

Discrete Mathematics 5

Boolean Algebra

Dr. Maged Kassab

Boolean Algebra Simplification

Name	AND form	OR form
Identity law	$1A = A$	$0 + A = A$
Null law	$0A = 0$	$1 + A = 1$
Idempotent law	$AA = A$	$A + A = A$
Inverse law	$A\bar{A} = 0$	$A + \bar{A} = 1$
Commutative law	$AB = BA$	$A + B = B + A$
Associative law	$(AB)C = A(BC)$	$(A + B) + C = A + (B + C)$
Distributive law	$A + BC = (A + B)(A + C)$	$A(B + C) = AB + AC$
Absorption law	$A(A + B) = A$	$A + AB = A$
De Morgan's law	$\overline{AB} = \bar{A} + \bar{B}$	$\overline{A + B} = \bar{A}\bar{B}$

Boolean Algebra Simplification

Boolean Algebra Simplification and how to simplify Boolean algebra expressions using some basic rules applied to their variables, literals and terms

Boolean Algebra Simplification

Name	AND form	OR form
Identity law	$1A = A$	$0 + A = A$
Null law	$0A = 0$	$1 + A = 1$
Idempotent law	$AA = A$	$A + A = A$
Inverse law	$A\bar{A} = 0$	$A + \bar{A} = 1$
Commutative law	$AB = BA$	$A + B = B + A$
Associative law	$(AB)C = A(BC)$	$(A + B) + C = A + (B + C)$
Distributive law	$A + BC = (A + B)(A + C)$	$A(B + C) = AB + AC$
Absorption law	$A(A + B) = A$	$A + AB = A$
De Morgan's law	$\overline{AB} = \bar{A} + \bar{B}$	$\overline{A + B} = \bar{A}\bar{B}$

Laws of Boolean Algebra

A set of rules or Laws of Boolean Algebra expressions have been invented to help reduce the number of logic gates needed to perform a particular logic operation resulting in a list of functions or theorems known commonly as the **Laws of Boolean Algebra**.

As well as the logic symbols “0” and “1” being used to represent a digital input or output, we can also use them as constants for a permanently “Open” or “Closed” circuit or contact respectively.

Laws of Boolean Algebra

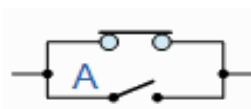
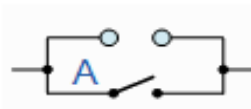
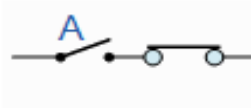
Boolean Algebra is the mathematics we use to analyse digital gates and circuits. We can use these “Laws of Boolean” to both reduce and simplify a complex Boolean expression in an attempt to reduce the number of logic gates required. *Boolean Algebra* is therefore a system of mathematics based on logic that has its own set of rules or laws which are used to define and reduce Boolean expressions.

The variables used in **Boolean Algebra** only have one of two possible values, a logic “0” and a logic “1” but an expression can have an infinite number of variables all labelled individually to represent inputs to the expression, For example, variables A, B, C etc, giving us a logical expression of $A + B = C$, but each variable can ONLY be a 0 or a 1.

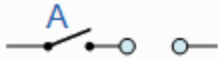
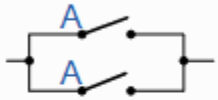
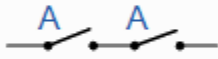
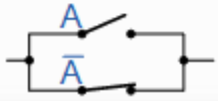
Laws of Boolean Algebra

Examples of these individual laws of Boolean, rules and theorems for Boolean Algebra are given in the following table.

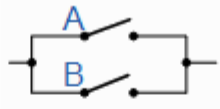
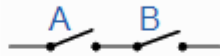
Truth Tables

Boolean Expression	Description	Equivalent Switching Circuit	Boolean Algebra Law or Rule
$A + 1 = 1$	A in parallel with closed = "CLOSED"		Annulment
$A + 0 = A$	A in parallel with open = "A"		Identity
$A \cdot 1 = A$	A in series with closed = "A"		Identity

Laws of Boolean Algebra

Boolean Expression	Description	Equivalent Switching Circuit	Boolean Algebra Law or Rule
$A \cdot 0 = 0$	A in series with open = "OPEN"		Annulment
$A + A = A$	A in parallel with A = "A"		Idempotent
$A \cdot A = A$	A in series with A = "A"		Idempotent
$\text{NOT } \overline{\overline{A}} = A$	NOT NOT A (double negative) = "A"		Double Negation
$A + \overline{A} = 1$	A in parallel with NOT A = "CLOSED"		Complement

Laws of Boolean Algebra

Boolean Expression	Description	Equivalent Switching Circuit	Boolean Algebra Law or Rule
$A+B = B+A$	A in parallel with B = B in parallel with A		Commutative
$A.B = B.A$	A in series with B = B in series with A		Commutative
$\overline{A+B} = \overline{A}. \overline{B}$	invert and replace OR with AND		de Morgan's Theorem
$\overline{A.B} = \overline{A} + \overline{B}$	invert and replace AND with OR		de Morgan's Theorem

Boolean Algebra Functions

Function	Description	Expression
1.	NULL	0
2.	IDENTITY	1
3.	Input A	A
4.	Input B	B
5.	NOT A	\overline{A}
6.	NOT B	\overline{B}

Boolean Algebra Functions

Function	Description	Expression
7.	A AND B (AND)	$A \cdot B$
8.	A AND NOT B	$A \cdot \overline{B}$
9.	NOT A AND B	$\overline{A} \cdot B$
10.	NOT AND (NAND)	$\overline{A \cdot B}$
11.	A OR B (OR)	$A + B$
12.	A OR NOT B	$A + \overline{B}$

Boolean Algebra Functions

Function	Description	Expression
13.	NOT A OR B	$\bar{A} + B$
14.	NOT OR (NOR)	$\overline{A + B}$
15.	Exclusive-OR	$A \cdot \bar{B} + \bar{A} \cdot B$
16.	Exclusive-NOR	$A \cdot B + \bar{A} \cdot \bar{B}$

Boolean Algebra Functions

Laws of Boolean Algebra Example No1

Using the above laws, simplify the following expression: $(A + B)(A + C)$

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

$$A.A + A.C + A.B + B.C \quad - \text{Distributive law}$$

$$A + A.C + A.B + B.C \quad - \text{Idempotent AND law (A.A = A)}$$

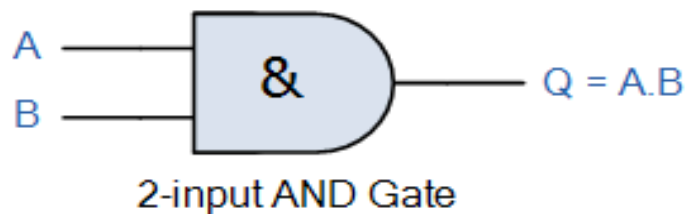
$$A(1 + C) + A.B + B.C \quad - \text{Distributive law}$$

$$A.1 + A.B + B.C \quad - \text{Identity OR law (1 + C = 1)}$$

$$A(1 + B) + B.C \quad - \text{Distributive law}$$

$$A.1 + B.C \quad - \text{Identity OR law (1 + B = 1)}$$

$$Q = A + (B.C) \quad - \text{Identity AND law (A.1 = A)}$$

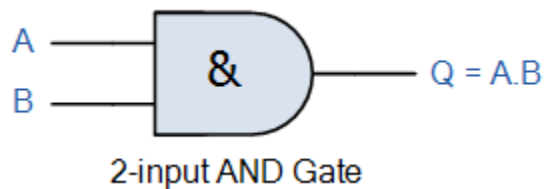
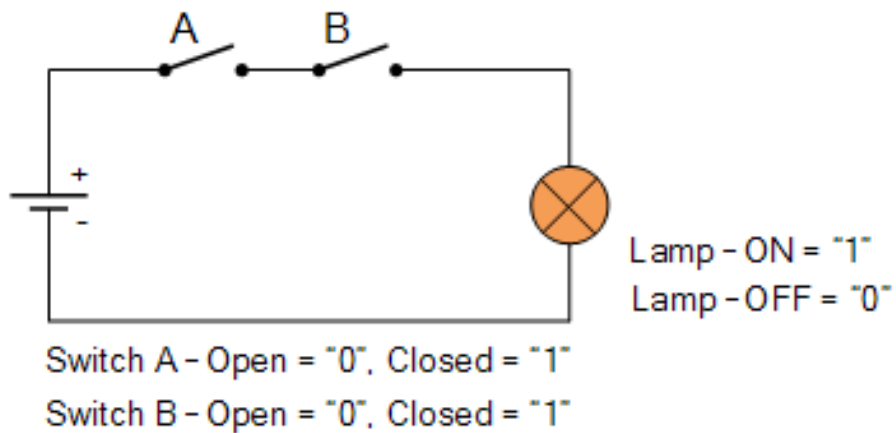


Logic AND Function

The Logic AND Function output is only true when all of its inputs are true, otherwise the output is false

Logic Gates

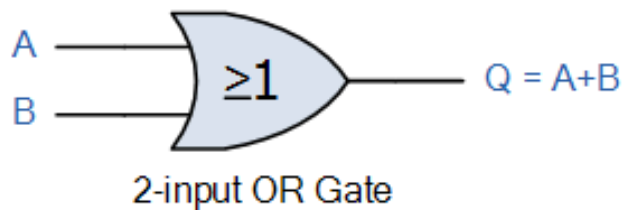
Switch Representation of the AND Function



Logic Gates

AND Function Truth Table

Switch A	Switch B	Output	Description
0	0	0	A and B are both open, lamp OFF
0	1	0	A is open and B is closed, lamp OFF
1	0	0	A is closed and B is open, lamp OFF
1	1	1	A is closed and B is closed, lamp ON
Boolean Expression (A AND B)			$A \cdot B$

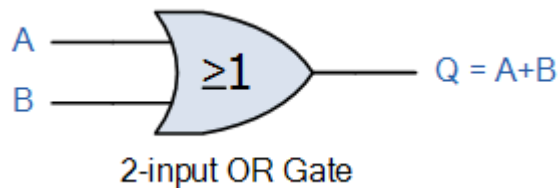
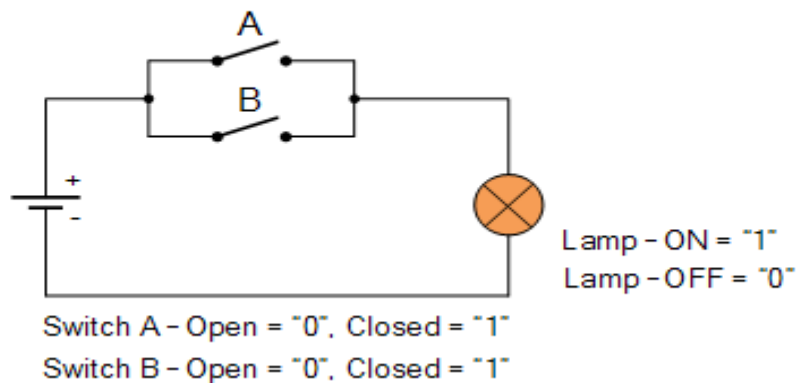


Logic OR Function

The Logic OR function output is only true if one or more of its inputs are true, otherwise the output is false

Logic Gates

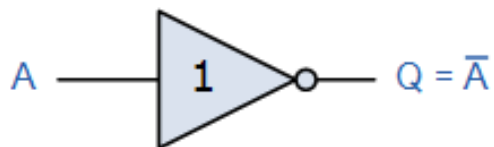
Switch Representation of the OR Function



Logic Gates

Logic OR Function Truth Table

Switch A	Switch B	Output	Description
0	0	0	A and B are both open, lamp OFF
0	1	1	A is open and B is closed, lamp ON
1	0	1	A is closed and B is open, lamp ON
1	1	1	A is closed and B is closed, lamp ON
Boolean Expression (A OR B)			$A + B$



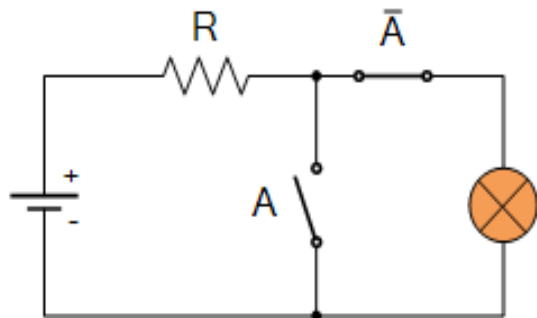
Inverter or NOT Gate

Logic NOT Function

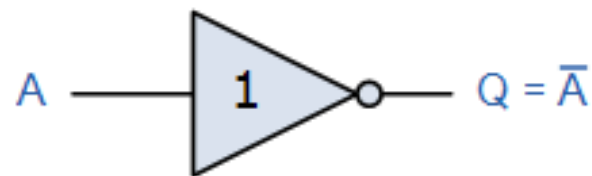
The Logic NOT Function output is true when its single input is false, and false when its single input is true

Logic Gates

Switch Representation of the Logic NOT Function



Switch A - Open = "0", Lamp - 0
Switch A - Closed = "1", Lamp - 0



Inverter or NOT Gate

Logic Gates

NOT Function Truth Table

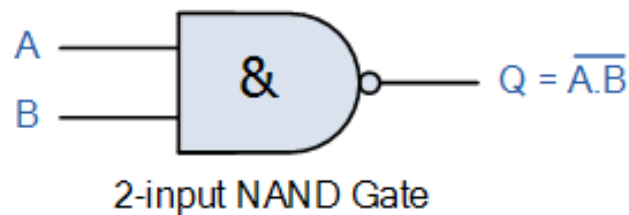
Switch	Output
1	0
0	1
Boolean Expression	not-A or \bar{A}

Logic Gates

Logic NOT gates or “Inverters” as they are more commonly called, can be connected with standard AND and OR gates to produce NAND and NOR gates respectively. Inverters can also be used to produce “Complementary” signals in more complex decoder/logic circuits for example, the complement of logic A is \bar{A} and two Inverters connected together in series will give a double inversion which produces at its output the original value of A.

Logic NOT Function Equivalents



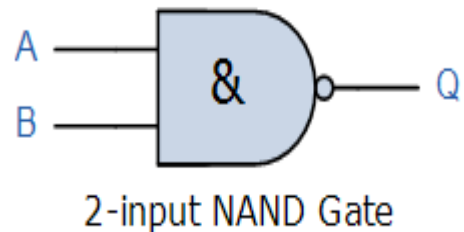
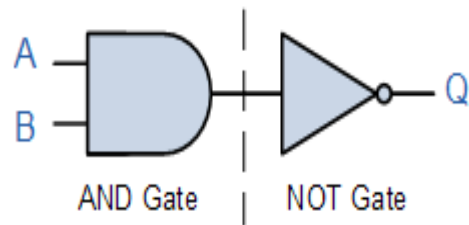
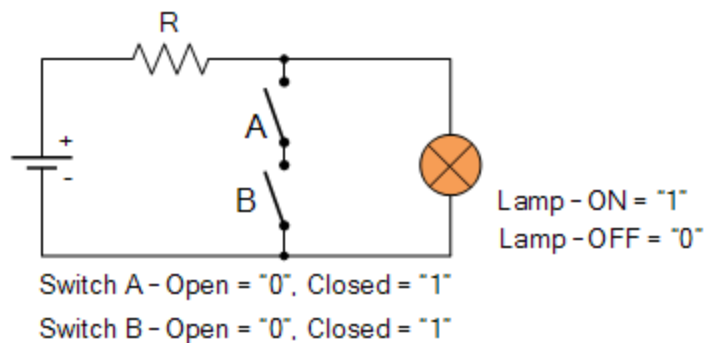


Logic NAND Function

The Logic NAND Function output is only false when all of its inputs are true, otherwise the output is always true

Logic Gates

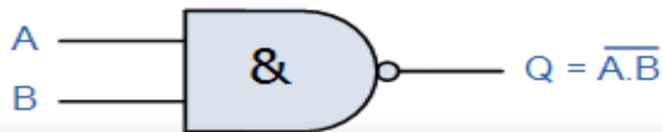
Switch Representation

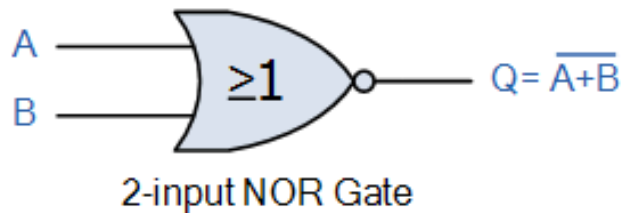


Logic Gates

NAND Function Truth Table

Switch A	Switch B	Output	Description
0	0	1	A and B are both open, lamp ON
0	1	1	A is open and B is closed, lamp ON
1	0	1	A is closed and B is open, lamp ON
1	1	0	A is closed and B is closed, lamp OFF
Boolean Expression ($\overline{A \text{ AND } B}$)			$\overline{A \cdot B}$



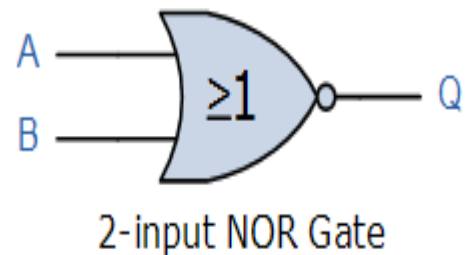
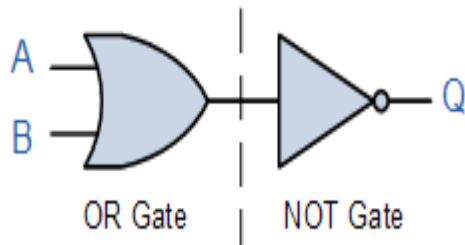
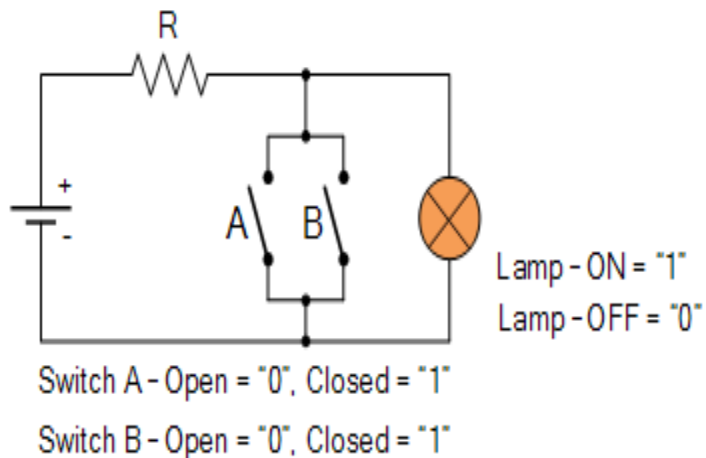


Logic NOR Function

The Logic NOR Function output is only true when all of its inputs are false, otherwise the output is always false

Logic Gates

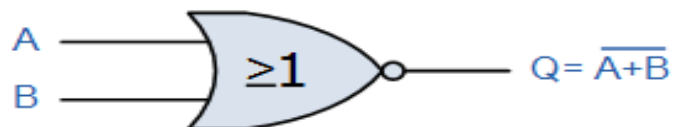
Switch Representation



Logic Gates

Logic NOR Function Truth Table

Switch A	Switch B	Output	Description
0	0	1	Both A and B are open, lamp ON
0	1	0	A is open and B is closed, lamp OFF
1	0	0	A is closed and B is open, lamp OFF
1	1	0	A is closed and B is closed, lamp OFF
Boolean Expression ($\overline{A \text{ OR } B}$)			$\overline{A + B}$

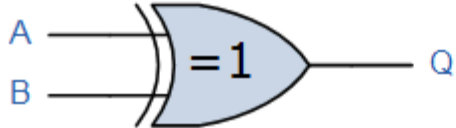


2-input NOR Gate

Logic Gates

2-input EX-OR (Exclusive OR) Gate

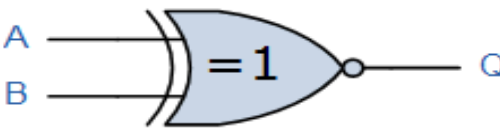
For a 2-input Ex-OR gate, the output Q is true if EITHER input A or if input B is true, but NOT both giving the Boolean Expression of: ($Q = (A \text{ and NOT } B) \text{ or } (\text{NOT } A \text{ and } B)$).

Symbol	Truth Table		
 2-input Ex-OR Gate	A	B	Q
	0	0	0
	0	1	1
	1	0	1
	1	1	0
Boolean Expression $Q = A \oplus B$			

Logic Gates

2-input EX-NOR (Exclusive NOR) Gate

For a 2-input Ex-NOR gate, the output Q is true if BOTH input A and input B are the same either true or false, giving the Boolean Expression of: ($Q = (A \text{ and } B) \text{ or } (\text{NOT } A \text{ and } \text{NOT } B)$).

Symbol	Truth Table		
 2-input Ex-NOR Gate	A	B	Q
	0	0	1
	0	1	0
	1	0	0
	1	1	1
Boolean Expression $Q = \overline{A \oplus B}$			

Logic Gates

Summary of 2-input Logic Gates

The following Boolean ALgebra Truth Tables compare the logical functions of the 2-input logic gates above.

Inputs		Truth Table Outputs For Each Gate					
A	B	AND	NAND	OR	NOR	EX-OR	EX-NOR
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1

Logic Gates

The following Boolean Algebra truth tables gives a list of the common logic functions and their equivalent Boolean notation.

Logic Function	Boolean Notation
AND	$A.B$
OR	$A+B$
NOT	\overline{A}
NAND	$\overline{A.B}$
NOR	$\overline{A+B}$
EX-OR	$(A.\overline{B}) + (\overline{A}.B) \text{ or } A \oplus B$
EX-NOR	$(A.B) + (\overline{A}.\overline{B}) \text{ or } \overline{A \oplus B}$



شكراً لحسن المتابعة