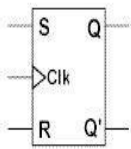
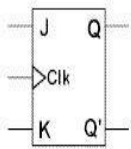
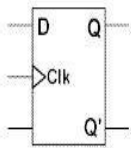
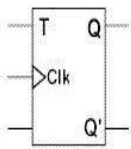


Chapter One

1- Background to the Flip Flop

A flip-flop is a binary storage device capable of storing one bit of information. In a stable state, the output of a flip-flop is either 0 or 1

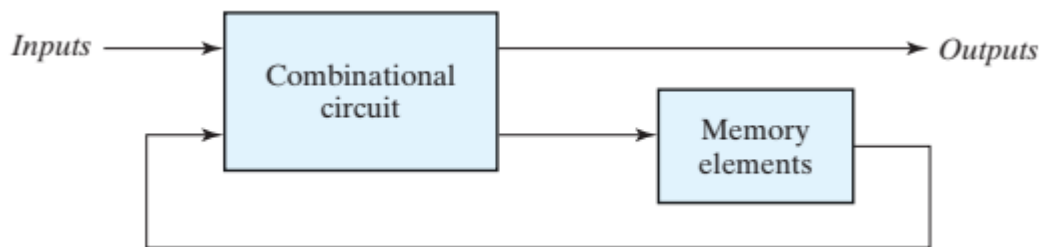
A change in state of the flip-flops is initiated only by a clock pulse transition

FLIP-FLOP NAME	FLIP-FLOP SYMBOL	CHARACTERISTIC TABLE	CHARACTERISTIC EQUATION	EXCITATION TABLE																																			
SR		<table><tr><th>S</th><th>R</th><th>Q_(next)</th></tr><tr><td>0</td><td>0</td><td>Q</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>?</td></tr></table>	S	R	Q _(next)	0	0	Q	0	1	0	1	0	1	1	1	?	$Q_{(next)} = S + R'Q$ $SR = 0$	<table><tr><th>Q</th><th>Q_(next)</th><th>S</th><th>R</th></tr><tr><td>0</td><td>0</td><td>0</td><td>X</td></tr><tr><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>X</td><td>0</td></tr></table>	Q	Q _(next)	S	R	0	0	0	X	0	1	1	0	1	0	0	1	1	1	X	0
S	R	Q _(next)																																					
0	0	Q																																					
0	1	0																																					
1	0	1																																					
1	1	?																																					
Q	Q _(next)	S	R																																				
0	0	0	X																																				
0	1	1	0																																				
1	0	0	1																																				
1	1	X	0																																				
JK		<table><tr><th>J</th><th>K</th><th>Q_(next)</th></tr><tr><td>0</td><td>0</td><td>Q</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>Q'</td></tr></table>	J	K	Q _(next)	0	0	Q	0	1	0	1	0	1	1	1	Q'	$Q_{(next)} = JQ' + K'Q$	<table><tr><th>Q</th><th>Q_(next)</th><th>J</th><th>K</th></tr><tr><td>0</td><td>0</td><td>0</td><td>X</td></tr><tr><td>0</td><td>1</td><td>1</td><td>X</td></tr><tr><td>1</td><td>0</td><td>X</td><td>1</td></tr><tr><td>1</td><td>1</td><td>X</td><td>0</td></tr></table>	Q	Q _(next)	J	K	0	0	0	X	0	1	1	X	1	0	X	1	1	1	X	0
J	K	Q _(next)																																					
0	0	Q																																					
0	1	0																																					
1	0	1																																					
1	1	Q'																																					
Q	Q _(next)	J	K																																				
0	0	0	X																																				
0	1	1	X																																				
1	0	X	1																																				
1	1	X	0																																				
D		<table><tr><th>D</th><th>Q_(next)</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table>	D	Q _(next)	0	0	1	1	$Q_{(next)} = D$	<table><tr><th>Q</th><th>Q_(next)</th><th>D</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	Q	Q _(next)	D	0	0	0	0	1	1	1	0	0	1	1	1														
D	Q _(next)																																						
0	0																																						
1	1																																						
Q	Q _(next)	D																																					
0	0	0																																					
0	1	1																																					
1	0	0																																					
1	1	1																																					
T		<table><tr><th>T</th><th>Q_(next)</th></tr><tr><td>0</td><td>Q</td></tr><tr><td>1</td><td>Q'</td></tr></table>	T	Q _(next)	0	Q	1	Q'	$Q_{(next)} = TQ' + T'Q$	<table><tr><th>Q</th><th>Q_(next)</th><th>T</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	Q	Q _(next)	T	0	0	0	0	1	1	1	0	1	1	1	0														
T	Q _(next)																																						
0	Q																																						
1	Q'																																						
Q	Q _(next)	T																																					
0	0	0																																					
0	1	1																																					
1	0	1																																					
1	1	0																																					

2- The sequential Machine

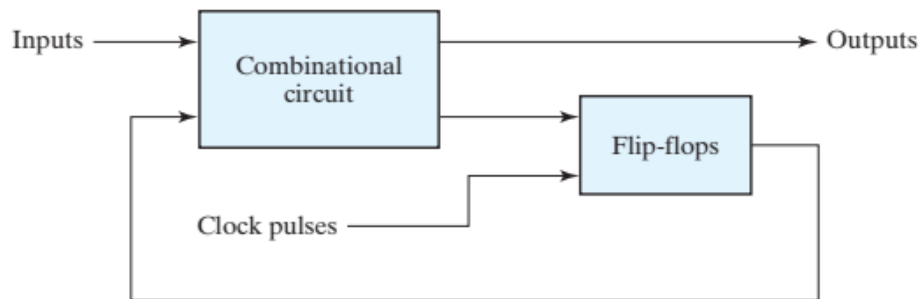
The sequential logic circuits are consist of a combinational circuit to which storage elements are connected to form a feedback path. The storage elements are devices capable of storing binary information. ,also they consist of :

- Synchronous or clocked sequential circuits
- An asynchronous logic circuits.



2-1 synchronous sequential circuits

In synchronous networks clocked flip- flops are used as memory elements, which change their individual states in synchronism with the periodic clock signal, the change in states of flip-flops and change in state of the entire circuit occurs at the transition of the clock signal.



(a) Block diagram



(b) Timing diagram of clock pulses

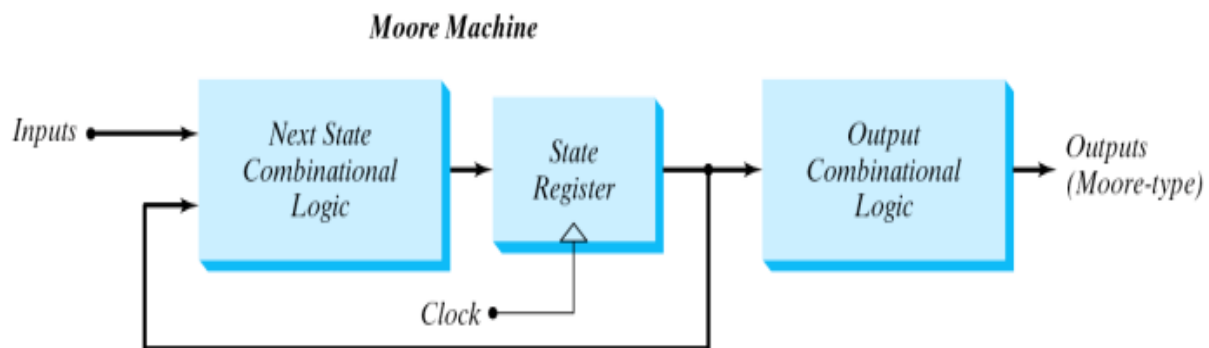
The synchronous or clocked sequential networks are represented by two models:

- **Moore model:** The output depends only on the present state of the flip-flops.
- **Mealy model:** The output depends on both the present state of the flip-flops and on the inputs.

1- Moore Model

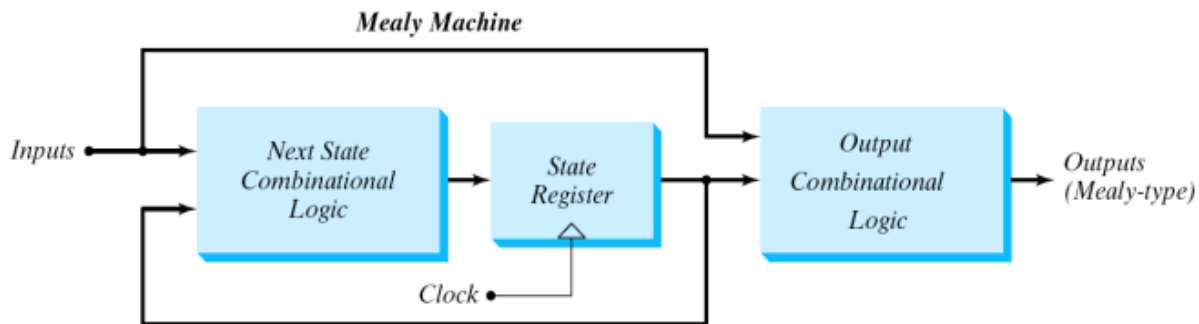
As mentioned earlier, when the output of the sequential network depends only on the present state of the flip-flop, the sequential network is referred to as Moore model.

In the Moore model, as output depends only on present state of flip-flops, it appears only after the clock pulse is applied, i.e. it varies in synchronism with the clock input.



2-Mealy Model

When the output of the sequential network depends on both the present state of flip-flops and on the inputs, the sequential circuit is referred to as Mealy model. In this type the output of the circuit is derived from the combination of present state of flip-flops and inputs of the circuit. The outputs may change if the inputs change during the clock pulse period



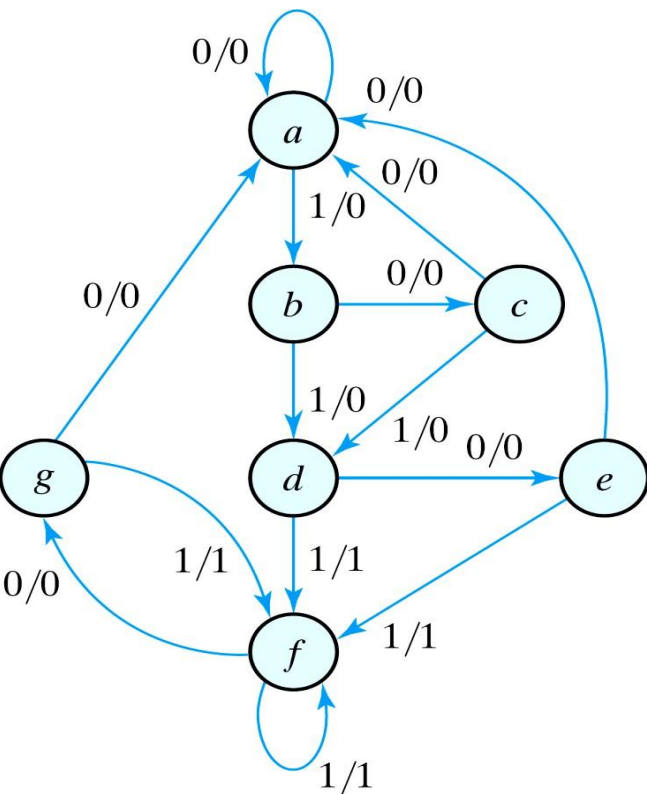
Sr. No.	Moore model	Mealy model
1.	Its output is a function of present state only.	Its output is a function of present state as well as present input.
2.	Input changes does not affect the output.	Input changes may affect the output of the circuit..
3.	Moore model requires more number of states for implementing same function.	It requires less number of states for implementing same function.

★ Design Procedure for sequential circuits:

- From the word description of the circuit behavior or from the state equation to get a state diagram.
- State table from the state diagram
- State reduction if necessary.
- Assign binary values to the states.
- Obtain the binary-coded state table.
- Choose the type of flip-flops.
- Derive the simplified flip-flop input equations and output equations.
- Draw the logic diagram.

Example : Design the sequential circuit that has the specification given in the state diagram shown, for the input sequence 01010110100 starting from the initial state a.

1- State diagram



2- State table

State Table

Present State	Next State		Output	
	x = 0	x = 1	x = 0	x = 1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	g	f	0	1
g	a	f	0	1

3- Reducing the state table:

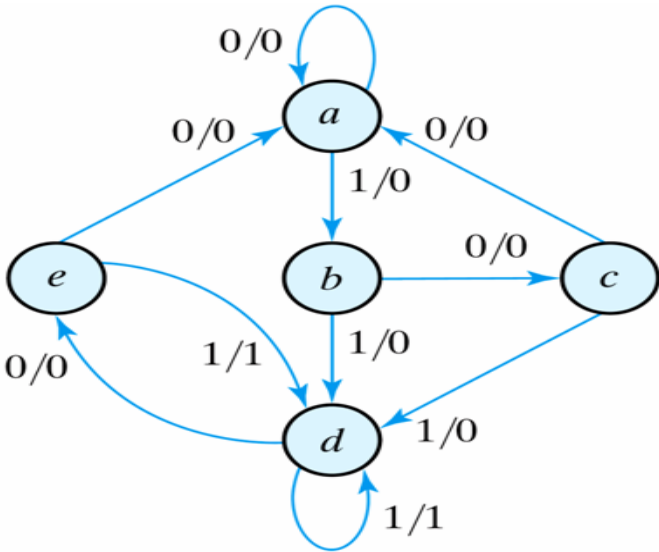
- a. e = g (remove g and keep e)
- b. d = f (remove f and keep d)

Reducing the State Table

Present State	Next State		Output	
	x = 0	x = 1	x = 0	x = 1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	e	f	0	1

Reduced State Table

Present State	Next State		Output	
	x = 0	x = 1	x = 0	x = 1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	d	0	1
e	a	d	0	1



"State diagram after reduction"

4- State Assignment:*Reduced State Table with Binary Assignment 1*

Present State	Next State		Output	
	$x = 0$	$x = 1$	$x = 0$	$x = 1$
000	000	001	0	0
001	010	011	0	0
010	000	011	0	0
011	100	011	0	1
100	000	011	0	1

Guidelines for State Assignment:

1. States which have the same next state for a given input should be given adjacent assignments.
2. States which are the next states of the same state should be given adjacent assignments.
3. States which have the same output for a given input should be given adjacent assignments.

Example : Selecting an Assignment.

	$X = 0$	1	$X = 0$	1
<i>a</i>	<i>a</i>	<i>c</i>	0	0
<i>b</i>	<i>d</i>	<i>f</i>	0	1
<i>c</i>	<i>c</i>	<i>a</i>	0	0
<i>d</i>	<i>d</i>	<i>b</i>	0	1
<i>e</i>	<i>b</i>	<i>f</i>	1	0
<i>f</i>	<i>c</i>	<i>e</i>	1	0

1. (*b, d*) (*c, f*) (*b, e*)
2. (*a, c*) (*d, f*) (*b, d*) (*b, f*) (*c, e*)
3. (*a, c*) (*b, d*) (*e, f*)

$Q_2Q_3 \backslash Q_1$		0	1
00		a	c
01			e
11		d	b
10			f

a = 000
b = 111
c = 100
d = 011
e = 101
f = 110

(b)

$Q_2Q_3 \backslash Q_1$		0	1
00		c	a
01			e
11		d	b
10		f	

a = 100
b = 111
c = 000
d = 011
e = 101
f = 010

(c)

$Q_1Q_2Q_3$	$Q_1^+Q_2^+Q_3^+$		$X = 0 \quad 1$	
	$X = 0$	1		
1 0 0	100	000	0	0
1 1 1	011	010	0	1
0 0 0	000	100	0	0
0 1 1	011	111	0	1
1 0 1	111	010	1	0
0 1 0	000	101	1	0

5- Choose the D type of flip-flops.

6- Derive the simplified flip-flop input equations and output equations.

7- Finally draw the logic diagram

$Q_2Q_3 \backslash XQ_1$		00	01	11	10
		00	01	11	10
00		0	1	0	1
01		X	1	0	X
11		0	0	0	1
10		0	X	X	1

$$Q_1^+ = D_1$$

$Q_2Q_3 \backslash XQ_1$		00	01	11	10
		00	01	11	10
00		0	0	0	0
01		X	1	1	X
11		1	1	1	1
10		0	X	X	0

$$Q_2^+ = D_2$$

$Q_2Q_3 \backslash XQ_1$		00	01	11	10
		00	01	11	10
00		0	0	0	0
01		X	1	0	X
11		1	1	0	1
10		0	X	X	1

$$Q_3^+ = D_3$$

$Q_2Q_3 \backslash XQ_1$		00	01	11	10
		00	01	11	10
00		0	0	0	0
01		X	1	0	X
11		0	0	1	1
10		1	X	X	0

Z

Example : Consider the state table given below:

Present state	Next state		Output (Z)	
	X=0	X=1	X=0	X=1
1	2	3	0	0
2	4	3	0	0
3	2	5	0	0
4	6	3	0	0
5	2	7	0	0
6	6	3	1	0
7	2	7	0	1

The state of adjacent states specified by guidelines 1&2 are:-

- 1. (1,3,5,7) (4,6) (1,2,4,6) (5,7)
- 2. (2,3) (3,4) (2,5) (3,6) (2,7)

state	Code from map(a)	Code from map (b)
1	000	000
2	101	010
3	010	101
4	111	110
5	110	111
6	011	100
7	100	011

	00	01	11	10
0	1	3	5	7
1		6	4	2

(a)

	00	01	11	10
0	1	2	4	6
1		7	5	3

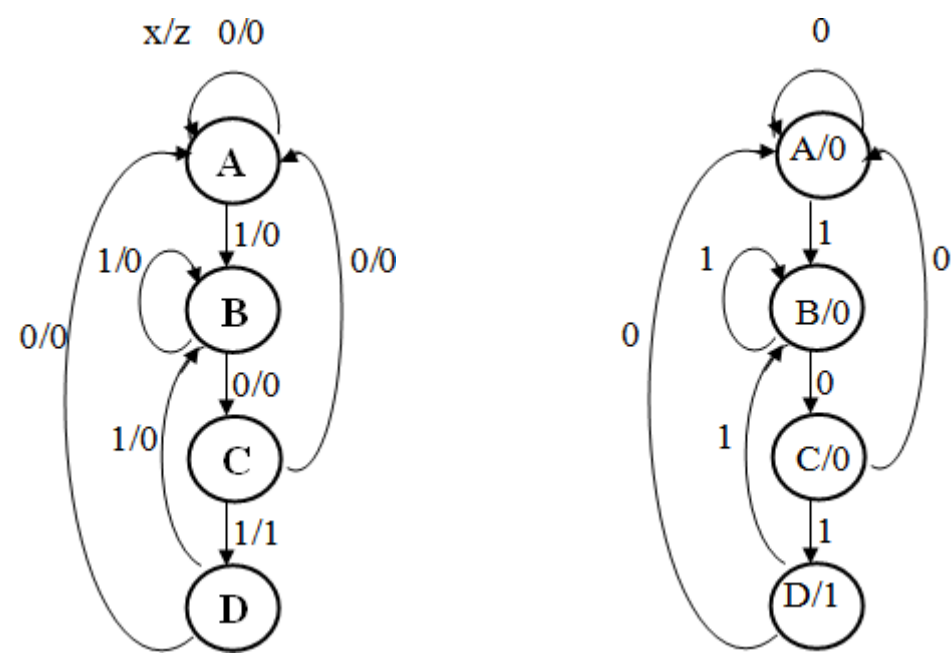
(b)

★ Designing of Synchronous State Machine

- Synchronous: all of the flip flops use the same clock signal

Example : Suppose a circuit is required to recognize the 3-bit pattern (101), and output (z=1) whenever it occurs in the continues serial input (X) to the circuit. then the circuit will reset and start testing again.

Solution:



Moore state diagram

State tables:

Mealy State table

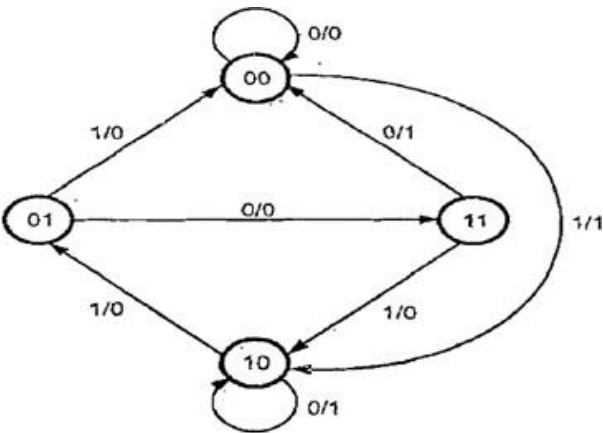
Present state	N.s X=0	N.S X=1	Output X=0	Output X=1
A	A	B	0	0
B	C	B	0	0
C	A	D	0	1
D	A	B	0	0

Moore State table

Present state	N.s X=0	N.S X=1	Output Z
A	A	B	0
B	C	B	0
C	A	D	0
D	A	B	1

Example: A sequential circuit has one input and one output. The state diagram is shown in Figure below .Design the sequential circuit with a) D flip-flops b) T flip- flops c) RS flip-flops and d) JK flip-flops.

1- State diagram



2- state table

Present state		Next state		Output	
		X = 0	X = 1	X = 0	X = 1
A	B	AB	AB	Y	Y
0	0	0 0	1 0	0	1
0	1	1 1	0 0	0	0
1	0	1 0	0 1	1	0
1	1	0 0	1 0	1	0

i) Design using D flip-flops

K-map simplification

AB \ X	0	1
00	0	1
01	1	0
11	0	1
10	1	0

(a) For flip - flop A

AB \ X	0	1
00	0	0
01	1	0
11	0	0
10	0	1

(b) For flip - flop B

AB \ X	0	1
00	0	1
01	0	0
11	1	0
10	1	0

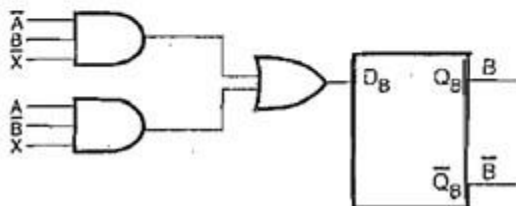
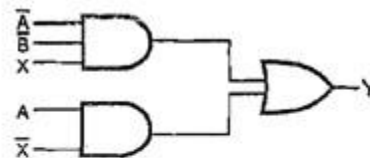
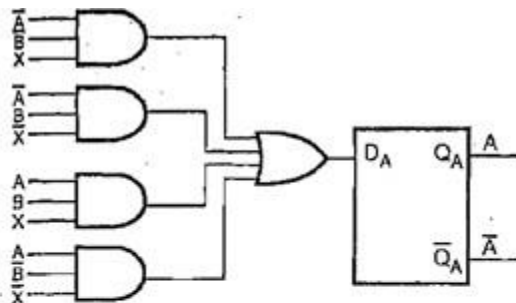
(c) For output

$$DA = \bar{A}\bar{B}X + \bar{A}BX + ABX + A\bar{B}\bar{X}$$

$$DB = \bar{A}\bar{B}\bar{X} + A\bar{B}\bar{X}$$

$$Y = \bar{A}\bar{B}X + A\bar{X}$$

"Logic diagram of given sequential circuit using D flip-flop"



ii) Design using T flip- flops :

Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

"Excitation table for T flip-flop"

Present state		Input	Next state		Flip-flop inputs		Output
A	B	X	A	B	T_A	T_B	Y
0	0	0	0	0	0	0	0
0	0	1	1	0	1	0	1
0	1	0	1	1	1	0	0
0	1	1	0	0	0	1	0
1	0	0	1	0	0	0	1
1	0	1	0	1	1	1	0
1	1	0	0	0	1	1	1
1	1	1	1	0	0	1	0

"Circuit Excitation Table"

K-map simplification

AB \ X	0	1
00	0	1
01	1	0
11	1	0
10	0	1

(a) For flip-flop A

AB \ X	0	1
00	0	0
01	0	1
11	1	1
10	0	1

(b) For flip-flop B

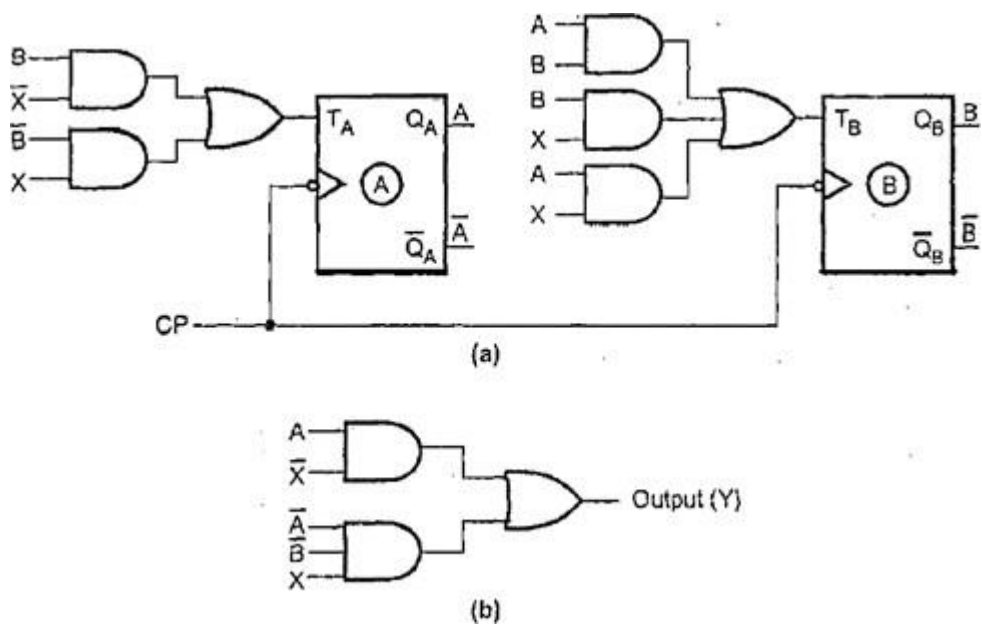
AB \ X	0	1
00	0	1
01	0	0
11	1	0
10	1	0

(c) For output

$$T_A = \bar{B}\bar{X} + \bar{B}XB =$$

$$AB + BX + AX$$

$$Y = \bar{A}\bar{B}X + A\bar{X}$$



a & b Logic diagram of given sequential circuit using T flip-flop"

iii) Design using RS flip-flops:

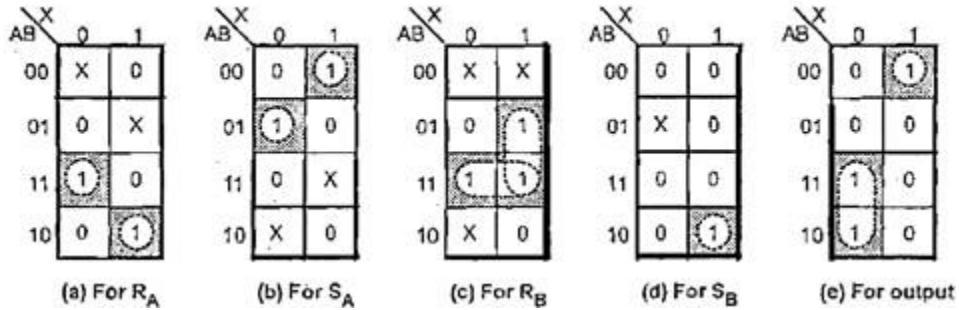
Q_n	Q_{n+1}	R	S
0	0	X	0
0	1	0	1
1	0	1	0
1	1	0	X

"Excitation table for RS flip-flop"

Present state		Input	Next state		Flip-flop inputs				Output
A	B	X	A	B	R_A	S_A	R_B	S_B	Y
0	0	0	0	0	X	0	X	0	0
0	0	1	1	0	0	1	X	0	1
0	1	0	1	1	0	1	0	X	0
0	1	1	0	0	X	0	1	0	0
1	0	0	1	0	0	X	X	0	1
1	0	1	0	1	1	0	0	1	0
1	1	0	0	0	1	0	1	0	1
1	1	1	1	0	0	X	1	0	0

Circuit excitation tale

K-map simplification



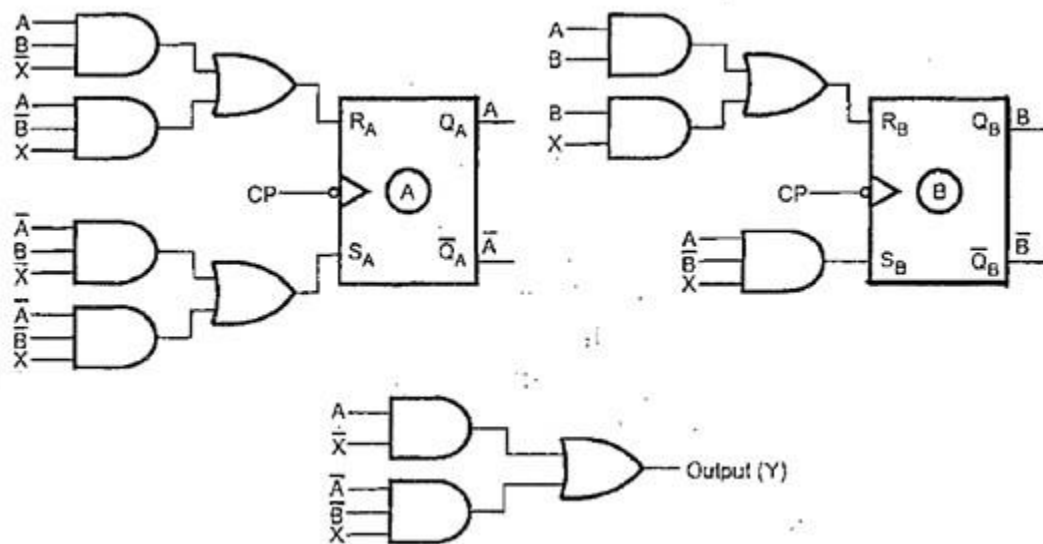
$$R_A = AB\bar{X} + \bar{A}\bar{B}X$$

$$= \bar{A}\bar{B}X + \bar{A}\bar{B}X R_B = A$$

$$B + BX$$

$$S_B = \bar{A}\bar{B}X$$

$$Y = \bar{A}\bar{B}X + A\bar{X}$$



"Logic diagram of given sequential circuit using RS flip-flop"

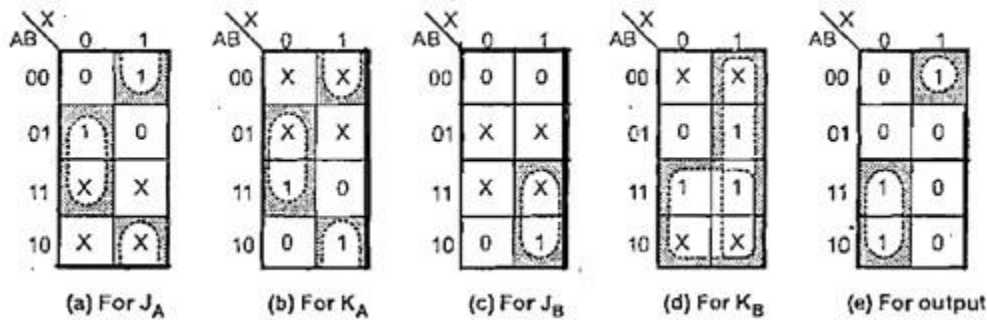
iv. Design using JK Flip-Flops:

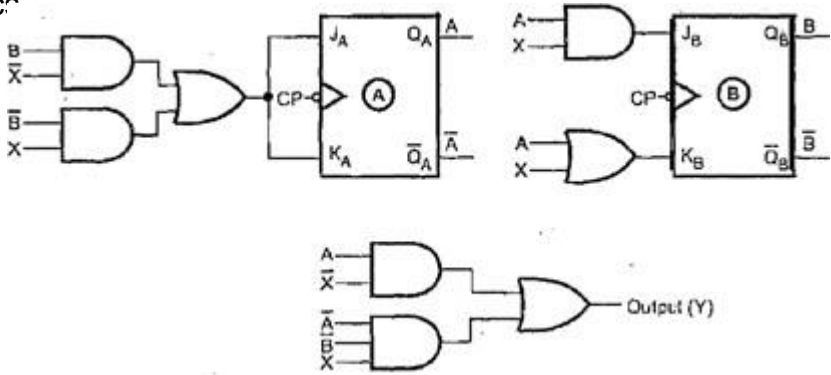
Q_n	Q_{n+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Present state		Input	Next state		Flip-flop inputs				Output
A	B	X	A	B	J_A	K_A	J_B	K_B	Y
0	0	0	0	0	0	X	0	X	0
0	0	1	1	0	1	X	0	X	1
0	1	0	1	1	1	X	X	0	0
0	1	1	0	0	0	X	X	1	0
1	0	0	1	0	X	0	0	X	1
1	0	1	0	1	X	1	1	X	0
1	1	0	0	0	X	1	X	1	1
1	1	1	1	0	X	0	X	1	0

"Circuit Excitation Table"

K-map simplification





"Logic diagram of given sequential circuit using JK flip-flop"

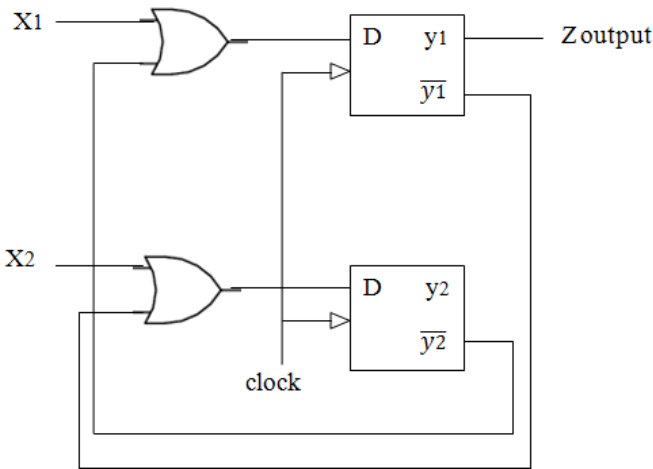
Analysis of Synchronous state machines

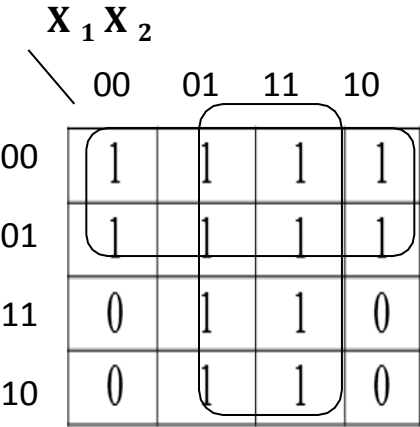
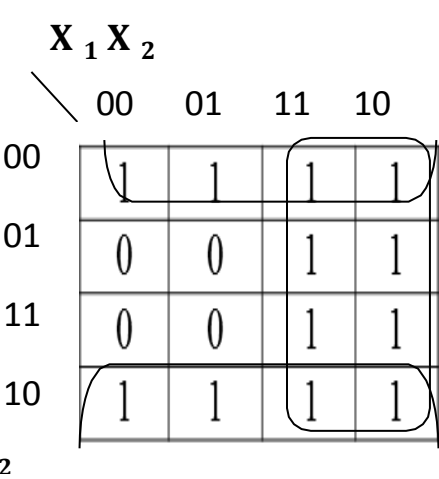
To obtain the excitation table for a given sequential circuit the characteristic equations shown in table below must be considered for the required type of flip- flops.

Flip Flop	Characteristics equation
SR	$Q^+ = S + \overline{R}Q$
Jk	$Q^+ = \overline{J} + KQ$
D	$Q^+ = D$
T	$Q^+ = T\overline{Q} + \overline{T}Q$ $= T + Q$

Example: Analyze the sequential circuit with inputs X1and X2 and output y which shown below, find: the related equations, state table and state diagram.

$Z = y_1$
 $Dy_1 = X_1 + \overline{y_2}$
 $Dy_2 = X_1 + \overline{y_1}$





$D y_1 = y_1^+ = X_1 + \bar{y}_2$

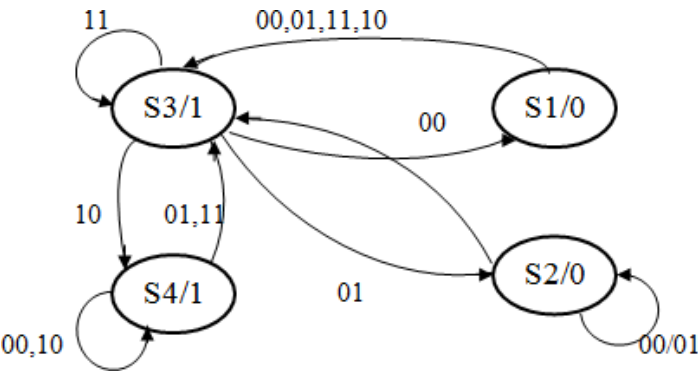
$D y_2 = y_2^+ = X_2 + \bar{y}_1$

Present state $y_1 y_2$	Next states $X_1 X_2$				Z output
	00 $y_1^+ y_2^+$	01	11	10	
00	11	11	11	11	0
01	01	01	11	11	0
11	00	01	11	10	1
10	10	11	11	10	1

Present state	Next state X_1X_2				Z output
	00	01	11	10	
S1	S3	S3	S3	S3	0
S2	S2	S2	S3	S3	0
S3	S1	S2	S3	S4	1
S4	S4	S3	S3	S4	1

S2	S2, S3		
S3	X	X	
S4	X	X	S1,S4
			S2,S3
	S1	S2	S3

No reduction



Example : a) give the state diagram for the logic circuit shown below
b) show how the circuit could be redesigned using D-type flip- flop.

a. $Jy_1 = X y_2$, $Ky_1 = X$

$Jy_2 = \bar{X}y_1$, $Ky_2 = X$

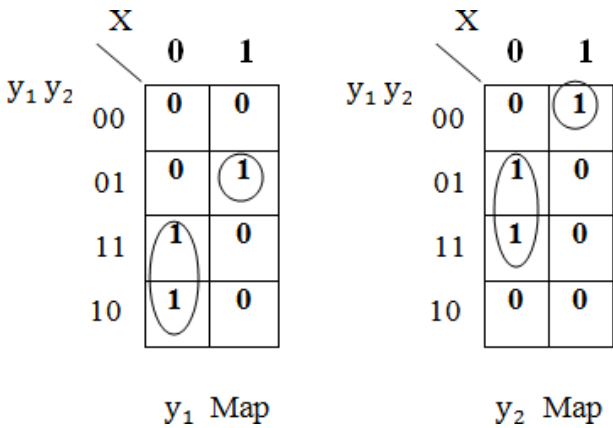
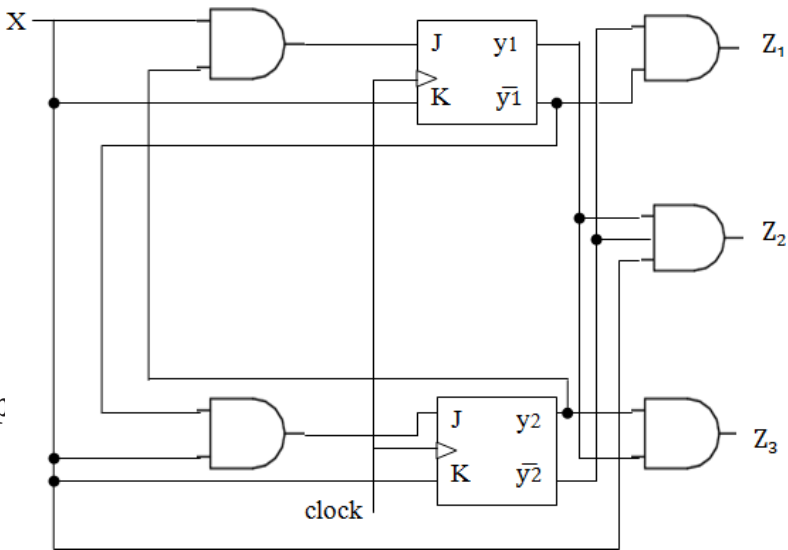
$Z_1 = \overline{y_1 y_2}$
 $Z_2 = y_1 \overline{y_2} X$
 $Z_3 = y_1 y_2$

Using the ch. Equation for the JK Flip-Flop

$Q^+ = JQ + \bar{K}Q$

60

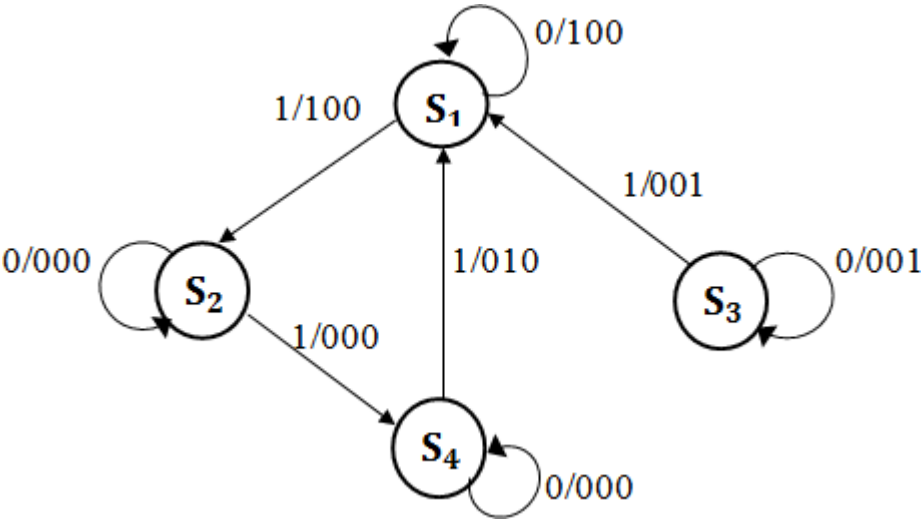
$y_1^+ = \bar{y_1}y_2X + y_1\bar{X}$
 $y_2^+ = y_1y_2 X + y_2\bar{X}$



Present state y ₁ y ₂	Next state		Z output	
	X=0	X=1	X=0	X=1
	y ₁ ⁺	y ₂ ⁺	Z ₁ Z ₂ Z ₃	Z ₁ Z ₂ Z ₃
00	00	01	100	100
01	01	10	000	000
11	11	00	001	001
10	10	00	000	010

Present state y_1y_2	Next state		Z output	
	X=0	X=1	X=0	X=1
	y_1^+	y_2^+	$Z_1Z_2Z_3$	$Z_1Z_2Z_3$
S_1	S_1	S_2	100	100
S_2	S_2	S_4	000	000
S_3	S_3	S_1	001	001
S_4	S_4	S_1	000	010

No reduction



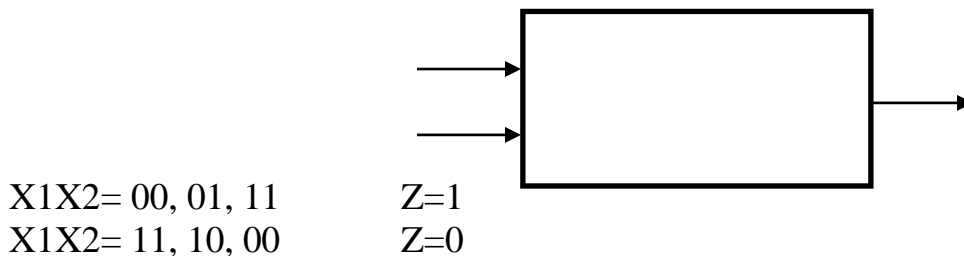
Chapter TwoDesign of Asynchronous State Machine

Design Steps:

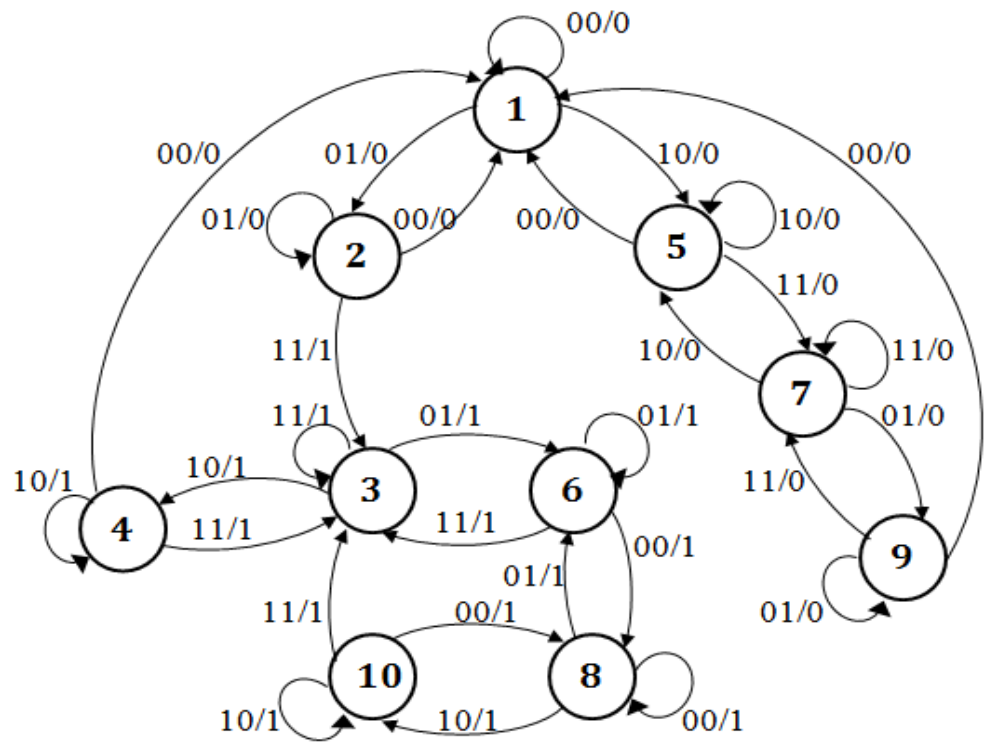
1. Prmitive State Diagram.
2. Primitive Flow Table.
3. Redution of Primitive flow Table.
4. Merging.
5. Merger Diagram.
6. Merged Flow Table.
7. Adjacency Sets.
8. Assignments Flow Table .
9. Logic Circuit.

Example 1: Design an asynchronous sequential logic circuit which has two inputs (X1&X2) and one output (Z). The output (Z) must go to (1) at the end of the input sequence (00,01,11). This output must be maintained (unchanged) for all input changes until the sequence (11,10,00) occurs, then the output must go to zero(0).

Solution:-



1. Prmitive State Diagram.



2. Primitive flow table.

State no.	Inputs X1X2				Output (Z)			
	00	01	11	10	00	01	11	10
1	1	2	—	5	0	—	—	—
2	1	2	3	—	—	0	—	—
3	—	6	3	4	—	—	1	—
4	1	—	3	4	—	—	—	1
5	1	—	7	5	—	—	—	0
6	8	6	3	—	—	1	—	—
7	—	9	7	5	—	—	0	—
8	8	6	—	10	1	—	—	—
9	1	9	7	—	—	0	—	—
10	8	—	3	10	—	—	—	1

3- Reduction of Primitive flow table using implication chart.

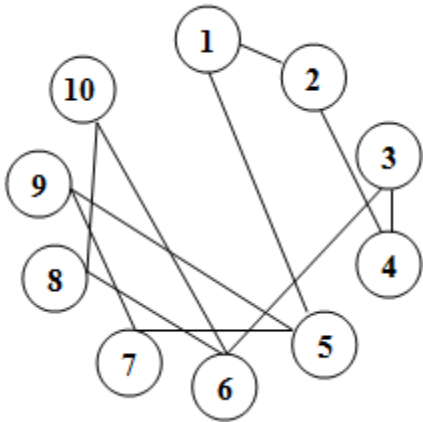
2	X								
3	X	X							
4	X	X	X						
5	X	X	X	X					
6	X	X	X	X	X				
7	X	X	X	X	X	X			
8	X	X	X	X	X	X	X		
9	X	3,7	X	X	X	X	X	X	
10	X	X	X	1,8	X	X	X	X	X
	1	2	3	4	5	6	7	8	9

N0 Reduction

4- Merging. and 5. The Merger Diagram.

m=(1,2) (1,5) (2,4) (3,4) (3,6) (5,7) (5,9) (6,8) (6,10) (7,9) (8,10).

Create the Merger Diagram, M=(1,2) (3,4) (5,7,9) (6,8,10)



6. Merged Flow Table.

	00	01	11	10	Z o/p
A	Ⓛ	Ⓜ	3	5	0
B	1	6	ⓓ	ⓔ	1
C	1	ⓖ	ⓗ	ⓙ	0
D	ⓚ	ⓛ	3	ⓞ	1

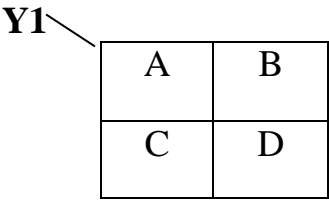
7. Adjacency Sets.

X1X2=00
(a,b,c);

X1X2=01
(b,d);

X1X2=11
(a,b,d);

X1X2=10
(a,c)



(Row Transition Map)

Note: The stable state take the code of the same row and the unstable state take the code of it's row where they are stable.

8. Assignments Flow Table .

		Inputs X1X2				Z o/p
	y1y2	00 y1y2	01 y1y2	11 y1y2	10 y1y2	
a	00	00	00	10	01	0
b	10	00	11	10	10	1
c	01	00	01	01	01	0
d	11	11	11	10	11	1

"Assigned Flow Table"

Circuit Relazation:-
Design the circuit using the S-R flip-flops.

		X1X2			
		00	01	11	10
y1y2	00	0	0	1	0
	01	0	0	0	0
	11	X	X	X	X
	10	0	X	X	X

$Sy1 = X1X2\bar{y}_2$

		X1X2			
		00	01	11	10
y1y2	00	X	X	0	X
	01	X	X	X	X
	11	0	0	0	0
	10	1	0	0	0


$Ry1 = \bar{X}1\bar{X}2\bar{y}2$

		X1X2			
y1y2 \		00	01	11	10
00		0	0	0	1
01		0	X	X	X
11		X	X	0	X
10		0	1	0	0

$Sy2 = \bar{X}1X2y1 + X1\bar{X}2\bar{y}1$

		X1X2			
		00	01	11	10
y1y2	00	X	X	X	0
	01	1	0	0	0
	11	0	0	1	0
	10	X	0	X	X

$$Ry2 = \bar{X}1\bar{X}2\bar{y}1 + X1X2y1$$

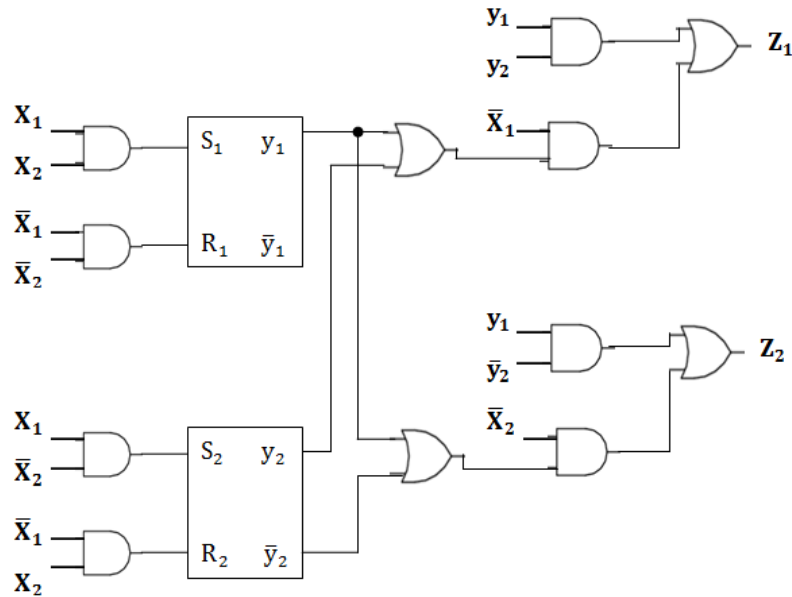
		X1X2				
y1y2	00	01	11	10		
	00	0	0	X		X
	01	X	0	0		0
	11	1	1	X		1
	10	X	X	1		1

|

Z=y1

Analysis of Asynchronous State Machine

Example 2: Let us analyze the following asynchronous sequential circuit.



Solution:

$$R_1 = \bar{x}_1 \bar{x}_2 \quad S_1 = x_1 x_2$$

$$R_2 = \bar{x}_1 x_2 \quad S_2 = x_1 \bar{x}_2$$

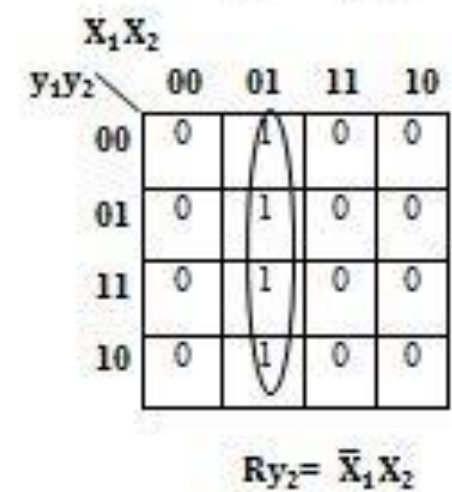
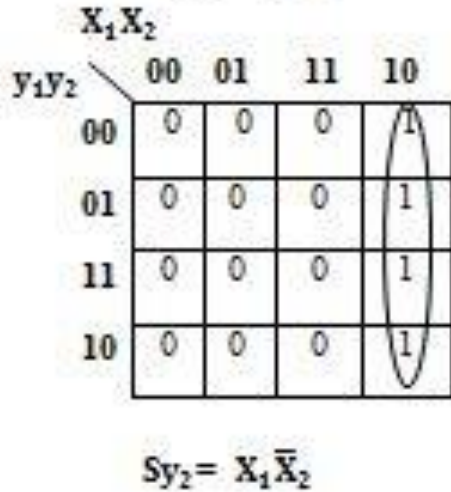
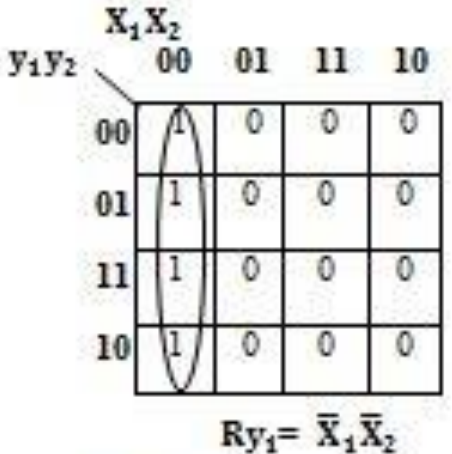
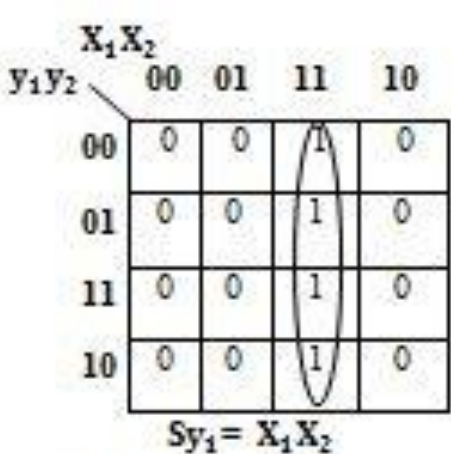
$$Z_1 = y_1 y_2 + \bar{x}_1 (y_1 + y_2)$$

$$Z_2 = y_1 \bar{y}_2 + \bar{x}_2 (y_1 + \bar{y}_2)$$

By using the characteristics equations of the S-R flip-flop.

$$Q^+ = S + \bar{R}Q$$

$$y^+ = S + \bar{R}y$$



Inputs $X_1 X_2$					
	$y_1 y_2$	00 $y_1^+ y_2^+$	01 $y_1^+ y_2^+$	11 $y_1^+ y_2^+$	10 $y_1^+ y_2^+$
a	00	00	00	10	01
b	01	01	00	11	01
c	11	01	10	11	11
d	10	00	10	10	11

The stable states are circuled.

Inputs X_1X_2				
y_1y_2	00 $y_1^+y_2^+$	01 $y_1^+y_2^+$	11 $y_1^+y_2^+$	10 $y_1^+y_2^+$
a	(a)	(a)	d	b
b	(b)	a	c	(b)
c	b	d	(c)	(c)
d	a	(d)	(d)	c

The merged flow table is given as follows:-

Inputs X_1X_2			
00	01	11	10
(1)	(2)	8	4
(3)	2	5	(4)
3	7	(5)	(6)
1	(7)	(8)	6

X_1X_2					
y_1y_2		00	01	11	10
00		0	0	0	0
01		1	1	0	0
11		1	1	1	1
10		1	1	0	0

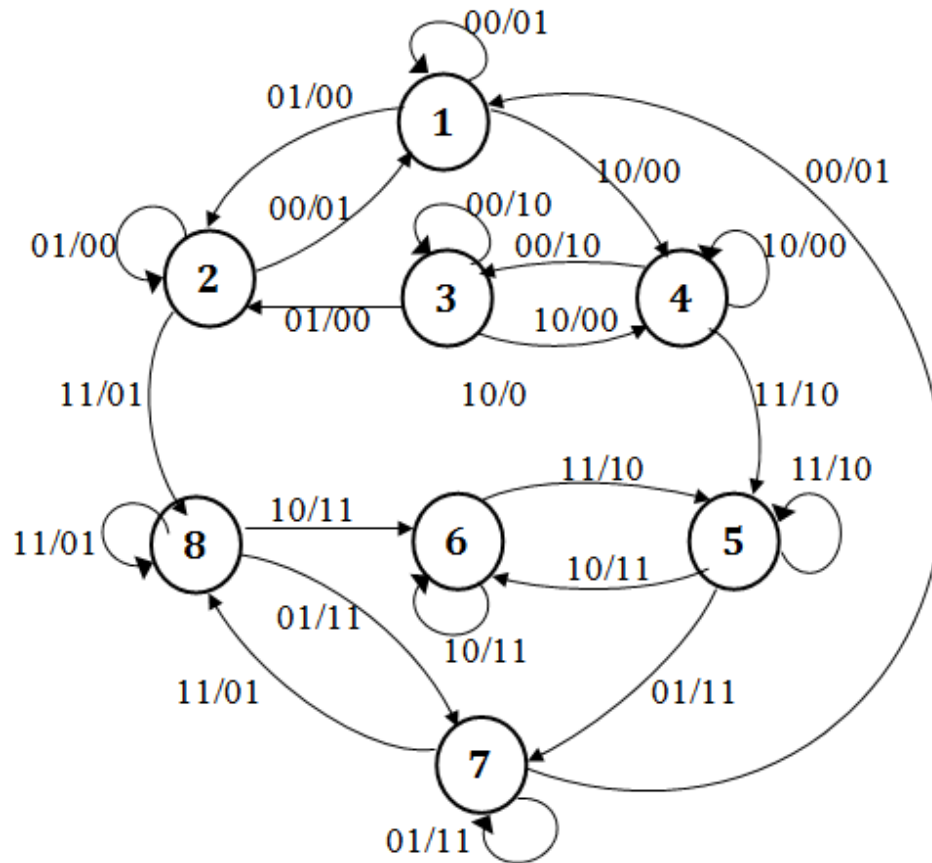
$Z_1 = y_1y_2 + \bar{X}_1(y_1 + y_2)$

X_1X_2					
y_1y_2		00	01	11	10
00		1	0	0	1
01		0	0	0	0
11		1	0	0	1
10		1	1	1	1

$Z_2 = y_1\bar{y}_2 + \bar{X}_2(y_1 + \bar{y}_2)$

Inputs X_1X_2				Z_1Z_2 o/p
00	01	11	10	
①	2	--	4	01
1	②	8	--	00
③	2	--	4	10
3	--	5	④	00
--	7	⑤	6	10
3	--	5	⑥	11
1	⑦	8	--	11
--	7	⑧	6	01

"Primitive flow table"

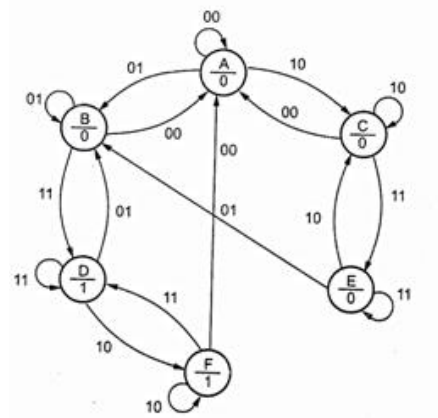


"Primitive state diagram"

Example 3: Design a circuit with primary inputs A and B to give an output Z equal to 1 when A becomes 1 if B is already 1. Once Z = 1 it will remain so until A goes to 0. Draw total state diagram, primitive flow table for designing this circuit.

Solution:

The total state diagram will be as shown in



this figure:

A primitive table is constructed from the state diagram as shown in the figure below.

Present state	Next State , Output Z for AB inputs			
	00	01	11	10
A	Ⓐ,0	B,-	-, -	C,-
B	A,-	Ⓑ,0	D,-	-, -
C	A,-	-, -	E,-	Ⓒ,0
D	-, -	B,-	Ⓓ,1	F,-
E	-, -	B,-	Ⓔ,0	C,-
F	A,-	-, -	D,-	Ⓕ,1

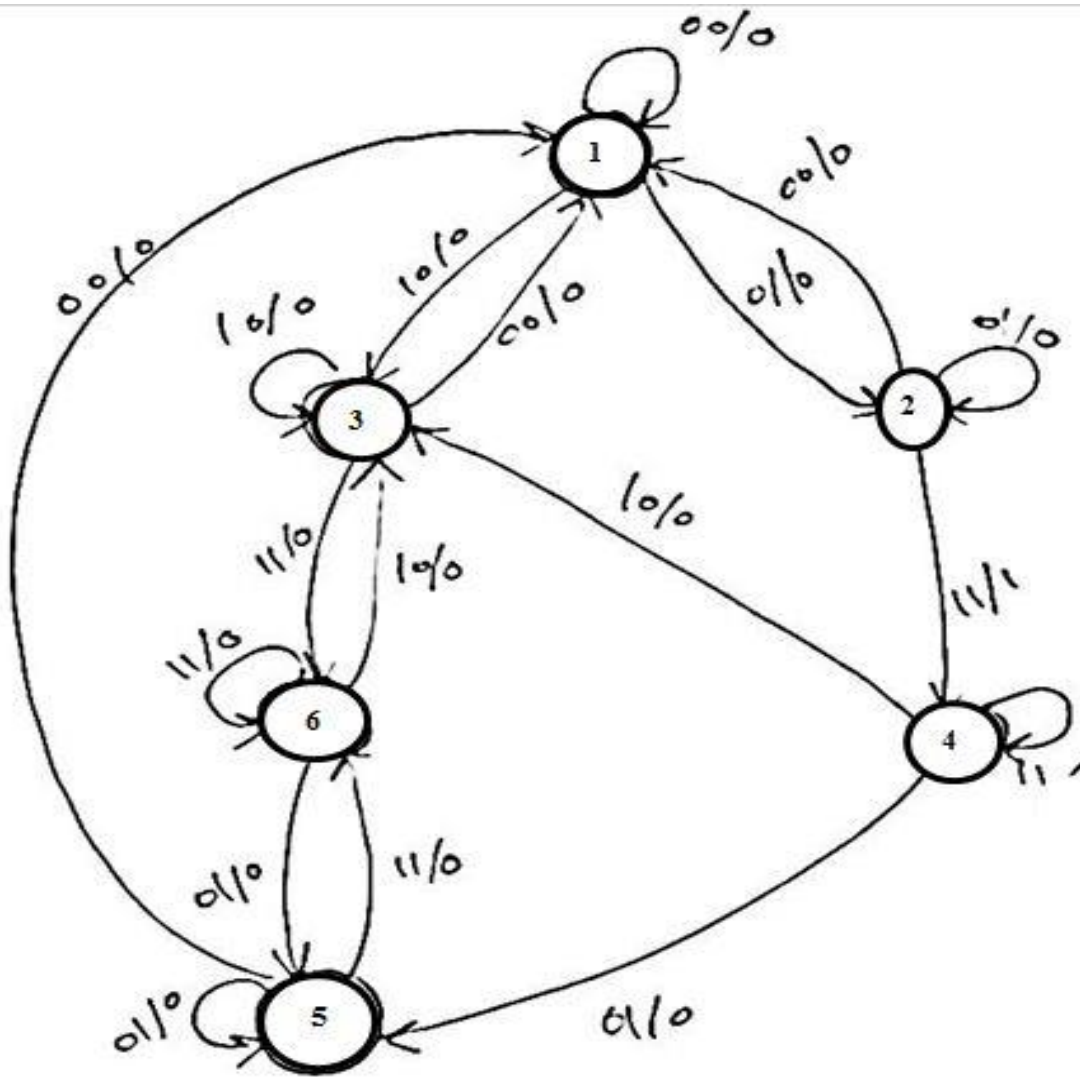
Example 4: Design an asynchronous sequential logic circuit which has two inputs (X1&X2) and one output (Z). The output (Z) = 1 when the sequence (00,01,11) occurs otherwise Z=0. Use mealy method in design.

Solution:-



X1X2 = 00, 01, 11 Z=1

1. Prmitive State Diagram.



2. Primitive flow table.

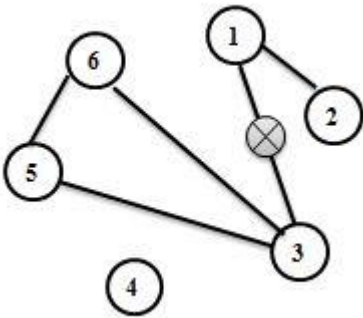
Inputs X_1X_2				Z_1 o/p
00	01	11	10	
①	2	--	3	0
1	②	4	--	0
1	--	6	③	0
--	5	④	3	1
1	⑤	6	--	0
--	5	⑥	3	0

3- Reduction of Primitive flow table using implication chart.

2	X				
3	X	X			
4	X	X	X		
5	X	4,6	X	X	
6	X	X	X	X	X
	1	2	3	4	5

NO Reduction

4- Merging. and 5. The Merger Diagram.



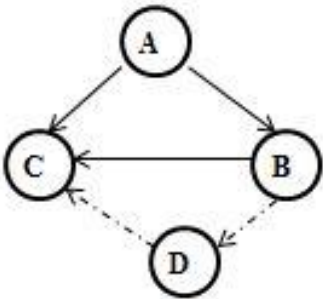
m=(1,2) (1, 3) (3,5) (3,6) (5,6)
Create the Merger Diagram, M=(1,2) (3,5,6)

6 Merged Flow Table.

	00	01	11	10	Z o/p
A	⓪	Ⓜ	4	3	0
B	1	Ⓟ	Ⓠ	Ⓡ	0
C	--	5	Ⓢ	3	1
D					

7. Adjacency Sets.

(A,B); (B,C); (A,C); (A,B,C)



Because there are many adjacenment between A,B ,C, so the stse D will add to solve this problem

A	00
B	10
C	01
D	11

	Y1	0	1
Y2	0	A	B
	1	C	D

(Row Transition Map)

8. Assignments Flow Table .

		Inputs X1X2				Z o/p
	y1y2	00 y1y2	01 y1y2	11 y1y2	10 y1y2	
A	0 0	0 0	0 0	0 1	1 0	0
B	1 0	0 0	1 0	1 0	1 0	0
C	0 1	--	1 0	0 1	1 0	1
D	1 1	--	--	--	--	--

"Assigned Flow Table"

Circuit Relaization:-
Design the circuit using the S-R flip-flops.
H.W.

Chapter 3 : Analog-To-Digital Conversion

An A/D converter is a very important building block and has numerous applications. It forms an essential interface when it comes to analyzing analogue data with a digital computer. It is an indispensable part of any digital communication system where the analogue signal to be transmitted is digitized at the sending end with the help of an A/D converter. It is invariably used in all digital read-out test and measuring equipment. Be it a digital voltmeter or a laser power meter, or for that matter even a pH meter, an A/D converter is the heart of all of them.

An A/D converter takes at its input an analogue voltage and after a certain amount of time produces a digital output code representing the analogue input. The A/D conversion process is generally more complex than the D/A conversion process.

1. A/D Converter Specifications

The major performance specifications of an A/D converter include resolution, accuracy, gain and offset errors, gain and offset drifts, the sampling frequency and aliasing phenomenon, quantization error, nonlinearity, differential nonlinearity, conversion time, aperture and acquisition times and code width.

1.1 Resolution

The resolution of an A/D converter is the quantum of the input analogue voltage change required to increment its digital output from one code to the next higher code. An n-bit A/D converter can resolve one part in $2^n - 1$. It may be expressed as a percentage of full scale or in bits. The resolution of an eight-bit A/D converter, for example, can be expressed as one part in 255 or as 0.4 % of full scale or simply as eight-bit resolution. If such a converter has a full-scale analogue input range of 10 V, it can resolve a 40 mV change in input.

1.2 Accuracy

The accuracy specification describes the maximum sum of all errors, both from analogue sources (mainly the comparator and the ladder resistors) and from the digital sources (quantization error) of the A/D converter. These errors mainly include the gain error, the offset error and the quantization error.

1.3 Sampling Frequency and Aliasing Phenomenon

If the rate at which the analogue signal to be digitized is sampled is at least twice the highest frequency in the analogue signal, which is what is embodied in the Shannon–Nyquist sampling theorem, then the analogue signal can be faithfully reproduced from its quantized values by using a suitable interpolation algorithm. The accuracy of the reproduced signal is, If the sampling rate is inadequate, i.e. if it is less than the Nyquist rate, then the reproduced signal is not a faithful reproduction of the original signal and these spurious signals, called aliases, are produced. The frequency of an aliased signal is the difference between the signal frequency and the sampling frequency. For example, if sampled at a 1.5 kHz rate, a 2 kHz sinewave would be reconstructed as a 500 Hz sine wave. This problem is called aliasing and, in order to avoid it, the analogue input signal is low-pass filtered to remove all frequency components above half the sampling rate. This filter, called an anti-aliasing filter, is used in all practical A/D converters.

1.4 Quantization Error

The quantization error is inherent to the digitizing process. For a given analogue input voltage range it can be reduced by increasing the number of digitized levels. An A/D converter having an n -bit output can only identify 2^n output codes while there are an infinite number of analogue input values adjacent to the LSB of the A/D converter that are assigned the same output code.

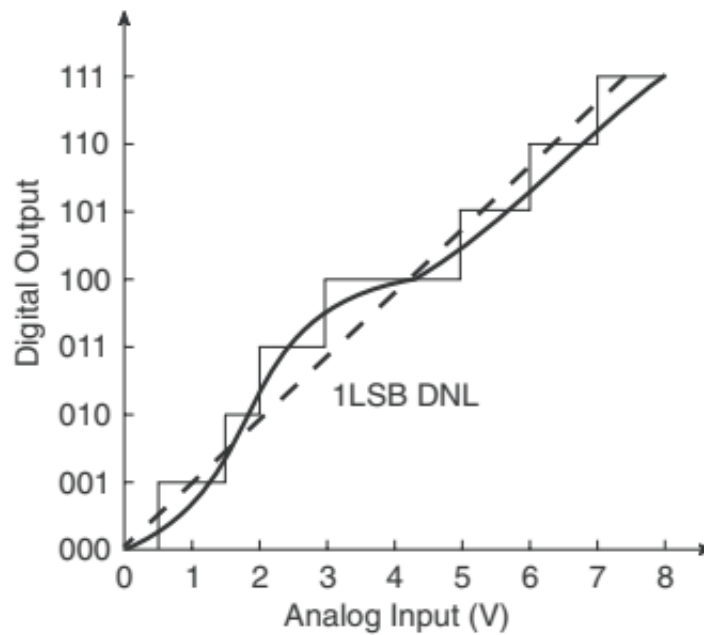


Figure :Transfer characteristics of a three-bit A/D converter (INL = one LSB, DNL = 1LSB).

1.5 Conversion Time

This is the time that elapses from the time instant of the start of the conversion signal until the conversion complete signal occurs. It ranges from a few nanoseconds for flash-type A/D converters to a few microseconds for successive approximation type A/D converters and may be as large as tens of milliseconds for dual-slope integrating A/D converters.

1.6 Aperture and Acquisition Times

The aperture and acquisition times are the parameters of the sample-and-hold circuit. The acquisition time is the time required for the electronic switch to close and the hold capacitor to charge, while the aperture time is the time needed for the switch completely to open after the occurrence of the hold signal. Ideally, both times should be zero.

2. A/D Converter Terminology

Some of the more commonly used terms while understanding the specifications and salient features of A/D converters are briefly described :

2.1 Unipolar Mode Operation

In the unipolar mode of operation, the analogue input to the A/D converter varies from 0 to full-scale voltage of one polarity only .

2.2 Bipolar Mode Operation

An A/D converter configured to convert both positive and negative analogue input voltages is said to be operating in bipolar mode .

Example 1

Determine the resolution of a 12-bit A/D converter having a full-scale analogue input voltage of 5 V .

Solution

- A 12-bit A/D converter resolves the analogue input voltage into $(2^{12} - 1)$ levels.
- The resolution = $5/(2^{12} - 1) = 5/(4096 - 1) = 5/4095 = 1.22\text{mV}$.

Example 2

The data sheet of a certain eight-bit A/D converter lists the following specifications: resolution eight bits; full-scale error 0.02 % of full scale; full-scale analogue input +5 V. Determine (a) the quantization error (in volts) and (b) the total possible error (in volts) .

Solution

(a) The eight-bit A/D converter has $2^8 - 1 = 255$ steps.

Therefore, the quantization error = $5/255 = 5/255 = 19.607 \text{ mV}$.

(b) The full-scale error = $0.02\% \text{ of full scale} = 0.02 \times 5/100 = 1 \text{ mV}$. Therefore, the total possible error = $19.607 + 1 = 20.607\text{mV}$.

3. Types of A/D Converter

Analogue-to-digital converters are often classified according to the conversion process or the conversion technique used to digitize the signal. Common types of A/D converter include flash or simultaneous or direct-conversion A/D converters, half-flash A/D converters, counter-type A/D converters, tracking A/D converters, successive approximation type A/D converters, single-slope, dual-slope and multislope A/D converters and sigma-delta A/D converters, and some of them are :

3.1 Flash (Simultaneous) Analog-to-Digital Converter

A comparator is an op. amp. configuration where the voltages of two inputs are compared. If the “+” input is greater than the “-” input, the output is a logic high. The flash method utilizes comparators that compare reference voltage with the analog input voltage. When the input voltage exceeds the reference voltage, a HIGH is generated. A comparator is not needed for all 0's condition.

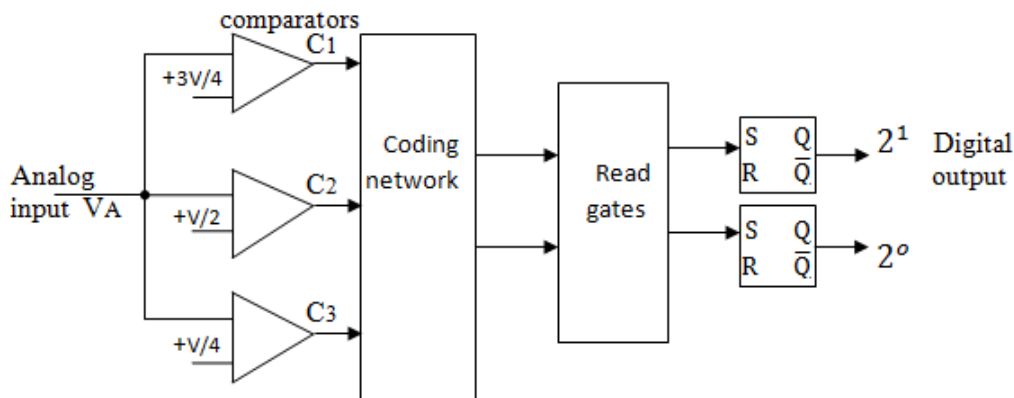
In general a $2^n - 1$ comparators are required for converting to an n- bit binary code. The number of bits in an ADC is its resolution.

Example 3: Design a 2-bit simultaneous A/D converter.

Solution:

The required number of comparators = $2^2 - 1 = 3$

where n is the number of bits of the digital output number.



Input Voltage (V)	Comparators output		
	C1	C2	C3
0 to +V/4	Low	Low	Low
+V/4 to +V/2	High	Low	Low
+V/2 to +3V/4	High	High	Low
+3V/4 to +V	High	High	High

Advantage:

Provides a fast conversion times because of a high through put measured in sps (samples per second.)

Disadvantage:

Large number of comparators necessary for a reasonable –sized binary number

no. of comparators = $2^n - 1$, Where n is the number of bits of the output.

Example4: Design a 3 bit Flash ADC.

Solution:

$$V_{IN} = 5.5V, V_{ref} = 8V$$

V_{IN} lies in between V_{comp5} & V_{comp6}

$$V_{comp5} = V_{ref} * 5/8 = 5V$$

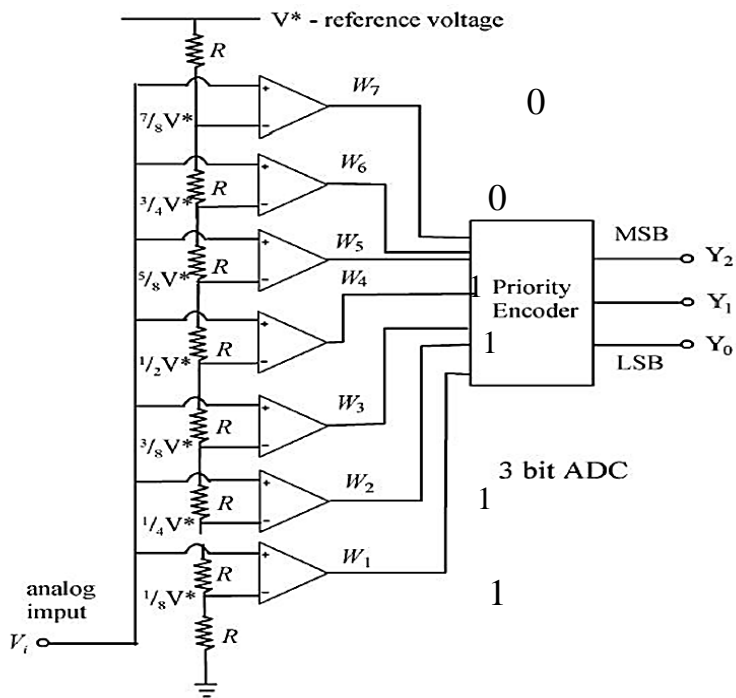
$$V_{comp6} = V_{ref} * 6/8 = 6V$$

Comparator 1 - 5 => output 1

Comparator 6 - 7 => output 0

Encoder Octal Input = sum (0011111) = 5

Encoder Binary Output = 1 0 1



Example 5: Determine the binary code output of the 3-bit flash ADC for the input with the encoder enable pulses shown in fig. 1. $V_{Ref} = +8\text{ V}$.

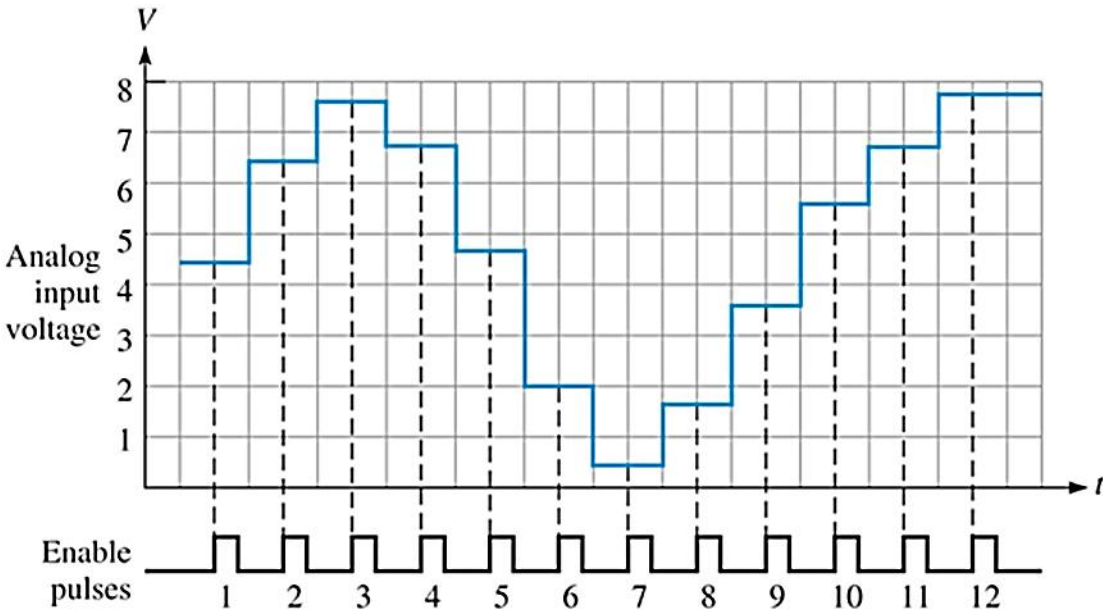
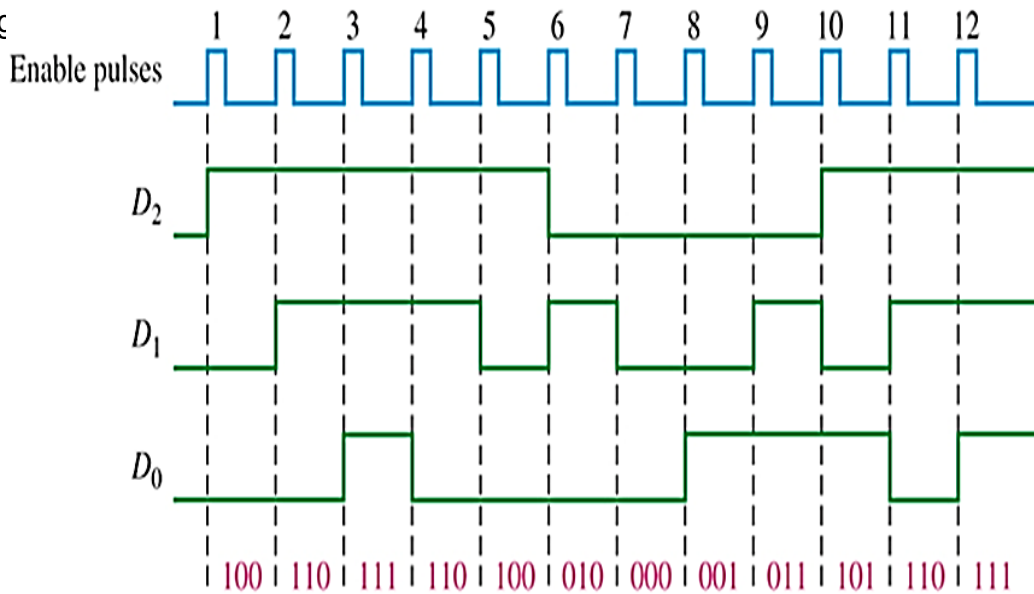


Fig. Sampling of values on a waveform for conversion to binary code.

The resulting digital output sequence: 100 (4), 110(6), 111(7), 110(6), 100 (4), 010(2), 000(0), 001(1), 011 (3), 101 (5), 110 (6), 111(7) Waveform of the resulting digital output sequence:



3.2 Half-Flash A/D Converter

The half-flash A/D converter, also known as the pipeline A/D converter, is a variant of the flash-type converter that largely overcomes the primary disadvantage of the high-resolution full-flash converter, namely the prohibitively large number of comparators required, without significantly degrading its high-speed conversion performance. Compared with a full-flash converter of certain resolution, while

the number of comparators and associated resistors is drastically reduced in a half-flash converter, the conversion time increases approximately by a factor of 2. For an n -bit flash converter the number of comparators required is $2^n[(2^n - 1)$ for encoding of amplitude and one comparator for polarity], while the same for an equivalent half-flash converter would be $2 \times 2^{n/2}$. In the case of an eight-bit converter, the number is 32 (for half-flash) against 256 (for full flash)

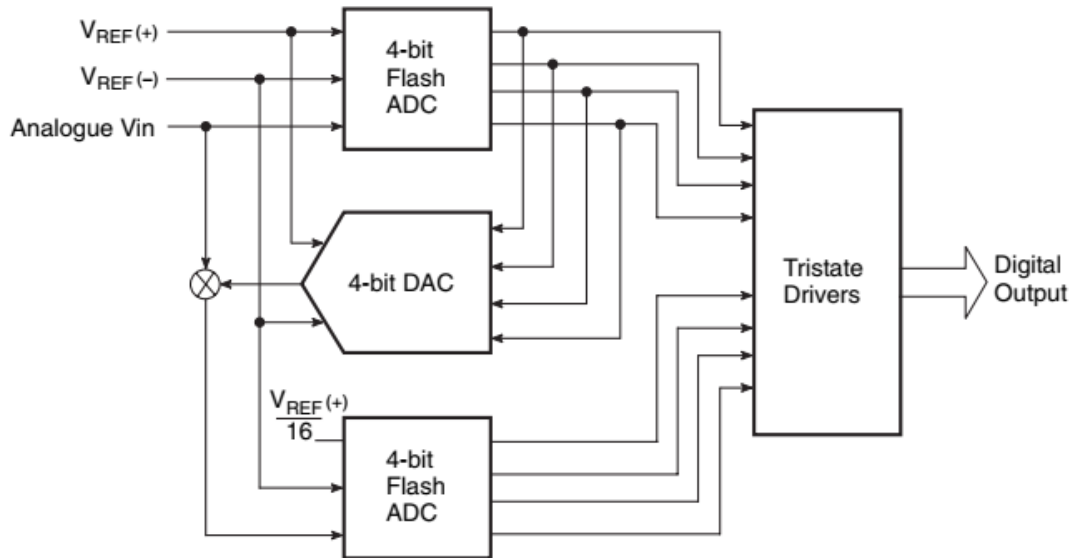


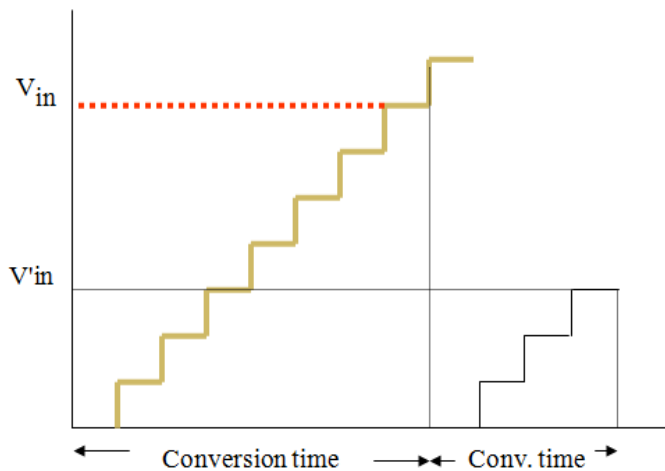
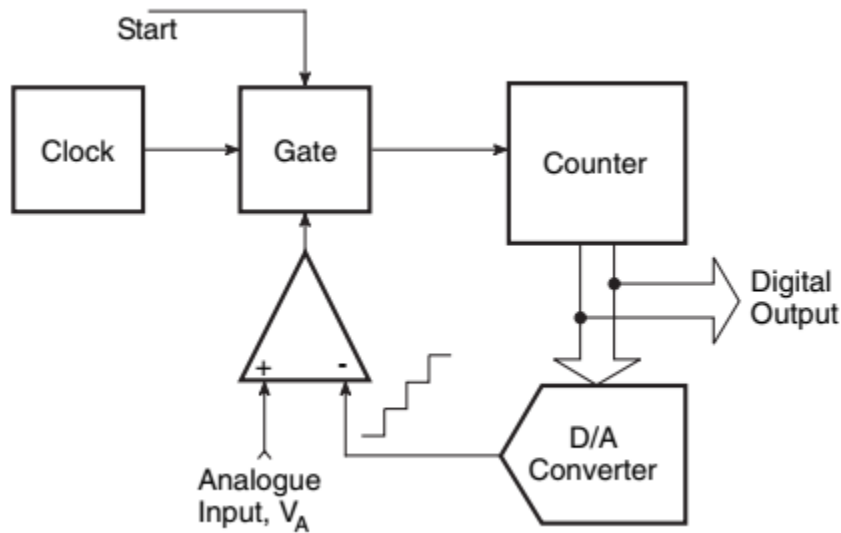
Figure Eight-bit half-flash A/D converter.

3.3 Counter Type A/D Converter:

It is possible to construct higher-resolution A/D converters with a single comparator by using a variable reference voltage. One such A/D converter is the counter-type A/D converter. A higher resolution A/D converter using only one comparator could be constructed if a variable reference voltage could then be applied to the comparator and when it became equal to the input analog voltage the conversion would be complete.

$$0 \leq V_A \leq V_{ref}$$

This method is much simpler than the simultaneous method for high resolution, but the conversion time required is longer



Since the counter always begins at zero and counts through its normal binary sequence, it may require as many as 2^n counts before conversion is complete. The average conversion time is 2^{n-1} counts.

Note that the conversion time depends on the size of the input signal

1. Step Size = $V_{f.s}/2^n-1$
2. Number of steps = $V_A / \text{step size}$
3. The conversion time = No. of steps * Time
4. The resolution in volt = $V_{f.s} / 2^n-1$

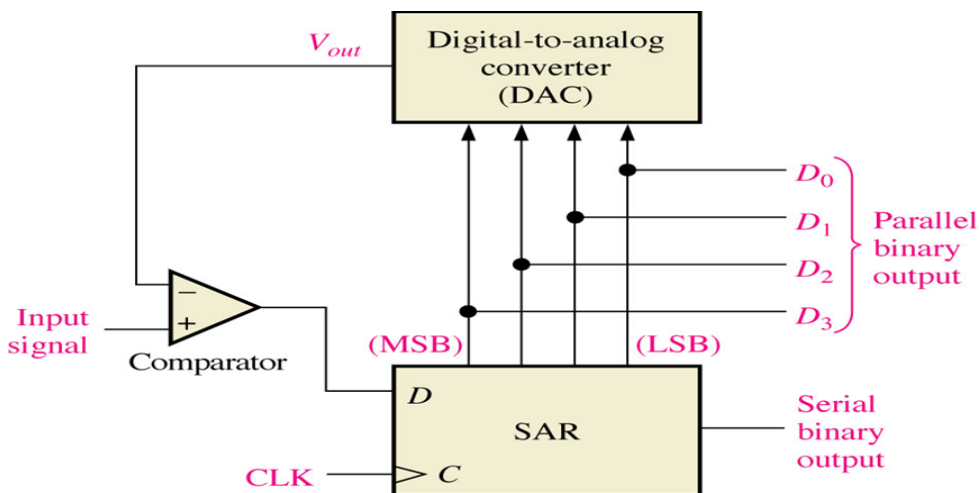
Example 6 : Determine the conversion time of a 12-bit A/D converter of the counter type for an input clock frequency of 1 MHz.

Solution

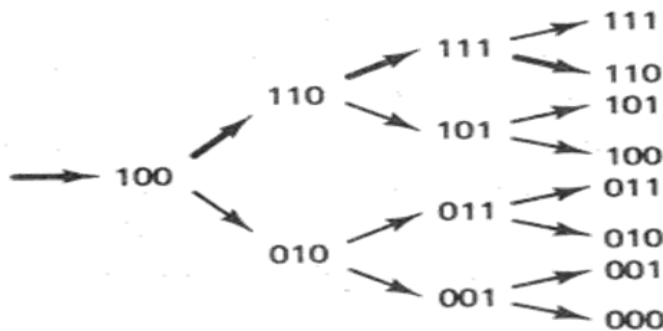
- The counter-type A/D converter shown in Fig. has a variable conversion time that is maximum when the input analogue voltage is just below the full-scale analogue input voltage .
- An average conversion time equal to half the maximum conversion time is usually defined in the case of such converters .
- The maximum conversion time equals the time taken by $2^{12} - 1 = 4095$ cycles of clock input .
- The clock time period = $1/(1 \times 10^6) = 1 \mu\text{s}$.
- Therefore, the maximum conversion time = $4095 \times 1 = 4095 \mu\text{s} = 4.095 \text{ ms}$.
- The average conversion time = $(4.095/2) = 2.047 \text{ ms}$.

3.4 Successive Approximation Analog-to-Digital Converter

The Successive-Approximation ADC is one of the most popular types in use today. It has a relatively simple configuration and an excellent conversion rate. The following Figure shows the basic block diagram of a 4 bit Successive - approximation ADC. It consists of a DAC, Successive-Approximation Register (SAR), and a comparator.



The operation diagram for a 3 bit successive - approximation converter ADC is shown below:



The time for one analog to digital conversion must depend on both the clocks period (T) and number of bits n. It is given as : $T_c = n \times T$, T_c : Conversion time. T: Clock period, n: Number of bits.

Example 7: An 8-bit successive approximation ADC is driven by 1MHz clock. Find its conversion time.

Solution: $F=1\text{MHz}$, $T=\frac{1}{f}$, $T=\frac{1}{1 \times 10^6}=1\mu\text{s}$
 $N=8$, $T_c = n \times T = 8 \times 1 = 8\mu\text{s}$

3.5 Tracking-Type A/D Converter

The tracking-type A/D converter, also called the delta-encoded A/D converter, is a modified form of counter-type converter that to some extent overcomes the shortcoming of the latter

Example 8: Compare the average conversion time of an eight-bit counter-type A/D converter with that of an eight-bit successive approximation type A/D converter if both are working at a 10 MHz clock frequency .

Solution • The clock time period = $0.1\mu\text{s}$.

- The average conversion time in the case of a counter-type A/D converter is given by: $[(2^8 - 1)/2] \times 0.1 = 1275\mu\text{s}$.
- The conversion time in the case of a successive approximation type A/D converter is given by $8 \times 0.1 = 0.8\mu\text{s}$.

4. Integrated Circuit A/D Converters

This section presents application-relevant information of some of the popular A/D converter IC type numbers, as it is not possible to give a detailed description of each one of them. The type numbers included for this purpose are ADC 0800, ADC 0808, ADC 80, ADC 84, ICL 7106/ICL 7107 and AD 7820.

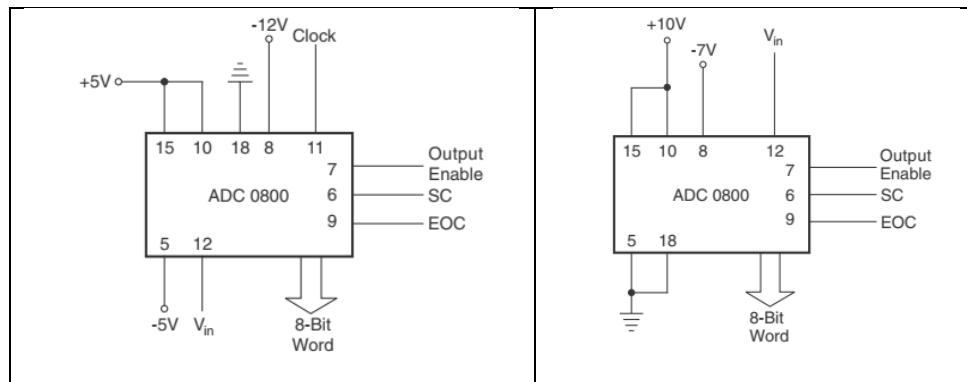


Figure Basic application circuits using AD 0800.

5. A/D Converter Applications

Like D/A converters, A/D converters have numerous applications. A/D converters are used in virtually all those applications where the analogue signal is to be processed, stored or transported in digital form. They form an essential interface when it comes to analysing analogue data with a digital computer, the process being known as 'data acquisition'. They are an indispensable component of any digital communication system where the analogue signal to be transmitted is digitized at the sending end with an A/D converter. They are invariably used in all digital readout test and measuring equipment such as digital multimeters (DMMs), digital storage oscilloscopes (DSOs), etc. Also, A/D converters are integral to contemporary music reproduction technology, as most of it is done on computers. In the case of analogue recording too, an A/D converter is needed to create the PCM data stream that goes onto a compact disc.

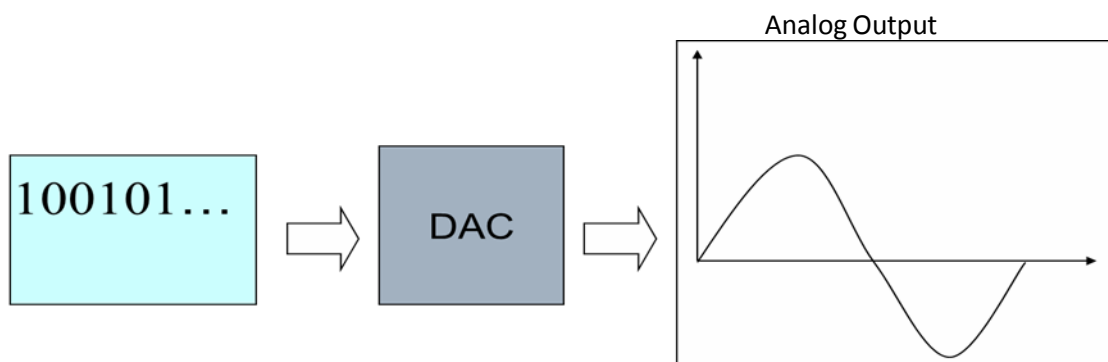
Chapter 4 : Digital –To- Analog Conversion

Digital-to-analogue (D/A) converter takes digital data at its input and converts them into analogue voltage or current that is proportional to the weighted sum of digital inputs ,A D/A converter is important not only because it is needed

at the output of most digital systems, but it is also important because it forms an indispensable part of the majority of A/D converter types.

- An A/D converter, too, has numerous applications. When it comes to transmitting analogue data, it forms an essential interface with a digital communication system where the analogue signal to be transmitted is digitized at the sending end with an A/D converter. It is invariably used in all digital read-out test and measuring equipment.

In the following paragraphs it is briefly explained how different bits in the digital input data contribute a different quantum to the overall output analogue voltage or current, and also that the LSB has the least and the MSB the highest weight.



1. D/A Converter Types

1.1 Simple Resistive Divider Network .

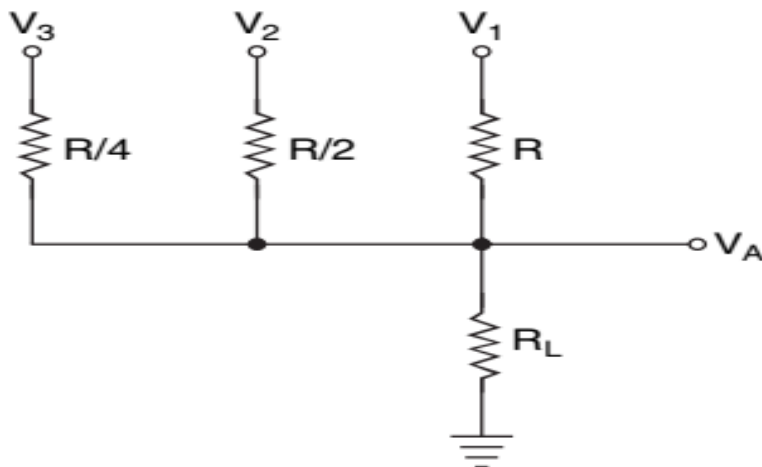
Simple resistive networks can be used to convert a digital input into an equivalent analogue output. Figure shows one such resistive network that can convert a three-bit digital input into an analogue output. if R_L is much larger than R it can be proved with the help of simple network theorems that the output analogue

voltage is given by

$$\begin{aligned} V_A &= \frac{[V_1/R] + [V_2/(R/2)] + [V_3/(R/4)]}{[1/R] + [1/(R/2)] + [1/(R/4)]} \\ &= \frac{[V_1/R] + [2V_2/R] + [4V_3/R]}{[1/R] + [2/R] + [4/R]} \\ &= \frac{V_1 + 2V_2 + 4V_3}{7} \end{aligned}$$

For **3-bit** which can be further expressed as

$$V_A = \frac{V_1 \cdot 2^0 + V_2 \cdot 2^1 + V_3 \cdot 2^2}{2^3 - 1}$$



Simple resistive network for D/A conversion

The generalized expression of Equation can be extended further to an **n-bit** D/A converter to get the following expression

$$V_A = \frac{V_1 \cdot 2^0 + V_2 \cdot 2^1 + V_3 \cdot 2^2 + \dots + V_n \cdot 2^{n-1}}{2^n - 1}$$

In the expression, If $V_1 = V_2 = \dots = V_n = V$

then a logic '1' at the LSB position would contribute $V/(2^n - 1)$ to the analogue output,

and a logic '1' in the next adjacent higher bit position would contribute $2V/(2^n - 1)$ then $4V/(2^n - 1)$, $8V/(2^n - 1)$, $16V/(2^n - 1)$ and so on.

The Limitations of simple resistive divider network are:

1- each resistor in the network is of a different value. Since these networks use precision resistors, the added expense becomes unattractive.

2- the resistor used for the most significant bit (MSB) is required to handle a much larger current than the LSB resistor. For example, in a 10-bit network, the current through the MSB resistor will be about 500 times the current through the LSB resistor.

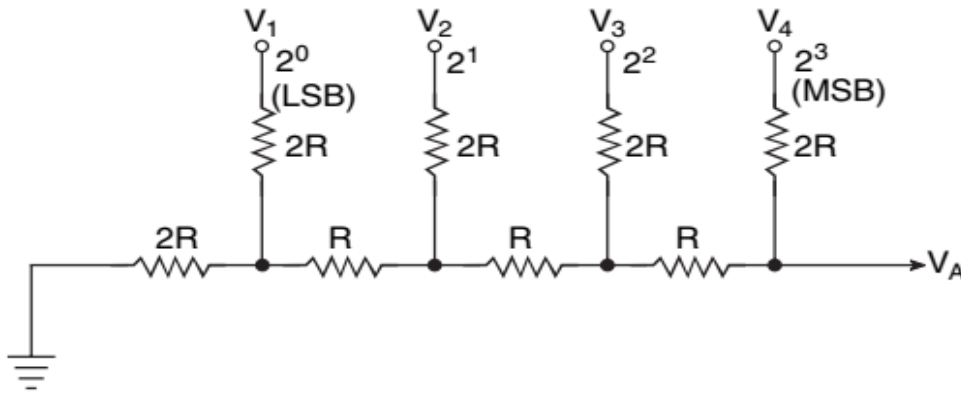
1.2 Binary Ladder Network

A second type of resistive network called the binary ladder (or $R/2R$

ladder) is used in practice. The binary ladder, too, is a resistive network that produces an analogue output equal to the weighted sum of digital inputs. Figure below shows the binary ladder network for a **four-bit** D/A converter. As is clear from the figure, the ladder is made up of only two different values

of resistor. This overcomes one of the drawbacks of the resistive divider network. It can be proved with the help of simple mathematics that the analogue output voltage V_A in the case of binary ladder network of the Fig. below is given by:

$$V_A = \frac{V_1 \cdot 2^0 + V_2 \cdot 2^1 + V_3 \cdot 2^2 + V_4 \cdot 2^3}{2^4}$$



Binary ladder network for 4- bit D/A conversion.

In general, for an n-bit D/A converter using a binary ladder network

$$V_A = \frac{V_1 * 2^0 + V_2 * 2^1 + V_3 * 2^2 + \dots + V_n * 2^{n-1}}{2^n}$$

- For $V_1 = V_2 = V_3 = \dots = V_n = V$, $V_A = \frac{2^n - 1}{2^n} * V$.
- For $V_1 = V_2 = V_3 = \dots = V_n = 0$, $V_A = 0$

Example 1 :What is the output for a(4 stage) ladder network using (5)volt for **logic 1** and (0)volt for **logic zero** for the following binary:

a)1000 b) 0001 c) 1101 .

$$\text{Sol: } V_A = \frac{V_1 * 2^0 + V_2 * 2^1 + V_3 * 2^2 + \dots + V_n * 2^{n-1}}{2^n}$$

For n= 4 bit

$$\text{a) For (1000) } V_A = \frac{0 * 2^0 + 0 * 2^1 + 0 * 2^2 + 5 * 2^3}{2^4} = 2.5 \text{ volt}$$

$$\text{b) For (0001) } V_A = \frac{5 * 2^0 + 0 * 2^1 + 0 * 2^2 + 0 * 2^3}{2^4} = 0.3125 \text{ volt}$$

$$\text{c) For (1101) } V_A = \frac{5 * 2^0 + 0 * 2^1 + 5 * 2^2 + 5 * 2^3}{2^4} = 4.0625 \text{ volt}$$

2.D/A Converter Specifications

The major performance specifications of a D/A converter include resolution, accuracy, conversion speed, dynamic range, and nonlinearity (NL).

2.1 Resolution:

The resolution of a D/A converter is the number of states (2^n) into which the full-scale range is divided or resolved

In general, for an n-bit D/A converter, the percentage resolution is given by;

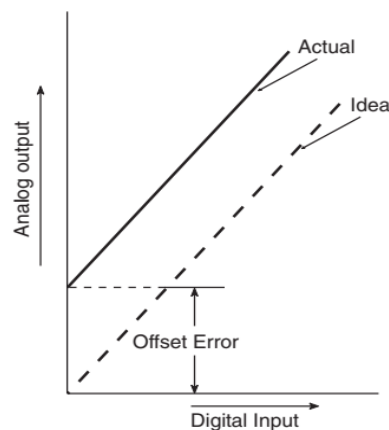
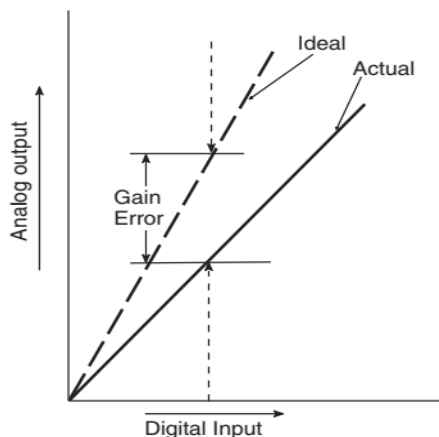
$$\frac{1}{2^n - 1} \times 100 \%$$

- An 8-bit D/A converter has 255 resolvable levels. It is said to have a percentage resolution of; $(1/255) \times 100 = 0.39\%$ or simply an eight-bit resolution.
- A 12-bit D/A converter would have a percentage resolution of $(1/4095) \times 100 = 0.0244 \%$.

The resolution in millivolts for the two cases for a full scale output of 5 V is approximately 20 mV (for an eight-bit converter) and 1.2 mV (for a 12-bit converter).

2.2 Accuracy

The accuracy of a D/A converter is the difference between the actual analogue output and the ideal expected output when a given digital input is applied. Sources of error include the gain error (or full-scale error), the offset error (or zero-scale error), nonlinearity errors and a drift of all these factors. The gain error is the difference between the actual and ideal output voltage, expressed as a percentage of full-scale output. It is also expressed in terms of LSB



2.3 Conversion Speed or Settling Time

The conversion speed of a D/A converter is expressed in terms of its settling time. The settling time is the time period that has elapsed for the analogue output to reach its final value within a specified error band after a digital input code change has been effected, General-purpose D/A converters have a settling time of several microseconds, to a few nanoseconds

Example 2 : An eight-bit D/A converter has a step size of 20 mV. Determine the full-scale output and percentage resolution.

• **Sol.:**

• $(\frac{1}{2^8}) \times V = 20 \times 10^{-3}$, where V is the voltage corresponding to a logic '1'.

• This gives $V = 20 \times 10^{-3} \times 2^8 = 5.12V$.

• The full-scale output $= (\frac{2^n - 1}{2^n} \times V) = (\frac{2^8 - 1}{2^8} \times 5.12)$

$$= (\frac{255}{256} \times 5.12) = 5.1V.$$

• The percentage resolution $= [\frac{1}{2^n - 1} \times 100] = \frac{100}{255} = 0.392\%$.

• Also The percentage resolution : $\frac{\text{Step size}}{\text{full-scale output}} \times 100$

$$= (\frac{20 \times 10^{-3}}{5.1}) \times 100 = 0.392\%.$$

Example 3: A certain eight-bit D/A converter has a full-scale output of 5 mA and a full-scale error of $\pm 0.25\%$ of full scale. Determine the range of expected analogue output for a digital input of 10000010.

Sol.:

$$\bullet \text{Step size} = \frac{\text{Full-scale output}}{\text{Number of steps}} = \frac{5 \times 10^{-3}}{2^8 - 1}$$

$$= 19.6 \mu A$$

- For a digital input of 10000010 ($= 130_{10}$) the analogue output is given by :
 $130 \times 19.6 = 2.548 \text{ mA}$.
- Error = $(0.25 \times 5 \times 10^{-3})/100 = \pm 12.5 \text{ } \mu\text{A}$,
- The expected analogue output will therefore be in the range 2.5355 to 2.5605 mA.

3. Types of D/A Converter

The type of D/A converters are : Multiplying-type D/A converters, Bipolar-output D/A converters, and Companding D/A converters .

And some of them are discussed below:

3.1 Multiplying-type D/A converters.

In a multiplying-type D/A converter, the converter multiplies an analogue reference by the digital input. Figure below shows the circuit representation.

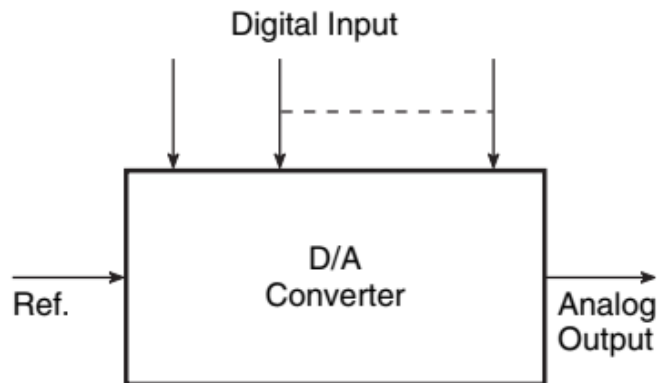


Figure : Multiplying-type D/A converter

3.2 Bipolar-output D/A converters.

In bipolar-output D/A converters the analogue output signal range includes both positive and negative values.

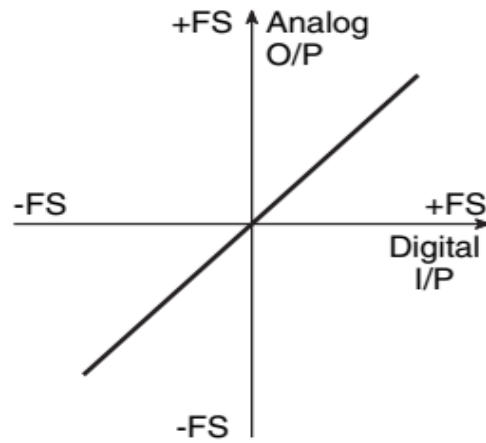


Figure: Bipolar-output D/A converter transfer characteristics

4. Modes of Operation

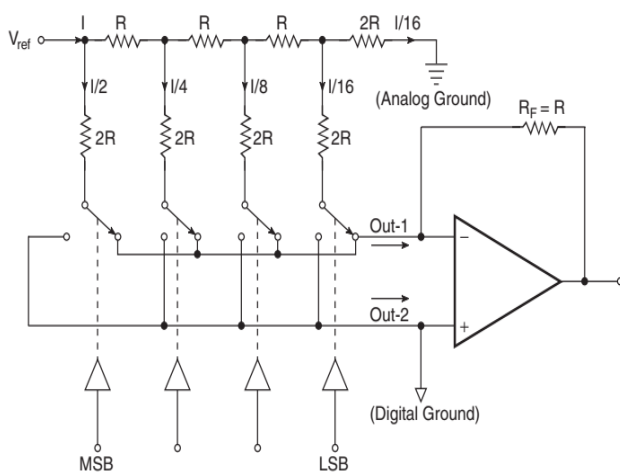
D/A converters are usually operated in either of the following two modes of operation: Current steering mode and Voltage switching mode

4.1. Current steering mode.

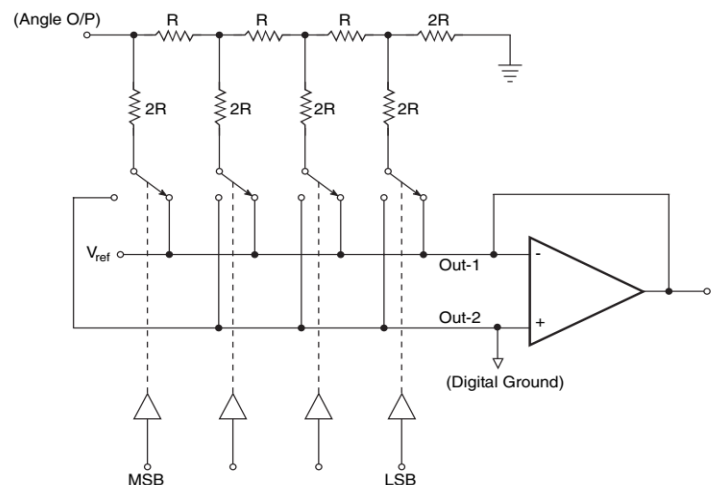
the analogue output is a current equal to the product of a reference voltage and a fractional binary value D of the input digital word

4.2 Voltage switching mode.

In the voltage switching mode of operation of a $R/2R$ ladder type D/A converter, the reference voltage is applied to the Out-1 terminal and the output is taken from the reference voltage terminal. Out-2 is joined to analogue ground



Current steering mode



Voltage switching mode

5.D/A Converter Applications .

In addition to being an integral part of some of the architectures of popular varieties of A/D converters, D/A converters are extensively used in a variety of other application circuits. Some common applications include multipliers, digitally controlled dividers, programmable integrators, low-frequency function generators and digitally controlled filters.

6. Integrated Circuit D/A Converters .

This section presents application-relevant information on some of the commonly used D/A converter IC type numbers, as it is not possible to give a detailed description of each one of them. The type numbers included for this purpose are DAC-08/0800, DAC-80, DAC-0808, AD 7524 and DAC-1408A/1508A

HW:An eight-bit D/A converter produces an analogue output of 12.5 mV for a digital input of 00000010.

Determine the analogue output for a digital input of 00000100.

Chapter 5 : Digital Integrate Circuits: Logic Families

1.Introduction to Logic Family

The Basic circuit in each family is either a NAND or a NOR or Inverter gate .The electronic components employed in the construction of the basic cct. are usually use to name the logic family .

The Logic Family :A group of compatible ICs with the same logic levels and supply voltages Fabrication using a specific circuit configuration Same basic logic gate during fabrication Use simple components like – Resistors, diodes, transistors and MOSFETs.

Significance of Logic Family

- Simplicity in the design
- Members with similar electrical characteristics
- Reduces the complexity in the design
- Provides compatibility in digital circuits
- Easy to expand

Classification and Types of Logic Families Logic families can be classified broadly according to the technologies they are built with. In earlier days we had vast number of these technologies, as you can see in the figure below:

- **DL:** Diode Logic.
- **RTL:** Resistor Transistor Logic.
- **DTL:** Diode Transistor Logic.
- **HTL:** High threshold Logic.
- **TTL:** Transistor Transistor Logic.
- **IIL(I²L):** Integrated Injection Logic.
- **ECL:** Emitter coupled logic.
- **MOS:** Metal Oxide Semiconductor Logic (PMOS and NMOS).
- **CMOS:** Complementary Metal Oxide Semiconductor Logic

Logic families are further classified based on number of logic gates fabricated per chip in IC form :

SI :Small Scale Integration	< 12 gates / chip
ISI :Medium Scale Integration	< 100 gates / chip
LSI :Large Scale Integration	...1K gates / chip
LSI: Very Large Scale Integration	...10 K gates/ chip

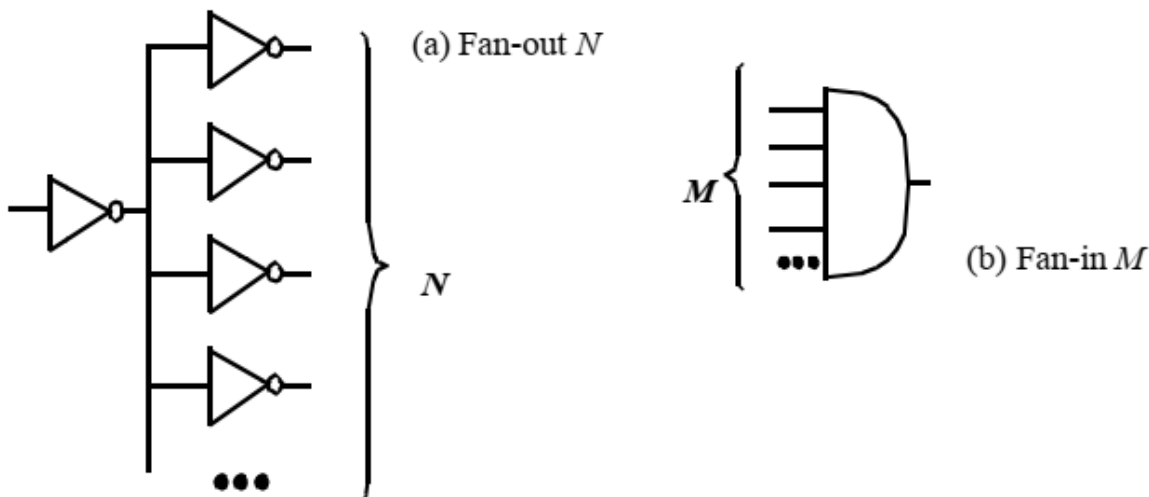
2.Characteristics of Logic Families :

2.1 Fan in

Fan in is the number of inputs connected to the gate without any degradation in the voltage level.

2.2 Fan-out

Fan out specifies the number of standard loads that the output of the gate can drive without impairment of its normal operation



2.3 Power dissipation

Power dissipation is measure of power consumed by the gate when fully driven by all its inputs.

$$I_{CC}(avg) = \frac{I_{CCH} + I_{CCL}}{2}$$

$$P_D (avg) = I_{CC} (avg) * V_{CC}$$

2.4 Propagation delay

Propagation delay is the average transition delay time for the signal to propagate from input to output when the signals change in value. It is expressed in ns.

Speed-power product(J)=Propagation delay (Seconds) x Power dissipation (joules/seconds)

Propagation delay increases with number of inputs. Therefore a 2 input NAND gate is faster than 4 input NAND gate.

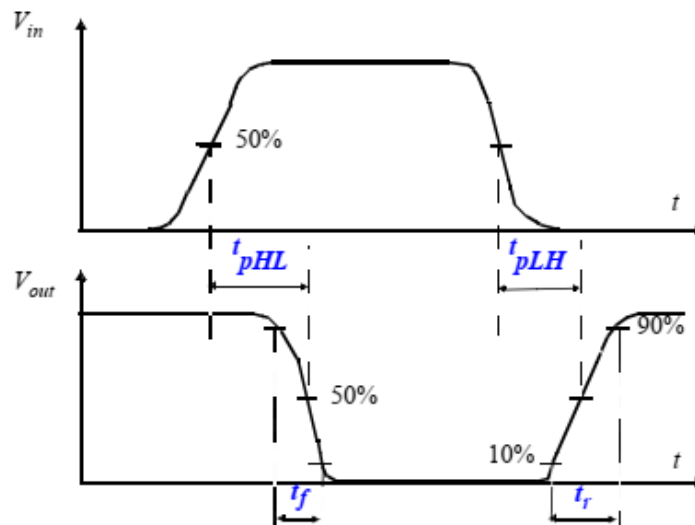
1. t_{PLH} : It is the delay time measured when output is changing from logic 0 to logic 1 state (LOW to HIGH).

2. t_{PHL} : It is the delay time measured when output is changing from logic 1 to logic 0 state (HIGH to LOW).

$$t_p = \max(t_{PLH}, t_{PHL})$$

Sometimes propagation delay t_p is taken to be the simple average of the two :

$$t_p = \frac{1}{2}(t_{pLH} + t_{pHL})$$



2.5 Noise margin

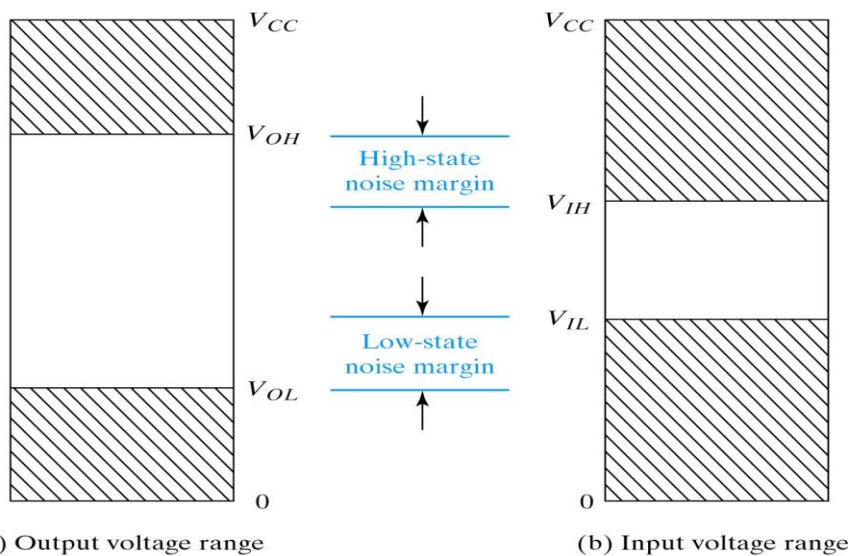
It is the maximum noise voltage added to an input signal of a digital circuit that does not cause an undesirable change in the circuit output. It is expressed in volts.

High state noise margin:

$$V_{NH} = V_{OH}(\min) - V_{IH}(\min)$$

Low state noise margin:

$$V_{NL} = V_{IL}(\max) - V_{IH}(\max)$$



Example 1:

LOW. What

(a) Output voltage range

Solution: The average I_{CC} is given as,

$$I_{CC}(\text{avg}) = \frac{I_{CCH} + I_{CCL}}{2} = 2.5 \text{ mA}$$

Then average power dissipation is given as,

$$PD(\text{avg}) = I_{CC}(\text{avg}) \times V_{CC} = 2.5 \text{ mA} \times 5 \text{ V} = 12.5 \text{ mW}$$

13.2 mA when its output is

Example 2: For a certain IC family, propagation delay is 10 ns with an average power dissipation of 6 mW. What is its speed power product?

Solution: The speed- power product is given as,

$$\begin{aligned} SPP &= \text{Propagation delay} \times \text{Average power dissipation} \\ &= 10 \text{ ns} \times 6 \text{ mW} = 60 \text{ pico joules (pJ)} \end{aligned}$$

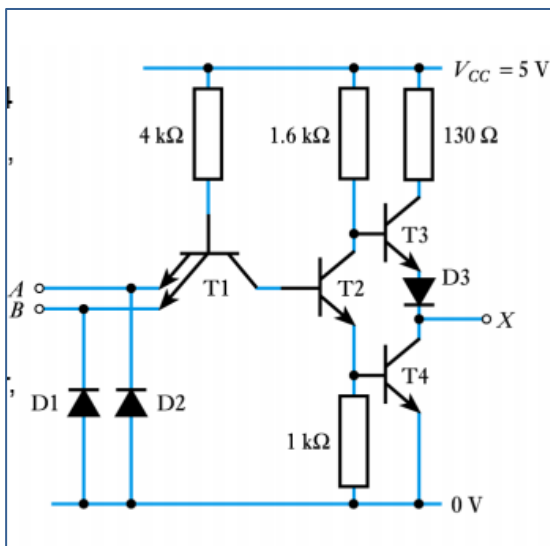
3. The most widely used families are:

3.1 Transistor Transistor Logic (TTL):

- based on bipolar transistors
- one of the most widely used families for small- and medium-scale devices rarely used for VLSI
- typically operated from 5V supply

Types of TTL:

It is a logic family implemented with bipolar process technology that combines or integrates NPN transistors, PN junction diodes and diffused resistors in a single monolithic structure to get the desired logic function. The NAND gate is the basic building block of this logic family



Truth Table
TTL
NAND Gate

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

1- Standard TTL

- typical gate propagation delay of 10ns
- and a power dissipation of 10 mW per gate,
- for a power–delay product (PDP) or switching energy of about 100 pJ

2-Low-power TTL (L)

- slow switching speed (33ns)
- reduction in power consumption (1 mW)
- (now essentially replaced by CMOS logic)

3- High-speed TTL (H)

- faster switching than standard TTL (6ns)
- but significantly higher power dissipation (22 mW)

4-Schottky TTL (S)

- used Schottky diode clamps at gate inputs to prevent charge storage and improve switching time.
- A Schottky diode has a very low forward-voltage drop of 0.15–0.45V approx (silicon diode has a voltage drop of 0.6–1.7V). This lower voltage drop can provide higher switching speed.
- Faster speed of (3ns) , but had higher power dissipation (19 mW)

Name	Abbreviation	Propagation Delay (ns)	Power dissipation(mw)	Speed Power Product (PJ)
Standard TTL	TTL	10	10	100
Low Power(TTL)	LTTL	33	1	33
High Speed TTL	HTTL	6	22	132
Schottky TTL	STTL	3	19	57
Low Power Schottky TTL	LSTTL	9.5	2	19

5- Low-power Schottky TTL (LS)

- used the higher resistance values of low-power TTL and the Schottky diodes to
- provide a good combination of speed (9.5ns), and reduced power consumption (2 mW), and PDP of about 20 pJ.

Note: The standard TTL gate was constructed with different resistor values to produce gate with lower dissipation or higher speed.

Note: The propagation delay of a saturated logic family depends largely on two factors, storage time and RC time constant.

3.2 Emitter-coupled logic (ECL)

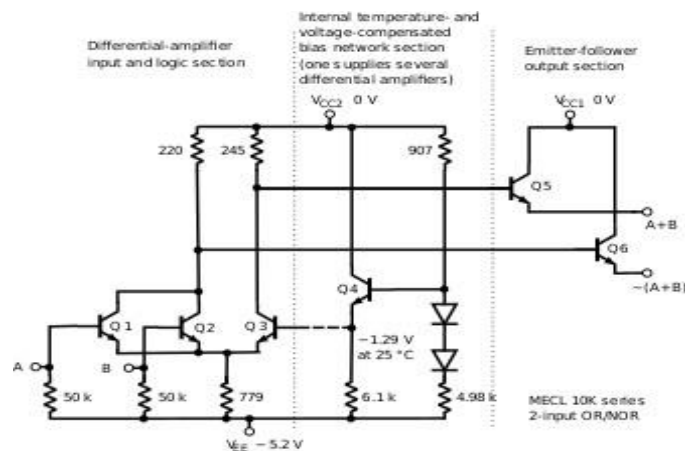
- based on bipolar transistors, but removes problems of storage time by preventing the transistors from saturating
- very fast operation - propagation delays of 1ns or less
- high power consumption, perhaps 60 mW/gate

Truth Table
ECL NOR gate

A	B	$Y = \overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

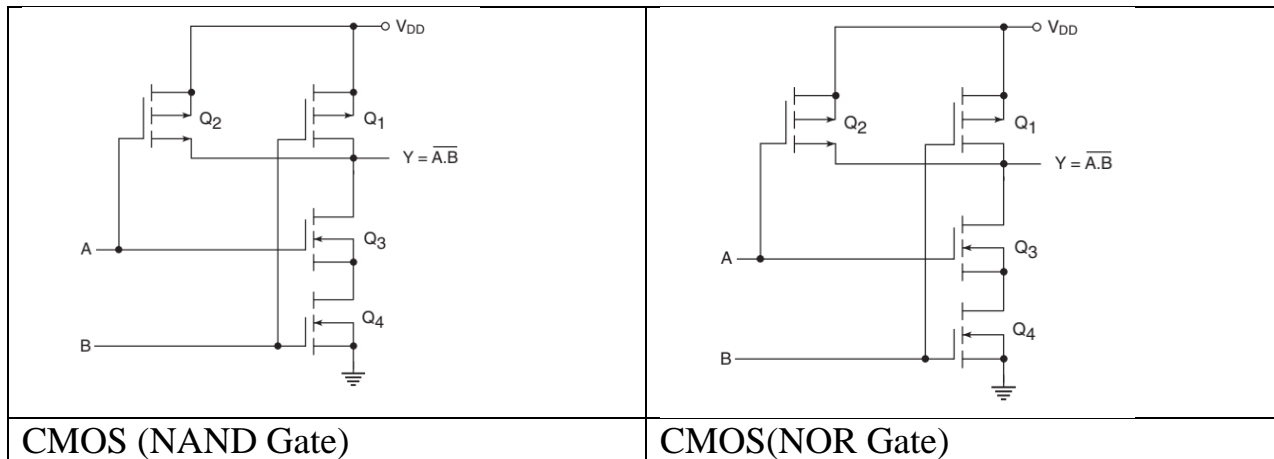
Characteristics:

- Propagation delay is very LOW(<1ns).
- ECL is fastest logic family.



3.3 Complementary metal oxide semiconductor (CMOS):

The logic families that use MOS devices as their basis are known as MOS families, and the prominent members belonging to this category are the PMOS family (using P-channel MOSFETs), the NMOS family (using N-channel MOSFETs) and the CMOS family (using both N- and P-channel devices) CMOS family



- Most widely used family for large-scale devices.
- Combines high speed with low power consumption.
- Usually operates from a single supply of 5 – 15 V.
- Excellent noise immunity of about 30% of supply voltage.
- Can be connected to a large number of gates (about 50).
- Many forms – some with t_{PD} down to 1 ns.
- Power consumption depends on speed (perhaps 1 mW).

Comparison between types of Logic families

	TTL	ECL	CMOS
Base Gate	NAND	OR/NOR	NAND/NOR
Fan-in	12-14	>10	>10
Fan-out	10	25	50
Power dissipation (mW)	10	175	0.001
Noise Margin	0.5V	0.16V (lowest)	1.5V (Highest)
Propagation Delay (ns)	10	<3 lowest	15 Highest
Noise immunity	Very good	Good	excellen

4. Semiconductor Memories:

A memory unit is a collection of cells capable of storing a large quantity of binary information, the process of storing a piece of data to a memory device is called writing, and the process of retrieving data is called reading.

There are two types of memories that are used in digital systems:

-Random-Access Memory (Ram)

-Read-Only Memory (ROM).

RAM stores new information for later use. The process of storing new information into memory is referred to as a memory write operation. The process of transferring the stored information out of memory is referred to as a memory read operation..

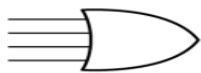
Random-access memory(RAM): perform both the write and read operations.

Read-only memory(ROM): perform only the read operation.

The read-only memory is a programmable logic device.

Other such units are the programmable logic array(PLA), the programmable array logic(PAL), and the field programmable gate array(FPGA).

- A typical programmable logic device may have hundreds to millions of gates interconnected through hundreds to thousands of internal paths.
- In order to show the internal logic diagram in a concise form, it is necessary to employ a special gate symbology applicable to array logic.



(a) Conventional symbol



(b) Array logic symbol

A

- **volatile.** Memory units that lose stored information when power is turned off , Both static and dynamic RAM, are of this category since the binary cells need external power to maintain the stored information.
- **Nonvolatile memory**, memory unit which retains its stored information after removal of power, such as magnetic disk, ROM.

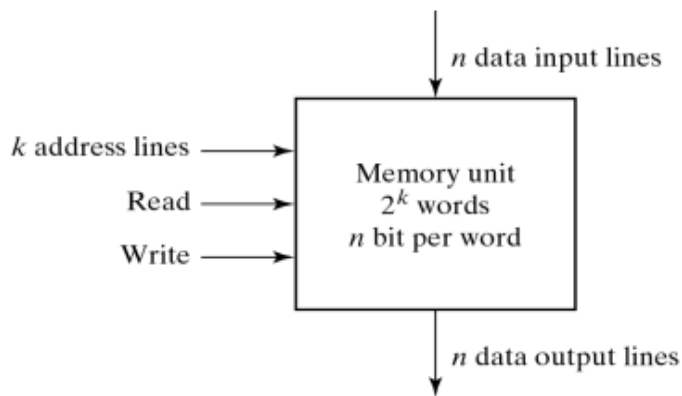
4.1 RAM

It is a type of memory which perform both the write and read operations and the memory unit stores binary information in groups of bits called words.

1 byte = 8 bits

1 word = 2 bytes

- The communication between a memory and its environment is achieved through data input and output lines, address selection lines, and control lines that specify the direction of transfer.



- Each word in memory is assigned an identification number, called an address, starting from 0 up to $2^k - 1$, where k is the number of address lines.
- The number of words in a memory with one of the letters

$K=2^{10}$, $M=2^{20}$, or $G=2^{30}$.

$64K = 2^{16}$ $2M = 2^{21}$, $4G = 2^{32}$

In random-access memory, the word locations may be thought of as being separated in space, with each word occupying one particular location.

In a random-access memory, the access time is always the same regardless of the particular location of the word.

Memory address		Memory contest
Binary	decimal	
0000000000	0	1011010101011101
0000000001	1	1010101110001001
0000000010	2	0000110101000110
	⋮	⋮
1111111101	1021	1001110100010100
1111111110	1022	0000110100011110
1111111111	1023	1101111000100101

Content of a 1024×16 Memory

Types of RAM :RAM units are available in two operating modes: **static** and **dynamic**:

- **Static RAM** consists essentially of internal latches that store the binary information.

1-The stored information remains valid as long as power is applied to the unit.

2- SRAM is easier to use and has shorter read and write cycles.

3- Low density, low capacity, high cost, high speed, high power consumption.

- **Dynamic RAM** stores the binary information in the form of electric charges on capacitors.

1- The capacitors are provided inside the chip by MOS transistors.

2- The capacitors tends to discharge with time and must be periodically recharged by refreshing the dynamic memory

3-DRAM offers reduced power consumption and larger storage capacity in a single memory chip.

4-High density, high capacity, low cost, low speed, low power consumption.

4.2 Read-Only Memory

A ROM is essentially a memory device in which permanent binary information is stored. The binary information must be specified by the designer and is then embedded in the unit to form the required interconnection pattern. Once the pattern is established, it stays within the unit even when power is turned off and on again.

A block diagram of a ROM is shown below. It consists of k address inputs and n data outputs.

- The number of words in a ROM is determined from the fact that k address input lines are needed to specify 2^k words.

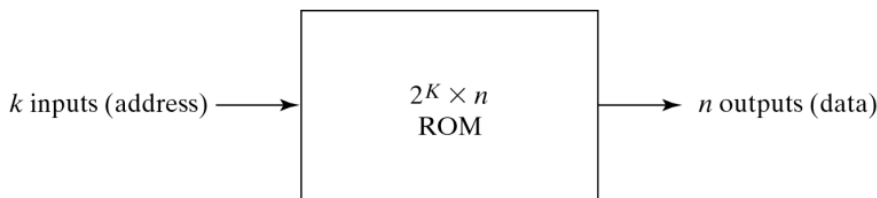


Fig. 7-9 ROM Block Diagram

- $k \times 2^k$ decoder input address
- n OR gates with 2^k input each
- Decoder output is connected to all n OR gates through fuses
- A $(2^k \times n)$ ROM will have an internal $k \times 2^k$ decoder and n OR gates.
- ROM $\rightarrow 2^k \times n$ programmable connections

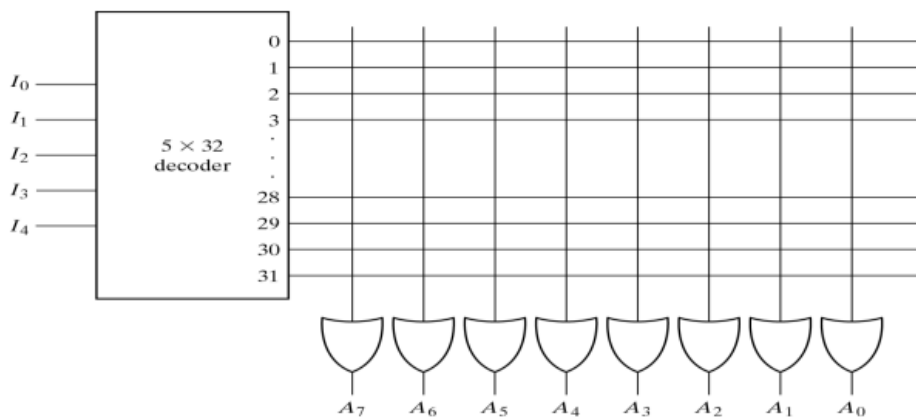


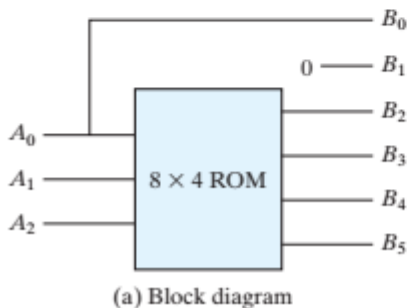
Fig. 7-10 Internal Logic of a 32×8 ROM

- A programmable connection between two lines is logically equivalent to a switch that can be altered to either be close or open.
- Intersection between two lines is sometimes called a crosspoint.

Example 3 : Design a combinational circuit using a ROM. The circuit accepts a 3-bit number and generates an output binary number equal to the square of the input number

Truth Table for Circuit of Example 7.1

Inputs			Outputs						Decimal
A_2	A_1	A_0	B_5	B_4	B_3	B_2	B_1	B_0	
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1	1
0	1	0	0	0	0	1	0	0	4
0	1	1	0	0	1	0	0	1	9
1	0	0	0	1	0	0	0	0	16
1	0	1	0	1	1	0	0	1	25
1	1	0	1	0	0	1	0	0	36
1	1	1	1	1	0	0	0	1	49



A_2	A_1	A_0	B_5	B_4	B_3	B_2
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	1
0	1	1	0	0	1	0
1	0	0	0	1	0	0
1	0	1	0	1	1	0
1	1	0	1	0	0	1
1	1	1	1	1	0	0

(b) ROM truth table

Types of ROMs

The type of ROM is determined by the way the switches are set or reset (i.e., programmed). The required paths in a ROM may be programmed in four different ways.

1. Mask programming: fabrication process

2. Read-only memory or PROM: blown fuse /fuse intact

3. Erasable PROM or EPROM: placed under a special ultraviolet light for a given period of time will erase the pattern in ROM.

4- UV EPROMs: You can recognize the UV EPROM device by the transparent quartz lid on the package. Erasure is done by exposure of the memory array chip to high-intensity ultraviolet

radiation through the quartz window on top of the package.

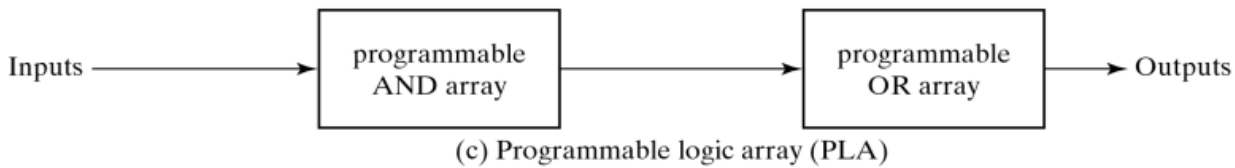
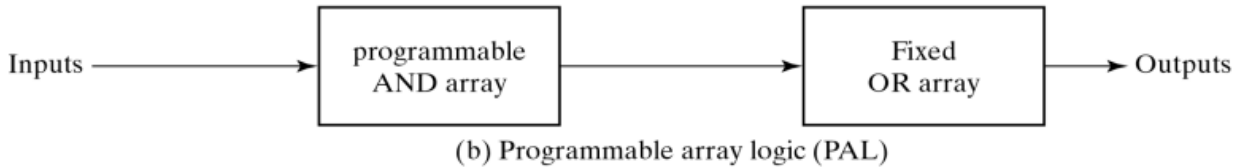
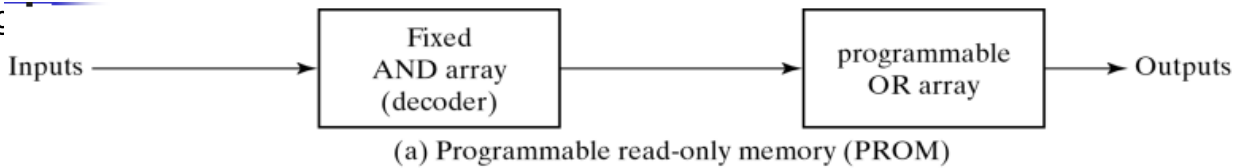
4. Electrically-erasable PROM(EEPROM): erased with an electrical signal instead of ultraviolet light.

5.Programmable Logic Devices (PLDs)

There are two broad categories of logic devices, namely fixed logic devices and programmable logic devices. Whereas a fixed logic device such as a logic gate or a multiplexer or a flip-flop performs a given logic function that is known at the time of device manufacture, a PLD can be configured by the user to perform a large variety of functions. The three major types of programmable logic are SPLD, CPLD, and FPGA..A combinational PLD is an integrated circuit with programmable gates divided into an AND array and an OR array to provide an AND-OR sum of product implementation.

- **PROM:** fixed AND array constructed as a decoder and programmable OR array.
- **PAL:** programmable AND array and fixed OR array.
- **PLA:** both the AND and OR arrays can be programmed

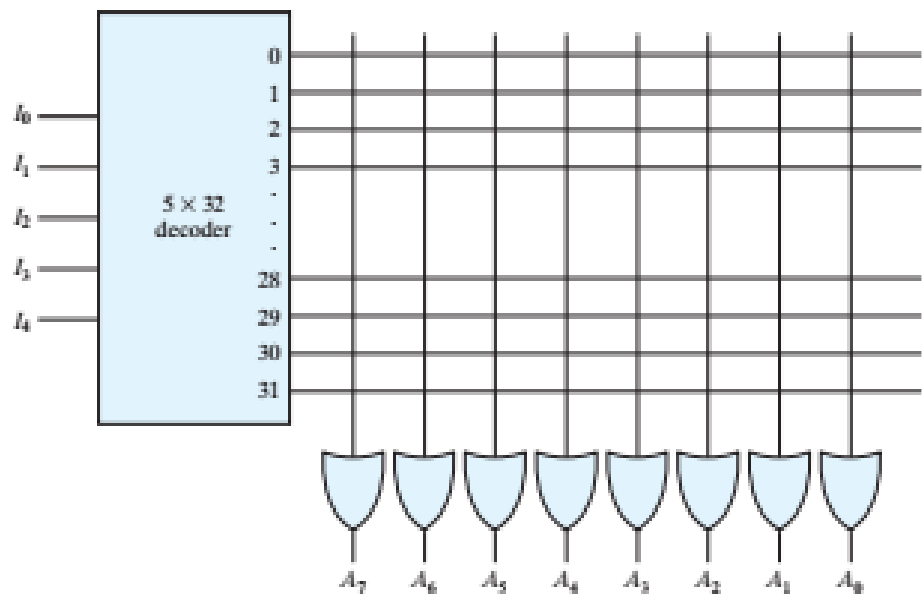
All PLDs consist of programmable arrays. A programmable array is essentially a grid of conductors that form rows and columns with a fusible link at each cross point. Arrays can be either fixed or programmable. The earliest type of programmable array, dating back to the 1960s, was a matrix with a diode at each cross point of the matrix.



- **FPGAs** (field-programmable gate arrays) have the greatest logic capacity in PLD. FPGAs consist of an array of anywhere from sixty-four to thousands of logic-gate groups that are sometimes called logic blocks. Two basic classes of FPGA are course-grained and fine-grained. The course-grained FPGA has large logic blocks, and the finegrained FPGA has much smaller logic blocks. FPGAs come in packages ranging up to 1000 pins or more.

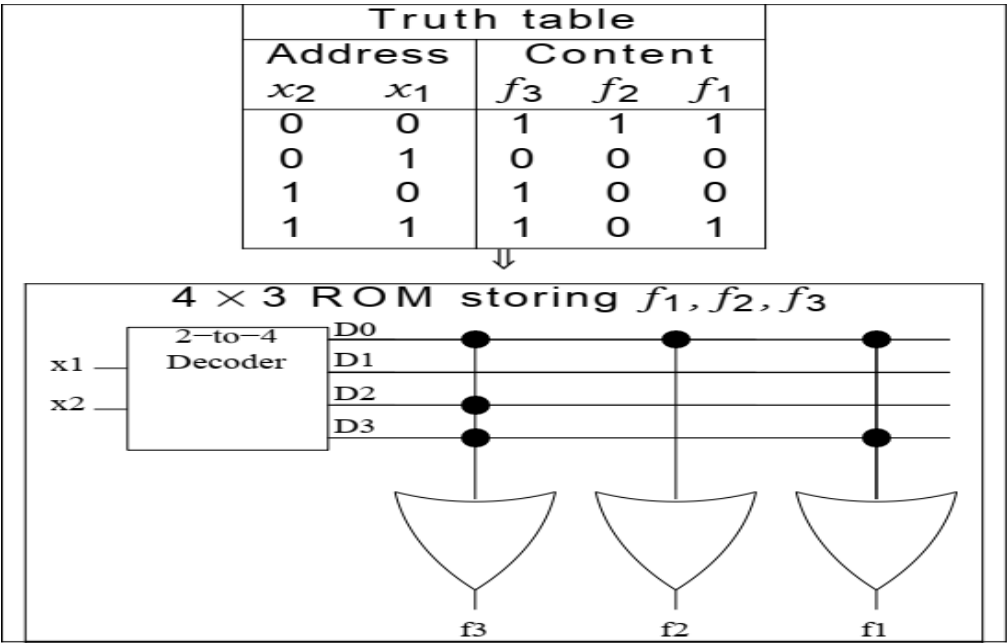
5.1 Programmable Read-Only Memory (PROM)

The **PROM** consists of a set of fixed (nonprogrammable) AND gates connected as a decoder and a programmable OR array, as shown in the generalized block diagram of The PROM is used primarily as an addressable memory and not as a logic device because of limitations imposed by the fixed AND gates.



Internal logic of a 32 x 8 ROM

Example 4 : Implement $f_1(x_2, x_1) = \Sigma 0,3$, $f_2(x_2, x_1) = \Sigma 0$, and $f_3(x_2, x_1) = \Sigma 0, 2,3$, using a 4×3 R O M



5.2 Programmable Logic Array (PLA)

The **PLA** is a PLD that consists of a programmable AND array and a programmable OR array.

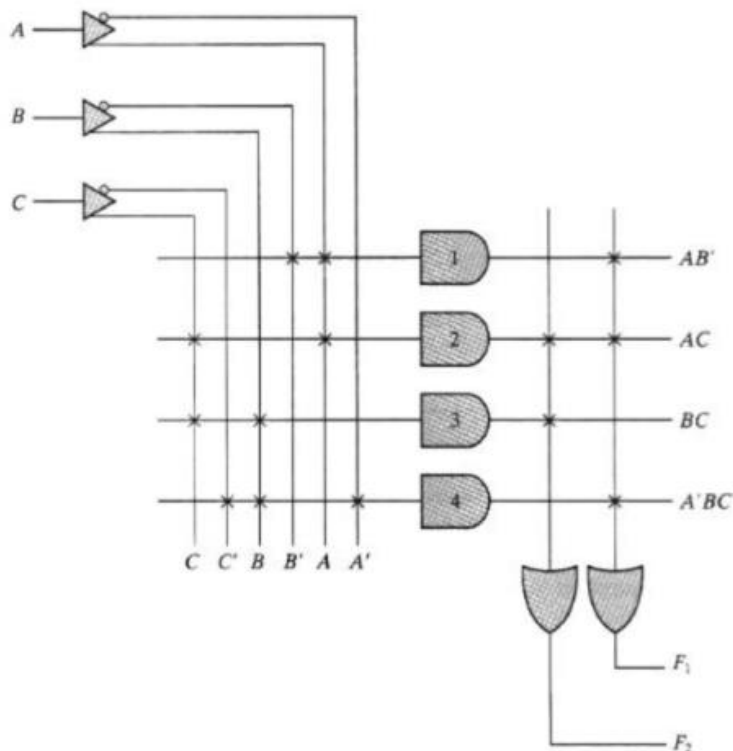
The PLA was developed to overcome some of the limitations of the PROM.

The PLA is also called an FPLA (field-programmable logic array) because the user in the field, not the manufacturer, programs it.

Logic expressions for content information to be stored in P L A must be obtained first, then minimized, and finally programmed into the P L A using a P L A program table. Each input goes through a buffer-inverter combination, shown in the diagram with a composite graphic symbol, that has both the true and complement outputs. Each input and its complement is connected to the inputs of each AND gate, as indicated by the intersections between the vertical and horizontal lines. The outputs of the AND gates are connected to the inputs of each OR gate.

Example 5: Implement the following functions using PLA

$$F_1 = AB' + AC + A'BC', F_2 = AC + BC$$

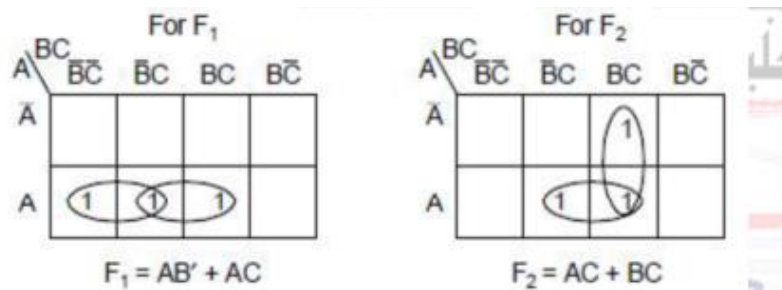


Example 6: Implement the following functions using 3-input, 3 product terms and 2-output PLA, $F_1 = \Sigma (4, 5, 7)$, $F_2 = \Sigma (3, 5, 7)$

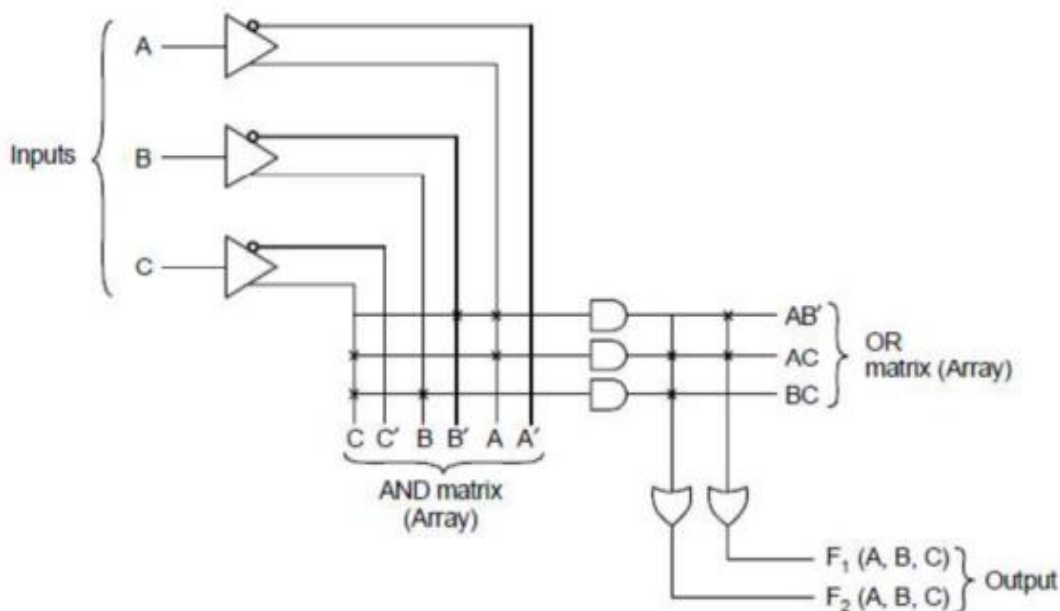
Solution: Step 1. Derive the truth table for the combinational circuit

Inputs			Outputs	
A	B	C	F_2	F_1
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	0
1	0	0	0	1
1	0	1	1	1
1	1	0	0	0
1	1	1	1	1

Step 2. Simplify functions using K-map.



Step 3. Draw the logic diagram



5.3 Programmable Array Logic (PAL)

The **PAL** is a PLD that was developed to overcome certain disadvantages of the PLA, such as longer delays due to the additional fusible links that result from using two programmable arrays and more circuit complexity.

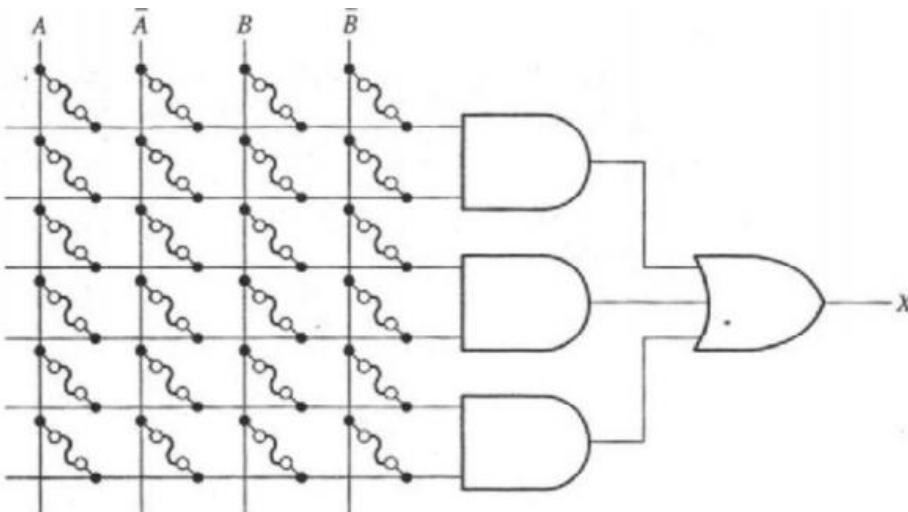
The basic PAL consists of a programmable AND array and a fixed OR array with output logic , this structure allows any sum-of-products (SOP) logic expression with a defined number of variables to be implemented

The PAL is the most common one-time programmable logic device (OTP) and is implemented with bipolar technology (TTL or ECL).

The basic structure of a PAL is illustrated in Figure (14) for two input variables and one output although most PALs have many inputs and many outputs.

A programmable array is essentially a grid of conductors forming rows and columns **with a fusible link at each cross point**. Each fused cross point of a row and column is called a cell and is the programmable element of a PAL. Each row is connected to the input of an AND gate and each column is connected to an input variable or its complement.

The number of product terms in each section is fixed, and if the number of terms in the function is too large, it may be necessary to use two sections to implement one Boolean function



Example 7 : using a PAL in the design of a combinational circuit, consider the following Boolean functions, given in sum-of-minterms form:

$$w(A, B, C, D) = g(2, 12, 13)$$

$$x(A, B, C, D) = g(7, 8, 9, 10, 11, 12, 13, 14, 15)$$

$$y(A, B, C, D) = g(0, 2, 3, 4, 5, 6, 7, 8, 10, 11, 15)$$

$$z(A, B, C, D) = g(1, 2, 8, 12, 13)$$

Simplifying the four functions to a minimum number of terms results in the following Boolean functions:

$$w = ABC' + A'B'CD'$$

$$x = A + BCD$$

$$y = A'B + CD + B'D'$$

$$z = ABC' + A'B'CD' + AC'D' + A'B'C'D$$

$$= w + AC'D' + A'B'C'D$$

PAL Programming Table

Product Term	AND Inputs					Outputs
	A	B	C	D	w	
1	1	1	0	—	—	$w = ABC' + A'B'CD'$
2	0	0	1	0	—	
3	—	—	—	—	—	
4	1	—	—	—	—	$x = A + BCD$
5	—	1	1	1	—	
6	—	—	—	—	—	
7	0	1	—	—	—	$y = A'B + CD + B'D'$
8	—	—	1	1	—	
9	—	0	—	0	—	
10	—	—	—	—	1	$z = w + AC'D' + A'B'C'D$
11	1	—	0	0	—	
12	0	0	0	1	—	

