DIGITAL ELECTRONICS

CHAPTER-5: LOGIC GATES

Reference:

- Malvino and Leach, Digital Principles & applications, 7th edition, TMH, 2010
- Morris Mano, “Digital design”, Prentice Hall of India, Third Edition.

MODULE -2

LOGIC GATES

- **OBJECTIVES**

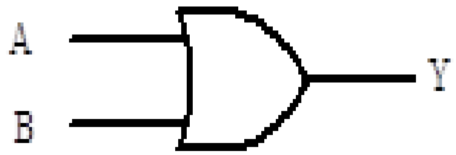
At the end of this module students will be able to :

- *Describe basic logic gates and the concept of universal logic.*
- *Build a logic circuit for the given Boolean expressions.*
- *Write Boolean expressions for the given logic circuit.*
- *Differentiate combinational and sequential circuits.*

- A logic gate is a digital circuit with one or more input signals and only one output signal.

- The input and output signals are either HIGH (1) or LOW (0).

- OR Gate:



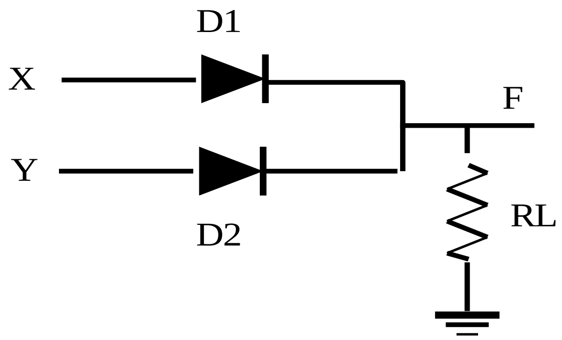
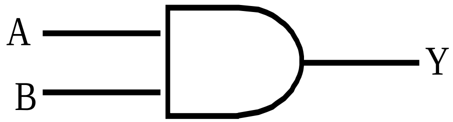


Table: Truth table for two input OR gate

Input		Output
A	B	Y= A+B
0	0	0
0	1	1
1	0	1
1	1	1

- AND Gate:



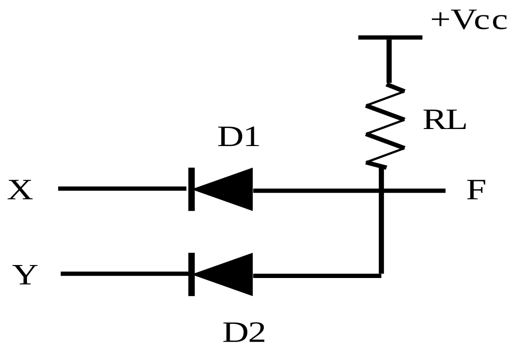
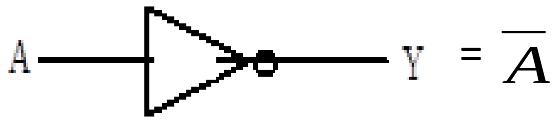
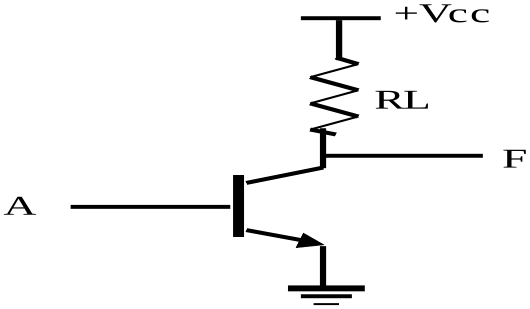


Table: Truth table for two input OR gate

Input		Output
A	B	Y=A.B
0	0	0
0	1	0
1	0	0
1	1	1

- Not Gate (Inverter):



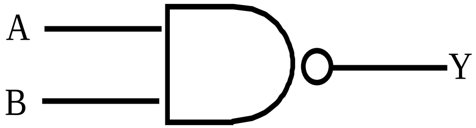


Truth Table for NOT gate

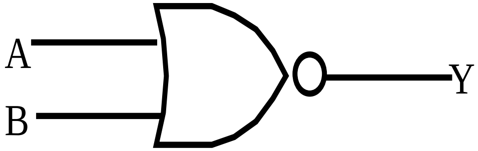
Input	output
A	Y=
0	1
1	0

Universal Gates

- NAND Gate: The output of a NAND gate is LOW only when all inputs are HIGH and output of the NAND is HIGH if one or more inputs are LOW.



- NOR Gate: The output of the NOR gate is HIGH only when all the inputs are LOW.

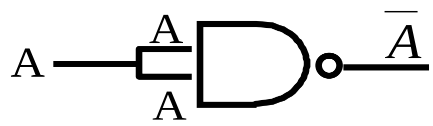


Universal gates: NAND and NOR gates are called Universal gates.

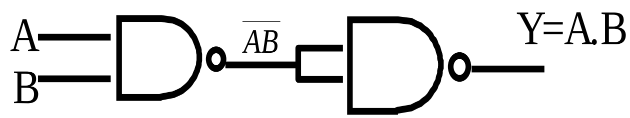
Universal Gates

- NAND gate as Universal gate:

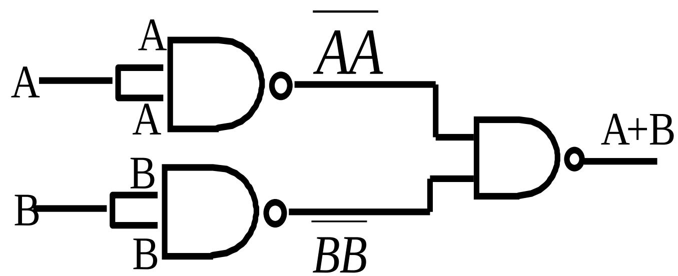
NOT operation:



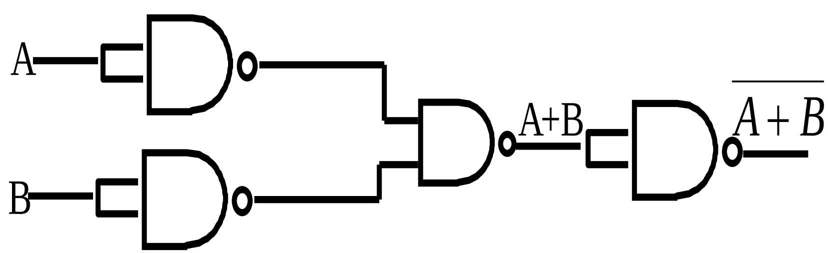
AND operation:



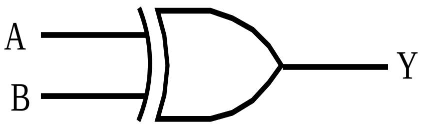
OR operation:



NOR operation:

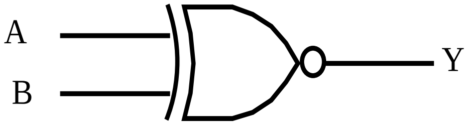


XOR Gate or Exclusive OR gate: output is HIGH only when any one of the input is HIGH. (inequality comparator)



XNOR Gate or Exclusive NOR Gate: complementary of XOR operation.

The output of XNOR gate is High, when all the inputs are identical; otherwise it is low.

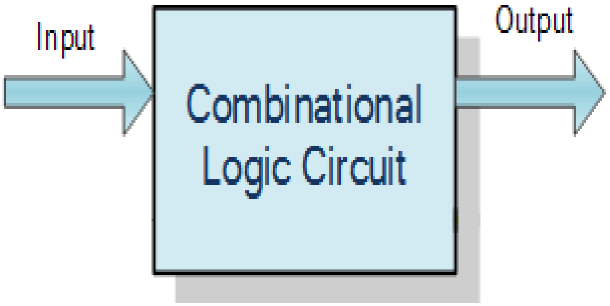


Digital circuits

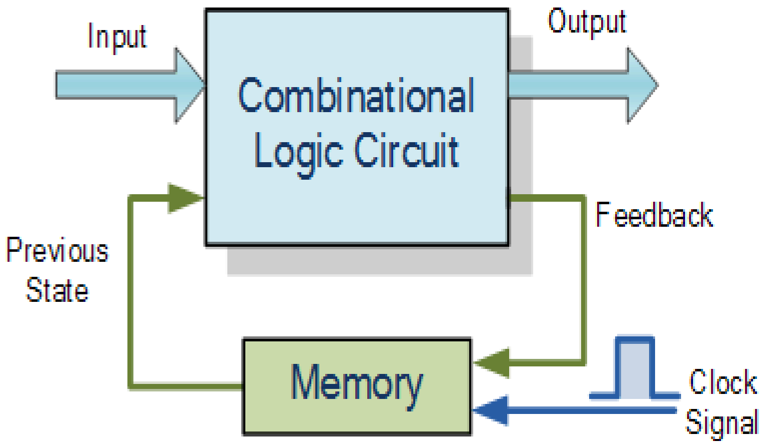
Types of Digital Circuits:

- Combinational Digital Circuits
- Sequential Digital Circuits

Digital circuits



(a)



(b)

Figure (a) Combinational Circuit (b) Sequential circuit

Elements of combinational logic:

- Literal

  Ex- X and X’ are both literals. Similarly ABCD’ consists of 4 literals A,B,C and D’.

- Product term

  Ex- X, XY’, XYZ are the product terms when X,Y,Z are Boolean variables.

- Sum term

  Ex- X+Y’, X+Y+Z are the sum terms when X,Y,Z are Boolean variables.

Continued……..

- Sum of products (SOP): Each product term is the logical AND of literals.

  Ex: Y+XY’+XYZ

- Products of Sums (POS): Each sum term is the logical OR of literals.

  Ex: (X+Y’)((X’Y+Z)(X+Y+Z)

Continued……..

- Canonical form: Canonical is defined as "conforming to a general rule". (Standard form)

  All the literals exist either complimented or non complimented form.

- Canonical Sum of Products:

  Ex: f(A,B,C)= A'B'C+A'BC'+A'BC+ABC'

- Canonical Product of Sums:

  Ex: f(X,Y)=(X+Y')(X+Y)(x'+y)

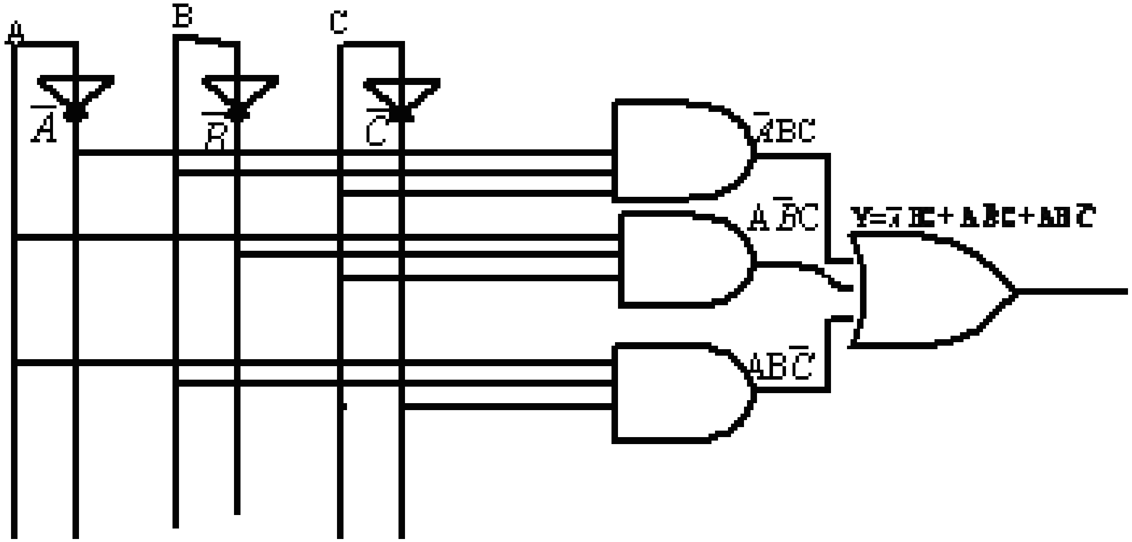
Continued……..

- Minterm: Each product term in the standard (canonical) SOP expression.

- Maxterm: Each sum term in the standard (canonical) POS expression

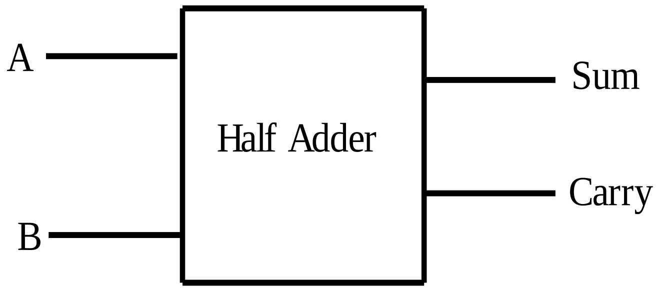
- Examples of combinational circuits

Draw the logic circuit for the Boolean expression. Y= A’BC+AB’C+ABC’.



(Do the reverse process also)

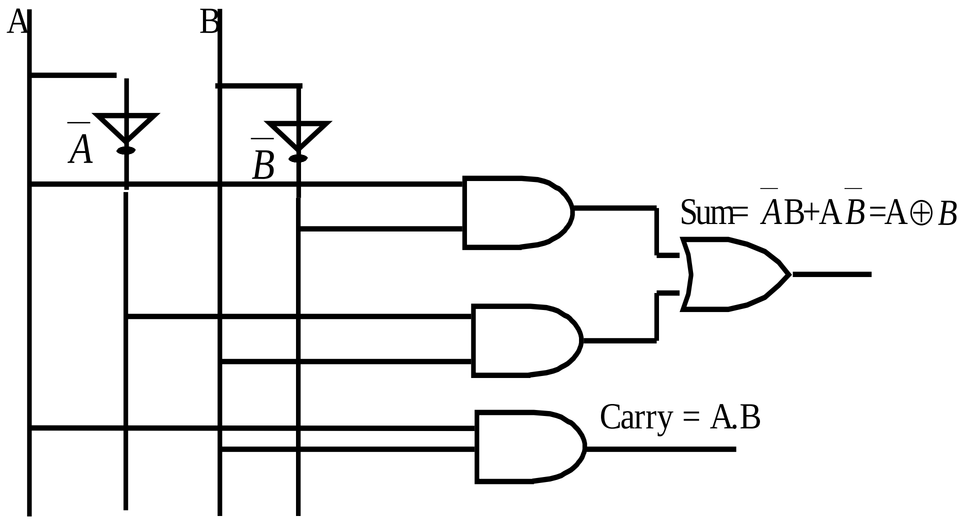
- Half adder circuit



Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Continued…….

- Sum= A’B+AB’ =A XOR B
- Carry= A.B



- Self test

1. Show that NOR is a universal gate.
2. Draw the logic circuit for the Boolean expression. Y= BC+A’ C+AB’C.
3. Implement the half adder circuit using XOR gates.
4. Implement the full adder circuit using logic gates.

Summary

- Logic gates are fundamental building blocks of digital systems
- The basic set of logic gates are AND, OR and NOT and this set is called Universal set.
- NAND and NOR are called Universal gates.
- Inputs and outputs of logic gates can occur in two levels. These two levels are termed as HIGH and LOW, or TRUE and FALSE, or ON and OFF, or simply 1 or 0.
- Logic circuits whose output at any instant of time is entirely dependent upon the input signals present at that time are known as combinational digital circuits.
- Logic circuits whose output at any instant of time depend, not only on the present input but also on the past outputs are called Sequential Circuits.

- **Exercise:**

1. Draw the logic circuit for the Boolean expression. Y= BC+A’ C+AB’C.
2. Show that AB+(A+B) is equivalent to A ʘ B. Also construct the corresponding logic diagrams.
3. The most suitable gate to check whether the number of 1s in a digital word is even or odd is -----------
4. a)X-OR b) NAND c) NOR d) AND, OR and NOT
5. Realize NOR and NAND gate using discrete components.
6. Implement Full Subtractor using Basic gates.
7. Implement full adder using two half adders.

Module-3

KARNAUGH MAP (K – MAP)

- **OBJECTIVES**

At the end of this module students will be able to :

- *Explain the standard form of Boolean expressions.*
- *Apply the K-map for Boolean expression simplification and design of logic circuits.*

Karnaugh map (k – map) method of simplifying the Boolean expressions:

➢ Boolean expression can be expressed in sum of product (SOP) form or product of sum (POS) form.

- Boolean expression in **SOP** form:
  Y = AB’C + ABC + A’BC
- Each of the product terms in the standard SOP form is called a **minterm**.

- Boolean expression in **POS** form:
  Y = (A + B + C’) (A’ + B + C) (A + B + C)
- Each sum term in the standard POS form is called a **maxterm**.

- Steps to convert SOP to canonical SOP:

- Find the missing literal in each product term.

- AND each product term having missing literals with terms by ORing the literal and its complement.

- Expand the terms and reduce the expression by removing repeated terms.

- Ex1: $F(A,B,C) = AC+AB+BC$

$$= A\,(B+B')C+AB(C+C')+(A+A')BC$$

$$= AB'C+ABC'+A'BC+ABC$$

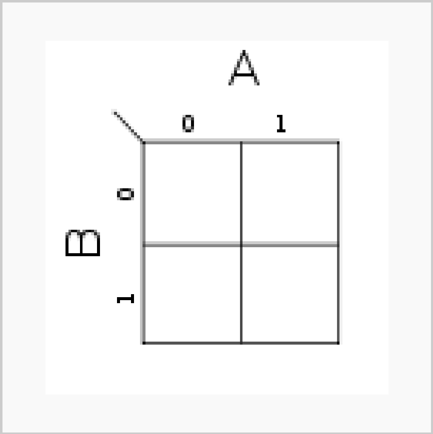
Self test

1. Determine the Boolean function of the truth table in canonical SOP form and simplify the expression.

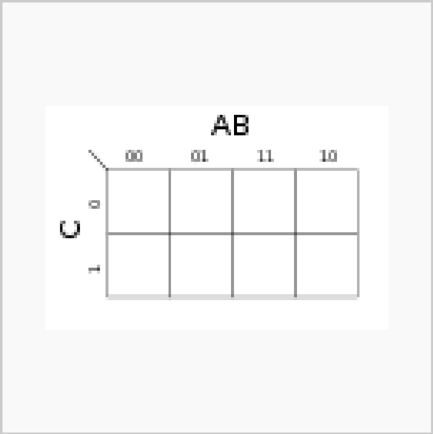
Inputs			Output
A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

- **Introduction**
- **Structure of a K - map :**

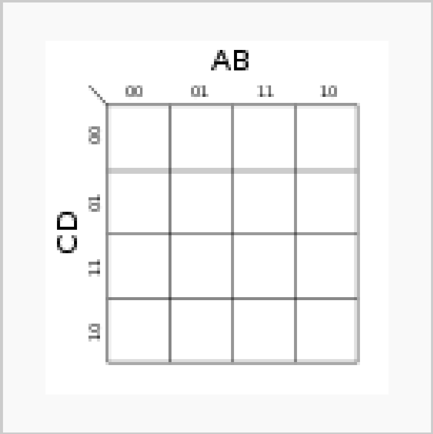
- Two variable K – map has $2^2 = 4$ cells
- Three variable K-map has $2^3 = 8$ cells
- Four variable K-map has $2^4 = 16$ cells.



2-variable *k*-map



3-variable *k*-map



4-variable *k*-map

2-Variable K-MAP

	$\bar{y}$	y
$\bar{x}$	00 0	01 1
x	10 2	11 3

This minterm is expressed as $f = \bar{x}y$.

3-Variable K-MAP

	$\bar{y}\bar{z}$	$\bar{y}z$	yz	$y\bar{z}$
$\bar{x}$	000 0	001 1	011 3	010 2
x	100 4	101 5	111 7	110 6

As an example, this minterm cell (011) represents the minterm $f = \bar{x}yz$.

4-Variable K-MAP

	$\bar{y}\bar{z}$	$\bar{y}z$	yz	$y\bar{z}$
$\bar{w}\bar{x}$	0000 0 0	0001 1 1	0011 3 3	0010 2 2
$\bar{w}x$	0100 4 4	0101 5 5	0111 7 7	0110 6 6
wx	1100 C 12	1101 D 13	1111 F 15	1110 E 14
$w\bar{x}$	1000 8 8	1001 9 9	1011 B 11	1010 A 10

Note that this is still an SOP K-map.

Simplification of Boolean expressions in SOP form:

1. Place logical 1s in the appropriate cells.
2. Two or Four or Eight adjacent logical '1s' can be grouped together.

Example

The minterms on the K-map can be labeled as $f=\Sigma m(5, 7, 13, 15)$ in decimal, or $f= \Sigma m(5, 7, D, F)$ in hex.*

	$\bar{y}\bar{z}$	$\bar{y}z$	yz	$y\bar{z}$
$\bar{w}\bar{x}$	0000 0 0	0001 1 1	0011 3 3	0010 2 2
$\bar{w}x$	0100 4 4	0101 1 5 5	0111 1 7 7	0110 6 6
wx	1100 C 12	1101 1 D 13	1111 1 F 15	1110 E 14
$w\bar{x}$	1000 8 8	1001 9 9	1011 B 11	1010 A 10

z

Observe that the Σ notations (in either SOP or POS) completely describe the Boolean function mapped on the K-map, as long as one knows what the input variables are.

- **PROBLEM** : Reduce the following Boolean expression using K-map:

$$f = AB + AB'C + A'BC' + BC'$$

Soln. The given Boolean expression is **not in SOP form**.

$$f = AB(C + C') + AB'C + A'BC' + BC'(A + A')$$

$$= ABC + ABC' + AB'C + A'BC' + ABC' + A'BC'$$

$$= ABC + ABC' + AB'C + A'BC'$$

$$= \Sigma m(7, 6, 5, 2)$$

Ans. f = AC + BC'

- Simplify using 3-variable K-map

Self test

Problem 1 Reduce the Boolean expression

$f = \Sigma m\ (0, 2, 3, 4, 5, 6)$ using K-map and implement it in AOI logic.

Ans: $f = C' + AB' + A'B$

Problem 2 Reduce the expression $f= A'B'+A'B+AB$ using mapping

Ans: $f=A'+B$

KARNAUGH MAP (K – MAP)

- Don’t care terms

- The combinations for which the values of the expression are not specified are called don’t care combinations.

- The don’t care terms are denoted by d or X.

Ex1: Simplify the following Boolean expression using K Map.
F(A,B,C) = Σm(3, 4) + d(2,5,6)

A \ BC	00	01	11	10
0	0	0	1	X
1	1	X	0	X

F = AB’ + A’B

Summary

1. The K-map is a chart or a graph, composed of an arrangement of adjacent cells, each representing a particular combination of variables in sum of product form.
2. It is a means of showing the relationship between the logic inputs and desired output.
3. K-map is limited to 6 variables.
4. Any Boolean expression can be expressed in a standard or canonical or expanded sum (OR) of products (AND) form –SOP form—or in a standard or canonical or expanded product (AND) of sums (OR) form – POS form.

Exercise:

1.Consider the truth table of a function. Transfer the outputs to the K map and write the Boolean expression.

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

2. Simplify the following Boolean expressions using K maps.

$F = \Sigma m(0,2,4,6)$

$F = \Sigma m(0,2,4,6) + d(5,7)$.