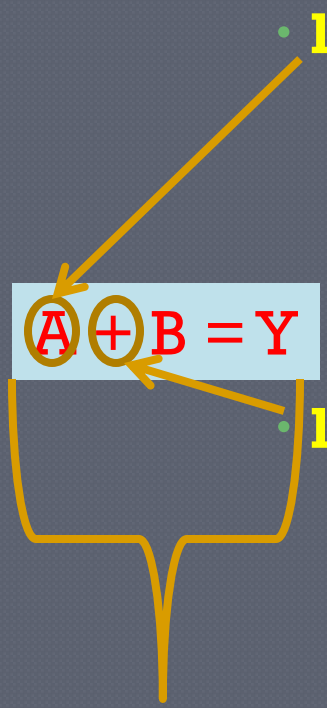
Digital Logic Design (CSNB163)

Module 3

Binary Logic

- Binary logic consists of :
  - **logic variables**
    - designated by alphabet letters, e.g. A, B, C… x, y, z, etc.
    - have ONLY 2 possible values: **1** or **0**
  - **logic operators**
    - 3 basic logical operators representing 3 basic logical operations: **AND, OR** and **NOT**



- The overall equation made up a **Boolean Equation**

Logic Gates

- Logic gates are **electronic circuits** that **operate on one or more binary input signals** to produce **a binary output signal**.
- Each logic gate represents a single processing unit that **performs binary logic operations**

Truth Tables

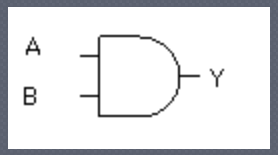
- Truth table is a **table of all possible combinations** of the input binary variables
- If there are ***n*** input variables, there are in total 2^n possible combination
- Truth tables show the relationship between the **input binary values** that the **output binary result of the operation**

Timing Diagram

- Timing diagram is a **graphical representation** that describe the **relationship** between **input(s)** and **output signals** in a digital circuit system.
- In actual implementation, the input and output signals are **electrical signals** that can be categorized into binary logic state of 1 or 0 based the defined range of voltages associated with each state.
- Input signal can either **static** (e.g. logic state 0 if grounded) or **dynamic** (e.g. if received input from flip flops)

Binary Logic Operation (AND)

- Symbol dot or none
- Boolean Equation
  A . B = Y or AB = Y
- Output variable is 1 if **all** input variables is 1
- Logic Gate

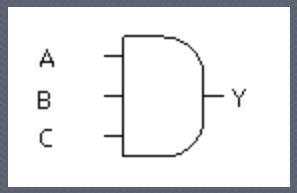


- Truth table

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

Binary Logic Operation (AND)

- Boolean Equation
  A . B . C = Y
- Logic Gate

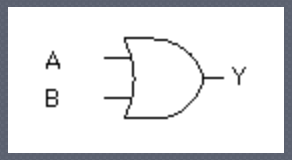


- Truth table

A	B	C	Y
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

Binary Logic Operation (OR)

- Symbol plus
- Boolean Equation
  A + B = Y
- Output variable is 1 if **any** input variables is 1
- Logic Gate

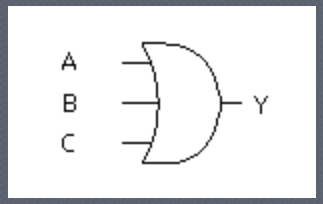


- Truth table

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

Binary Logic Operation (OR)

- Boolean Equation
  A + B + C = Y
- Logic Gate

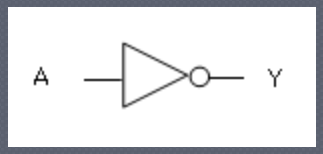


- Truth table

A	B	C	Y
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

Binary Logic Operation (NOT)

- Symbol prime or overbar
  $\overline{A} = Y$ or $A' = Y$
- Output variable is **always opposite** of input variable
- Logic Gate

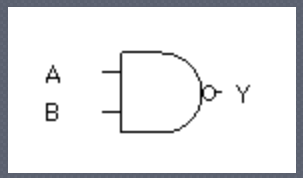


- Truth table

A	Y
0	1
1	0

Binary Logic Operation (NAND)

- A combination of **AND** operation **followed by NOT** operation
- Symbol
  $A \cdot B = Y$ or $\overline{AB} = Y$
- Logic Gate



- Truth table

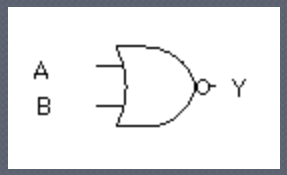
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Binary Logic Operation (NOR)

- A combination of **OR** operation **followed by NOT** operation
- Symbol

$$\overline{A + B} = Y$$

- Logic Gate

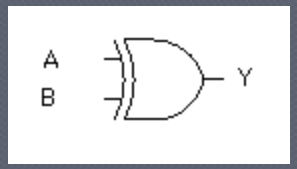


- Truth table

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Binary Logic Operation (XOR)

- Exclusive OR (XOR)
- Symbol ⊕
- Boolean Expression
  AB' + A'B = Y
- Logic Gate

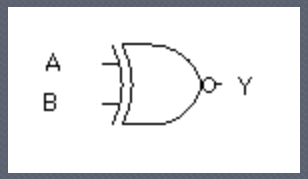


- Truth table

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Binary Logic Operation (XNOR)

- Exclusive NOR (XNOR)
- Symbol $(A \oplus B)'$
- Boolean Expression $AB + A'B' = Y$
- Logic Gate



- Truth table

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

NAND and NOR gates

- Digital circuits are more frequently constructed with NAND or NOR gates rather than with AND or OR gates.
- The main reason behind this is that NAND and NOR gates are easier to fabricate with electronic components.

XOR gates

- Exclusive OR gates are often used for error detection and correction purposes.
- This is done by using XOR gates in building the parity generator and checker digital logic circuits.

Digital Logic Design (CSNB163)

End of Module 3