

**Ministry of high Education and Scientific Research  
Southern Technical University  
Technological institute of Basra  
Department of Electronic Techniques**



## **Learning package**

# **Digital circuits 1 (E112)**

For

First year students

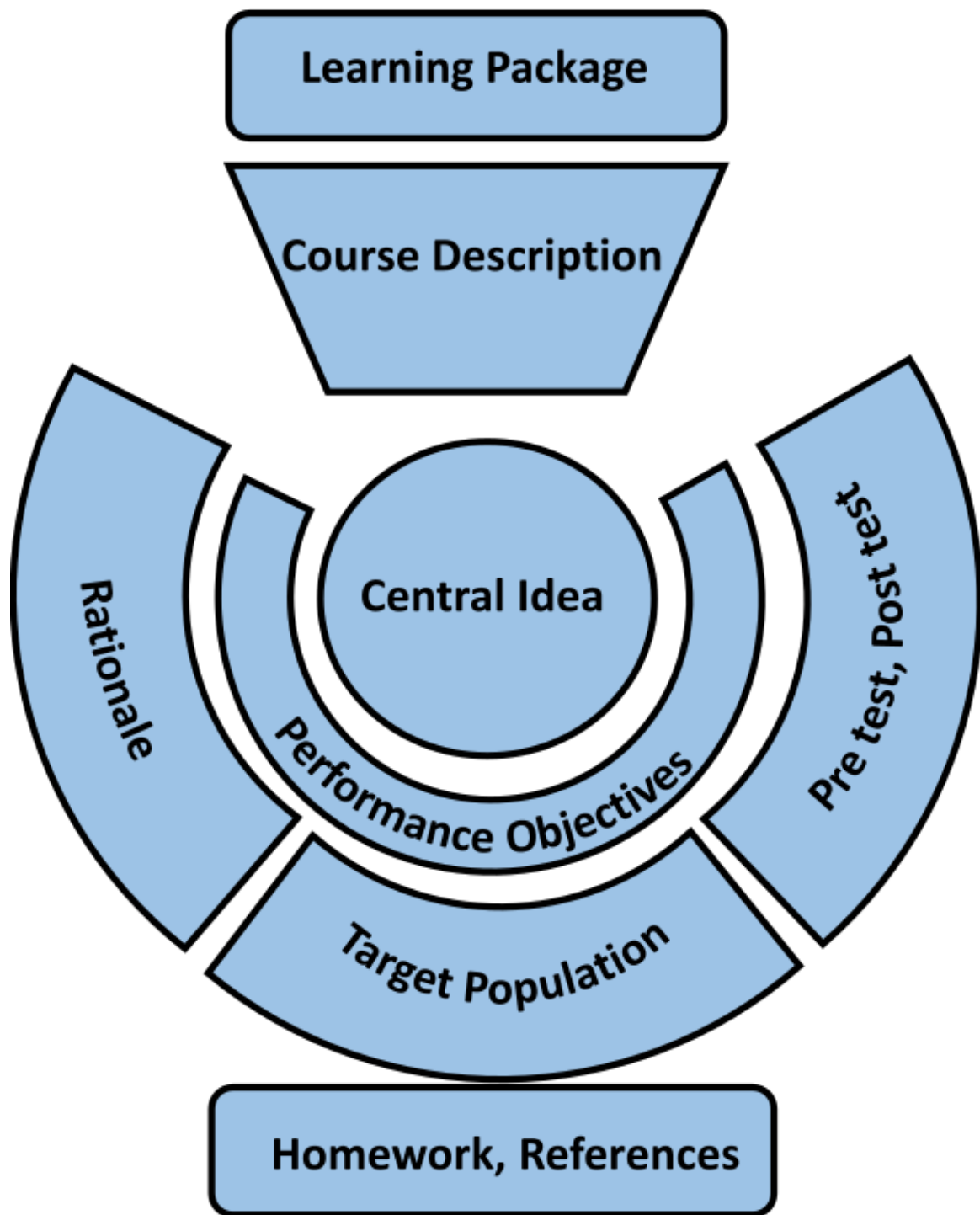
By

**Dr. Haider Mohammed**

**Assistant Professor**

**Dep. Of Electronic Techniques**

**2025**



## Course Description

Course Name:		
Digital circuit 1		
Course Code:		
E112		
Semester / Year:		
Semester		
Description Preparation Date:		
14/ 05/ 2025		
Available Attendance Forms:		
Attendance only		
Number of Credit Hours (Total) / Number of Units (Total)		
60 hours/4 hour weekly/4 unit		
Course administrator's name (mention all, if more than one name)		
Name: Dr. Haider Mohammed		
Email: <a href="mailto:h.m.a.alrudainy@stu.edu.iq">h.m.a.alrudainy@stu.edu.iq</a>		
Course Objectives		
<p><b>1. Developing Basic Understanding of digital circuits:</b> Enabling students understand the fundamental principles of digital circuits, including basic electronic components such <b>as adders, digital gates, and comparator.</b></p> <p><b>2. Applying Theoretical Concepts:</b> Enhancing the ability to apply theoretical concepts in the design and analysis of electronic circuits.</p> <p><b>3. Developing Practical Skills:</b> Providing hands-on training through laboratory experiments, allowing students to acquire the skills necessary to build and test electronic circuits.</p>	<ul style="list-style-type: none"> <li>•</li> <li>•</li> <li>•</li> </ul>	<p>.....</p> <p>.....</p> <p>.....</p>

4. Information and Communication Technology: Understanding the role of digital circuits in information and communication technology and its practical applications.					
5. Enhancing Critical Thinking: Encouraging students to engage in critical and analytical thinking when solving problems related to electronics.					
Teaching and Learning Strategies					
1. Cooperative Concept Planning Strategy.					
2. Brainstorming Teaching Strategy.					
3. Note-taking Sequence Strategy.					
Course Structure					
Weeks	Hours	Required Learning Outcomes	Unit or subject name	Learning method	Evaluation method
1	4hours	1.Understanding digital circuits Applications 2.Developing Critical Thinking and Problem-Solving Skills through Circuit Analysis and Fault Detection. 3.to Use Electronic Laboratory Tools, such as Multimeter, Signal Generators, and Oscilloscopes. 4.Analyzing Electronic Circuits	1- Number Systems	1.Conducting laboratory experiments to build and test digital circuits. This enhances theoretical understanding and develops practical skills.  2.Seeking feedback from instructors and peers to identify strengths and weaknesses.  3.Reviewing concepts periodically and applying them to new problems to reinforce memory and understanding.	Weekly, Monthly, Daily and Written Exams, and Final Term Exam.
2	4hours		2- Number Systems		
3	4hours		3.Logic Gates		
4	4hours		4.Representation of Logic Gates		
5	4hours		5.Boolean Algebra		
6	4hours		6.DeMorgan’s Theorem		
7	4hours		7.Karnaugh Map		
8	4hours		8.Karnaugh Map for Three Variables		
9	4hours		9.Karnaugh Map for Four Variables		
10	4hours		10.Digital Comparator		
11	4hours		11.Two-Level Comparison		
12	4hours		12.Codebreaker		
13	4hours		13.Encoding		
14	4hours		14-15. Decimal to Binary Encoding		

				<p><b>4.</b>Using educational software and interactive applications to better understand concepts, such as circuit simulations.</p> <p><b>5.</b>Encouraging self-research on new topics in electronics and exploring recent developments.</p>	
Course Evaluation					
Distribution as follows: 20 points for Midterm Theoretical Exams for the first semester, 20 points for Midterm Practical Exams for the first semester, 10 points for Daily Exams and Continuous Assessment, and 50 points for the Final Exam.					
Learning and Teaching Resources					
Required textbooks (curricular books, if any)				Holdsworth, Brian, and Clive Woods. Digital logic design. Elsevier, 2002.	
Main references (sources)				Alam, Mansaf, and Bashir Alam. Digital Logic Design. PHI Learning Pvt. Ltd., 2015.	
Recommended books and references (scientific journals, reports...)				Dally, William James, and R. Curtis Harting. Digital design: a systems approach. Cambridge University Press, 2012.	
Electronic References, Websites				<a href="https://zlibrary-asia.se/">https://zlibrary-asia.se/</a>	

Ministry of high Education and Scientific Research  
Southern Technical University  
Technological institute of Basra  
Department of Electronic Techniques



**Learning package**  
**In**  
**Numbers Systems**  
**For**  
Students of First Year



**By**  
**Dr. Haider Mohammed**  
Assistant Professor  
Dep. Of Electronic Techniques  
2025

# **1/ Overview**

## **1 / A –Target population :-**

For students of First year  
Technological institute of Basra  
Dep. Of Electronic Techniques

## **1 / B –Rationale :-**

Understanding number systems is crucial for gaining comprehensive knowledge of digital electronics, which is why I have created this modular unit to facilitate learning about this subject.

## **1 / C –Central Idea :-**

- 1 – Types of numbers systems
- 2 – Representation of numbers systems
- 3 –Conversion numbers systems

## **1 / D – Performance Objectives**

After studying the first unit, the student will be able to:-

1. Know the types of Number systems
2. Representations of numbers systems
3. Convert between the types of numbers systems

## **2/ Pretest**

**:**



### **3/ Numbers Systems :-**

#### **1. The types of number system**

##### **A-Decimal Number System :-**

This system is composed of 10 numbers or symbols, these 10 symbols are:

0   1   2   3   4   5   6   7   8   9

These symbols are called digits.

The decimal system, also called base 10 system, because it has 10 digits which is a naturally result of the fact that man has 10 fingers.

##### **B- Binary Number System**

In this system there are only two symbols or possible digit values , 0 or 1 . Even so , this base-2 system.

## **C- Octal Number System**

**This system is composed of 8 numbers or symbols:**

**0 1 2 3 4 5 6 7**

**This is a base -8 system.**

## **D- Hexa- Decimal System**

**This system is composed of 16 numbers or symbols  
(digit):**

**0 1 2 3 4 5 6 7 8 9 A B C D E F**

**It is a base – 16 systems**

## **2. Representation of numbers**

**1) Decimal :-**

$$(124)_{10} = 4 \times 10^0 + 2 \times 10^1 + 1 \times 10^2$$

$$(252.512)_{10} = 2 \times 10^0 + 5 \times 10^1 + 2 \times 10^2 + 5 \times 10^{-1} + 1 \times 10^{-2} + 2 \times 10^{-3}$$

**2) Binary :-**

$$(1011101)_2 = 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 1 \times 2^3 + 1 \times 2^4 + 0 \times 2^5 + 1 \times 2^6$$

$$= (93)_{10}$$

$$(101.11)_2 = 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 1 \times 2^{-1} + 1 \times 2^{-2}$$

$$= (5.75)_{10}$$

### 3) Octal :-

$$(537)_8 = 7 \times 8^0 + 3 \times 8^1 + 5 \times 8^2$$

$$= (351)_{10}$$

### 4) Hexa- Decimal :-

$$(A01B)_{16} = 11 \times 16^0 + 1 \times 16^1 + 0 \times 16^2 + 10 \times 16^3$$

$$= (40987)_{10}$$

## 3. Convert between the types of numbers systems

***any base-to-decimal conversion :-*** just use the definition given above.

### ***Decimal-to-binary :-***

divide decimal value by 2 (the base) until the value is 0

example: convert the following decimal numbers to the equivalent binary numbers (36 , 39.5).

$$\begin{array}{rcl}
 36/2 = 18 & r=0 & \\
 18/2 = 9 & r=0 & \\
 9/2 = 4 & r=1 & \\
 4/2 = 2 & r=0 & \\
 2/2 = 1 & r=0 & \\
 1/2 = 0 & r=1 & 
 \end{array}$$

$$36 \text{ (base 10)} = 100100$$

$$\begin{array}{rcl}
 39/2 = 19 & r=1 & \\
 19/2 = 9 & r=1 & \\
 9/2 = 4 & r=1 & \\
 4/2 = 2 & r=0 & \\
 2/2 = 1 & r=0 & \\
 1/2 = 0 & r=1 & 
 \end{array}$$

$$39.5 \text{ (base 10)} = 100111$$

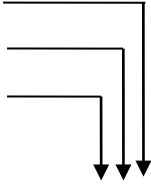
$$\begin{array}{rcl}
 0.5 \times 2 = 1 & & \uparrow \\
 0 \times 2 = 0 & & \uparrow
 \end{array}$$

The binary equivalent of  $(39.5)_{10}$  is  $(100111.10)_2$

### ***Decimal-to-Octal :-***

A decimal integer can be converted to octal by using the same repeated-division method that we used in the decimal-to-binary conversion, but with a division of 8 instead of 2.

*Example:* convert the following decimal number to equivalent octal number  $(266)_{10}$  &  $(20.75)_{10}$

$$\begin{array}{rcl}
 266/8 = 33 & r=2 & \\
 33/8 = 4 & r=1 & \\
 4/8 = 0 & r=4 & \\
 \hline
 (266)_{10} = & & 412
 \end{array}$$


$$\begin{array}{rcl}
 20/8 = 2 & r=4 & 0.75 \times 8 = 6.0 \\
 2/8 = 0 & r=2 &
 \end{array}$$

The equivalent octal number is  $(24.6)_8$

### ***Octal-to-binary :-***

The conversion from octal to binary is performed by converting each octal digit to its 3-bit binary equivalent. The eight possible digits are converted as indicated in the following table:

Octal digit	0	1	2	3	4	5	6	7
Binary digit	000	001	010	011	100	101	110	111

*Example:* convert the following octal number to its equivalent binary number  $(472)_8$

$$\begin{array}{ccc}
 4 & 7 & 2 \\
 100 & 111 & 010
 \end{array}$$

The equivalent binary number is  $(100111010)_2$

### ***Binary-to-octal :-***

1. group into 3's starting at least significant symbol (if the number of bits is not evenly divisible by 3, then add 0's at the most significant end)
2. write 1 octal digit for each group

*example:*

$$\begin{array}{ccc} \underline{100} & \underline{010} & \underline{111} \text{ (binary)} \\ 4 & 2 & 7 \text{ (octal)} \end{array}$$

$$\begin{array}{ccc} \underline{10} & \underline{101} & \underline{110} \text{ (binary)} \\ 2 & 5 & 6 \text{ (octal)} \end{array}$$

*example:-*

convert  $(177)_{10}$  to its 8-bit binary equivalent by first converting to octal.

Solution:-

$$177/8 = 22 + \text{remainder of } 1$$

$$22/8 = 2 + \text{remainder of } 6$$

$$2/8 = 0 + \text{remainder of } 2$$

$$(177)_{10} = (261)_8$$

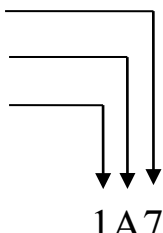
$$\begin{array}{ccc} \underline{2} & \underline{6} & \underline{1} \\ 010 & 110 & 001 \end{array}$$

$$(177)_{10} = (261)_8 = (010110001)_2$$

### ***Decimal-to-hex :-***

This conversion can be done using repeated division by 16.

*Example:* convert  $(423)_{10}$  to hex.

$$\begin{array}{rcl} 423/16 = 26 & r=7 & \\ 26/16 = 1 & r=10 & \\ 1/16 = 0 & r=1 & \end{array}$$


1A7

### ***Hex-to-binary :-***

Each hex digit is converted to its 4-bit binary equivalent as shown in the table below :

Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Binary	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

*example:*

3    9    C    8  
0011 1001 1100 1000

### ***Binary-to-hex :-***

This conversion is just the reverse of the hexa-to-binary conversion process.

*example:*

$$\begin{array}{cccc} \underline{1001} & \underline{1110} & \underline{0111} & \underline{0000} \\ 9 & E & 7 & 0 \end{array}$$

$$\begin{array}{cccc} \underline{1} & \underline{1111} & \underline{1010} & \underline{0011} \\ 1 & F & A & 3 \end{array}$$

### ***Binary-coded-decimal code :-***

If each digit of a decimal number is represented by its binary equivalent, this produces a code called binary-coded-decimal (BCD). Since a decimal digit can be as large as 9, 4-bits are required to code each digit. The table below shows each decimal digit and its binary equivalent.

Decimal	0	1	2	3	4	5	6	7	8	9
BCD	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

*example:* The BCD of decimal 874

$$\begin{array}{ccc} 8 & 7 & 4 \\ \underline{1000} & \underline{0111} & \underline{0100} \end{array}$$

### **Excess-3-code :-**

It is performed in the same manner as BCD except that 3 is added to each decimal digit before encoding it in binary. The following table shows this code.

Decimal	0	1	2	3	4	5	6	7	8	9
---------	---	---	---	---	---	---	---	---	---	---



<b>Ex-3code</b>	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100
-----------------	------	------	------	------	------	------	------	------	------	------

### **Gray code :-**

The Gray code belongs to a class of codes called minimum change codes, in which only from one step to the next. The following table shows this code.

<b>Decimal</b>	0	1	2	3	4	5	6	7	8	9
<b>Gray-code</b>	0000	0001	0011	0010	0110	0111	0101	0100	1100	1101

## **4/ Post test :-**

**1- Convert of (177) to its 8-bit binary equivalent by first converting to octal**

**2- Convert (423) to Hexa**

## **key answer :-**

**post test :-**

1. 010110001
2. 1A7

## **5/ HomeWorks:-**

- 1- Convert  $(641)_8$  to decimal (Ans. 369).
- 2- Convert  $(146)_{10}$  to octal then from octal to binary (Ans. 222 and 010010010).
- 3- Convert  $(10011101)_2$  to octal (Ans. 235).
- 4- Write the next three numbers in this octal counting sequence:  
624, 625, 626, ....., ....., .....
- 5- Convert  $(975)_{10}$  to binary by first converting to octal (Ans. 1111001111).
- 6- Convert binary 1010111011 to decimal by first converting to octal (Ans. 699).
- 7-Convert  $(24CE)_{16}$  to decimal (Ans. 9422).
- 8-Convert  $(3117)_{10}$  to hex, then from hex to binary (Ans. C2D and 110000101101)

- 9-Convert  $(1001011110110101)_2$  to hex (Ans 97B5).  
10-Write the next four numbers in this hex counting sequence:  
E9A, E9B, E9C, E9D, ....., ....., ....., .....  
11-Convert  $(3527)_8$  to hex (Ans.  $(757)_{16}$ ).



## 6/References :-

1. FLOYD / DIGITAL FUNDAMENTALS
2. MALVINO/ DIGITAL PRINCIPLES AND APPLICATION

Ministry of high Education and Scientific Research  
Southern Technical University  
Technological institute of Basra  
Department of Electronic Techniques



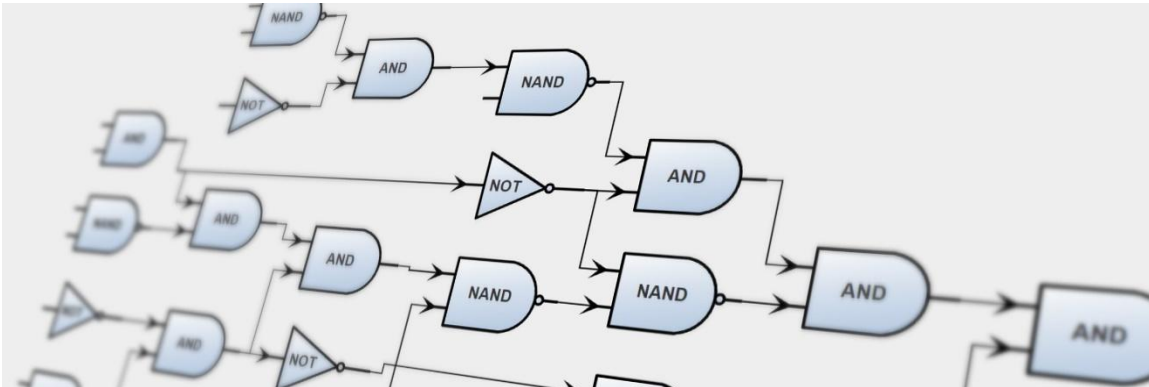
## Learning package

In

## Logic gates

For

First year students



By

**Dr. Haider Mohammed**

Assistant Professor

Dep. Of Electronic Techniques

2025

# **1/ Overview**

## **1 / A –Target population :-**

For students of First year  
Technological institute of Basra  
Dep. Of Electronic Techniques

## **1 / B –Rationale :-**

A logic gate executes a logical operation on one or more inputs and yields a single output. Typically, this operation is Boolean logic and is most often utilized in digital circuits. These gates are mainly built electronically using diodes or transistors, but can also be assembled through electromagnetic relays (relay logic), fluidic logic, pneumatic logic, optics, molecular logic, or even mechanical components.

## **1 / C –Central Idea :-**

**1-NOT gate**

**2- AND gate**

**3-OR gate**

**4-NAND gate**

**5-NOR gate**

**6-XOR gate**

**7-XNOR**

## **1 / D – Performance Objectives**

After studying this unit, the student will be able to:-

1- Know the types of logic gates

2- Build simple and complex logic circuits

3- Represent logic circuits in Boolean algebra

## **2/ Pretest**

**:**

### **3/ Logic gates :-**

#### **Truth tables :-**

Many logic circuits have more than one input and one or more outputs. A truth table shows how the logic circuit's output responds to the various combinations of logic states at the inputs. The formal for two, three, and four input with one output truth tables are shown below :

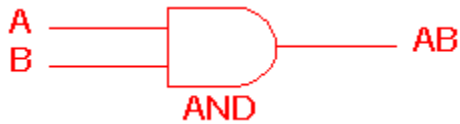
B	A	X
0	0	
0	1	
1	0	
1	1	

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

D	C	B	A	X
0	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	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- AND gate



2 Input AND gate		
A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

The AND gate is an electronic circuit that gives a **high** output (1) only if **all** its inputs are high. A dot (.) is used to show the AND operation i.e.

A.B. Bear in mind that this dot is sometimes omitted i.e. AB

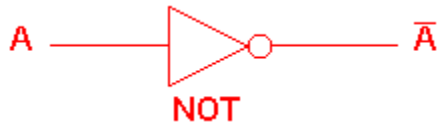
## 2- OR gate



2 Input OR gate		
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

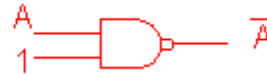
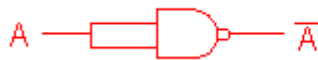
The OR gate is an electronic circuit that gives a high output (1) if **one or more** of its inputs are high. A plus (+) is used to show the OR operation.

### 3- NOT gate



NOT gate	
A	$\bar{A}$
0	1
1	0

The NOT gate is an electronic circuit that produces an inverted version of the input at its output. It is also known as an *inverter*. If the input variable is  $A$ , the inverted output is known as NOT  $A$ . This is also shown as  $A'$ , or  $A$  with a bar over the top, as shown at the outputs. The diagrams below show two ways that the NAND logic gate can be configured to produce a NOT gate. It can also be done using NOR logic gates in the same way.



### 4- NAND gate



2 Input NAND gate		
A	B	$\bar{A}\bar{B}$
0	0	1
0	1	1
1	0	1
1	1	0

This is a NOT-AND gate which is equal to an AND gate followed by a NOT gate. The outputs of all NAND gates are high if **any** of the inputs are low. The symbol is an AND gate with a small circle on the output. The small circle represents inversion.

## 5- NOR gate



2 Input NOR gate		
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

This is a NOT-OR gate which is equal to an OR gate followed by a NOT gate. The outputs of all NOR gates are low if **any** of the inputs are high. The symbol is an OR gate with a small circle on the output. The small circle represents inversion.

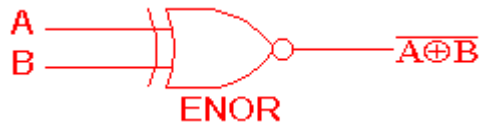
## 6- XOR gate



2 Input EXOR gate		
A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

The '**Exclusive-OR**' gate is a circuit which will give a high output if **either, but not both**, of its two inputs are high. An encircled plus sign ( $\oplus$ ) is used to show the EOR operation.

## 7- XNOR gate



2 Input EXNOR gate		
A	B	$\overline{A \oplus B}$
0	0	1
0	1	0
1	0	0
1	1	1

The 'Exclusive-NOR' gate circuit does the opposite to the EOR gate. It will give a low output if **either, but not both**, of its two inputs are high. The symbol is an EXOR gate with a small circle on the output. The small circle represents inversion.

The NAND and NOR gates are called *universal functions* since with either one the AND and OR functions and NOT can be generated.

Note:

A function in *sum of products* form can be implemented using NAND gates by replacing all AND and OR gates by NAND gates.

A function in *product of sums* form can be implemented using NOR gates by replacing all AND and OR gates by NOR gates.

### Table 1: Logic gate symbols

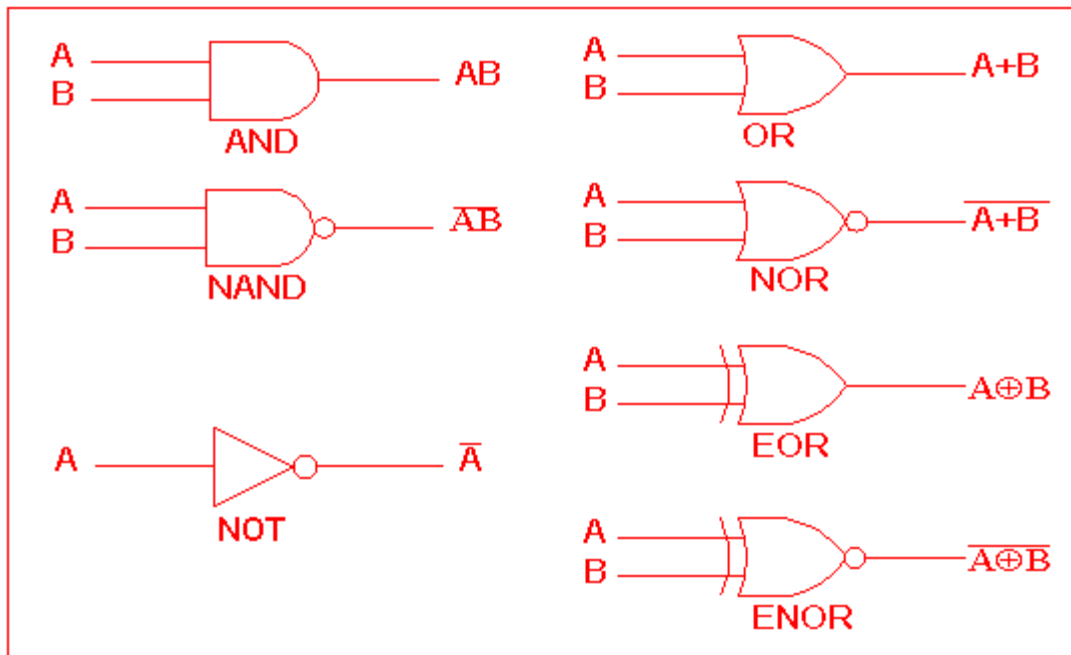


Table 2 is a summary truth table of the input/output combinations for the NOT gate together with all possible input/output combinations for the other gate functions. Also note that a truth table with 'n' inputs has  $2^n$  rows. You can compare the outputs of different gates.

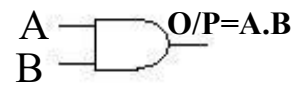
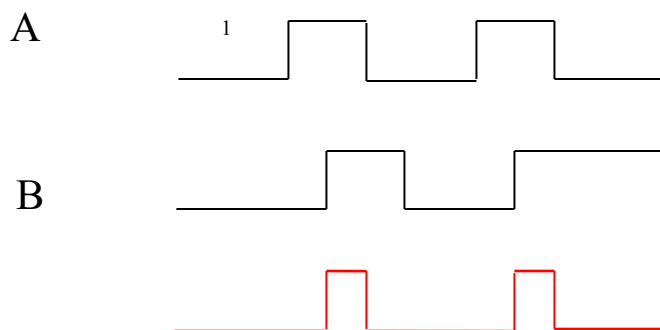
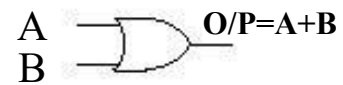
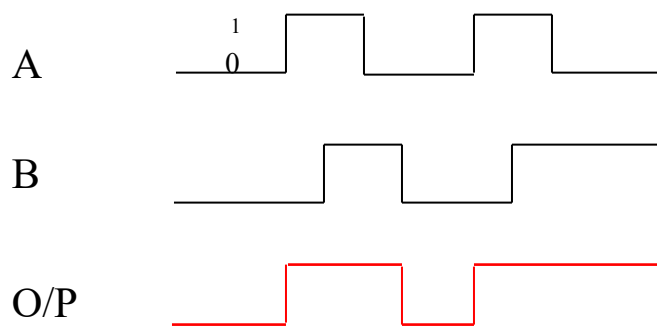
**Table 2: Logic gates representation using the Truth table**

		INPUTS		OUTPUTS					
		A	B	AND	NAND	OR	NOR	EXOR	EXNOR
<b>NOT gate</b>	A	0	0	0	1	0	1	0	1
	$\overline{A}$	0	1	0	1	1	0	1	0
	0	1	0	0	1	1	0	1	0
	1	1	1	1	0	1	0	0	1

*Example:*

Determine the OR gate output in the fig. shown below. The OR gate inputs A and B are varying according to the timing diagrams shown.

*Example:* Repeat the previous example for AND gate

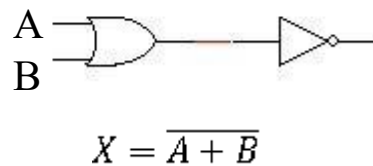
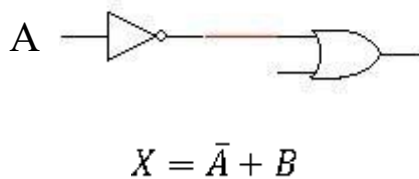
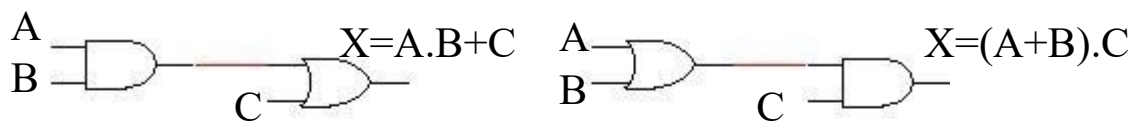


## **Describing logic circuits algebraically :-**

Any logic circuit, no matter how complex, may be completely described using the Boolean operations previously defined.

Example:-

Determine the output expression for the logic indicated below: -



## **Implementing circuits from Boolean expressions :-**

If the operation of a circuit is defined by a Boolean expression, a logic circuit can be implemented directly from that expression.

Example :

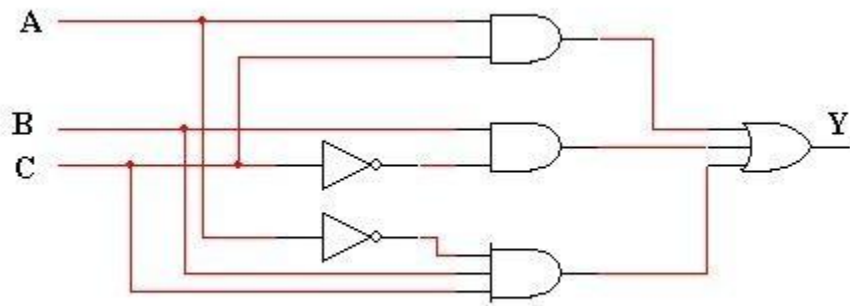
Implement the logic circuits defined by the following Boolean expressions :

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

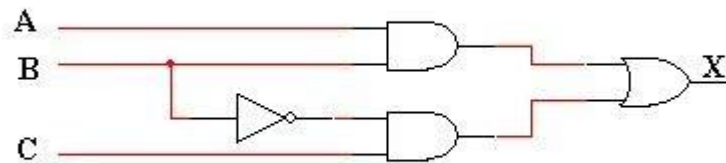
b)  $X = AB + \bar{B}C$

Solution :-

a)



b)



#### 4/ Post test :-

Draw the circuit diagrams to show how a **NOR gate** can be made into a **NOT gate**.



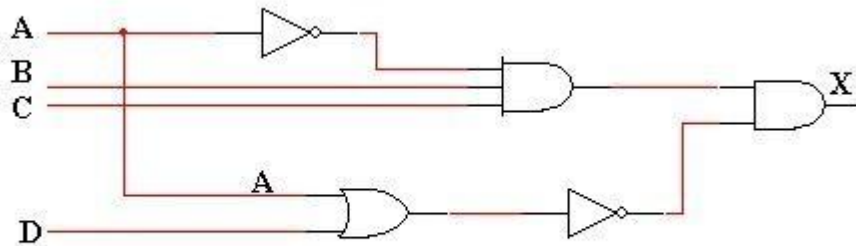
**key answer :-**

**1- post test :-**



**5/ HomeWorks: -**

a) determine the output expression for the following circuit



b) determine the output logic level if A=1, B=1 and C=0, D=0

## **6 / Resferences**

- 1- FLOYD / DIGITAL FUNDAMENTALS
- 2- MALVINO/ DIGITAL PRINCIPLES AND APPLICATION