

3. Gates and Boolean Algebra

Objectives

1. Analyze the INVERTER circuit.
2. Describe the operation of and construct the truth tables for the AND, NAND, OR, and NOR gates.
3. Draw timing diagrams for the various logic-circuit gates.
4. Simplify complex logic circuits by applying various Boolean algebra laws and rules.
5. Simplify intricate Boolean equations by applying DeMorgan's theorems.
6. Use either of the universal gates (NAND or NOR) to implement the circuit represented by a Boolean expression.
7. Explain the advantages of constructing a logic circuit diagram using the alternate gate symbols, versus the standard logic-gate symbols.
8. Describe the concept of active-Low and active-High signals.

Introduction

All Digital (logic) circuits operate in the binary mode, i.e.

Digital circuits operate only in 2 states which are called the: High (abbreviated H) state and Low (L) state.

What this means is that *input* & *output* voltages are within one of two voltage ranges which are denoted as Logic H, also referred to as Binary 1, and Logic L or Binary 0

This characteristic of digital circuits allows us to use Boolean algebra as a tool to analyze and design logic circuits.

The topics which follow introduces the basic logic gates which are the building blocks of Digital Electronics and the theory of Boolean Algebra, which is used in the design and analysis of digital circuits.

Boolean Constants and Variables

In Boolean algebra, constants and variables can **ONLY** have two values:

Logic 1 or Logic 0

Boolean 1 or 0 do not actually represent numbers.

They are synonymous with:

True	or	False
On	or	Off
Yes	or	No
High	or	Low

With High - Low being the most appropriate in describing the behaviour of digital circuits

In Digital circuits, Boolean 1 and 0 thus represent voltage levels, which are also referred to as the **LOGIC LEVELS**.

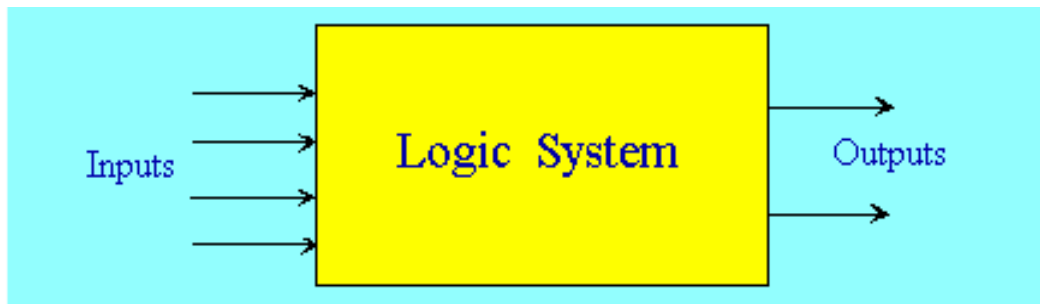
Boolean algebra can be used to express the effects of digital circuits on logic inputs.

Electronic circuits that operate on binary data are called:

Digital Logic Systems

There are in general, 2 types of Digital Logic Systems:

- (i) Combinational Logic systems *
- (ii) Sequential Logic systems *



Circuits within the system that carry out elementary logic operations are called ***gates***.

NB:

- *Combinational circuits: the output(s) at any instant is a direct response to the input combinations at that instant. There is no feedback between the output(s) and input(s).*
- *Sequential circuits: the output(s) at any instant depends not only on the current values (states) of the inputs but also their previous values, i.e. there is an element of memory.*

Truth-Tables

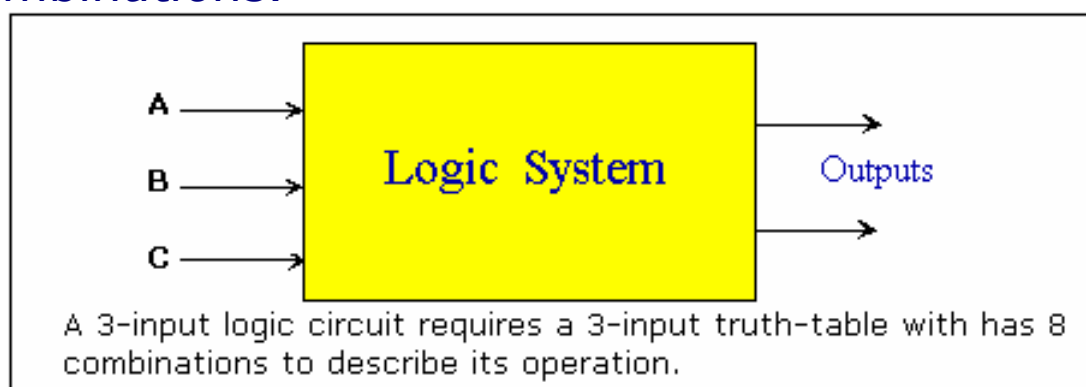
A truth table is a means to show how the outputs of a logic system respond to a given input combination of logic levels.

It lists all the possible input combinations of the logic levels present and the corresponding output logic levels.

Truth tables are typically used to describe the behaviour of Combinational Logic circuits.

A truth table used to describe a N-input circuit will have a 2^N different input combinations.

In a 3-input truth table there are 8 different input combinations.



In a 4-input truth table there are 16 different input combinations.

Question

How many possible input combinations will there be in a truth-table that is used to describe a combinational logic circuit with 6 inputs and 2 outputs?

(a)			(b)				(c)					
A	B	x	A	B	C	x	A	B	C	D	x	
0	0	?	0	0	0	?	0	0	0	0	?	Truth Tables for (a) two-input, (b) three-input, (c) four-inputs.
0	1	?	0	0	1	?	0	0	0	1	?	
1	0	?	0	1	0	?	0	0	1	0	?	
1	1	?	0	1	1	?	0	0	1	1	?	
			1	0	0	?	0	1	0	0	?	&
			1	0	1	?	0	1	0	1	?	
			1	1	0	?	0	1	1	0	?	
			1	1	1	?	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	?	

Answer : There will be _____ input combinations.

Logic Gates

A logic gate is a two state device that relates the outputs of a binary system to its input.

The fundamental logic gates are:

AND, **OR** and the **NOT**.

Any Digital circuit can be synthesized by using a combination of these basic gates.

OR is also called *Logical Addition* or *OR addition*
Equation symbol used is a: **+** (plus sign)

AND is also known as *Logical Multiplication* or *AND multiplication*.

Equation symbol for AND is: **.** (multiplication sign)

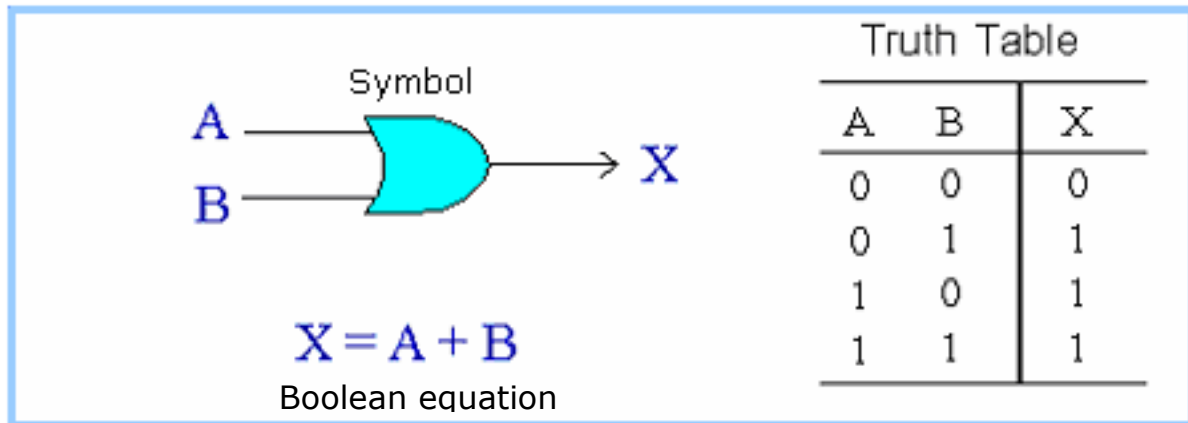
NOT is also called *Logical Complementation* or *inversion*.

Equation symbol for NOT is: **—** (bar over the variable names)

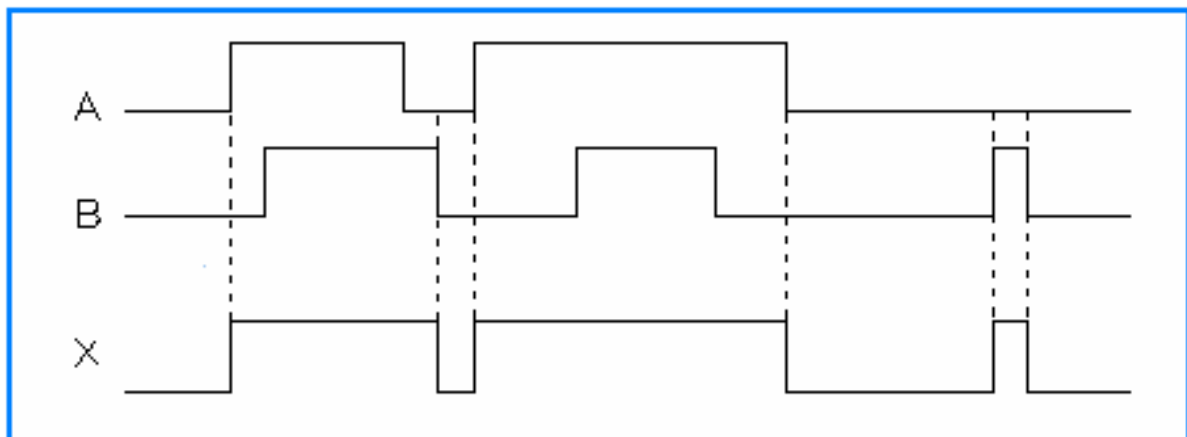
These three logic gates are the basic 'building blocks' of both *Combinational Logic* and *Sequential Logic* circuits.

OR Gate

- An OR gate has a minimum of 2 inputs.
- It has only 1 output.

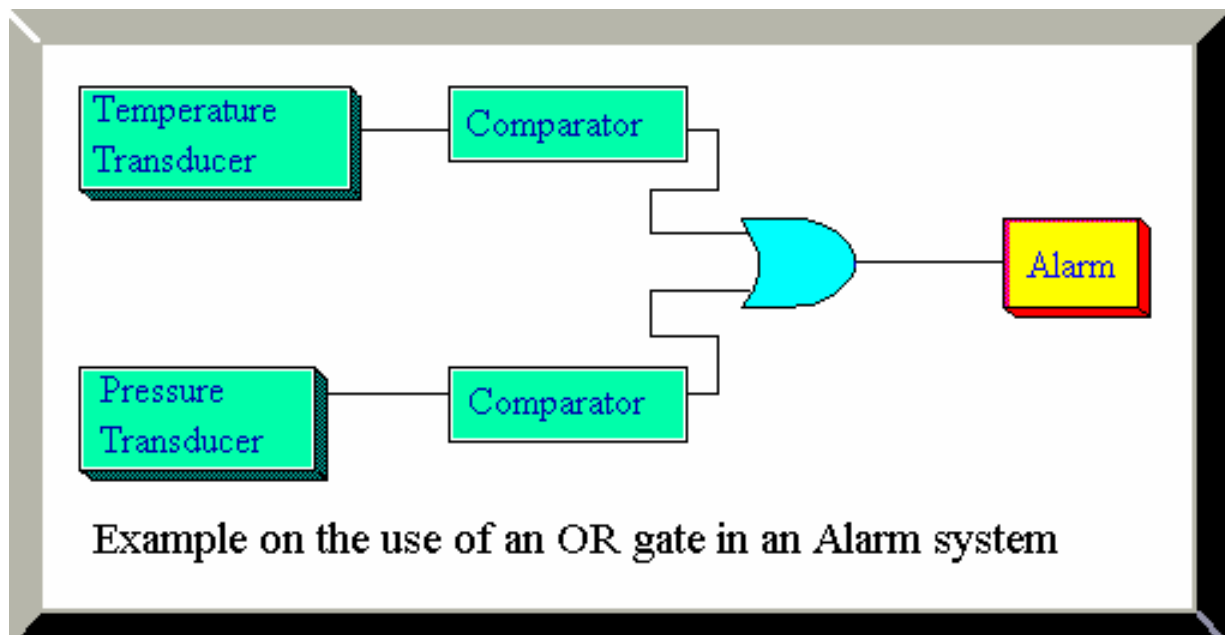


- The OR gate output is logic H or 1 whenever any of its inputs is at logic H or 1.
- The output is only L when all its inputs are L.



Example of waveforms of input signals & output response of a 2 input OR gate.

An Application example of the OR gate



Process of monitoring temperature & Pressure

*The alarm is to be activated whenever the Temperature **OR** Pressure exceeds a certain limit.*

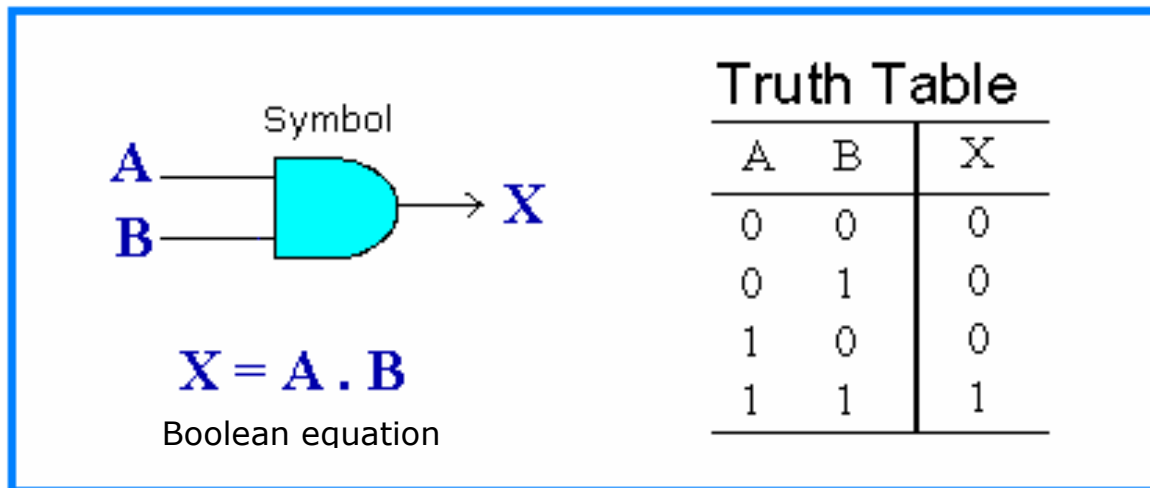
The temperature or pressure transducer produces an electrical signal which is proportional to the respective quantity monitored.

The amplitude of the signals are compared with a reference value at the comparators. Whenever the reference is exceeded, a logic High or 1 will be produced.

The Alarm will thus be triggered if pressure OR temperature OR both exceeds the reference value.

AND Gate

- An AND gate has a minimum of 2 inputs.
- It has only 1 output.



- The AND gate output is at logic H or 1 only when all its inputs are at logic H or 1.
- The AND output is Low or 0 when any of its inputs is Low.

Question

For a 3-input AND gate, what will be its outputs for the following input values?

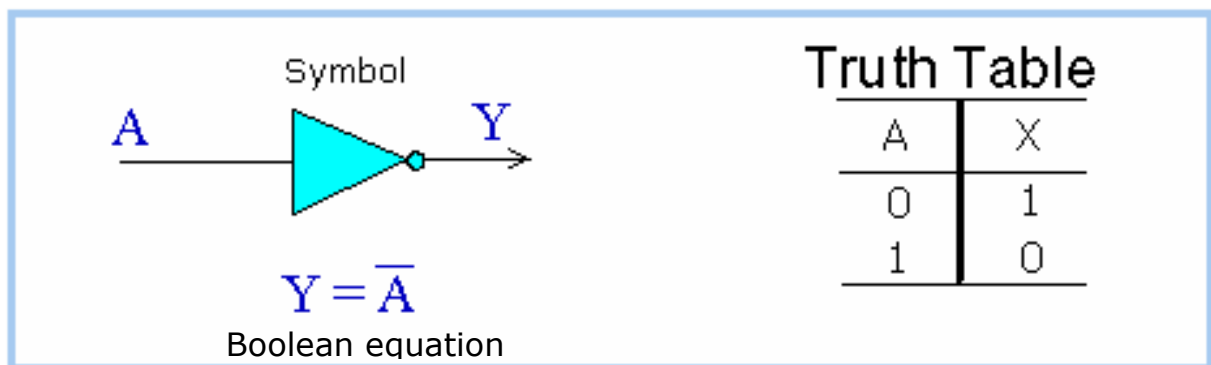
Inputs -> 0 0 1, output = _____

Inputs -> 0 1 1, output = _____

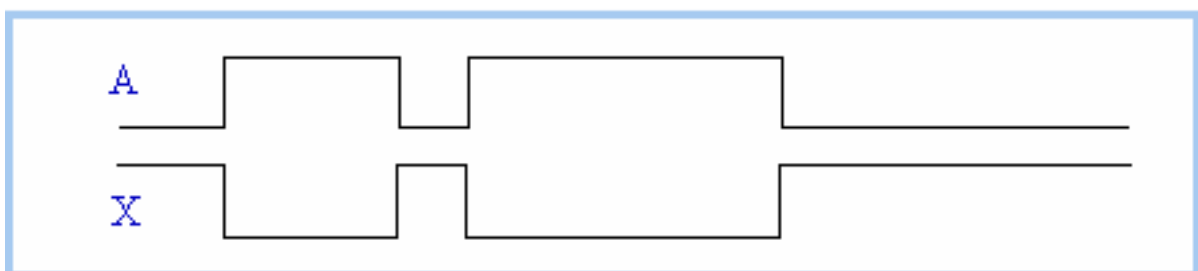
& for inputs 1 1 1, output = _____

NOT Gate

- Is the simplest possible logic gate.
- Is also known as the inverter
- Has only one input and one output.



- The NOT gate inverts or complements the input logic level.
- A logic H or 1 applied at the input becomes a L or 0 and conversely, a L applied becomes a H.



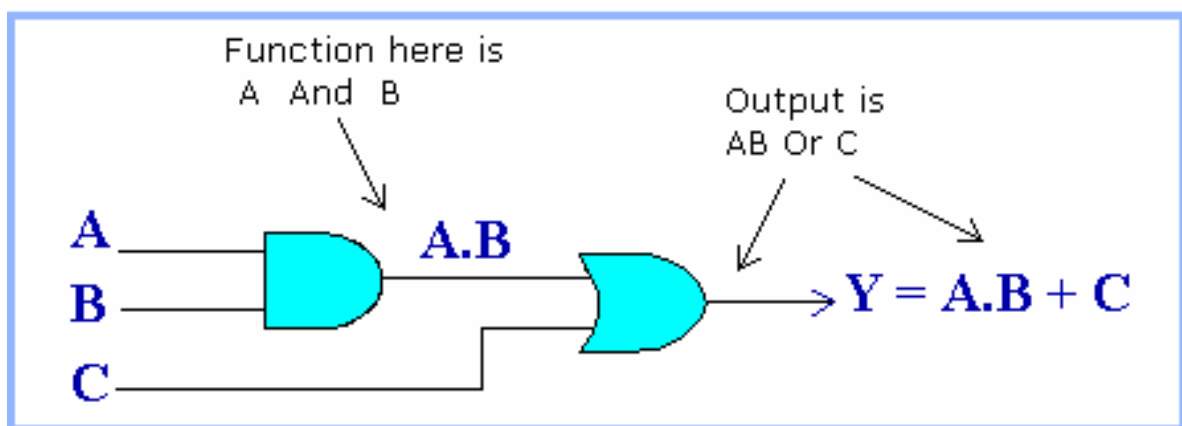
Example of input waveform and output response from an OR gate.

Describing Logic Circuits Algebraically

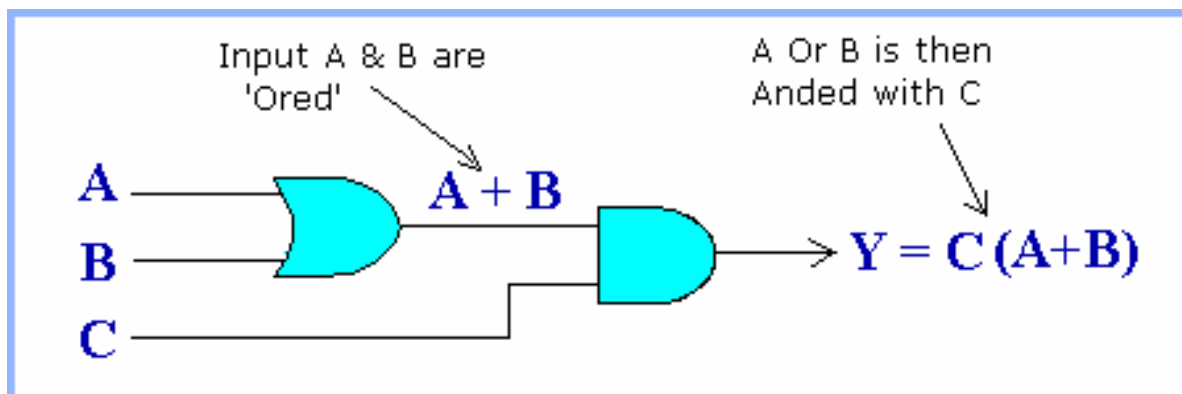
Any logic circuit can be described using the boolean operations of OR, AND, and NOT gates.

Two Examples:

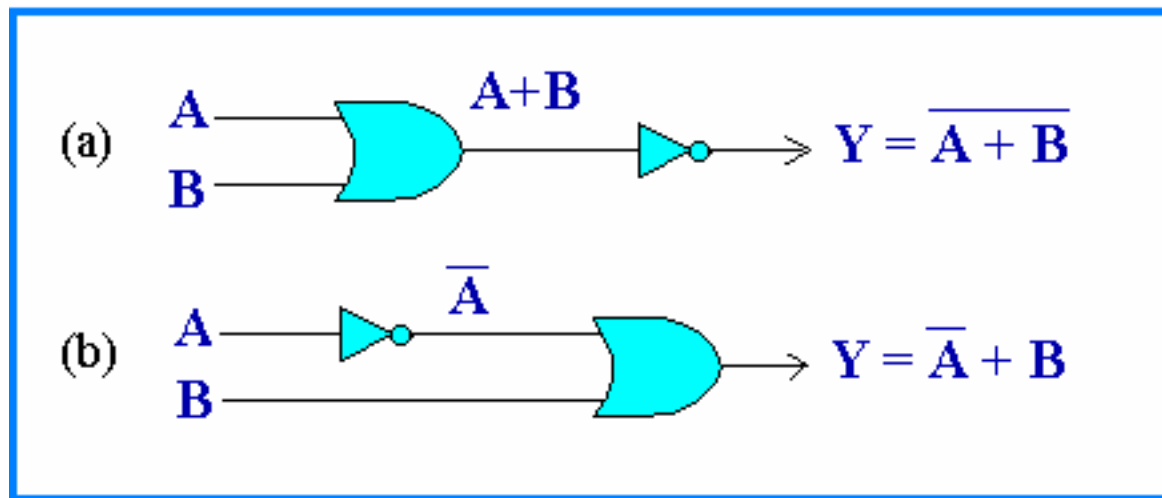
An AND gate followed by an OR gate



An OR gate followed by an AND gate



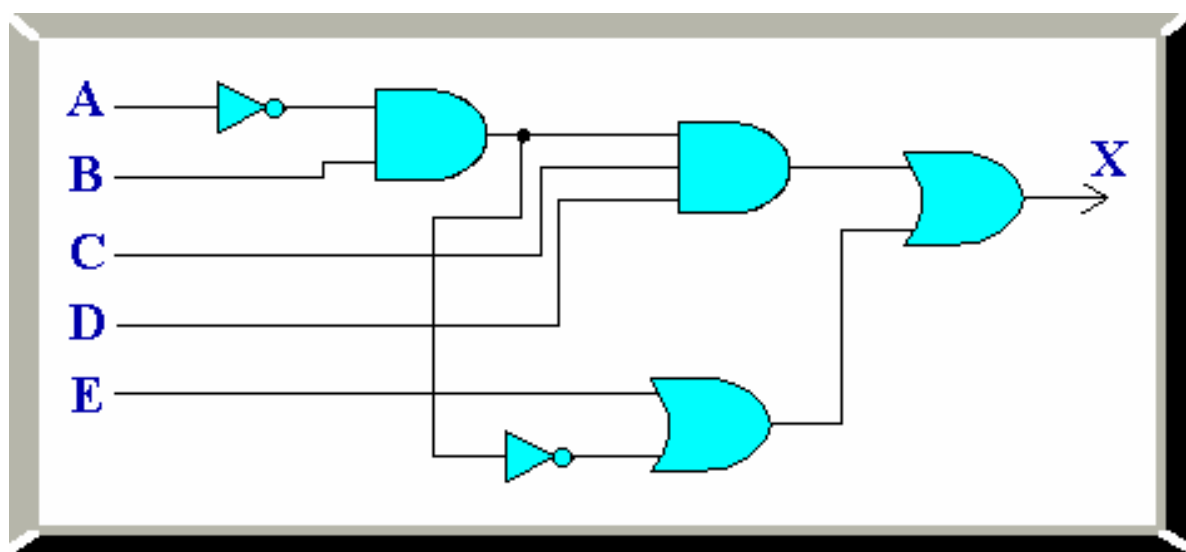
Circuits containing INVERTERS



In (a) above, inputs A and B are ORed together with *output* INVERTED

In (b), inputs A and B are ORed together with A *input* INVERTED.

Try this example: What is output Boolean expression?



The Output Boolean expression of the circuit shown previously is:

$$X = \bar{A}BCD + (\overline{\bar{A}B}) + E$$

The Boolean Expression is obtained as follows:

"Working from input to output, each logic term (of the final expression) is obtained progressively at the output of each gate."

"For example, at the output of the first inverter, NOT A is obtained and this variable is then combined with input B at the 2-input AND gate to yield NOT A AND B."

"Progressing in this manner, the term at the output of the 4-input AND gate is derived by ANDing NOT A AND B with inputs C AND D. This intermediate product which forms one part of the final expression, is applied to the upper input of the output OR gate."

"Similarly, the other logic term which is applied at the lower input of the final OR gate is likewise derived. These 2 logic terms are then ORed together to yield the Final Boolean expression."

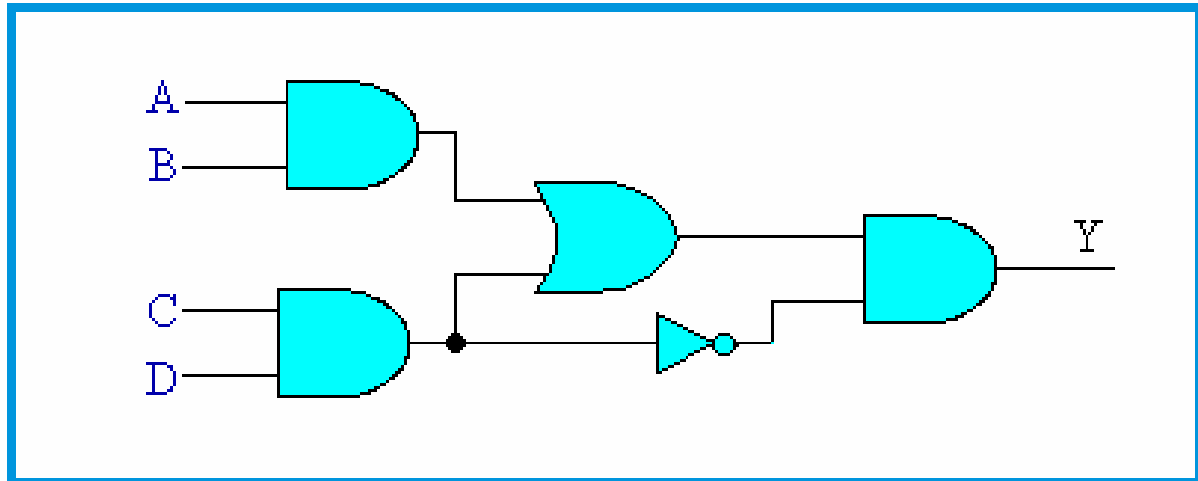
Evaluating Logic - Circuit Outputs

Given a logic circuit, the steps for evaluating or determining the output logic state given a set of input combinations, are:

	Methodology
1	Obtain the Boolean expression.
2	Substitute each variable with the applied logic state.
3	Perform all inversions of single terms.
4	Perform operations within parentheses.
5	Perform AND operation before OR operation unless parentheses indicate otherwise.
6	If an expression has a bar over it, perform the operation of the expression first and then invert the result.

Example

For the circuit shown below, evaluate the output for $A = 1$, $B = 1$, $C = 0$ & $D = 1$.



1. First determine the Boolean expression which is:

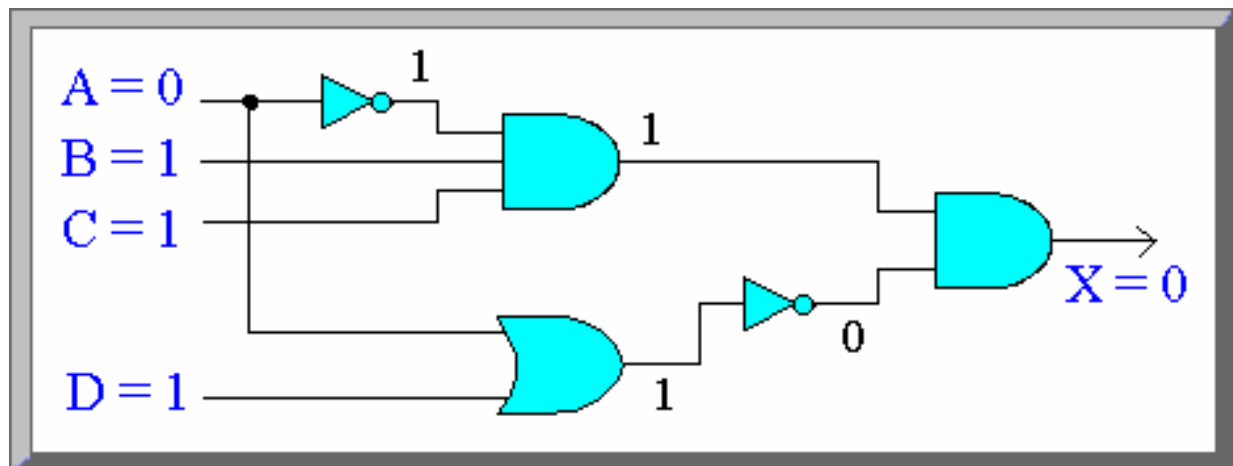
$$Y = (A B + C D) \overline{(C D)}$$

2. Substituting the variables with the input logic levels, yields:

$$Y = (1.1 + 0.1) \overline{(0.1)}$$

which is = 1

Determining Output level from a Circuit Diagram can be done without using the Boolean expression as shown below:



In this case, the input logic levels are propagated from the inputs to the output on a gate by gate basis, i.e.

the output at each intermediate gate is evaluated before it is considered as an input to the following gate.

Question

If input D is changed to logic 0 will the output X remains at the same logic 0 level?

Answer: Yes or No?

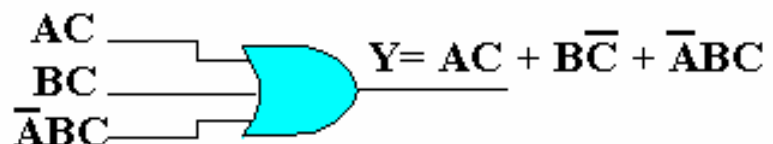
Implementing Circuits from Boolean Expressions

To implement means to draw the circuit from the given expression.

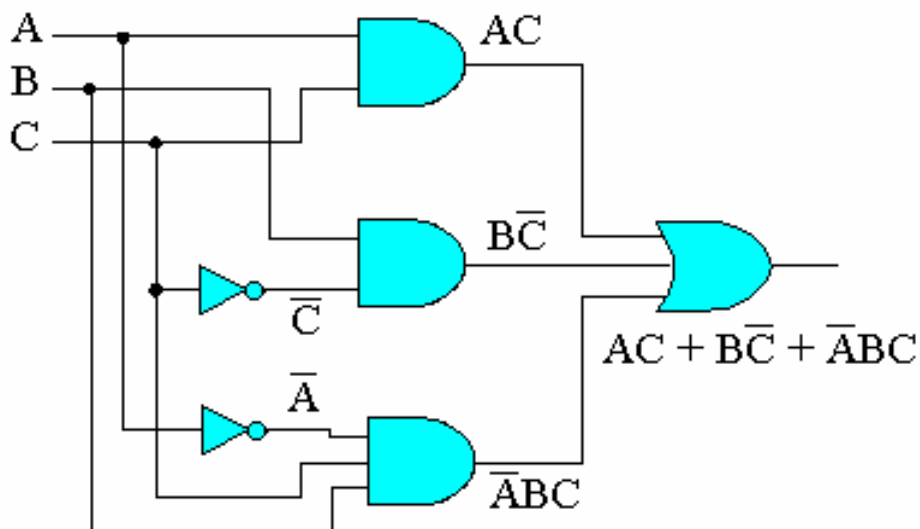
E.g. Implement the Boolean expression:

$$Y = AC + B\bar{C} + \bar{A}BC$$

- One approach is to draw the circuit starting from the output (final) stage:

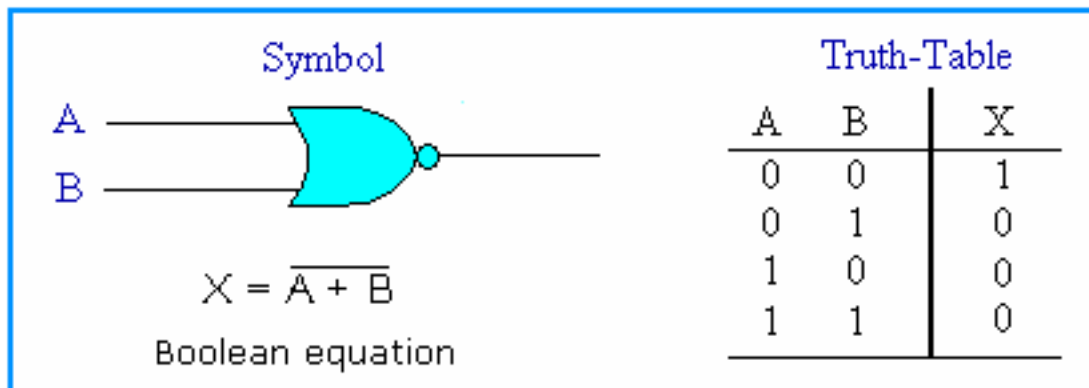


- Working backwards, the inputs are then derived.

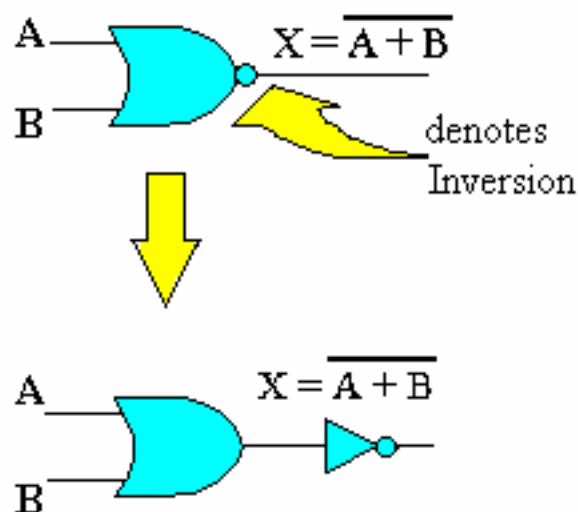


NOR: Universal gate.

- Abbreviated from NoT-OR.
- Has one output & a minimum of 2 inputs.



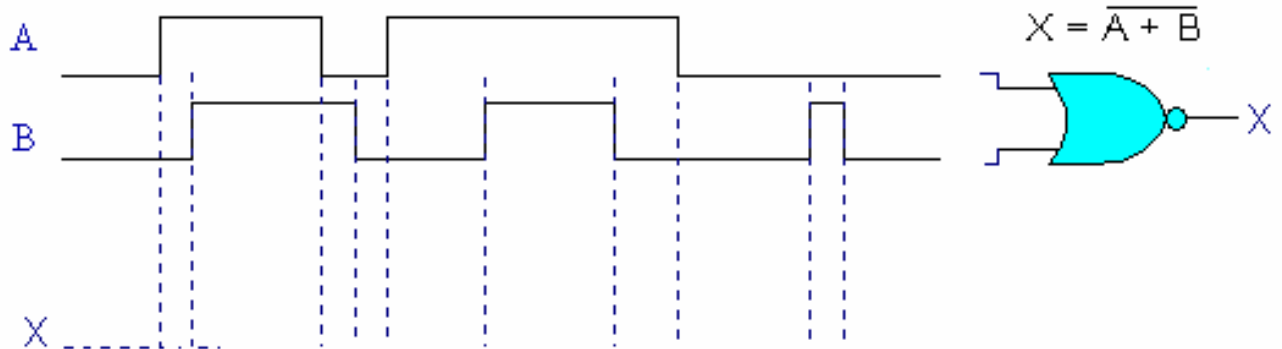
- Is an OR gate connected to a NOT gate.



- The NOR gate output is H (or 1) only when all its inputs is L (or 0).
- The NOR gate output is L when any of its inputs is H.

Questions:

1. Given: what is the output waveform X?

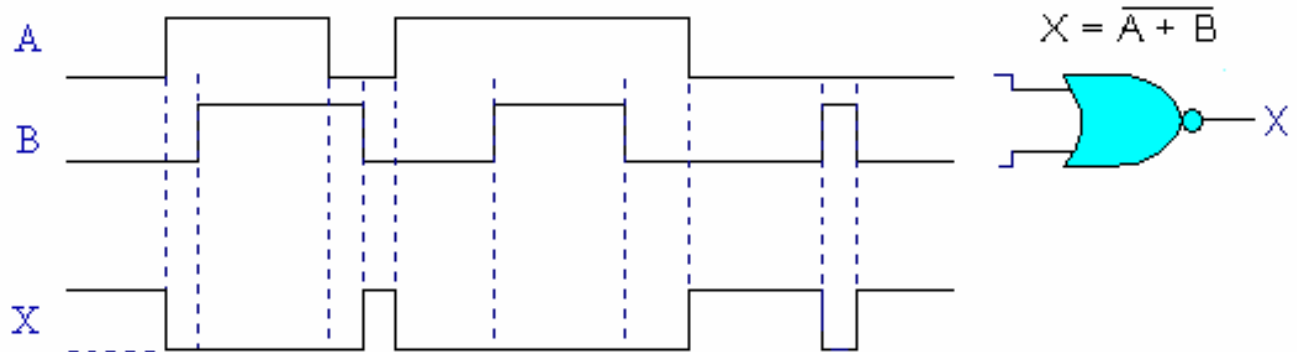


2. Complete the truth-table for a 3-input NOR gate?

Inputs			Output
A	B	C	X
L	L	L	
L	L	H	
L	H	L	
L	H	H	
H	L	L	
H	L	H	
H	H	L	
H	H	H	

Answers to Previous Questions

1.

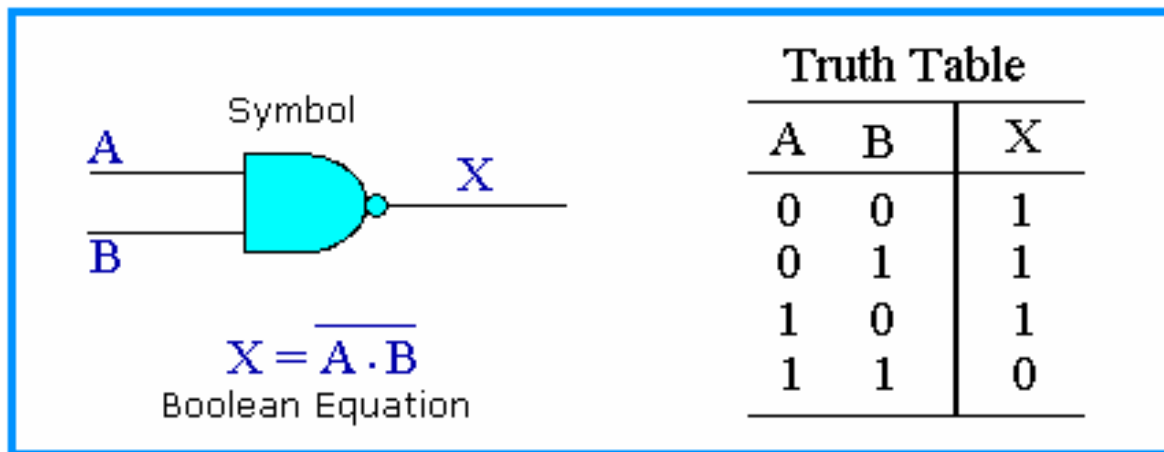


2. Truth-table for 3-input NOR

Inputs			Output
A	B	C	X
L	L	L	H
L	L	H	L
L	H	L	L
L	H	H	L
H	L	L	L
H	L	H	L
H	H	L	L
H	H	H	L

NAND: Universal gate.

- Abbreviated from NOT-AND.
- Has only one output & a minimum of 2 inputs.



- Is derived from an AND gate in connected cascade with a NOT gate.
- The NAND output is only L (or 0) when all its inputs are H (or 1).

Question:

For a 3-input NAND gate, what is the output logic level, if all the inputs are at logic High?

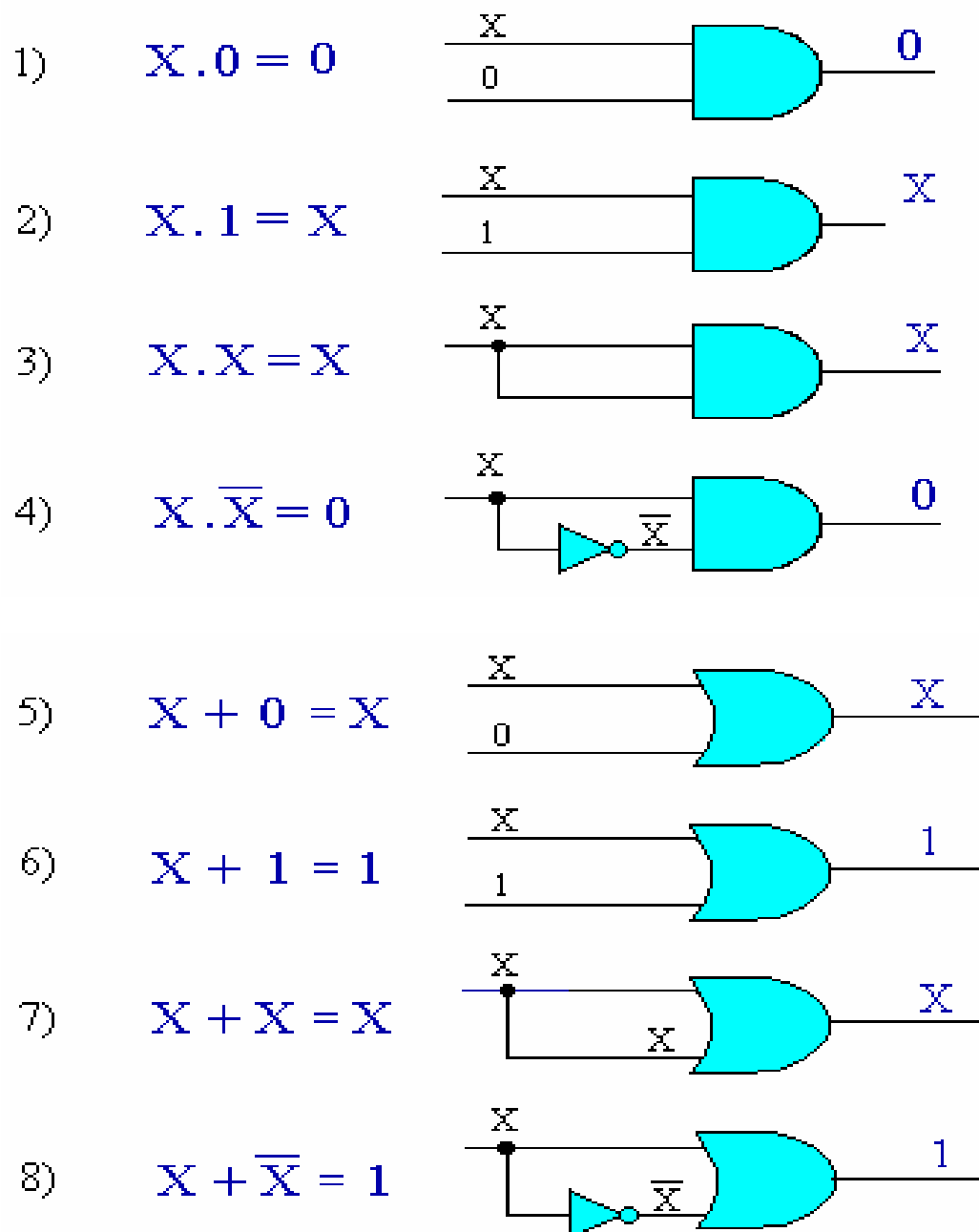
Answer:

The output will be L (or 0)

Boolean Theorems

- Two groups:
Single variable and multi-variables theorems.

Single variable theorems:



Multivariable Theorems:

- These theorems involve more than one variable.

	Theorems	
9	$x + y = y + x$	theorems 9 & 10 are the commutative laws.
10	$x \cdot y = y \cdot x$	
11	$x + (y + z) = (x + y) + z = x+y+z$	theorems 11 & 12 are the associative laws.
12	$x(yz) = (xy)z = xyz$	
13a	$x(y + z) = xy + xz$	theorem 13 is the distributive law.
13b	$(w + x)(y + z) = wy+xy+wz+xz$	
14	$x + xy = x$	
15a	$x + \overline{x}y = x + y$	
15b	$\overline{x} + xy = \overline{x} + y$	

- Theorems 9 to 15 are similar to those found in ordinary algebra.
- Theorems 14 and 15 have no equivalence in ordinary algebra.

Example: Simply the following Boolean equations.

$$(1) \quad y = A\bar{B}D + A\bar{B}\bar{D}$$

$$y = A\bar{B}(D + \bar{D})$$

$$y = A\bar{B} \cdot 1$$

$$y = A\bar{B}$$

$$(2) \quad x = \bar{A}CD + ABCD$$

$$x = CD(\bar{A} + AB) \quad \text{- theorem 15}$$

$$x = CD(\bar{A} + B)$$

$$x = \bar{A}CD + BCD$$

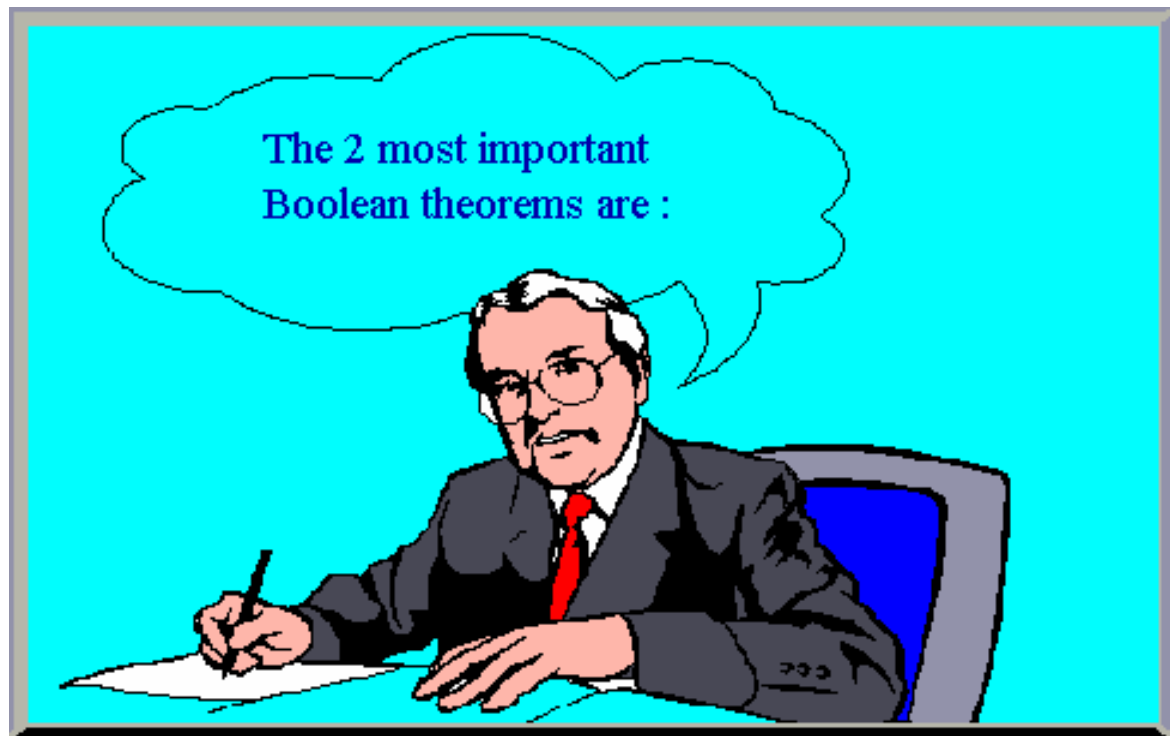
Question:

Using the Boolean theorems, simply the expression:

$$Z = A\bar{B}\bar{C}D + AB + A\bar{B}CD$$

Express your **Answer** in the simplest form.

Answer: $Z = AB + AD$



DeMorgan's Theorems:

$$16) \quad \overline{(X + Y)} = \bar{X} \cdot \bar{Y}$$

$$17) \quad \overline{(X \cdot Y)} = \bar{X} + \bar{Y}$$

Schematically equivalent to:

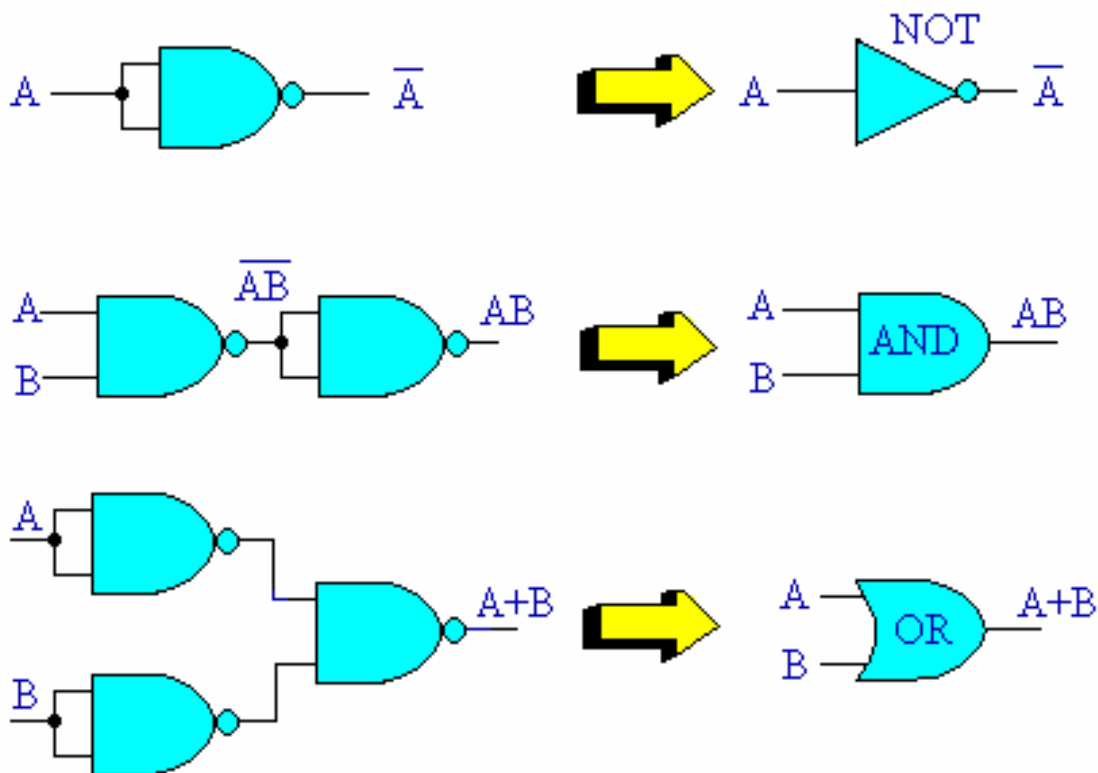


Universality of NAND and NOR Gates

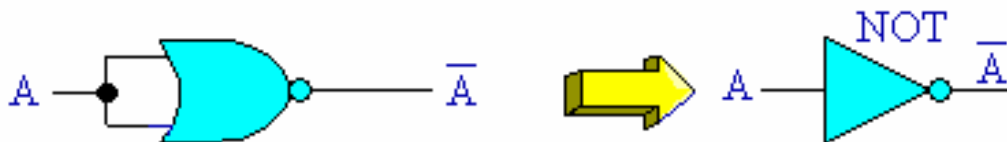
All Boolean expressions can be implemented using AND, OR, and NOT gates.

Since the AND, OR, and NOT functions can be configured from the NAND function, then any digital circuit can be implemented using only NAND gates.

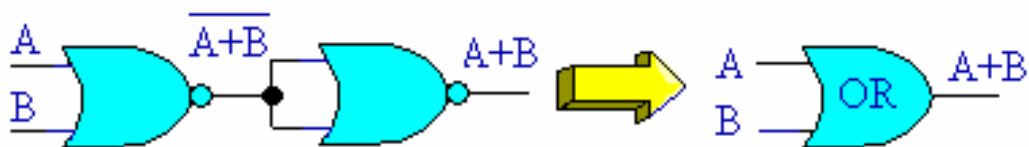
For this reason, the NAND gate is referred to as a 'Universal' gate.



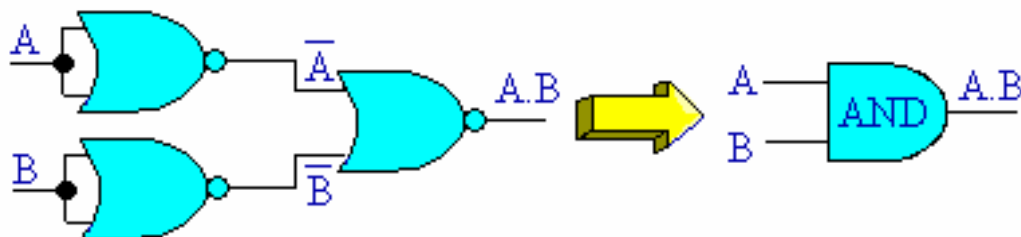
Similarly, NOR gates are also universal in that any digital circuit can be implemented solely from it.



Joining the inputs of a NOR gate together turns it into a NOT gate.



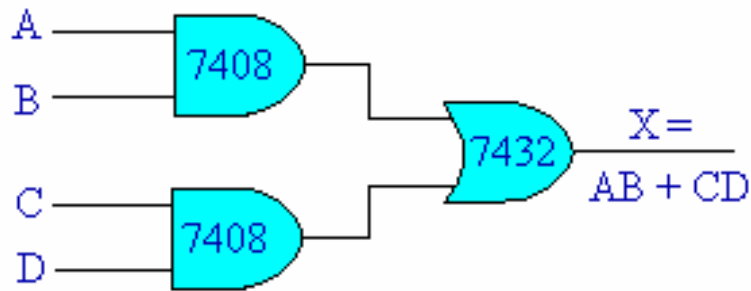
A NOR followed by NOT is equivalent to the OR function.



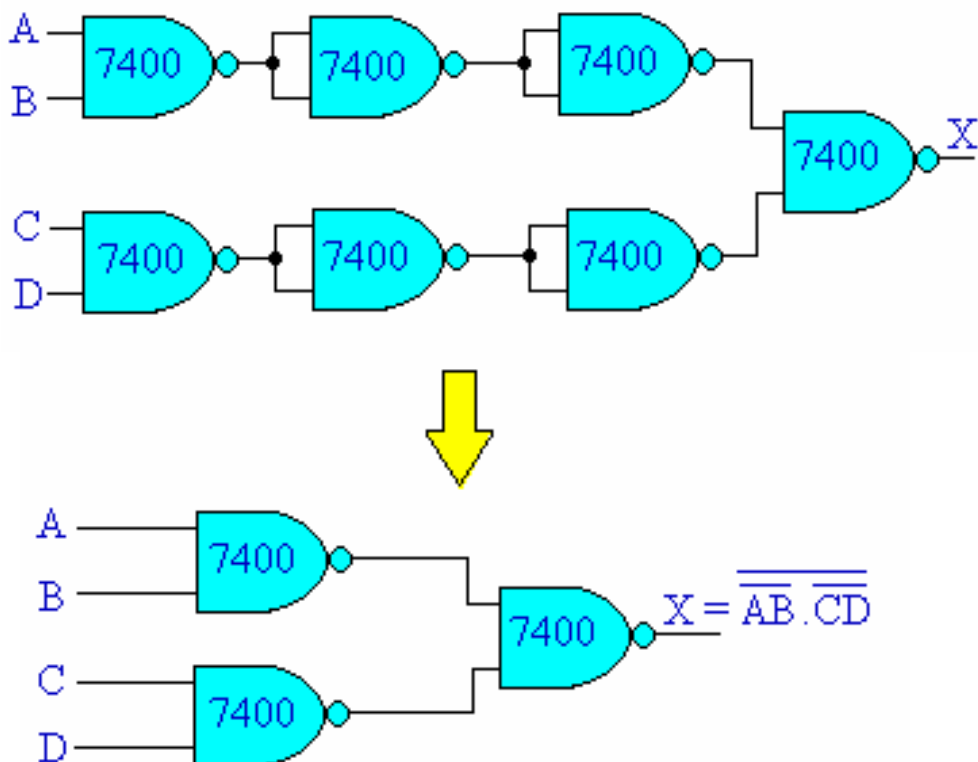
From DeMorgan's theorems, inverting the inputs to the NOR function is equivalent to the AND function.

Example:

Given a typical AND-OR circuit combination as shown below:



Replacing each AND and OR gate with the equivalent NAND gate configuration, yields:

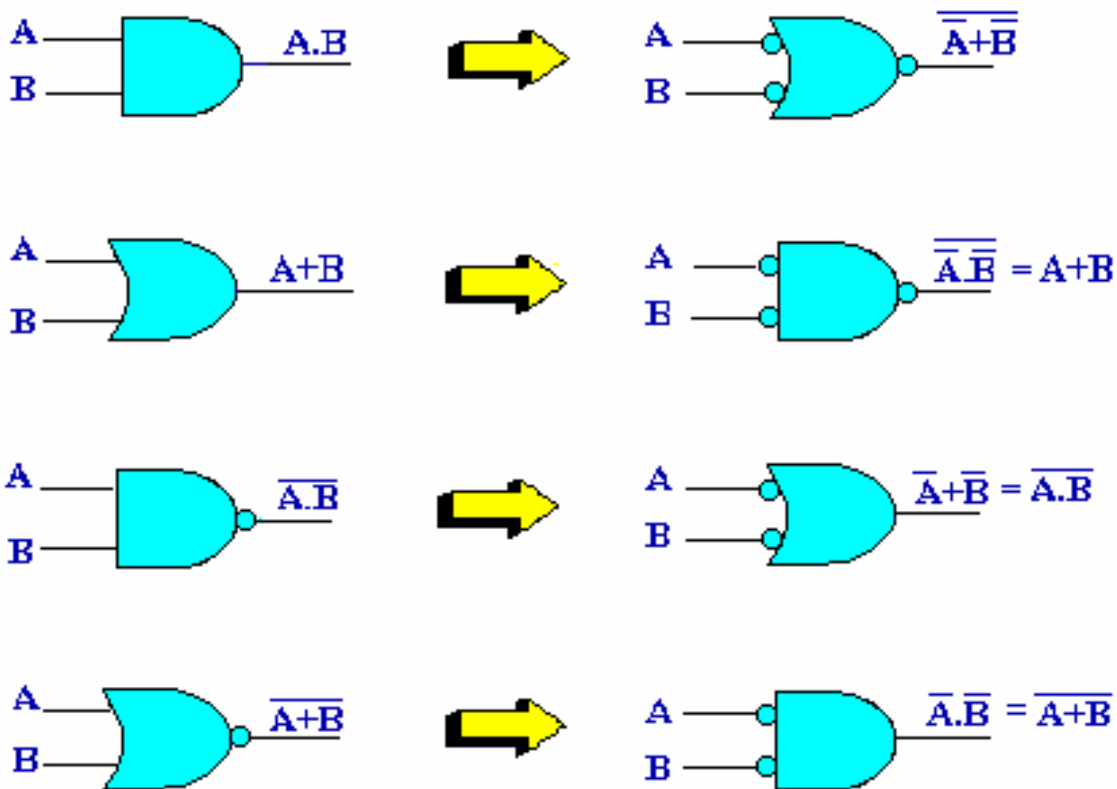


By eliminating double inversions, a circuit requiring only 3 NAND gates is obtained.

Alternative Logic - Gate Representation

Alternate logic symbols are sometimes use in place of the standard logic symbols.

These symbols when used appropriately, allows the operation of a logic circuit to be determined more readily.

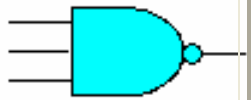



Easy way to obtain alternate symbol is:

1. AND is replaced by OR & vice versa.
2. Bubbles at inputs &/or output are replaced with no bubbles & vice versa.

Example

To obtain the alternate symbol for a 3-input NAND gate,

Conventional symbol of 3-input NAND gate.	
replace it with the OR symbol complement its inputs and output. This yields -->	

Alternate symbols allows the operation of a logic gate to be interpreted from a different perspective.

Taking the NAND gate example, if the standard symbol (bubble at the output) is used, the operation can be described as :

- *The output goes Low only when all inputs are High.*

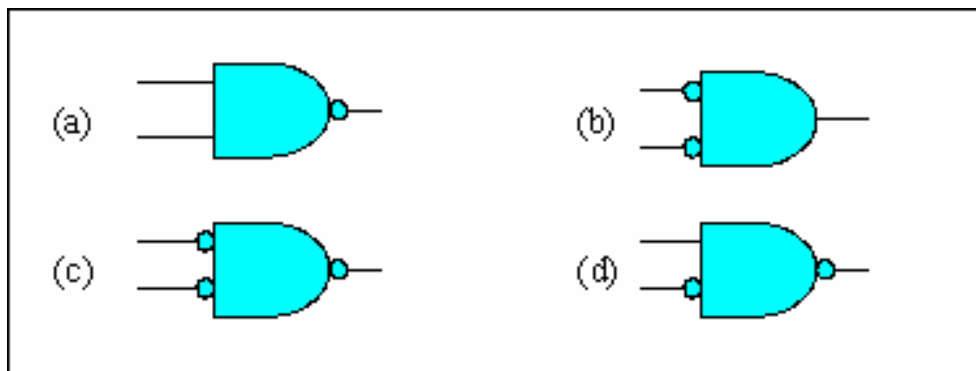
If the alternate symbol (shown above) is used, the gate operation would be described as:

- *The output is High when any input is Low.*

By judiciously using the appropriate symbols in a digital circuit, be it the standard or alternate symbol, the operation of the circuit can be made more readily understandable.

Question

What is the alternate symbol of the OR gate?



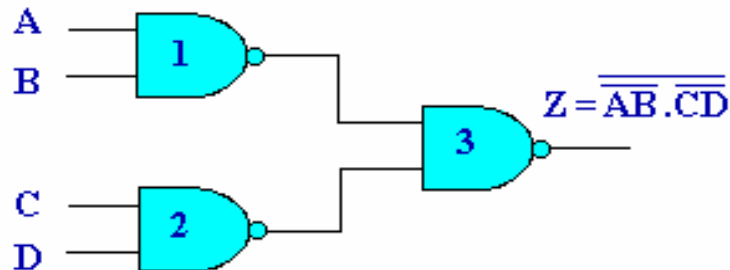
Answer by selecting (a), (b), (c), (d)

Hint

- *AND is replaced by OR and OR is replaced by AND.*
- *Where there are 'bubbles', remove them.*
- *Where there are 'no bubbles', replaced with 'bubbles'.*

Which Gate Representation to use?

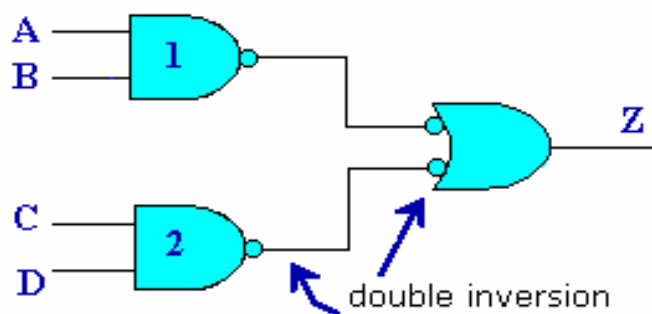
A typical NAND implementation is shown below:



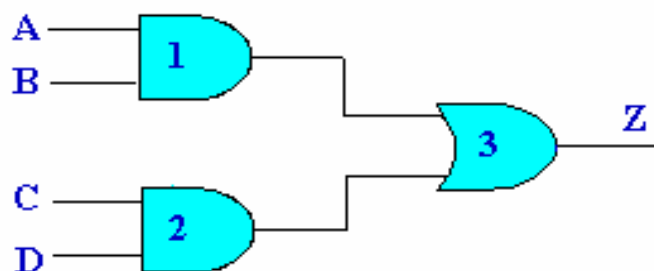
Output $Z = \overline{\overline{AB} \cdot \overline{CD}}$

Expression is logically correct but does not clearly show the circuit operation.

Replacing NAND 3 with the equivalent symbol,



Double inversion between the NAND gates 1\2 & NAND gate 3 is equivalent to no inversion:



The operation can now be easily determined.

Question

What logic levels are required at A B C D for output Z to be = 1 ?

Answer:

$A = H$ and $B = H$ or,

$C = H$ and $D = H$ or,

All inputs = H

In order for Z to be H