Basic Digital Circuits & Minimization using Boolean ALgebra

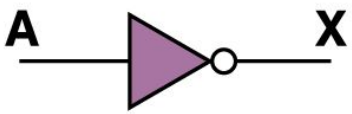
NOT Gate

- A NOT gate accepts one input value and produces one output value

Boolean Expression

$X = A'$

Logic Diagram Symbol



Truth Table

A	X
0	1
1	0

NOT Gate

- By definition, if the input value for a NOT gate is 0, the output value is 1, and if the input value is 1, the output is 0
- A NOT gate is sometimes referred to as an ***inverter*** because it inverts the input value

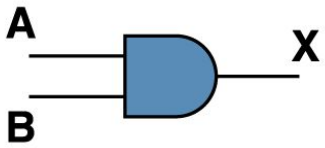
AND Gate

- An AND gate accepts two input signals
- If the two input values for an AND gate are both 1, the output is 1; otherwise, the output is 0

Boolean Expression

$$X = A \cdot B$$

Logic Diagram Symbol



Truth Table

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

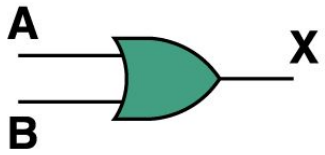
OR Gate

- If the two input values are both 0, the output value is 0; otherwise, the output is 1

Boolean Expression

$$X = A + B$$

Logic Diagram Symbol



Truth Table

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

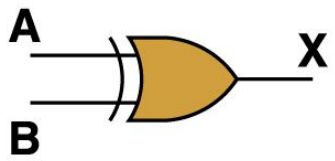
XOR Gate

Boolean Expression

$X = A \oplus B$

X= A’ . B + A . B’

Logic Diagram Symbol



Truth Table

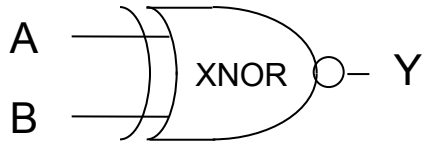
A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

- XOR, or *exclusive* OR, gate
  - An XOR gate produces 0 if its two inputs are the same, and a 1 otherwise

XNOR Gate

Also called **Coincidence gate**.
The Output is High when both the inputs are same or Coincident.

$Y = A \cdot B + A' \cdot B'$



A	B	A’	B’	AB	A’B’	Y=AB+ A’B’
0	0	1	1	0	1	1
0	1	1	0	0	0	0
1	0	0	1	0	0	0
1	1	0	0	1	0	1

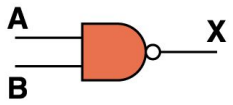
NAND and NOR Gates

- The NAND and NOR gates are essentially the opposite of the AND and OR gates, respectively

Boolean Expression

$X = (A \cdot B)'$

Logic Diagram Symbol



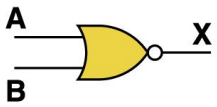
Truth Table

A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

Boolean Expression

$X = (A + B)'$

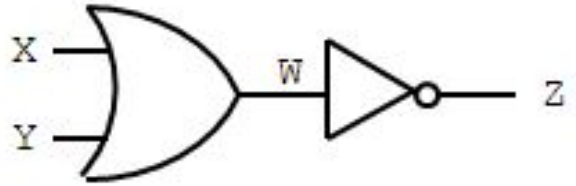
Logic Diagram Symbol

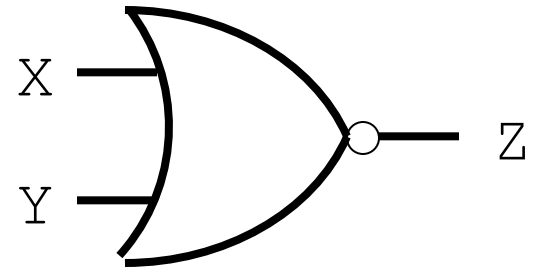


Truth Table

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

NOR Gate

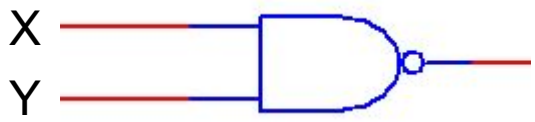




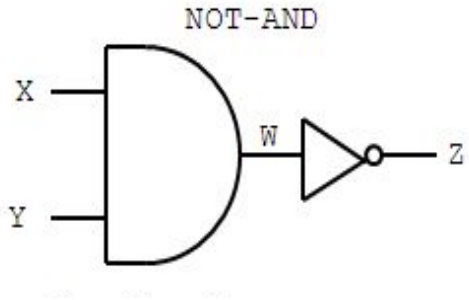
$Z = (X+Y)'$

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

NAND Gate



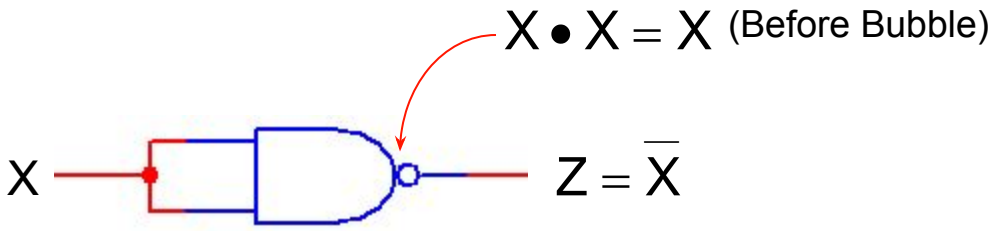
$$Z = \overline{X\,Y} = \overline{X} + \overline{Y}$$



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

Universal Property - NAND & NOR

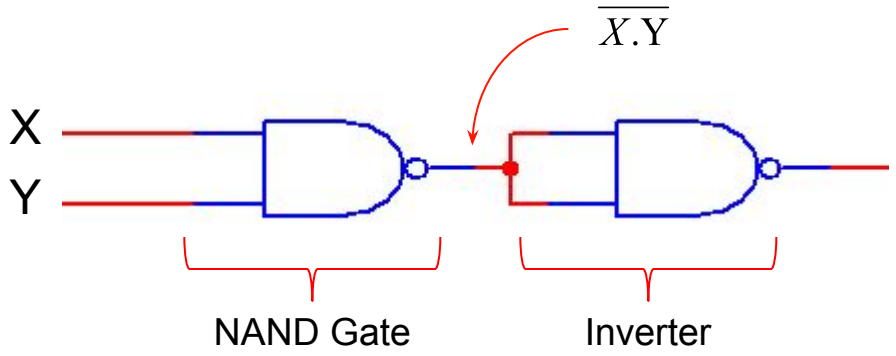
NAND Gate as an Inverter Gate



X	Z
0	1
1	0

Equivalent to Inverter

NAND Gate as an AND Gate

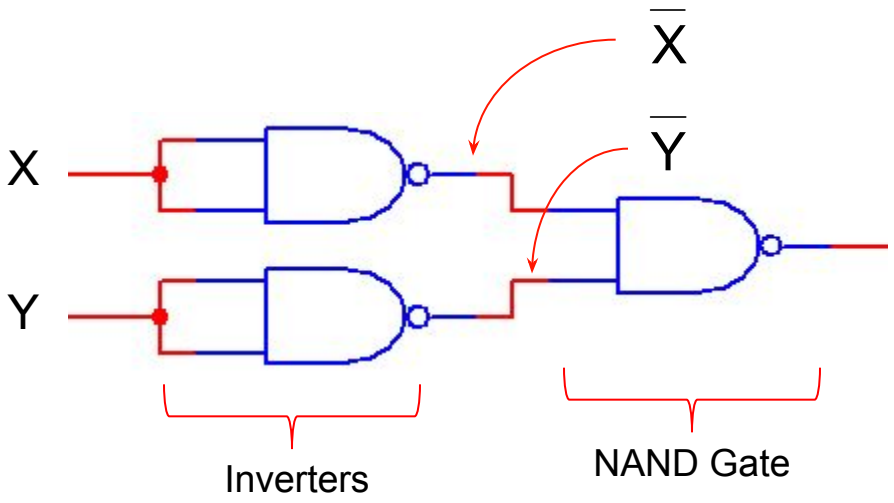


$$Z = \overline{\overline{X.Y}} = X.Y$$

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

Equivalent to AND Gate

NAND Gate as an OR Gate



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

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

Equivalent to OR Gate

Universal Property - NAND & NOR

NOR GATE AS A UNIVERSAL LOGIC ELEMENT

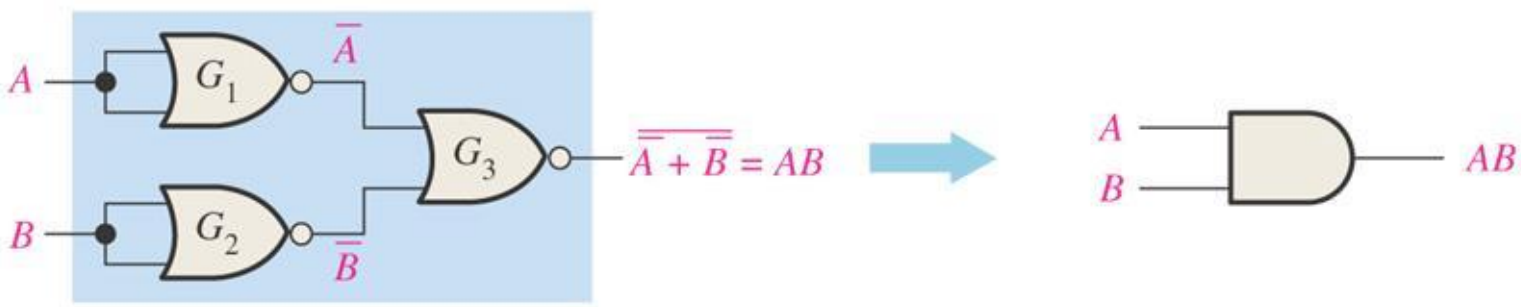
NOR gate is also a universal gate because it can be used to produce the NOT, AND, OR and NAND functions.

(a) One NOR gate used as an inverter

(b) Two NOR gates used as an OR gate

Universal Property - NAND & NOR

NOR GATE AS A UNIVERSAL LOGIC ELEMENT (Cont.)



(c) Three NOR gates used as an AND gate

Introduction to Boolean Algebra

- Developed by English Mathematician *George Boole* in between 1815 - 1864.
- It is described as *an algebra of logic or an algebra of two values i.e True or False.*
- The term *logic* means a statement having binary decisions i.e *True/Yes or False/No*.

Application of Boolean algebra

- It is used to perform the logical operations in digital computer.
- In digital computer True represent by '1' (high volt) and False represent by '0' (low volt)
- Logical operations are performed by logical operators. The fundamental logical operators are:
    1. AND (conjunction)
    2. OR (disjunction)
    3. NOT (negation/complement)

Truth Table

- Truth table is a table that contains all possible values of logical variables/statements in a Boolean expression.

No. of possible combination =

2^n, where n=number of variables used in a Boolean expression.

Truth Table

- The truth table for XY + Z is as follows:

Dec	X	Y	Z	XY	XY+Z
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	0	0
3	0	1	1	0	1
4	1	0	0	0	0
5	1	0	1	0	1
6	1	1	0	1	1
7	1	1	1	1	1

Boolean LAWS

(Existence of 1 and 0 element)

(1) $x + 0 = x$

(2) $x \cdot 0 = 0$

(3) $x + 1 = 1$

(4) $x \cdot 1 = x$

(Existence of complement)

(5) $x + x = x$

(6) $x \cdot x = x$

(7) $x + x' = 1$

(8) $x \cdot x' = 0$

(Commutativity):

(9) x + y = y + x

(10) xy = yx

(Associativity):

(11) x + (y + z) = (x + y) + z

(12) x (yz) = (xy) z

(Distributivity):

(13) $x (y + z) = xy + xz$

(14) $x + yz = (x + y)(x + z)$

(DeMorgan's Theorem)

(15) $(x + y)' = x' y'$

(16) $(xy)' = x' + y'$

(Absorption Law)

1)X+XY=X

LHS: X+XY

=> X(1+Y) Using 1+Y=1

=> X

2) X+X'Y=X+Y

LHS: X+X'Y

=>X+XY+X'Y Using X=X+XY i.e First Absorption Law

=>X+Y(X+X') Using X+X'=1

=>X+Y

(Involution)

(17) $(x')' = x$

Minimization of Boolean Expression

- Minimization of Boolean expression is needed to simplify the Boolean expression and thus reduce the circuitry complexity as it uses less number of gates to produce same output that can by taken by long expression.

Algebraic Method

- The different Boolean rules and theorems are used to simplify the Boolean expression in this method.

Minimization of Boolean Expression (Contd…)

Solved Problem

Minimize the following Boolean Expression:

1. a'bc + ab'c' + ab'c + abc' +abc

   = a'bc + ab'(c' + c) + ab(c' + c) Using X + X' = 1

   = a'bc + ab' + ab

   = a'bc + a(b' + b)

   = a'bc + a

2. AB'CD' + AB'CD + ABCD' + ABCD

   = AB'C(D'+D) + ABC(D'+D) Using X + X' = 1

   = AC(B'+B)

   = AC

A	B	C	D	MINTERM	DESIGNATION	MAXTERM	DESIGNATION
0	0	0	0	A’B’C’D’	m_0	A+B+C+D	M_0
0	0	0	1	A’B’C’D	m_1	A+B+C+D’	M_1
0	0	1	0	A’B’CD’	m_2	A+B+C’+D	M_2
0	0	1	1	A’B’CD	m_3	A+B+C’+D’	M_3
0	1	0	0	A’BC’D’			
0	1	0	1	A’BC’D			
0	1	1	0				
0	1	1	1				
1	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				

1.Prove A’+AB = (A’ +B)

2.(A+ B).(A’+C)= AC +A’B+BC

3.AB +A’C+BC =(A+C).(A’+B)