Computer System Architecture
COMP201TH
Lecture-1

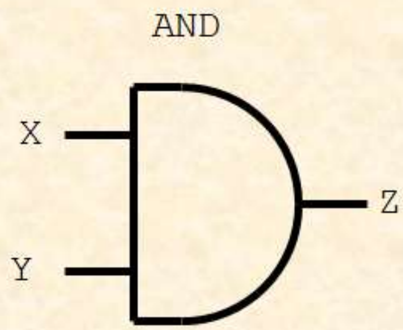
"If people do not believe that mathematics is simple, it is only because they do not realize how complicated life is."

.... John von Neumann

- **Compute Architecture:**
  - Computer Architecture is a specification describing how hardware and software technologies interact to create a computer platform or system.
  - Computer architecture consists of three main categories:
    - System Design:
      - This includes all the hardware parts, such as CPU, data processors, multiprocessors, memory controllers and direct memory access. This part is the actual computer system.
    - Instruction Set Architecture:
      - This includes the CPU's functions and capabilities; the CPU's programming language, data formats, processor register types and instructions used by computer programmers. This part is the software that makes it run, such as Windows or Photoshop etc.
    - Microarchitecture:
      - This defines the data processing and storage element or data paths and how they should be implemented into the instruction set architecture.
- **Digital Computers:**
  - The digital computer is a digital system that performs various computational tasks.
  - Digital Computers use the binary number system, which has two digits: 0 and 1.
    - A binary digit is called a bit.
  - Information is represented in digital computers in groups of bits.
  - By using various coding techniques, groups of bits can be made to represent not only binary numbers but also discrete symbols such as decimal digits or letters of the alphabet.
- **Logic Gates:**
  - Binary information is represented in digital computers by physical quantities called signals.
  - Electric signals such as voltage exist throughout the computer in either one of two recognizable states.

        - The two states represent a binary variable that can be equal to 1 or 0.
            - E.g. in a particular computer; a signal of 3 V may represent binary 1 while a signal of 0.5V may represent binary 0.
    - The manipulation of binary information is done by logic circuits called gates.
        - Gates are blocks of hardware that produce signals of binary 1 or 0 when input logic requirements are satisfied.
    - The input-output relationship of the binary variables for each gate can be represented in tabular form by a truth table.
    - The various logic gates are:
        - AND
        - OR
        - NOT
        - NAND
        - NOR
        - XOR
        - XNOR
- **Hardware Description Language(HDL):**
    - HDL is a textual language that is specifically intended to clearly and concisely capture the defining features of digital design to describe the digital circuits.
        - Another approach to describe a digital circuit is visual approach called schematic. In theory, we could interpret a CPU as a vast sea of transistors, but it is much easier to organize transistors into logic gates, logic gates into adders or registers or timing modules, registers into memory banks and so forth.
            - This hierarchical structure allows us to effectively represent a digital circuit by means of interconnected diagrams.
    - The most popular hardware description languages are Verilog and VHDL (VHIC-HDL, Very High Speed Integrated Circuit Hardware Description Language).
        - ABEL (Advanced Boolean Expression Language)→ now obsolete.

- **AND Gate:**
    - In AND gate; the output is 1 if input A and input B are both equal to 1, otherwise the output is 0.



Z = X & Y

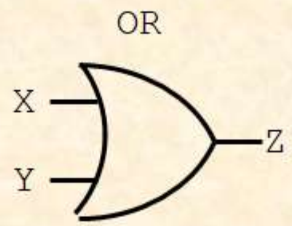
X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

The algebra operation symbol of the AND function is same as the multiplication symbol of ordinary arithmetic. We can either use a dot between the variables or concatenate the variables without an operation symbol between them. Other representations of AND function is as below:

- X & Y (Verilog and ABEL)
- X **and** Y (VHDL)
- X ∧ Y
- X ∩ Y
- X * Y
- XY
- **and**(Z,X,Y) (Verilog)

AND gates may have more than two inputs, and by definition, the output is 1 if and only if all inputs are 1.

- **OR Gate:**
  - The OR gate produces the inclusive-OR function i.e. the output is 1 if input A or input B or both inputs are 1; otherwise the output is 0.



Z = X | Y

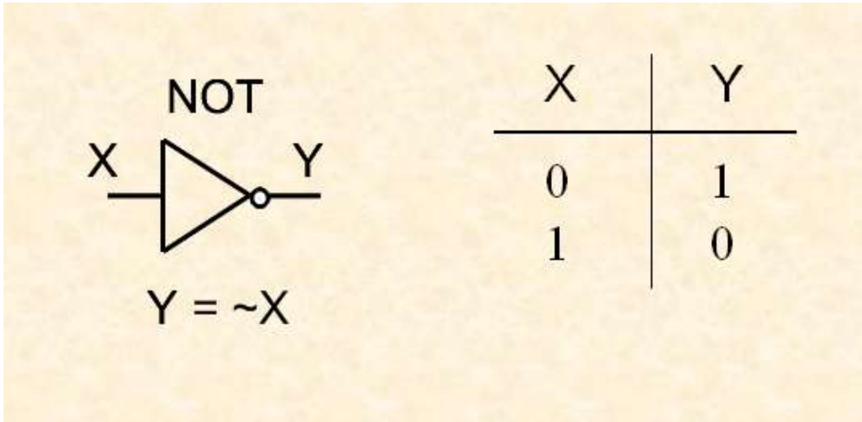
X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

The algebraic symbol of the OR function is +, similar to arithmetic addition. Other representations of OR function is as below:

- X | Y (Verilog)
- X # Y (ABEL)
- X **or** Y (VHDL)
- X + Y
- X V Y
- X U Y
- **or**(Z,X,Y) (Verilog)

OR gates may have more than two inputs, and by definition, the output is 1 if any input is 1.

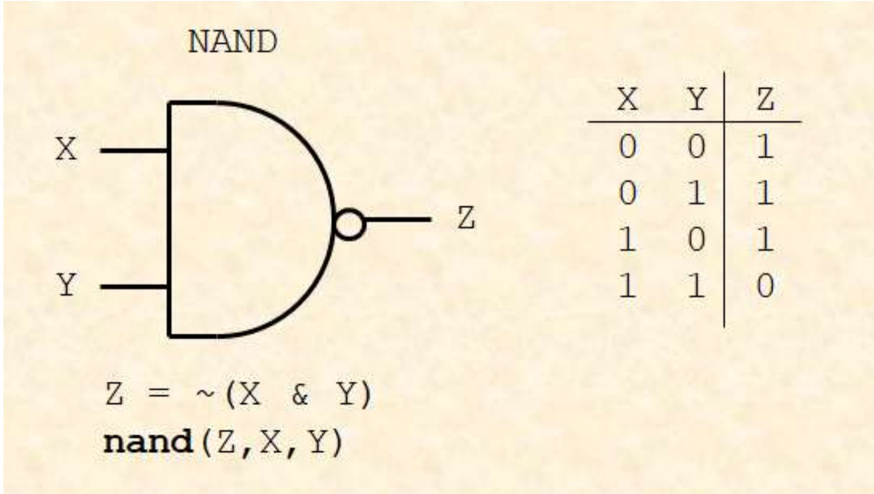
- **NOT Gate:**
    - NOT gate also called as inverter circuit inverts the logic sense of a binary signal i.e. if input is 1 it will produce output as 0 and vice versa.



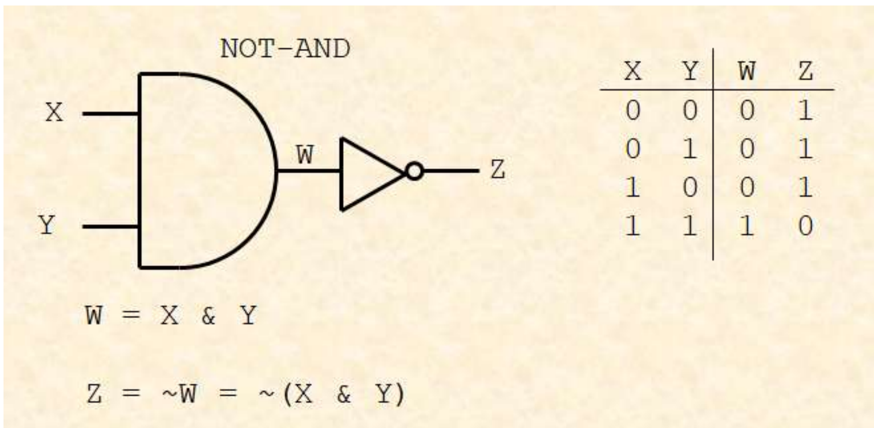
The algebraic symbol used for the logic complement is either a prime or a bar over the variable symbol. Other representations of NOT function are as below:

- Y = ~X (Verilog)
- Y = !X (ABEL)
- Y = **not** X (VHDL)
- Y = X′
- Y = ¬X
- Y = $\overline{X}$
- **not**(Y,X) (Verilog)

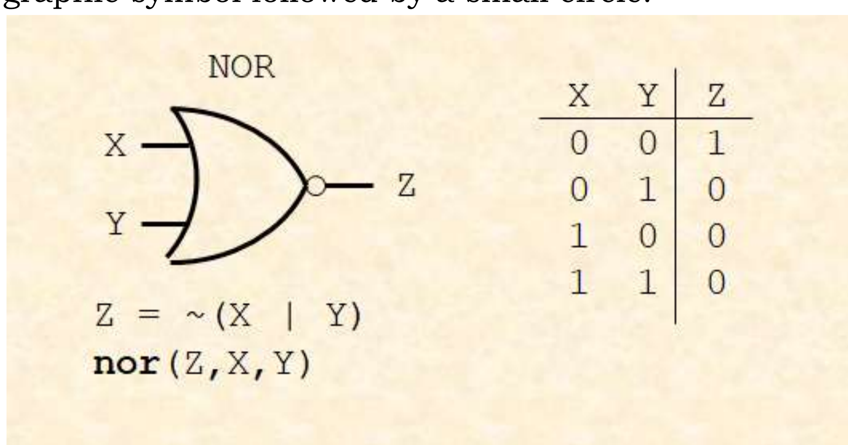
- **NAND Gate:**
    - The NAND function is the complement of the AND function, as indicated by the graphic symbol, which consists of an AND graphic symbol followed by a small circle.

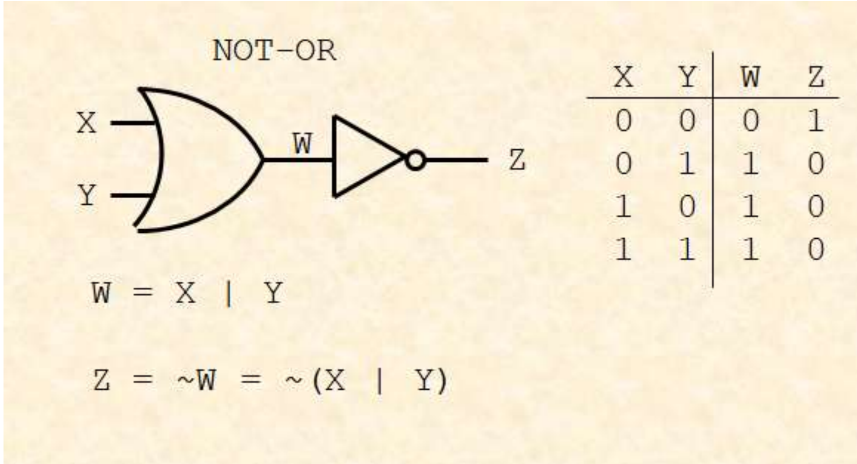


The designation NAND is derived from the abbreviation of NOT-AND.

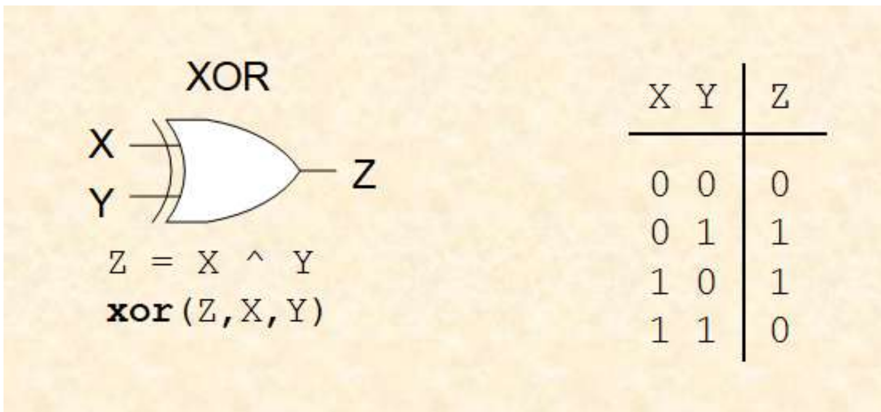


- **NOR Gate:**
  - The NOR gate is the complement of the OR gate and uses an OR graphic symbol followed by a small circle.

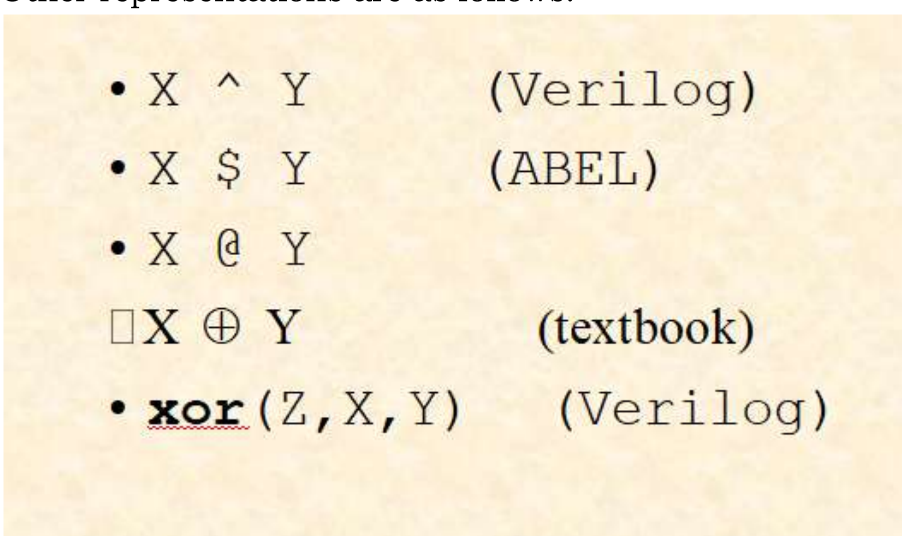




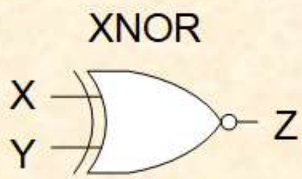
- **Exclusive- OR (XOR) Gate:**
  - The output of XOR gate is 1 if any input is 1 but excludes the combination when both inputs are 1.



XOR function can also be written as Z= X`Y+XY`.
Other representations are as follows:



- **Exclusive-NOR (XNOR) Gate:**
  - The exclusive- NOR is the complement of the exclusive-OR. The output of this gate is 1 only if both inputs are equal to 1 or both inputs are equal to 0.



Z = ~(X ^ Y)
Z = X ~^ Y
xnor(Z,X,Y)

X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	1

The exclusive-NOR operation is also termed as an odd function because its output is 1 if an odd number of inputs are 1.
Other representations of XNOR are:

- X ~^ Y (Verilog)
- !(X $ Y) (ABEL)
- X @ Y
- **xnor**(Z,X,Y) (Verilog)