Logic Design

1st class

أستاذة المادة: م.م جمانه وليد صالح

Contents

6- Product of Sum (POS).

Lectured Five: Karnaugh map

1- Three – variable karnaugh map.

2- Four – variable karnaugh map.

Lectured Six: Combinational Logic

1- The NAND Gate as a Universal Logic Element.

2- The NOR Gate as a Universal Logic Element.

3- Bit Parallel Adder.

4- Example.

Lectured Seven:

1- Decoders.

2-Encoders.

3- Multiplexer.

Lectured Eight: Flip-Flop

1- SR Flip-Flops.

2- D Flip-Flops.

3- JK Flip-Flops.

References

1- Computer System Architecture *Third Edition*

M. Morris Mano

2- Digital Fundamentals *Eight Edition*

FLOYD

Lectured One

: Number Systems Operation-1

1- Decimal Numbers.
2- Binary Numbers.
3- Octal Numbers.
4- Hexadecimal Numbers.

In the decimal number system each of the ten ***: Decimal Numbers-1***digits (10digits), 0 through 9 (0, 1, 2, 3, 4, 5, 6, 7, 8, and 9).

Decimal weight … …10^4 10^3 10^2 10^1 10^0. 10^{-1} 10^{-2} 10^{-3} ….

Example (1): **$(345)_{10}$**

$300+40+5=10^2*3+10^1*4+10^0*5=345= (345)_{10}$

3 4 5

Example (2): **$23.5 = (23.5)_{10}$**

$2*10^1 + 3*10^0 +5*10^{-1} = 20+3+0.5=23.5$

Where $10^0 =1$

2- Binary Numbers: The binary number system its two digits a base-two system. The two binary digits (bits) are 1 and 0 (1,0).

Binary weight 2^3 2^2 2^1 2^0

Weight value 8 4 2 1

A- Binary – to – Decimal Conversion:

***Binary number 1101101 where $2^0 =1$**

1 1 0 1 1 0 1

$2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0 = 2^6 *1+ 2^5*1+2^4*0+2^3*1+2^2*1+2^1*0+2^0*1$

$= 64+32+0+8+4+0+1=96+13=109$ □$(109)_{10}$

***The fractional binary number 0.1011**

0. 1 0 1 1

$2^{-1}\ 2^{-2}\ 2^{-3}\ 2^{-4} = 1*2^{-1}+ 0*2^{-2}+1*2^{-3}+1*2^{-4} =$

$0.5+0+0.125+0.0625=0.6875$ □ $(0.6875)_{10}$

B- Decimal – to – Binary Conversion:

1- Convert a decimal whole number to binary using the repeated division – by – 2 method.

2- Convert a decimal fraction to binary using the repeated Multiplication – by – 2 method.

Example (1):

Number $(58)_{10}$ ====□$(111010)_2$

2 58 mod LSB
2 29 ==0
2 14 ==1
2 7 == 0 ======= (111010)$_2$
2 3 ==1
2 1 ==1
0 ==1
MSB
Example (2):
Number (0.3125)$_{10}$ ======(0101)$_2$
MSB carry
0.3125*2
0 0.6250*2
1 0.2500*2
0 0.5000*2
1 0.0000
LSB
(0101)$_2$

3- Octal Numbers: The octal number system is composed of eight digits, which are 0, 1, 2, 3, 4, 5, 6, and 7.
To count above 7, begin another column and start over:
10, 11, 12, 13, 14, 15, 16, and 17.
20, 21, 22, 23, 24, 25, 26, and 27.
30, 31, … … … … …37.

A- Octal – to – Decimal conversion:
Weight … … 8^3 8^2 8^1 8^0
Octal number 2374 ==== (1276)$_{10}$
Example:
(2374)$_8$ = $2*8^3+3*8^2+7*8^1+4*8^0$
= 2*512+3*64+7*8+4*1
= 1024+192+56+4
= (1276)$_{10}$
B- Decimal – to – Octal Conversion:
Example:
Decimal number (359)$_{10}$ ======= (547)$_8$
8 359 mod LSB
8 44 ==7

8 5 ==□4 ====□(547)$_8$
0 ==□5
MSB

C- Octal – to – Binary Conversion:
Octal digit can be represented by a 3-bit binary number.
Octal digit binary
0 1 2 3 4 5 6 7
000 001 010 011 100 101 110 111
***Example*s:**
(25)$_8$ (140)$_8$
(2 5)$_8$ (1 4 0)$_8$
(010101)$_2$ (001100000)$_2$
D- Binary – to – Octal Conversion:
Conversion binary number to octal number is start with right – most group of three bits and moving from right to left.
***Example*s:**
(110101)$_2$ (101111001)$_2$
110 101 101 111 001
6 5 5 7 1
(6 5)$_8$ (5 7 1)$_8$
(65)$_8$ (571)$_8$

4- Hexadecimal Numbers: The hexadecimal number system has a base of sixteen; it is composed of 16 digits and alphabetic characters.

Decimal	Binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

A- __Binary – to – Hexadecimal conversion:__

4-bit groups, starting at the right-most bit.

Example: **$(1100101001010111)_2$ =======→ $(CA57)_{16}$**

1100 1010 0101 0111

C A 5 7

B- __Hexadecimal – to – Binary Conversion:__

Example: **$(10A4)_{16}$ =========→ $(1000010100100)_2$**

1 0 A 4

0001 0000 1010 0100

C- *__Hexadecimal – to –Decimal Conversion:__* **By to method**

*** First method:**

Example: **$(A85)_{16}$ ====→ $(2693)_{10}$**

1- **Convert to binary number.**

2- **Convert from binary number to decimal number.**

A 8 5

1010 1000 0101 =

$2^{11}*1+2^{10}*0+2^{9}*1+2^{8}*0+2^{7}*1+2^{6}*0+2^{5}*0+2^{4}*0+2^{3}*0+2^{2}*1+2^{1}*0+2^{0}*1=$

$2^{11}+2^{9}+2^{7}+2^{2}+2^{0}=2048+512+128+4+1=2693= (2693)_{10}$

*** Second method:**

Example: **$(E5)_{16}$ ========→ $(229)_{10}$**

$(E5)_{16}=E*16^{1}+5*16^{0}=14*16+5*1=224+5=229= (229)_{10}$

D- ***Decimal – to – Hexadecimal Conversion:***

Example: **Convert the decimal number 650 to hexadecimal by repeated division by 16.**

$(650)_{10}$ =====➔ $(28A)_{16}$

		Mod	
			LSD ↑
16	**650**		
16	**40**	======➔**A**	
16	**2**	======➔**8**	
	0	======➔**2**	
			MSD

MSD 2 8 A LSD = $(28A)_8$

Lectured Two

2-Binary Arithmetic:

1- Binary Addition.

2- Binary Subtraction.

3- Binary Multiplication.

4- Binary Division.

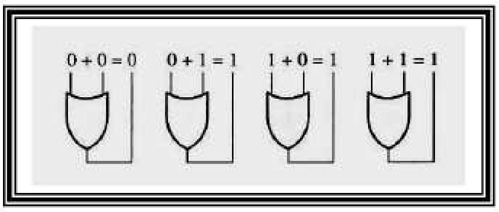
1- ***Binary Addition:*** The four basic rules for adding binary digits (bits) are as follows.

0+0=0 Sum of 0 with a carry 0

0+1=1 Sum of 1 with a carry 0

1+0=1 Sum of 1 with a carry 0

1+1=1 0 Sum of 0 with a carry 1



Examples:

110	**6**	**111**	**7**
+ 100	**+4**	**+011**	**+3**
1010	**10**	**1010**	**10**

1111	**15**
+ **1100**	**+12**
11011	**27**

2- Binary Subtraction: The four basic rules for subtracting are as follows.

0-0=0

1-1=0

1-0=1

0-1=1 0-1 with a borrow of 1

Examples:

11	3	11	3	101	5
- 01	- 1	- 10	- 2	- 011	- 3
10	2	01	1	010	2

110	6	101101	45
- 101	- 5	- 001110	- 14
001	1	011111	31

3- *1's And 2's Complement of Binary Number:*

The 1's complement and the 2's complement of binary number are important because they permit the representation of negative numbers.

Binary Number **1 0 1 1 0 0 1 0** **0 1**

1'sComplement **0 1 0 0 1 1 0 1** **o**

1 0

2's Complement of a binary number is found by adding 1 to the LSB of the 1's Complement.

2's Complement= (1's Complement) +1

Binary number	**10110010**
1'scomplement	**01001101**
Add 1	**+ 1**
2's complement	**01001110**

In decimal number complement such as:

0====➔9

7====➔2

6====➔3

9====➔0

4====➔5

1====➔8

<u>Signed Numbers:</u> Signed binary number consists of both sign and magnitu
information.

The sign bit → **0 positive numbers**

The sign bit → **1 negative numbers**

00011001

0   0011001

sign bit (0)   **magnitude bits** (0011001)

<u>Example:</u> **Express the decimal number - 39 as an 8-bit number in the sign-magnitude, 1's complement, and 2's complement forms.**

Solution:

1- **Write the 8-bit number for +39** **00100111**
2- **1's complement** **11011000**
3- **Add 1** **1**

1 1011001 = - 39

sign bit negative (the first 1)

4- Hexadecimal Addition & Subtraction:

Hexadecimal Addition:

2A7	**2AB**	**2B**
+ 317	**+317**	**+ 84**
5BE	**5C2**	**AF**

Hexadecimal subtraction:

CA2	**47C**
- A1B	**- 2BE**
287	**1BE**

5- Octal Addition & Subtraction:

325	**247**	**325**
+ 117	**+ 123**	**- 117**
444	**372**	**206**

Binary Coded Decimal (BCD):

Binary coded decimal means that each decimal digit, 0 through 9, is represented by a binary code of four bits.

The 8 4 2 1 (BCD) Code:

The 8 4 2 1 code is a type of (BCD) code. The 8 4 2 1 indicates the binary weights of the four bits (2^3, 2^2, 2^1, 2^0).

6 -*<u>The Gray Code:</u>*

Example:

Convert binary to Gray

10110 Binary ====➔1+ 0+1+1+ 0

11101 Gray ====➔ 1 1 1 0 1

Convert Binary to Gray

11011 Gray ====➔1 1 0 1 1

+ + + +

10010 Binary ====➔1 0 0 1 0

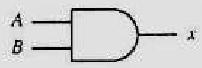
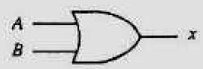
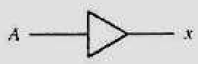
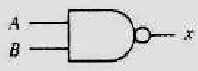
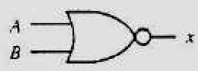
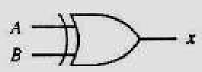
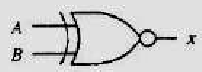
Decimal	Binary	Gray
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	.	.
.	.	.
.	.	.

7- Excess^{-3} Code: Addition three to any number in decimal number or binary number such as in table.

Decimal	BCD	Excess^{-3}	Excess^{-3}Gray
0	0000	0011	0010
1	0001	0100	0110
2	0010	0101	0111
3	0011	0110	0101
4	0100	0111	0100
5	0101	1000	1100
6	0110	1001	1101
7	0111	1010	1111
8	1000	1011	1110
9	1001	1100	1010
.	.	.	.
.	.	.	.
.	.	.	.

Lectured Three

Logic Gats: 1- Set of Gets

Name	Graphic symbol	Algebraic function	Truth table
AND		$x = A \cdot B$ or $x = AB$	A B | x 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1
OR		$x = A + B$	A B | x 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1
Inverter		$x = A'$	A | x 0 | 1 1 | 0
Buffer		$x = A$	A | x 0 | 0 1 | 1
NAND		$x = (AB)'$	A B | x 0 0 | 1 0 1 | 1 1 0 | 1 1 1 | 0
NOR		$x = (A + B)'$	A B | x 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 0
Exclusive-OR (XOR)		$x = A \oplus B$ or $x = A'B + AB'$	A B | x 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 0
Exclusive-NOR or equivalence		$x = (A \oplus B)'$ or $x = A'B' + AB$	A B | x 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 1

<u>*2- Half – Adder:*</u> The basic digital arithmetic circuit is the addition of two binary digits. Input variables of a half-adder call augends & addend bits. The output variables the sum & carry.

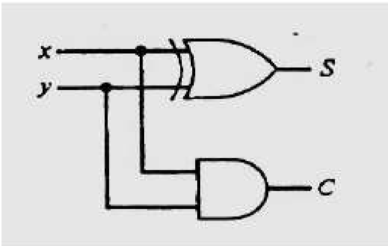


Figure (1-a) Logic diagram for half adder

X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Figure (1-b) Truth table for half adder

<u>Half- Adder questions:</u>

$S=\overline{X}Y+X\overline{Y}$

S=X (+)Y

C=X*Y

<u>*3-Full-Adder:*</u> A full - adder is a combinational circuit that forms the arithmetic sum of three input bits. It consists of three inputs &two outputs.

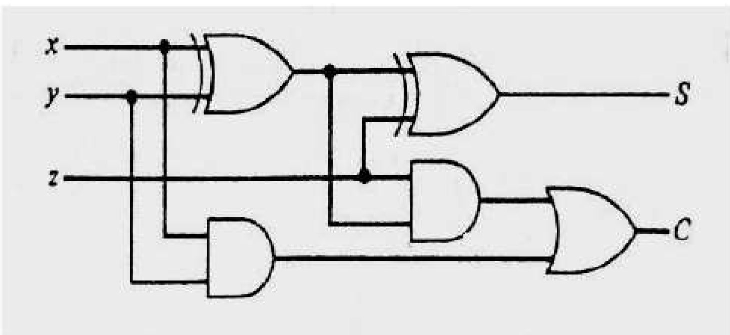


Figure (2-a) Logic diagram for full adder

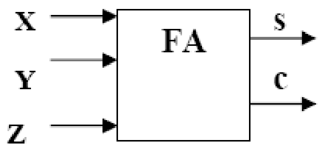


Figure (2-b) Block diagram for full adder

Inputs			Out puts	
X	Y	Z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0		1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

Figure (2-c) Truth table for full adder

Full - Adder questions:

S=x (+) y (+) z

C=XY+ (XZ (+) YZ)

C=X*Y+ (X (+) Y) Z

Lecture Four

Boolean Algebra &Logic Simplification:

1-Rules of Boolean algebra:

1- $A+0=A$
2- $A+1=1$
3- $A*0=0$
4- $A*1=A$
5- $A+A=A$
6- $A+\overline{A}=1$
7- $A*A=A$
8- $A*\overline{A}=0$
9- $\overline{\overline{A}}=A$ ======➔**Demoragan's theorems**
10- $A+BA=A$
11- $A+\overline{A}B=A+B$
12- $(A+B)(A+C)=A+BC$

2- Examples:

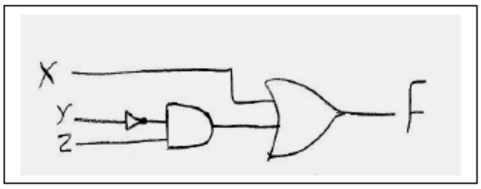
Example 1:

F = X + ý

Determine the truth table and logic diagram

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

Truth table

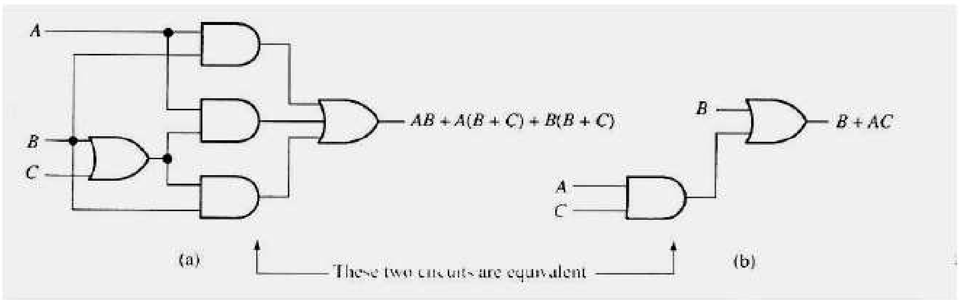


Logic diagram

Example 2:

AB+ A (B+C)+ B(B+C)

1. **AB+AB+AC+BB+BC**
2. **AB+AB+AC+B+BC**
3. **AB+AC+B+BC**
4. **AB+AC+B**
5. **B+AC**



Example 3:

F=ABC+ABĆ+ĂC

F= AB(C+Ć) +ĂC

F= AB+ĂC

Example 4:

Simplify the following Boolean expression:

$$\overline{AB + AC} + \overline{A}\,\overline{B}C$$

Solution **Step 1.** Apply DeMorgan's theorem to the first term.

$$(\overline{AB})(\overline{AC}) + \overline{A}\,\overline{B}C$$

Step 2. Apply DeMorgan's theorem to each term in parentheses.

$$(\overline{A} + \overline{B})(\overline{A} + \overline{C}) + \overline{A}\,\overline{B}C$$

Step 3. Apply the distributive law to the two terms in parentheses.

$$\overline{A}\,\overline{A} + \overline{A}\,\overline{C} + \overline{A}\,\overline{B} + \overline{B}\,\overline{C} + \overline{A}\,\overline{B}C$$

Step 4. Apply rule 7 ($\overline{A}\,\overline{A} = \overline{A}$) to the first term, and apply rule 10 [$\overline{A}\,\overline{B} + \overline{A}\,\overline{B}C = \overline{A}\,\overline{B}(1 + C) = \overline{A}\,\overline{B}$] to the third and last terms.

$$\overline{A} + \overline{A}\,\overline{C} + \overline{A}\,\overline{B} + \overline{B}\,\overline{C}$$

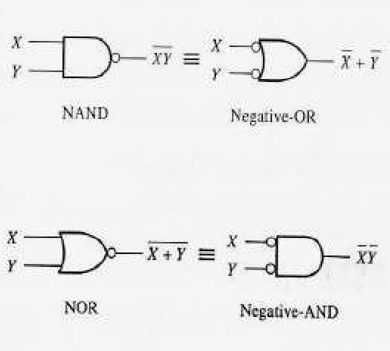
Step 5. Apply rule 10 [$\overline{A} + \overline{A}\,\overline{C} = \overline{A}(1 + \overline{C}) = \overline{A}$] to the first and second terms.

$$\overline{A} + \overline{A}\,\overline{B} + \overline{B}\,\overline{C}$$

Step 6. Apply rule 10 [$\overline{A} + \overline{A}\,\overline{B} = \overline{A}(1 + \overline{B}) = \overline{A}$] to the first and second terms.

$$\overline{A} + \overline{B}\,\overline{C}$$

3- Demorgan's theorems:



Inputs		Output	
X	Y	$\overline{XY}$	$\overline{X}+\overline{Y}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

Inputs		Output	
X	Y	$\overline{X+Y}$	$\overline{X}\,\overline{Y}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

Demorgan's theorems

4- Example:

Example 1:

a- $\overline{\overline{(A+B)}+\overline{C}} = \overline{\overline{(A+B)}}\ \overline{\overline{C}} = (A+B)C$

b- $\overline{(\overline{A}+B)+CD} = \overline{(\overline{A}+B)}\ \overline{CD} = (A\overline{B})\ (\overline{C}+\overline{D}) = A\overline{B}\,(\overline{C}+\overline{D})$

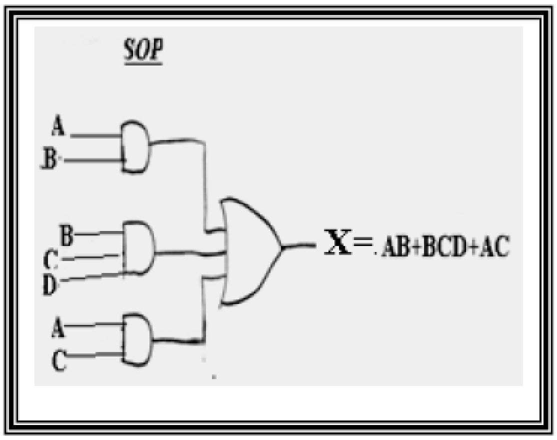
c- $\overline{(A+B)\overline{C}\ \overline{D}+E+\overline{F}} = (\overline{(A+B)\overline{C}\ \overline{D}})\ (\overline{E+\overline{F}})$

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

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

5- Sum – Of – Products (SOP):

X=AB+BCD+AC



Example:

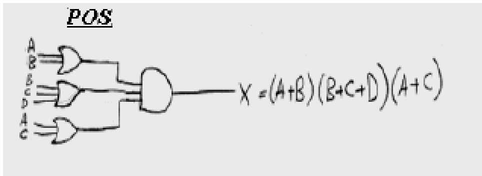
a- AB+B(CD+EF)=AB+BCD+BEF

b- (A+B)(B+C+D)=AB+AC+AD+BB+BC+BD

c- $\overline{\overline{(A+B)}+C} = \overline{\overline{(A+B)}}*\overline{C} = (A+B)\overline{C} = A\overline{C}+B\overline{C}$

6- Product – Of – Sum(POS):

(A+B)(B+C+D)(A+C)



Example: SOP

A	B	X	F
0	0	0	
0	1	1	$\bar{A}$ B
1	0	1	A $\bar{B}$
1	1	0	

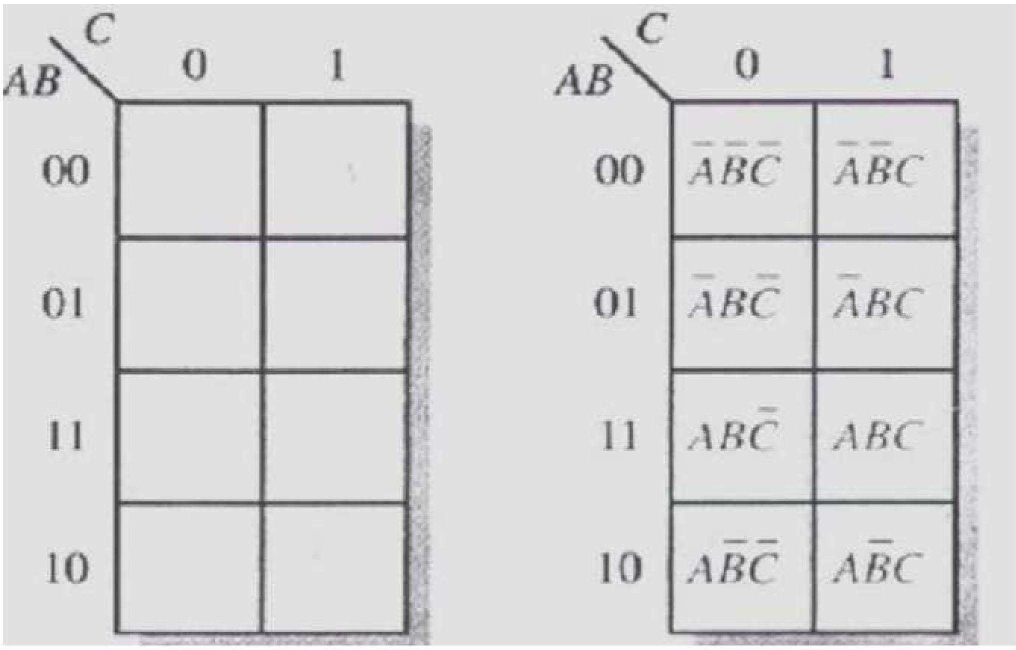
Example: POS

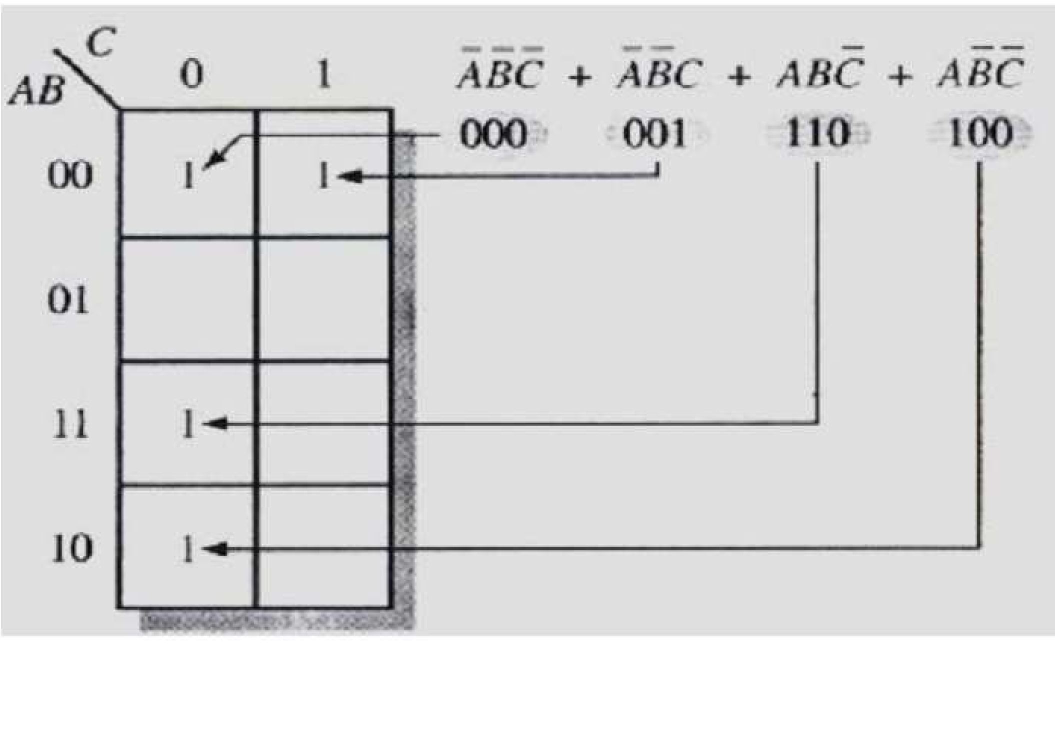
A	B	X	F
0	0	0	$\bar{A}+\bar{B}$
0	1	1	
1	0	1	
1	1	0	A+ B

Lectured Five

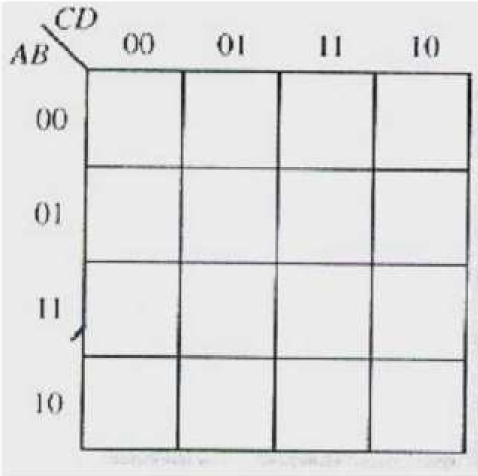
Karnaugh map:

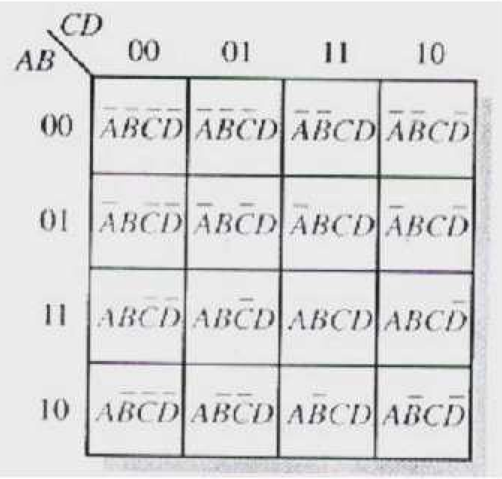
1- Three – variable karnaugh map.

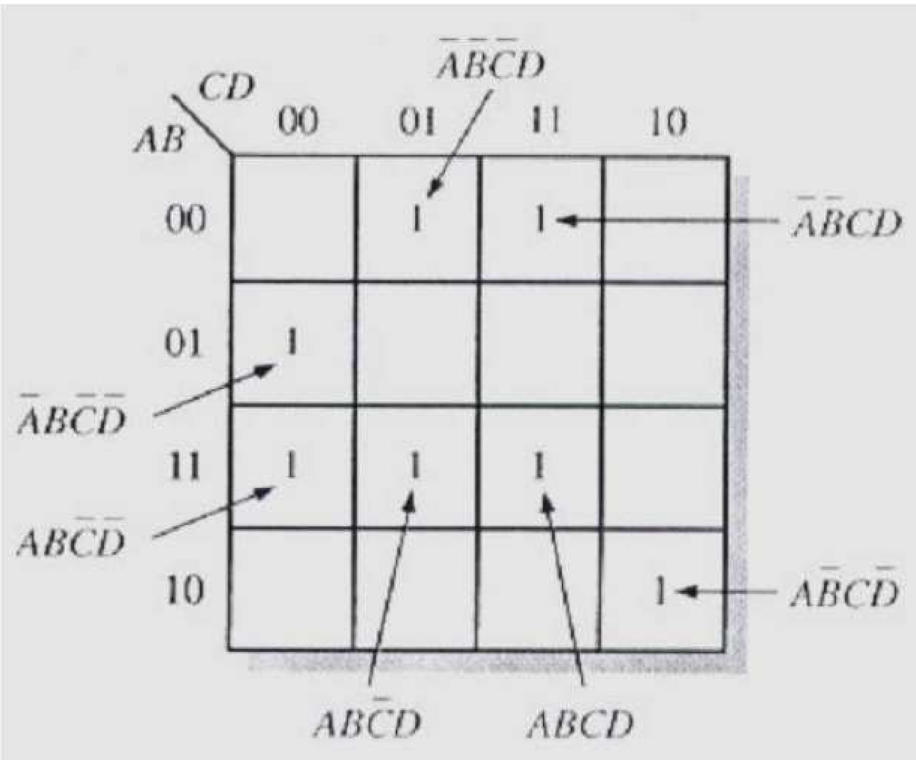




2- Four – variable karnaugh map.







Lectured Six

Combinational Logic:

1-The NAND Gate as a Universal Logic Element:

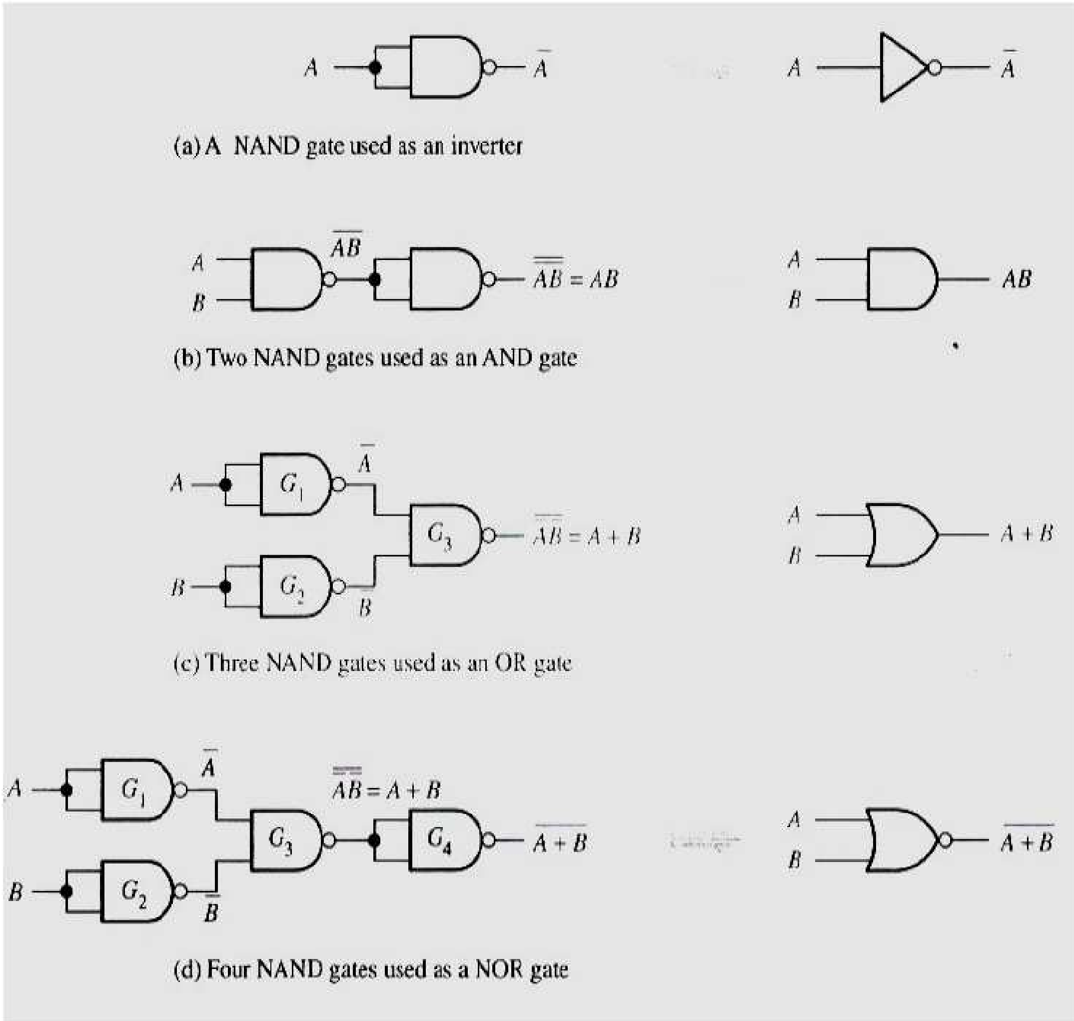


Figure (3) NAND Gates

2-The NOR Gate as a Universal Logic Element:

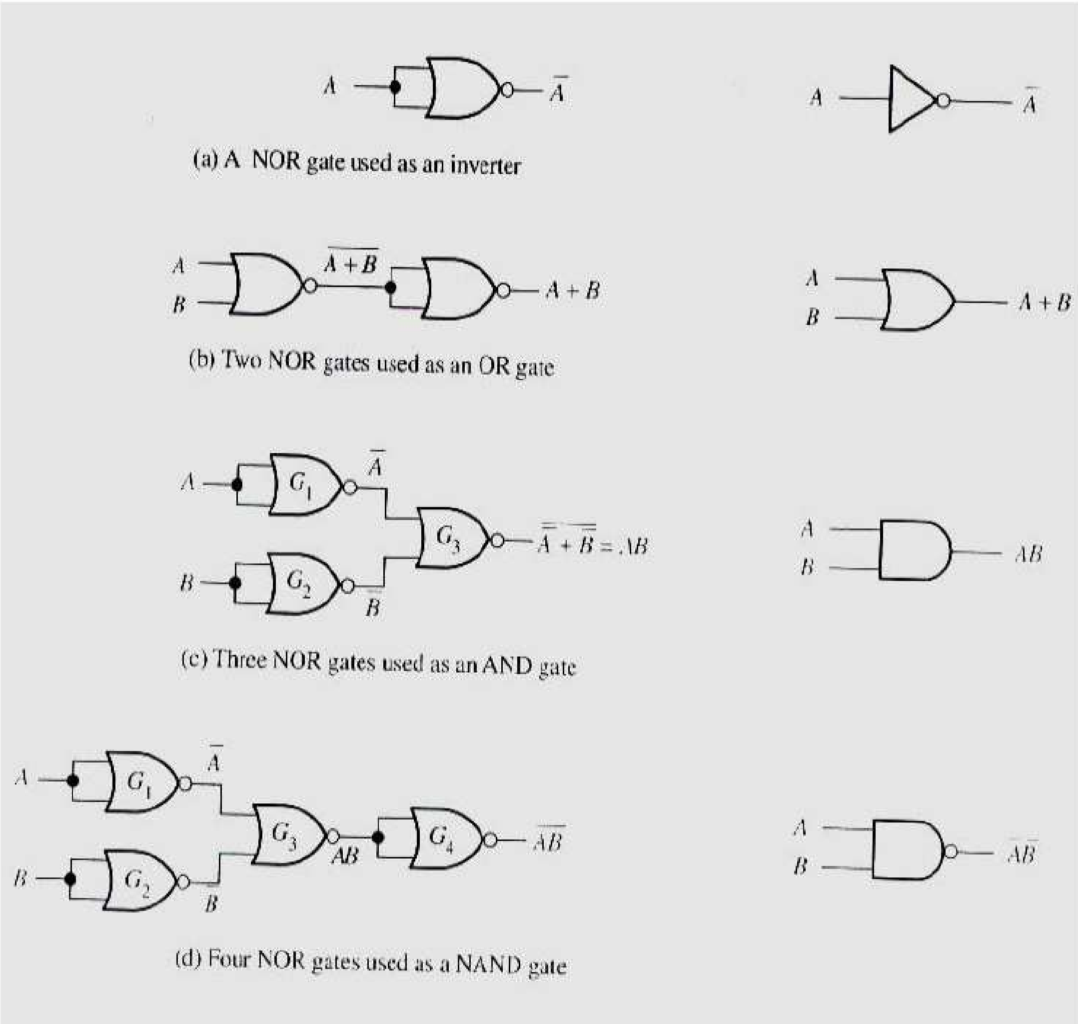


Figure (4) NOR Gates

3- Bit Parallel Adder:

A group of four bits is a nibble. A basic 4-bit parallel adder i implementation with four full adder stages.

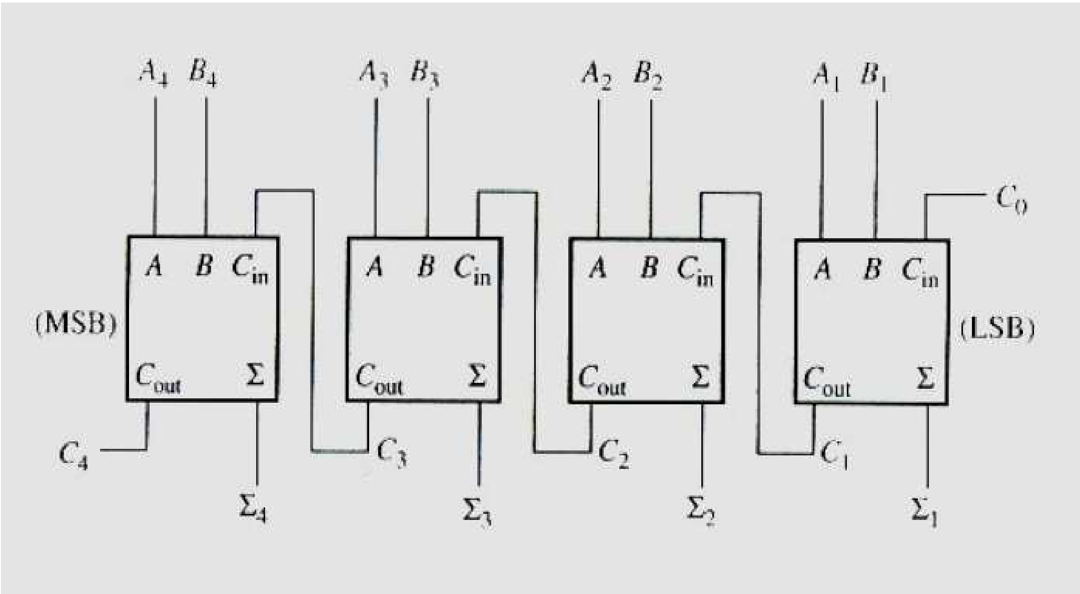


Figure (5) 4-bit parallel adder

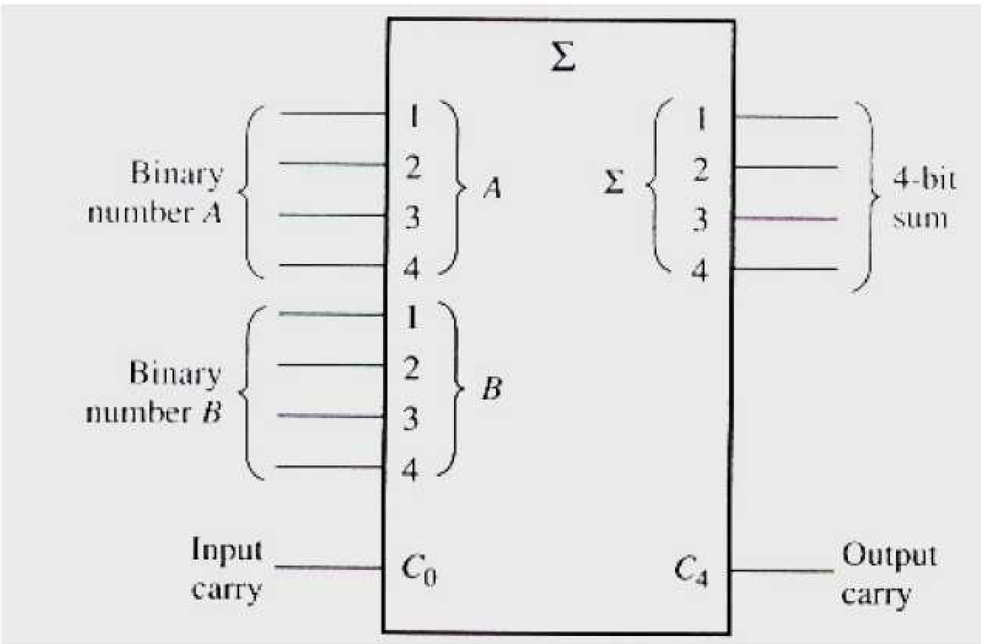


Figure (6) Symbol Logic

4- Example:

Draw the 4-bit parallel adder, find the sum and output carry for the addition of the following two 4-bit numbers if the input carry (C_{n-1}) is 0:

A4A3A2A1=1010 and B4B3B2B1=1011

<u>Solution:</u>

For n=1

A1=0, B1=1, C_{n-1}=0

$\sum$ =1, and C1=0

For n=2

A2=1, B2=1, C_{n-1}=0

$\sum$=0, and C2=1

For n=3

A3=0, B3=0, C_{n-1}=1

$\sum$=1, and C3=0

For n=4

A4=1, B4=1, C_{n-1}=0

$\sum$=0, and C4=1

Lectured Seven

Decoders & encoders:

1- Decoder:

A decoders is combinational circuit that converts binary information form the n coded inputs to a maximum of 2^n unique outputs.

That decoders are called n-to-m line decoders where m <=2^n.

The logic diagram of a 3-to-8 line decoder is three data inputs,A0,A1,and A2 are decoded into eight out puts, each out puts representing one of the combinations of the three binary input variables.

This decoder is a binary – to – octal conversion.

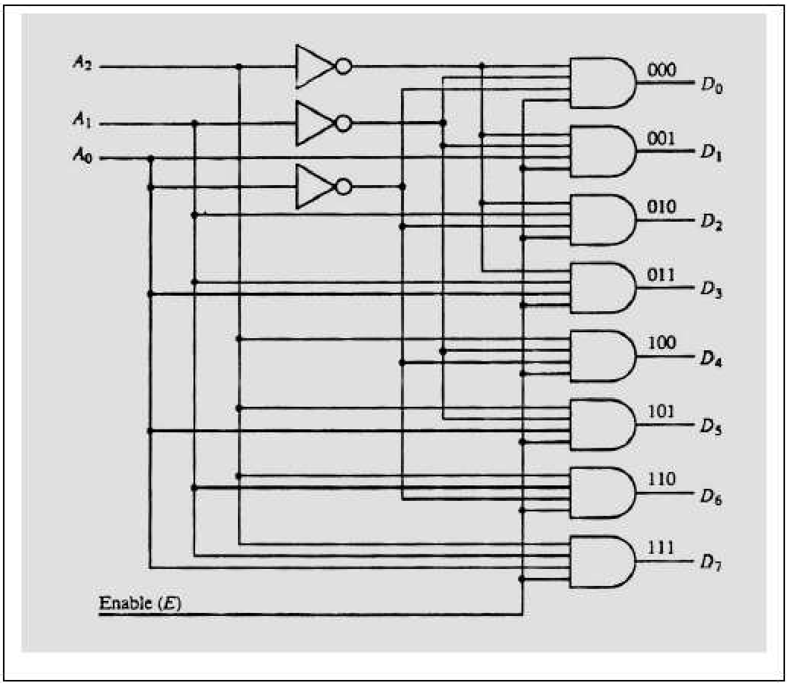


Figure (7) 3-to-8 line decoder

Enable	Inputs			Outputs							
E	**A2**	**A1**	**A0**	**D7**	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**
0	X	X	X	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0		0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0

Truth table for 3-to-8 line decoder

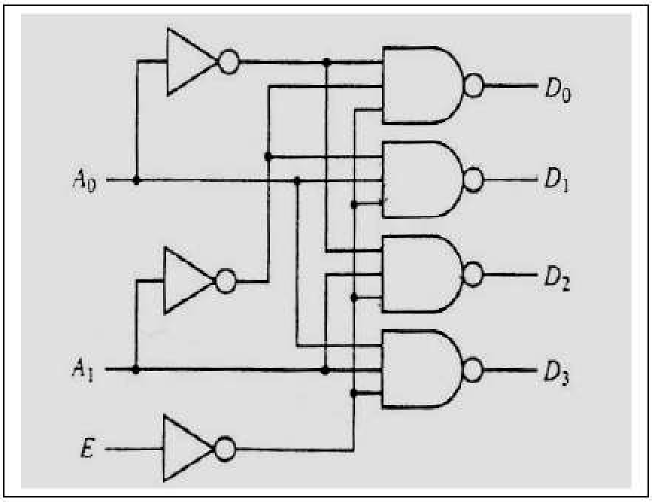


Figure (8) 2-to-4 line decoder

Enable	Inputs		Outputs			
E	A1	A0	D0	D1	D2	D3
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0
0	X	X	1	1	1	1

Truth table for 2-to-4 line decoder

2- Encoder:

An encoder is a digit circuit that performs the inverse operation of a decoder. An encoder has 2^n (or less) input lines and n output lines. An encoder is the octal – to – binary encoder.

It has eight inputs, one for each of the octal digits, and three outputs that generate the corresponding binary number.

Ao = D1+D3+D5+D7

A1 = D2+D3+D6+D7

A2 = D4+D5+D6+D7

(Implementation in three OR gates)

Inputs								Outputs		
D7	D6	D5	D4	D3	D2	D1	D0	A2	A1	A0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

Truth table for octal – to – binary encoder

3- Multiplexers:

A multiplexer is a combinational circuit that receiver binary information form one of 2^n input data lines and directs it to a single out put line.

The selection of a particular input data line for the output is determined by a set of selection inputs. A 2^n- to- 1 , A 4-to-1. Multiplexer is called Data Selector.

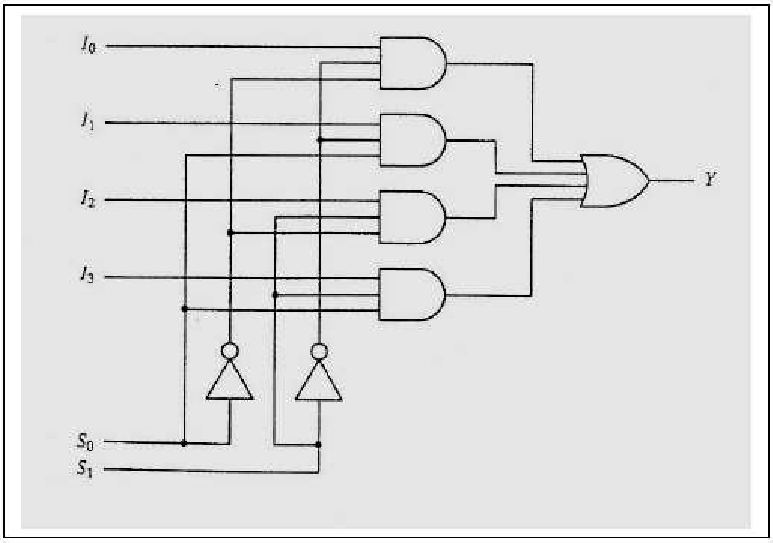


Figure (9) 4-to-1 line multiplexer

Inputs		Outputs
0	0	Y1
0	1	Y2
1	0	Y3
1	1	Y4

Truth table for 4-to-1 multiplexer

Lectured Eight

Flip-Flop:

The storage elements employed in clocked sequential circuits are called flip-flops. A flip-flops is a binary cell capable of storing one bit of information. It has two outputs, one for the normal value and one for the complement value of the bit stored in it.

Type of flip-flops:

1- **SR flip-flops.**
2- **D flip-flops.**
3- **JK flip-flops.**

1- **SR FLIP-FLOPS:**

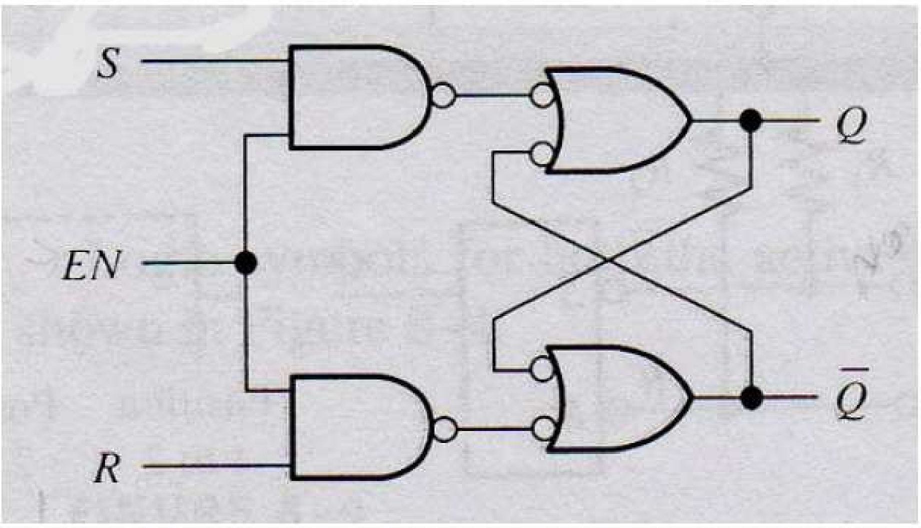


Figure (10) Logic diagram for SR flip-flop

Inputs		Outputs		
S	R	Q	$\bar{Q}$	Comments
0	0	1	1	Invalid condition
0	1	1	0	Latch set
1	0	0	1	Latch reset
1	1	N.C	N.C	No change

Truth table for SR flip-flop

2- D FLIP-FLOPS:

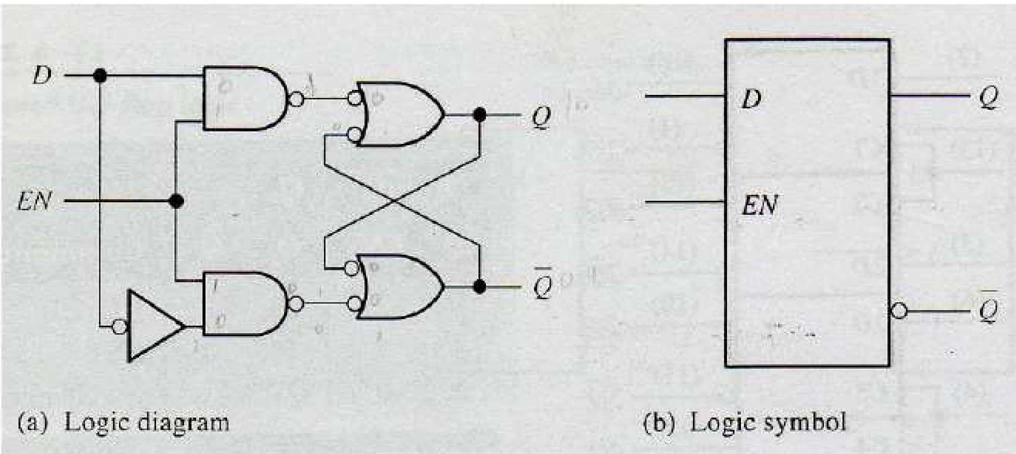


FIGURE (11) D Flip-flop

Inputs		Outputs		
D	CLK	Q	$\bar{Q}$	Comments
1	↑	1	0	Set(stor1)
0	↑	0	1	Reset(stor0)

Truth table for D flip-flop

3- **<u>J K FLIP-FLOPS:</u>**

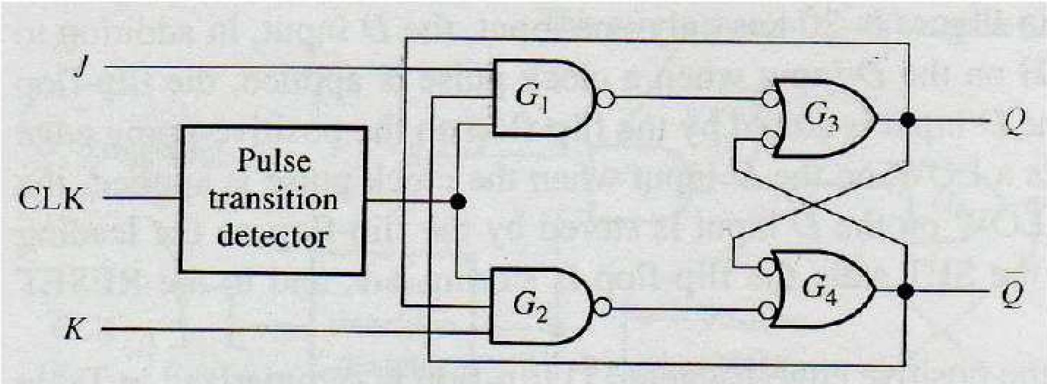


Figure (12) JK Flip-flop

Inputs			Outputs		
J	K	CLK	Q	$\bar{Q}$	Comments
0	0	↑	Q0	$\bar{Q}0$	No change
0	1	↑	0	1	Reset
1	0	↑	1	0	Set
1	1	↑	$\bar{Q}0$	Q0	Toggle

Truth table for JK flip-flop