = not A or $\overline{A}$	**Read as inverse of A gives Q**	

OR Gate

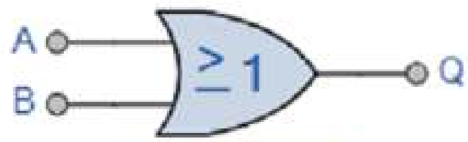
If either A or B is true, then Q is true

- 2-input Transistor OR Gate

B	A	OUT
0	0	0
0	1	1
1	0	1
1	1	1

- The 2-input Logic OR Gate

Symbol



2-input OR Gate

Truth Table

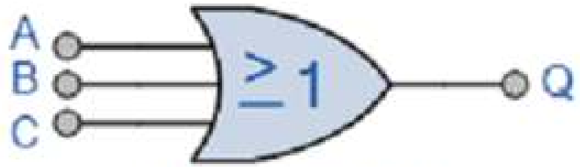
B	A	Q
0	0	0
0	1	1
1	0	1
1	1	1

Boolean Expression Q = A+B

Read as A OR B gives Q

The 3-input Logic OR Gate

Symbol



3-input OR Gate

Truth Table

C	B	A	Q
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

Boolean Expression Q = A+B+C

Read as A OR B OR C gives Q

AND Gate

If both A and B are true, then Q is true

Symbol

2-input AND Gate

Truth Table

B	A	Q
0	0	0
0	1	0
1	0	0
1	1	1

Boolean Expression Q = A.B

Read as A AND B gives Q

Symbol

3-input AND Gate

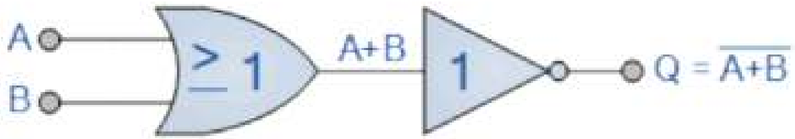
Truth Table

C	B	A	Q
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

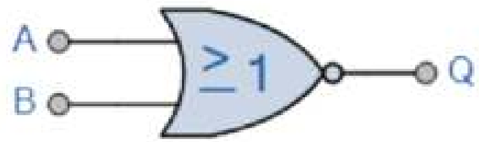
Boolean Expression Q = A.B.C

Read as A AND B AND C gives Q

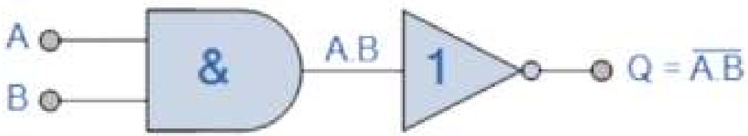
NOR Gate



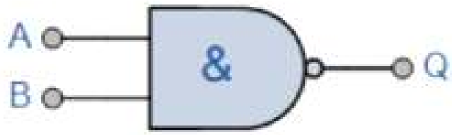
2-input "OR" gate plus a "NOT" gate



NAND Gate

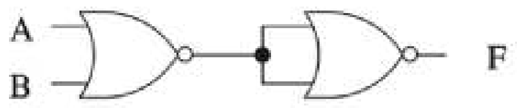


2-input "AND" gate plus a "NOT" gate

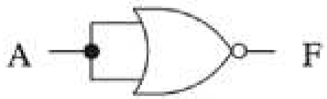


NOR & NAND Universal Gate

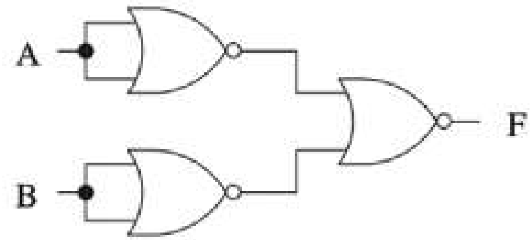
- Any gate can be constructed using them.



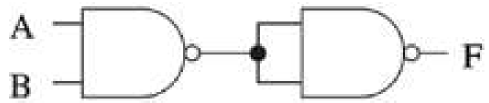
OR gate



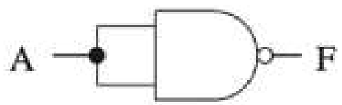
NOT gate



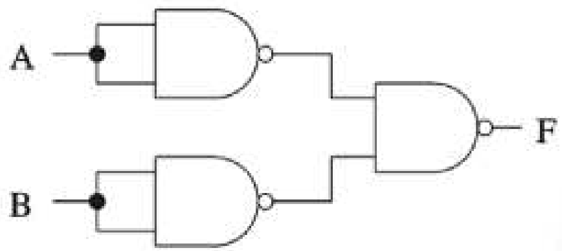
AND gate



AND gate



NOT gate



OR gate

Example of a Logic Function

3-input majority function

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

- Logical expression form

  F = A B + B C + A C