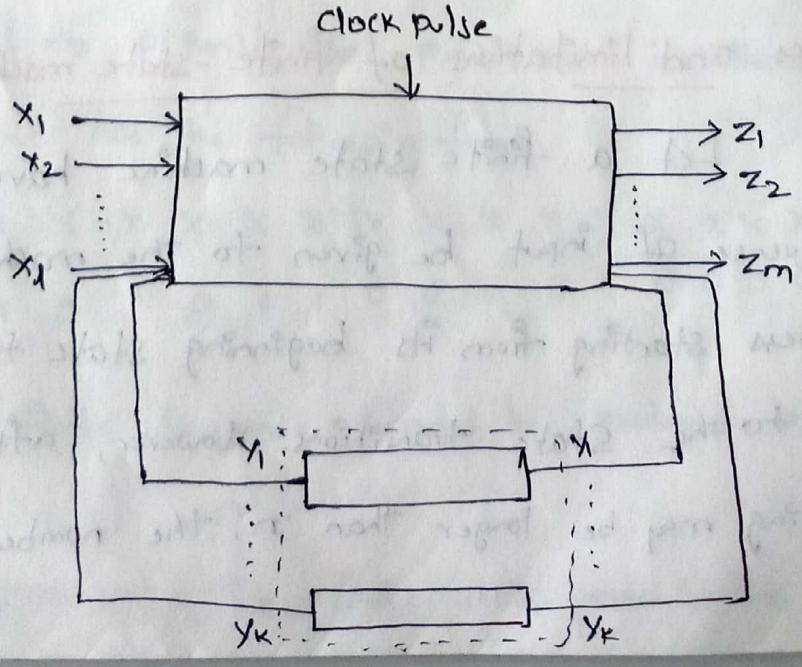
Finite state machines and programmable logic Devices

Finite state machine:-

Finite state machine can be defined as a type of machine whose past histories can affect its future behaviour in a finite number of ways. To clarify, consider for example of binary full adder. Its output depends on the present input and the carry generated from the previous input. It may have a large number of previous input histories but they can be divided into two types (i) Input (ii) output.

The most general model of a sequential circuit has inputs, outputs and internal states. A sequential circuit is referred to as a finite state machine (FSM). A finite state machine is abstract model that describes the synchronous sequential machine. The fig. shows the block diagram of a finite state model. $x_1, x_2, \ldots x_l$ are inputs. $z_1, z_2, \ldots z_m$ are outputs. $Y_1, Y_2, \ldots Y_k$ are state variables and $Y_1, Y_2, \ldots Y_k$ represent the next state.



Types of FSM:-

Finite state machines are two types. They differ in the way the output is generate they are:-

1. Mealy type model: In this model, the output is function of the present state and the present input.
2. Moore type model: In this model, the output is a function of the present state only.

Mathematical representation of Synchronous Sequential machines

The relation between the present state S(t), present input X(t), and next state S(t+1) can be given as

$$S(t+1) = f\{S(t), X(t)\}$$

The value of output Z(t) can be given as

$$Z(t) = g\{S(t), X(t)\} \quad : \text{for mealy model}$$

$$Z(t) = G\{S(t)\} \quad : \text{for moore model.}$$

Because, in a mealy machine, the output depends on the present state and input, where as in a moore machine, the output depends only the present state.

capabilities and limitations of finite-state machine:-

Let a finite state machine have 'n' states. Let a long sequence of input be given to the machine. The machine will progress starting from its beginning state to the next state according to the state transitions. However, after sometime the input string may be longer than 'n', the number of states.

As there are only 'n' states in the machine, it must come to a state it was previously been in and from this phase if the input remains the same, the machine will function in a periodically repeating fashion. from here a conclusion that for a 'n' state machine the output will become periodic after a number of clock pulses less than equal to 'n' can be drawn. States are memory elements. As for a finite state machine the number of states is finite, so finite number of memory elements are required to design a finite state machine.

Limitations:-

1. Periodic sequence and limitations of finite states:- with n-state machines, we can generate periodic sequences of n states are smaller than 'n' states. For example, in a 6-state machine, we can have a maximum Periodic sequence as 0,1,2,3,4,5,0,1.....

2. No infinite sequence:- Consider an infinite sequence such that the output is '1' when and only when the numbers of inputs received so far is equal to $P(P+1)/2$ for $P=1,2,3...$ i.e, the desired input-output sequence has the following form:

Input: x x x x x x x x x x x x x x x x x x x x x x

output: 1 0 1 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1

Such an infinite sequence cannot be produced by a finite state machine.

3. Limited memory:- The finite state machine has a limited memory

and due to limited memory it cannot produce certain outputs. Consider a binary multiplier circuit for multiplying two arbitrarily large binary numbers. the memory is not sufficient to store arbitrarily large partial products resulted during multiplication.

Comparison between Moore machine and mealy machine:-

Moore machine

1. Its output is a function of present state only $z(t) = g\{s(t)\}$.
2. Input changes do not affect the output.
3. It requires more number of states for implementing same function.

Mealy machine

1. Its output is a function of present state as well as present input $z(t) = g\{s(t), x(t)\}$.
2. Input changes may affect the output of the circuit.
3. It requires less number of states for implementing same function.

Mealy Model:-

when the output of the sequential circuit depends on the both the present state of the flip-flops and on the inputs, the sequential circuit is referred to as mealy circuit (or) mealy machine.

The fig. shows the logic diagram of the mealy model. Notice that the output depends up on the present state as well as the present inputs. we can easily realize that changes in the input during the clock pulse cannot affect the state of the flip-flop. They can affect the output of the circuit.

If the input variations are not synchronized with a clock, the derived output will also not be synchronized with the clock and we get false output. The false outputs can be eliminated by allowing input to change only at the active transition of the clock.

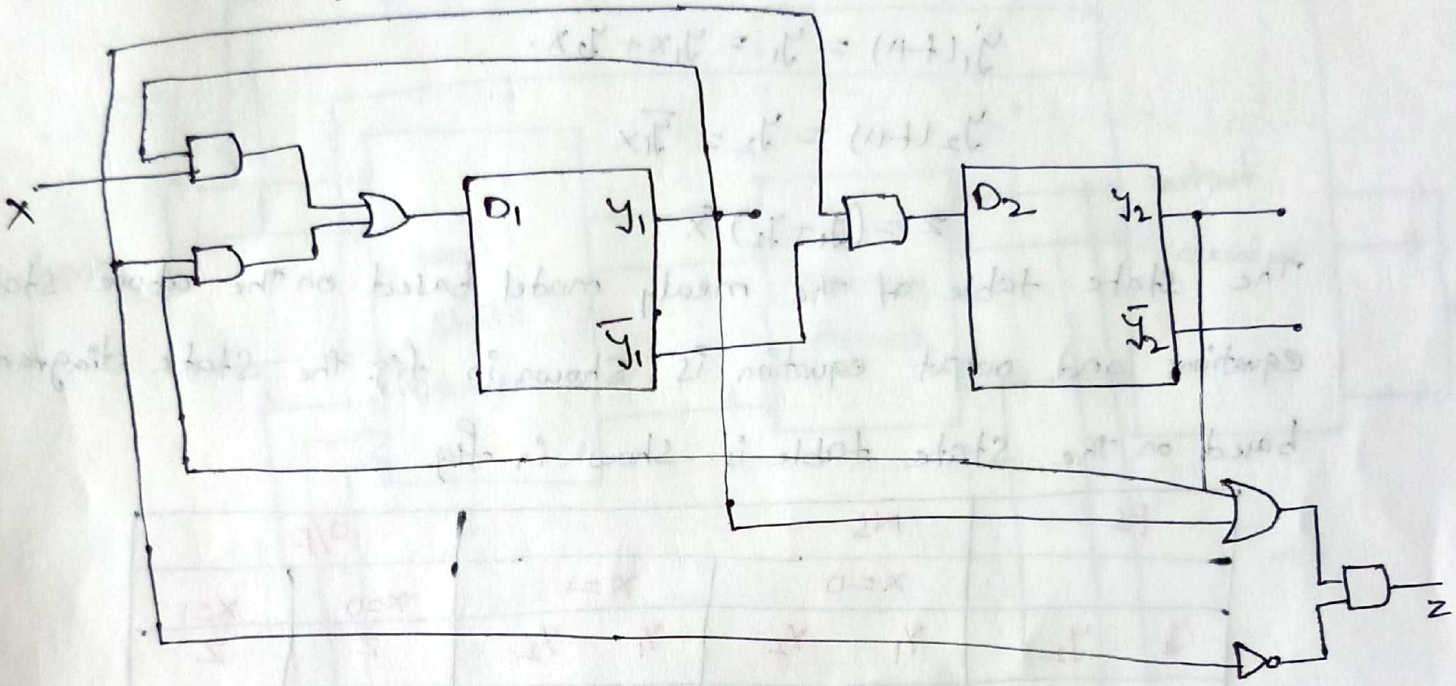


fig:- Logic diagram of a mealy model.

The behaviour of a clocked sequential circuit can be described algebraically by means of state equations. A state equation specifies the next state as a function of the present state and inputs. The mealy model shown in fig. consists of two 'D' flip-flops, an input x and an output z. Since the 'D' input of a flip-flop determines the value of the next state, the state equations for the model can be written as

$$y_1(t+1) = y_1(t)\,x(t) + y_2(t)\,x(t).$$

$$y_2(t+1) = y_1(t)\,x(t)$$

and the output equation is

$$z(t) = \{y_1(t) + y_2(t)\}\,x'(t)$$

where $y(t+1)$ is the next state of the flip flop one clock edge later, $x(t)$ is the present input and $z(t)$ is the present output. If $y_1(t+1)$ are represented by $y_1(t)$ and $y_2(t)$, in more compact form, the equations are

$$Y_1(t+1) = Y_1 = Y_1x + Y_2x.$$

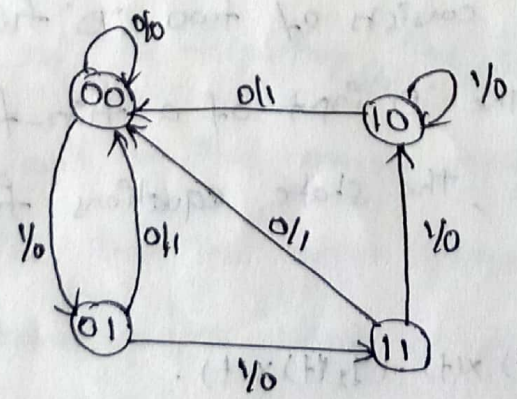
$$Y_2(t+1) = Y_2 = \bar{Y}_1x$$

$$z = (Y_1 + Y_2)\bar{x}.$$

The state table of the mealy model based on the above state equations and output equation is shown in fig. the state diagram based on the state table is shown in fig.

PS		NS				O/P	
		x=0		x=1		x=0	x=1
Y_1	Y_2	Y_1	Y_2	Y_1	Y_2	z	z
0	0	0	0	0	1	0	0
0	1	0	0	1	1	1	0
1	0	0	0	1	0	1	0
1	1	0	0	1	0	1	0

fig:- (a) State table.



(b). State diagram.

In general form, the mealy circuit can be represented with its block schematic as shown in below fig.

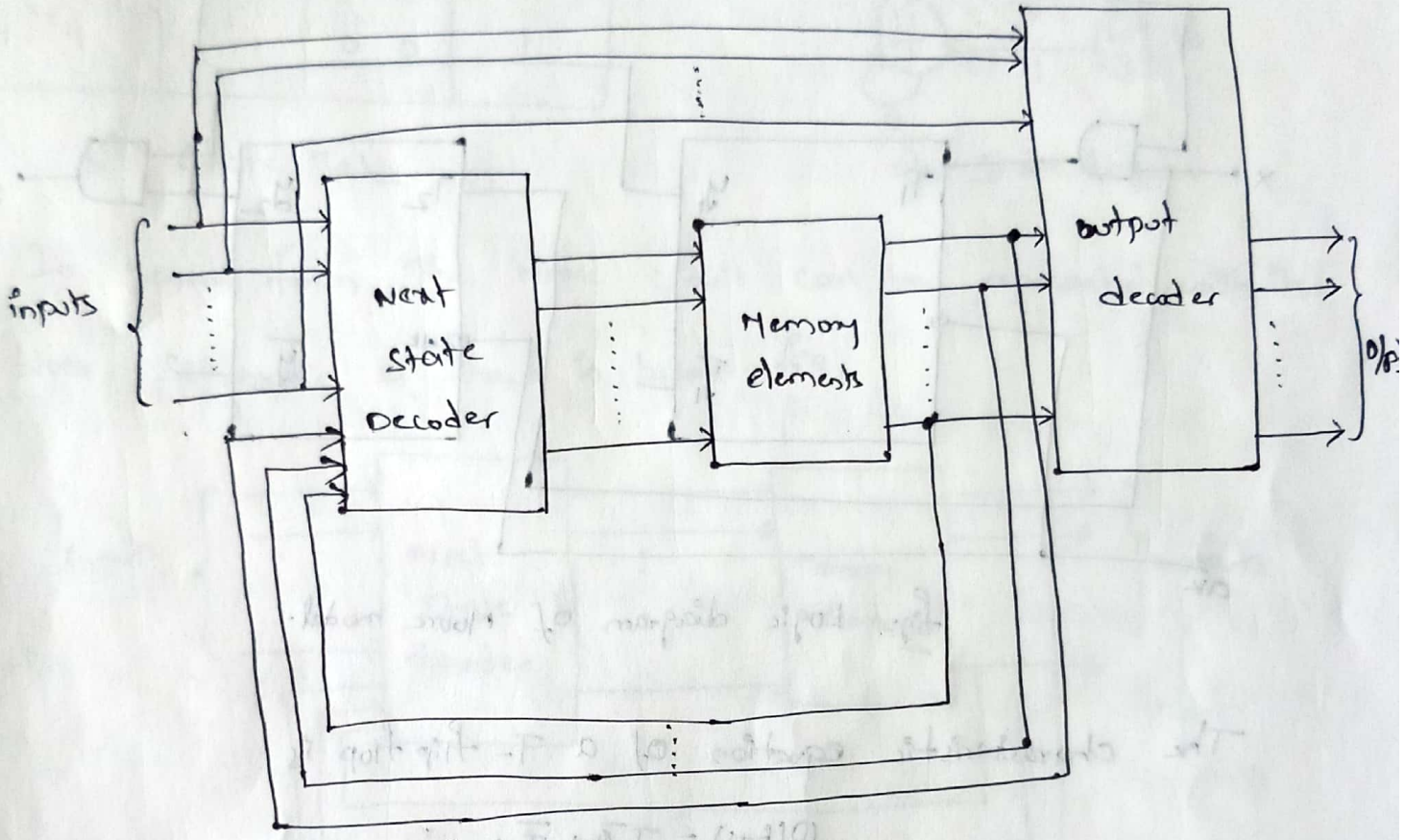


fig:- Schematic of mealy model.

Moore Model:-

When the output of the sequential circuit depends upon only on the present state of the flip-flop, the sequential circuit is referred as to as the moore circuit or the moore machine.

Notice that the output depend only on the present state. It does not depend upon the input at all. The input is used only to determine the inputs of flip-flops. It is not used to determine the output. The circuit shown has two T flip-flops, one input x, and one output z. It can be described algebraically by two input equations an output equation.

$$T_1 = y_2 x$$

$$T_2 = x$$

$$z = y_1 y_2$$

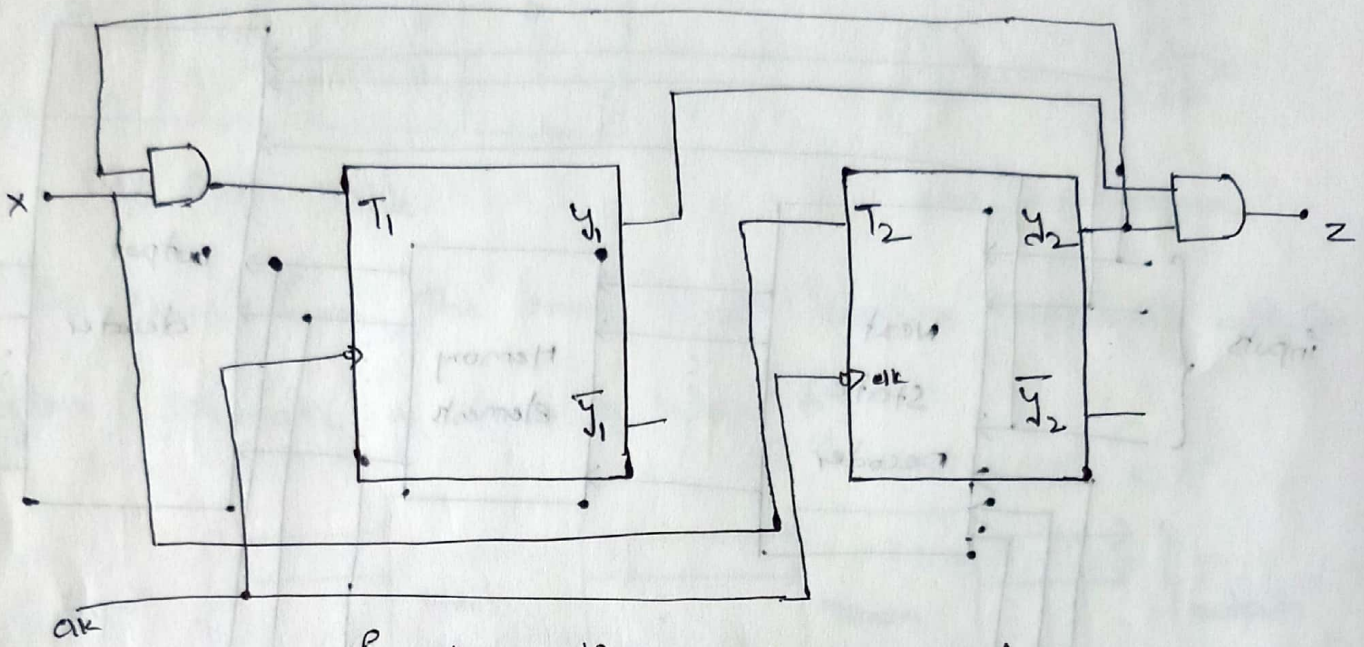


fig:- Logic diagram of Moore model.

The characteristic equation of a T-flip flop is

$$Q(t+1) = T\overline{Q} + \overline{T}Q$$

The values for the next state can be derived from the state equations by substituting T_1 and T_2 in the characteristic equation yielding

$$Y_1(t+1) = Y_1 = (y_2 x) \oplus = (y_2 x) y_1 + (y_2 x) y_1 x$$

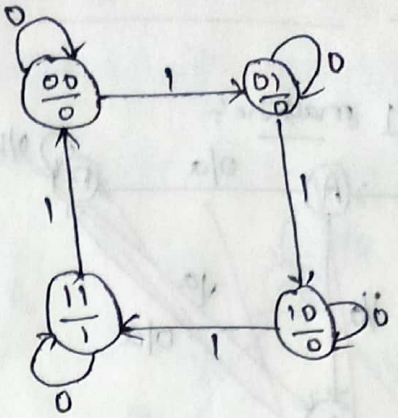
$$= y_1 y_2 + y_1 x + y_1 y_2 x$$

$$= y_2(t+1) = x \oplus y_2 = x y_2 + x y_2.$$

The state table of the Moore model based on the above state equations and output equation is shown in below fig.

PS		NS				O/P
		x=0		x=1		
y_1	y_2	y_1	y_2	y_1	y_2	z
0	0	0	0	0	1	0
0	1	0	1	1	0	0
1	0	1	0	1	1	0
1	1	1	1	0	0	1

(a) State Table.



(b) State diagram

In general form, the Moore circuit can be represented with its block schematic as shown in below fig.

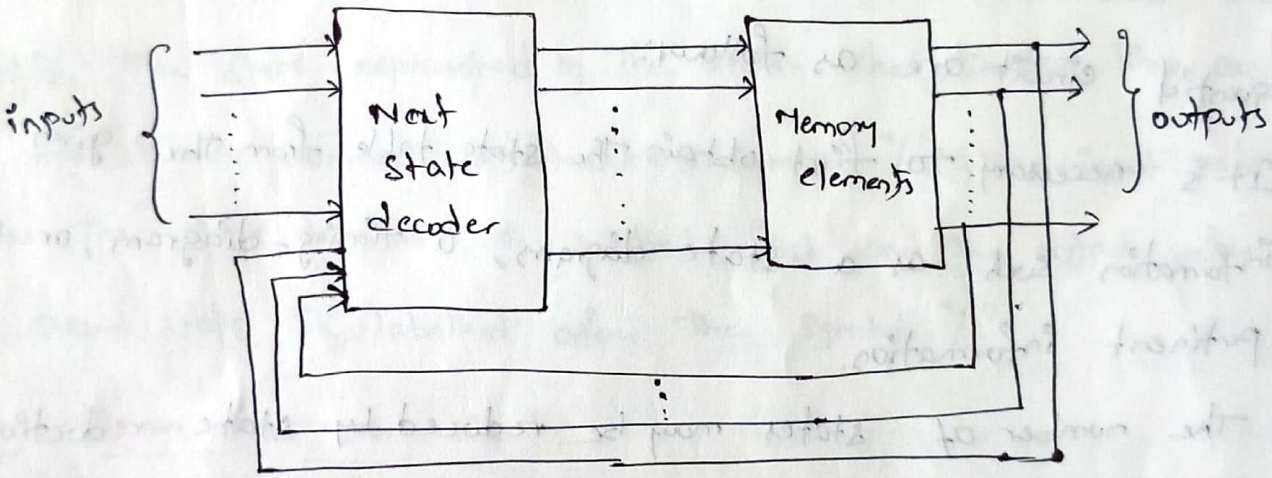


fig:- Moore circuit model.

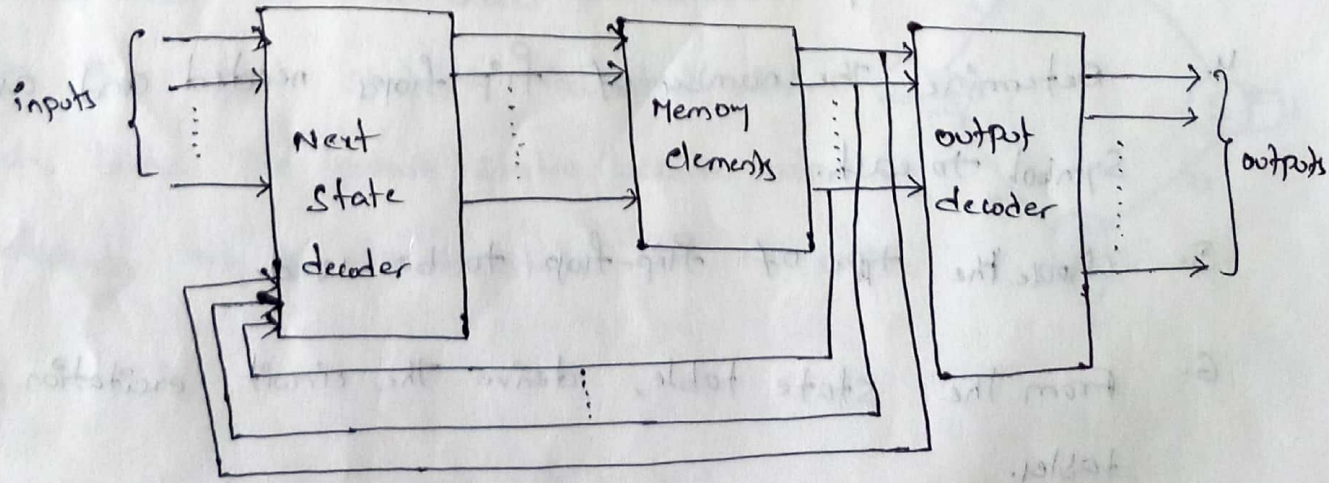


fig:- Moore circuit model with an output decoder.

Basic design steps:-

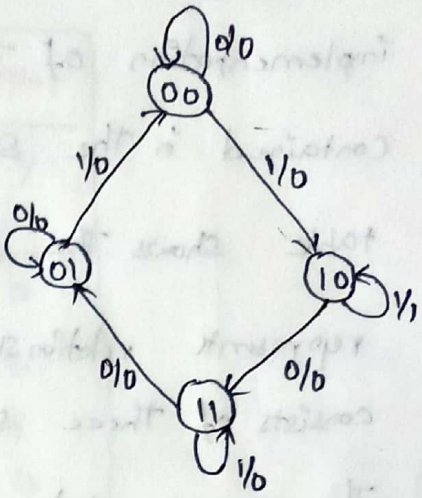
The recommended steps for the design of a clocked synchronous Sequential circuit are as follows:

1. It is necessary to first obtain the state table from the given circuit information such as a state diagram, a timing-diagram, or other pertinent information.
2. The number of states may be reduced by state reduction technique if the sequential circuit can be categorized by input-output relationships independent of the number of states.
3. Assign binary values to each state in the state table.
4. Determine the number of flip-flops needed and assign a letter symbol to each.
5. Choose the type of flip-flop to be used.
6. From the state table, derive the circuit excitation and output tables.
7. Using the K-map or any other simplification method, derive the circuit output functions and the flip-flop input functions.

8. Draw the logic diagram

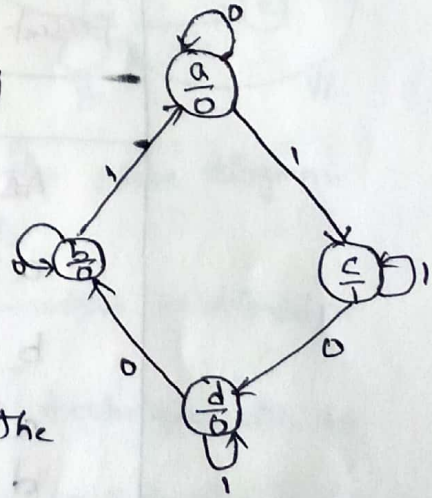
State diagram for Mealy circuit:-

State diagram is a pictorial representation of a behaviour of sequential circuit. The state is represented by the circle, and the transition between states is indicated by directed lines connecting the circles. A directed line connecting the circle with itself indicates that next state is same as present state. The binary number inside each circle identifies the state represented by the circle. The directed lines are labelled with two binary numbers seperated by a symbol "/". The input value that causes the state transition is labelled first and the output value during the present state is labelled after the symbol "/".



State diagram for Moore circuit:-

In case of Moore circuit, the directed lines are labelled with only one binary number representing the state of the input that causes the state transition. The output state is indicated within the circle, below the present state because output state depends only on present state and not on the input.



State Table:-

Although the state diagram provides a description of the behaviour of a sequential circuit that is easy to understand, to proceed with the implementation of the circuit, It is convenient to translate the information contained in the state diagram into a tabular form. The following table shows the state table for state diagram of Mealy circuit. It represents relationship between input, output and flip-flop states. It consists of three sections labelled present state, next state and output. The present state designates the state of flip-flops before the occurrence of a clock pulse. The next state is the state of flip-flop after the application of clock pulse, and the output section gives the values of the output variables during the present state. Both the next state and output sections have two columns representing two possible input conditions : $X=0$ and $X=1$.

present state	next state		output	
	$X=0$	$X=1$	$x=0$	$X=1$
AB	AB	AB	Y	Y
a	a	c	0	0
b	b	a	0	0
c	d	c	0	1
d	b	d	0	0

Table:- State Table for Mealy circuit.

In case of Moore circuit the output section has only one column since output does not depend on input. The following table shows state table for Moore circuit whose state diagram is shown in above.

present state	Next state		output
	x=0	x=1	y
AB	AB	AB	
a	a	e	0
b	b	a	0
c	d	c	1
d	b	d	0

State Reduction:-

The state reduction technique basically avoids the introduction of redundant states. The reduction in redundant states reduce the number of required flip-flops and logic gates, reducing the cost of the final circuit. The two states are said to be redundant or equivalent, if every possible set of inputs generate exactly same output and same next state. when two states are equivalent, one of them can be removed without altering input-output relationship.

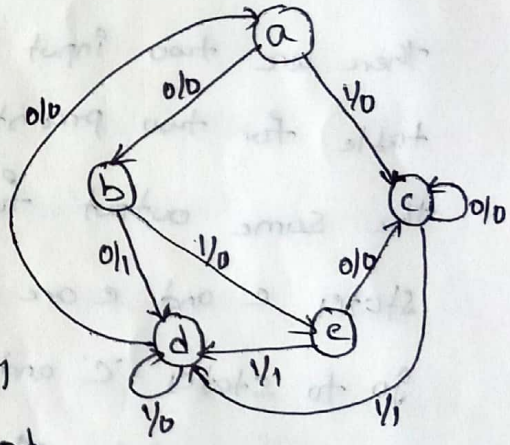


fig: state diagram.

Let us illustrate the state reduction technique with an example. we start with a sequential circuit whose specification is given in the state diagram. As shown in the diagram, the states are denoted by letter symbols instead of their binary values, because in state reduction technique internal states are not important; but only input-output sequences are important.

Step 1:- Determine the state table for given state diagram.

Present state	Next state		output	
	X=0	X=1	X=0	X=1
a	b	c	0	0
b	d	e	1	0
c	c	d	0	1
d	a	d	0	0
e	c	d	0	1

Table:- State table.

Step 2:- Find equivalent states

As mentioned earlier, in equivalent states every possible set of inputs generate exactly same output and same next state. In the given circuit there are two input combinations X=0 and X=1. Looking at the state table for two present states that go to the same next state and have the same output for both input combinations, we can easily find that states c and e are equivalent. This is because 'c' and 'e' both states go to states 'c' and 'd' and have outputs of 0 and 1 for X=0 and X=1, respectively. Therefore, state 'e' can be removed and replaced by 'c'. The final reduced table is shown in below.

Present state	Next state		output	
	X=0	X=1	X=0	X=1
a	b	c	0	0
b	d	c	1	0
c	c	d	0	1
d	a	d	0	0

Table:- Reduced State Table

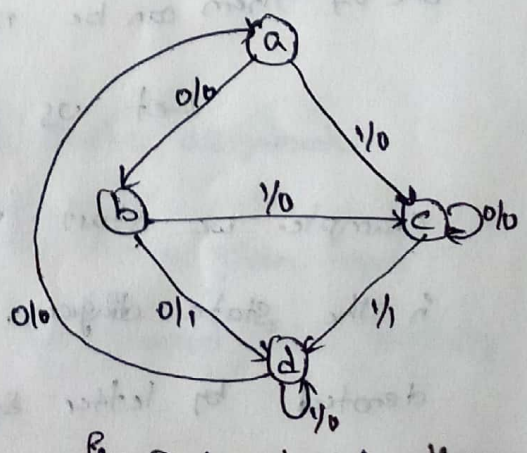


fig:- Reduced state diagram

State Assignment:-

In sequential circuits we know that the behaviour of the circuit is defined interms of its inputs, present states, next states and outputs. To generate desired next state at particular present state and inputs, it is necessary to have specific flip-flop inputs. These flip-flop inputs are described by a set of Boolean functions called flip flop input functions. To determine the flip-flop input functions, it is necessary to represent states in the state diagram using binary values instead of alphabets. This procedure is known as state assignment. we assign binary values to the states in such a way that it is possible to implement flip-flop input functions using minimum logic gates.

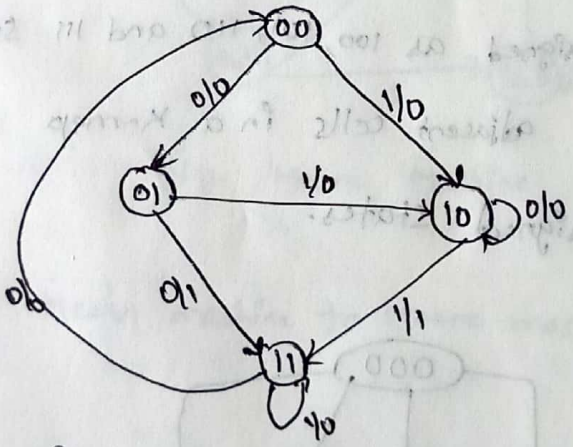


fig:- State diagram with binary states

Rules for state assignments:-

There are two basic rules for making state assignments.

Rule 1:- states having the same NEXT STATES for a given input condition should have assignments which can be grouped into logically adjacent cells in a K-map.

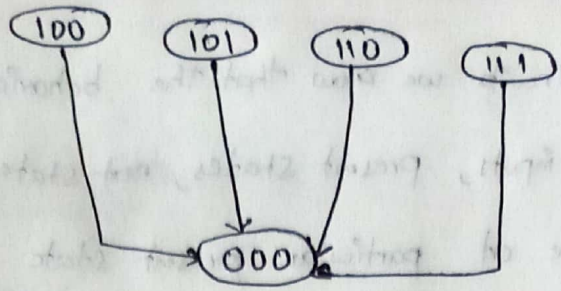


fig:- Example of using Rule 1

As shown in the fig. there are four states whose next state is same. Thus state assignments for these states are 100, 101, 110 and 111, which can be grouped into logically adjacent cells in a K-map.

Rule 2:-

States that are the NEXT STATES of a single state should have assignment which can be grouped into logically adjacents cells in a K-map.

As shown in the fig. for state 000, there are four next states. These states are assigned as 100, 101, 110 and 111 so that they can be grouped into logically adjacent cells in a K-map and table shows the state table with assigned states.

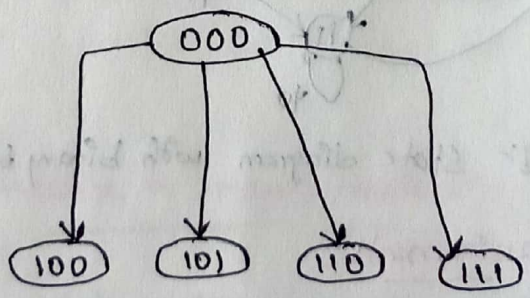


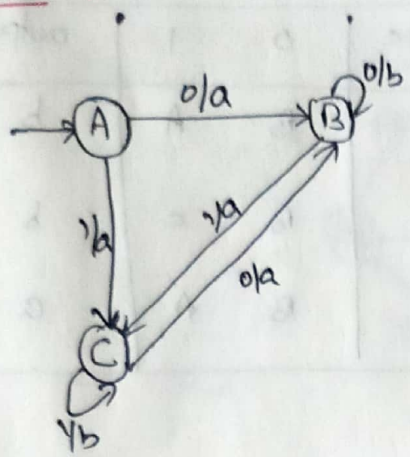
fig: Example of using Rule 2.

present state	Next state		output	
	x=0	x=1	x=0	x=1
00	01	10	0	0
01	11	10	1	0
10	10	11	0	1
11	00	11	0	0

Table: state table with assigned states.

Conversion of Mealy Machine to Moore Machine :-

Example :-1



inputs $\Sigma = \{0,1\}$

outputs $\Delta = \{a,b\}$

fig. Mealy machine.

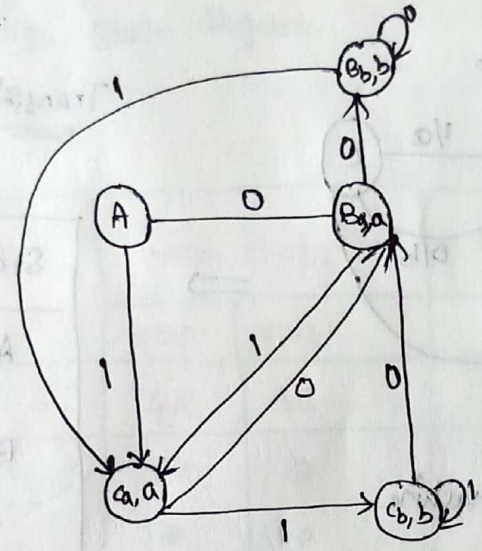


fig. Moore Machine.

when we convert Mealy machine to Moore machine, the number of states is increased.

Conversion of Moore machine to Mealy machine :-

Construct a Moore that prints 'a' whenever the sequence '01' is encountered in any input binary string and then convert it to its equivalent Mealy machine.

inputs $\Sigma = \{0,1\}$

outputs $\Delta = \{a,b\}$

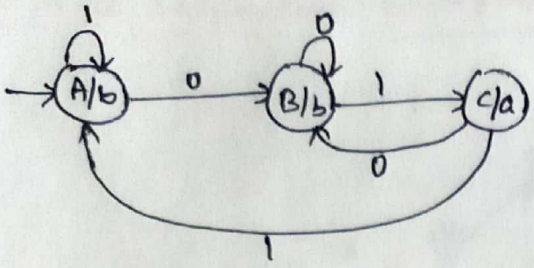


fig: Moore Machine

⟹

Transition Table:-

State	0	1	output
A	B	A	b
B	B	C	b
C	B	A	a

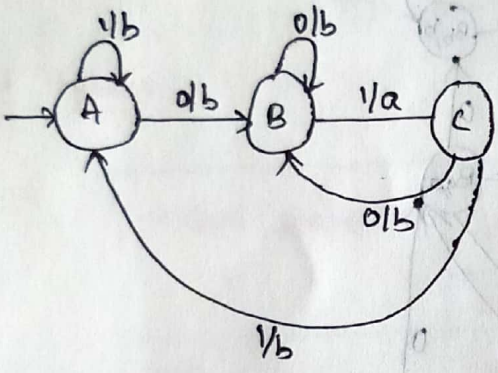


fig:- Mealy Machine.

⟹

Transition Table:-

State	0	1	output
A	B,b	A,b	b
B	B,b	C,a	b
C	B,b	A,b	a

Design a sequential circuit using flip flops:-

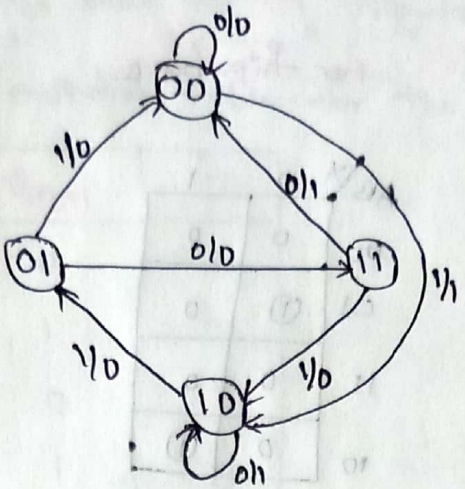


fig:- State diagram.

State table:-

present state		Next state		output	
		X=0	X=1	X=0	X=1
A	B	AB	AB	Y	Y
0	0	00	10	0	1
0	1	11	00	0	0
1	0	10	01	1	0
1	1	00	10	1	0

As seen from the state table there are no equivalent states. Therefore, no reduction is the state diagram. The state table shows that circuit goes through four states, therefore we require 2 flip-flops (number of states = 2^m, where m = number of flip-flops).

Design using 'D' flip-flops:-

For D flip-flops next states are nothing but the new present states. Thus, we can directly use next states to determine the flip-flop input with the help of K-map simplification.

K-map simplification:-

For flip-flop 'A'

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

For flip-flop B

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

For output

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

$$D_A = \bar{A}\bar{B}x + \bar{A}B\bar{x} + ABx + A\bar{B}\bar{x}$$

$$D_B = \bar{A}B\bar{x} + A\bar{B}x$$

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

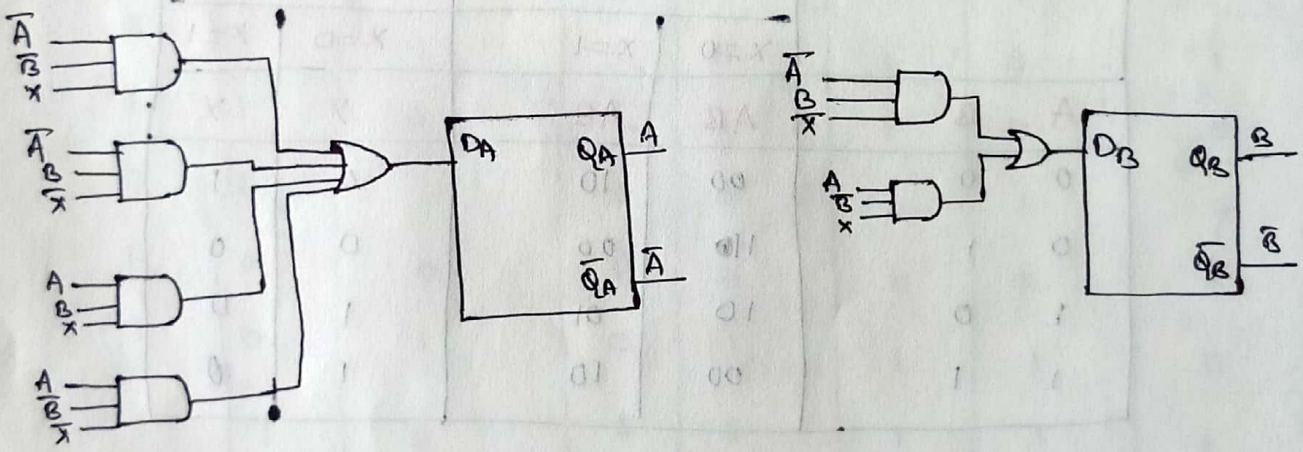


fig:- Logic diagram of given sequential circuit using 'D' Flip-flop

(ii) Design using T flip-flops :-

using the excitation table for T flip-flop shown in below table. we can determine the excitation table for the given circuit.

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

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	0	1	0	1	0
0	1	0	1	0	1	1	0
0	1	1	1	1	0	0	0
1	0	0	0	0	1	0	0
1	0	1	1	0	0	0	1
1	1	0	0	1	1	1	0
1	1	1	1	0	0	1	0

Table: Circuit excitation table.

The first row of circuit excitation table shows that there is no change in the state for both flip-flops. The transition from 0→0 for T flip-flop requires input T to be at logic '0'. The second row shows that flip-flop A has transition 0→1. It requires the input T_A to be at logic 1. It requires the input T_A to be at logic 1. Similarly we can find inputs for each flip-flop for each row in the table by referring present state, next state and 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) flip-flop B

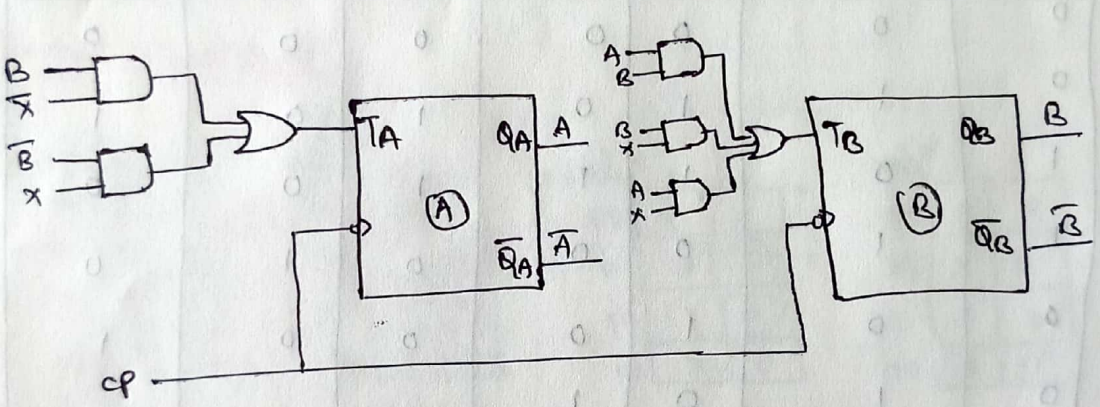
AB \ X	0	1
00	0	1
01	0	0
11	1	0
10	1	0

(c) for output

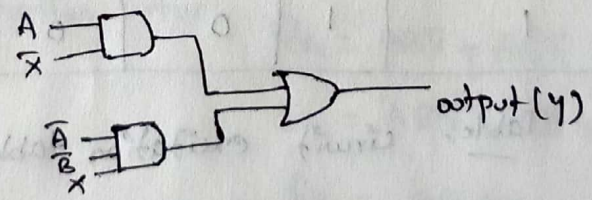
input function for $T_A = B\bar{X} + \bar{B}X$

$T_B = AB + BX + AX$, and

circuit output function $= A\bar{X} + \bar{A}BX$



(a)



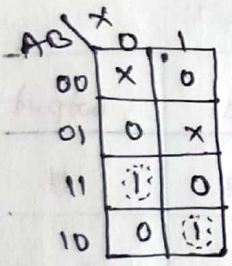
(b)

Design using ~~SR~~ JK Flip-flops:-

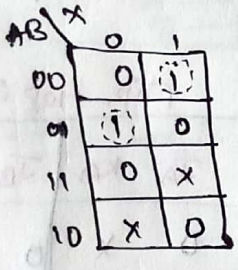
Q_n	Q_{n+1}	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Present state		Input	Next state		Flip-flop inputs				output
A	B	x	A	B	S_A	R_A	S_B	R_B	y
0	0	0	0	0	0	x	0	x	0
0	0	1	1	0	1	0	0	x	1
0	1	0	1	1	1	0	x	0	0
0	1	1	0	0	0	x	0	1	0
1	0	0	1	0	x	0	0	x	1
1	0	1	0	1	0	1	1	0	0
1	1	0	0	0	0	1	0	1	1
1	1	1	1	0	x	0	0	1	0

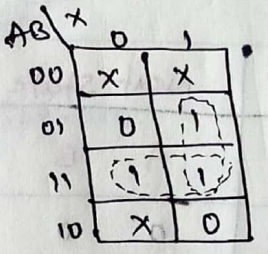
K-map simplification:-



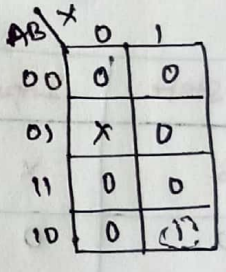
for R_A



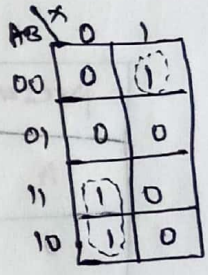
for S_A



for R_B



for S_B



for output

input function for

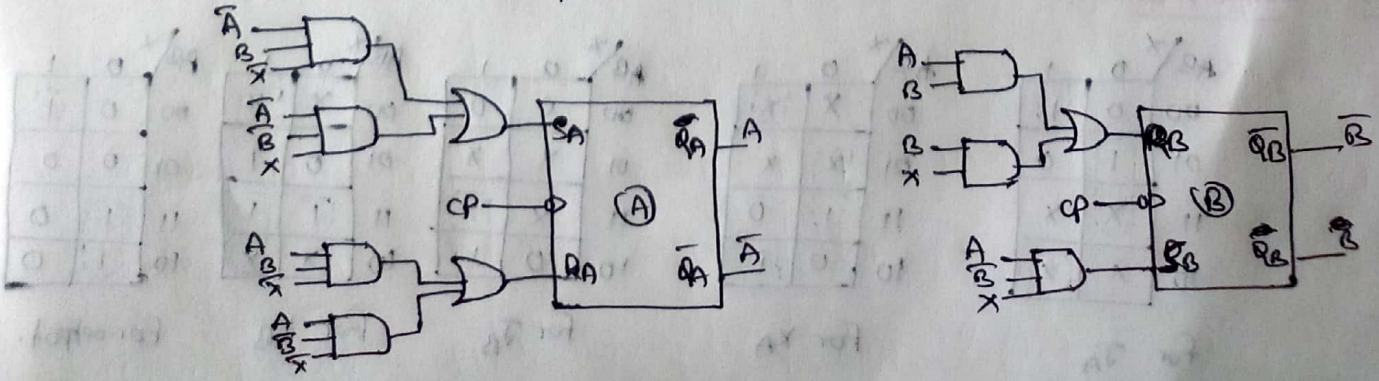
$R_A = AB\bar{x} + A\bar{B}\bar{x}$

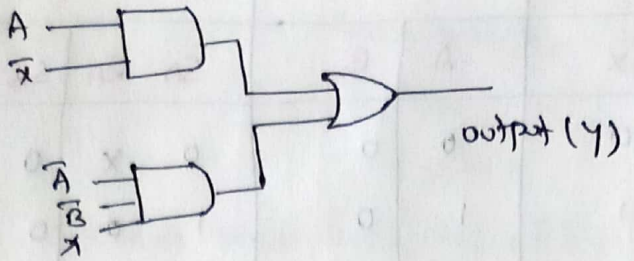
$S_A = \bar{A}B\bar{x} + \bar{A}\bar{B}x$

$R_B = AB + Bx$

$S_B = A\bar{B}x$ and

Circuit output function $= A\bar{x} + \bar{A}\bar{B}x$





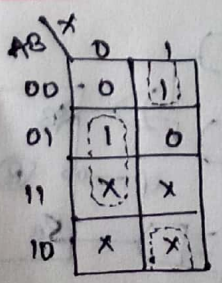
(iv) Design using JK flip-flops:-

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

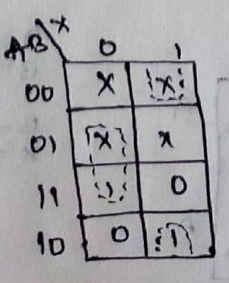
Table: Excitation Table for JK flip-flop.

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

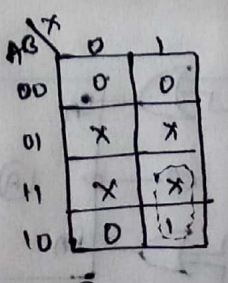
K-map Simplification:-



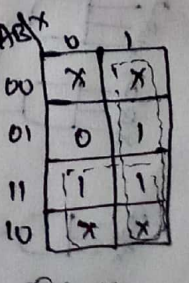
for J_A



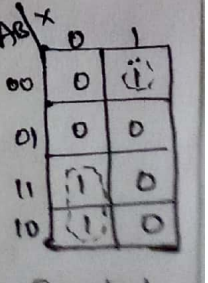
for K_A



for J_B



for K_B



for output

Therefore, Input function for $J_A = B\bar{x} + \bar{B}x$

$K_A = B\bar{x} + \bar{B}x$

$J_B = Ax$

$K_B = A + x$

Circuit output function $= A\bar{x} + \bar{A}\bar{B}x$

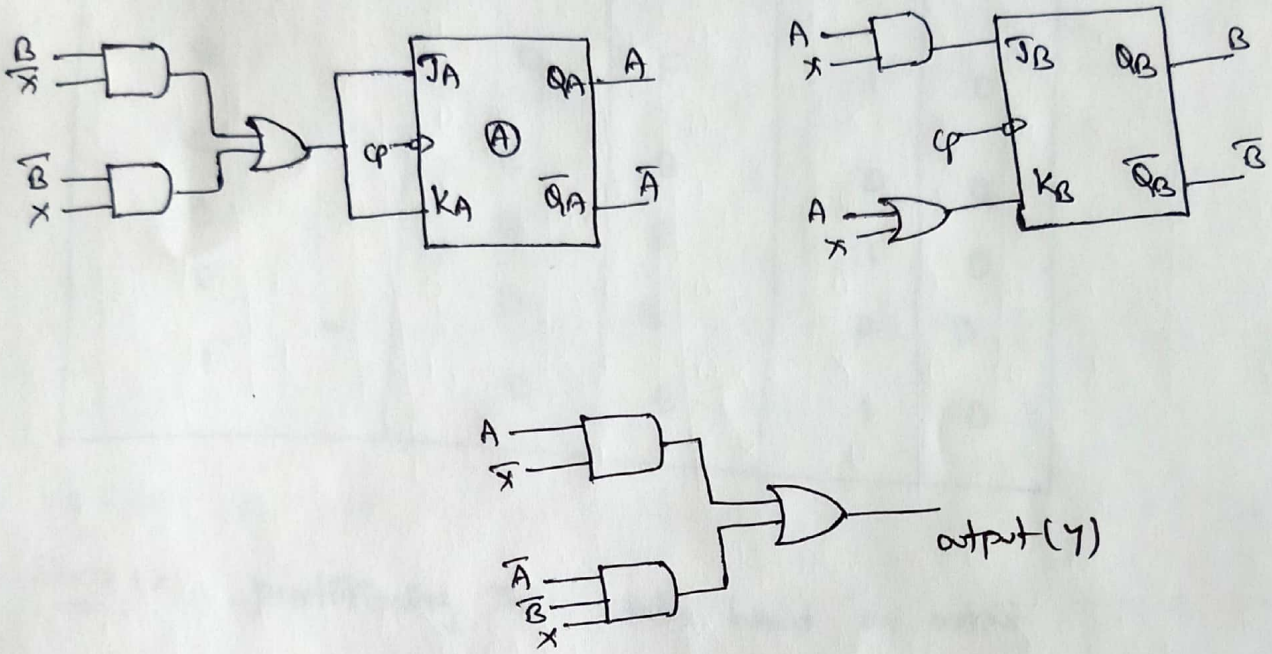


Fig: Logic diagram.

State Table Reduction using partition Technique:-

Example:-

Present state	Next state		output	
	x=0	x=1	x=0	x=1
A	C	F	0	0
B	D	F	1	0
C	E	B	0	0
D	B	E	1	0
E	D	B	0	0
F	D	B	1	0

Step 1:- partitioning the states based on output.

$P_1 = (A, C, E)\ (B, D, F)$

Step 2:- partitioning the states based on NEXT STATES

$P_1 = (A, C, E)\ (B, D, F)$

$P_2 = (A, C)\ (E)\ (B, F)\ (D)$

$P_3 = (A)\ (C)\ (E)\ (B, F)\ (D)$

Step 3:-

$P_1 = (A, C, E)\ (B, D, F)$

$P_2 = (A, C)\ (E)\ (B, F)\ (D)$

$P_3 = (A)\ (C)\ (E)\ (B, F)\ (D)$

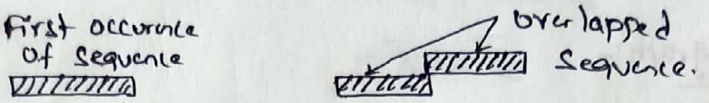
$B = F$

State Table after partitioning:-

present state	Next state		output	
	X=0	X=1	X=0	X=1
A	C	B	0	0
B	D	B	1	0
C	E	B	0	0
D	B	E	1	0
E	D	B	0	0

Sequence Detector :- (or) pattern detector :-

The specified input sequence can be detected using a Sequential Machine called sequence detector. In this circuit output goes high when a prescribed input sequence occurs. A typical input Sequence and the corresponding output sequence for desired input sequence 101 are:-



X = 0 0 1 1 0 1 1 0 0 1 0 1 0 1 1

Z = 0 0 0 0 0 1 0 0 0 0 0 1 0 1 0

As shown above the detection of required input sequence can occur in a longer data string and the desired input sequence can overlap with another input sequence. It is assumed that input can change only between clock pulses. once we know the sequence which is to be detected, we can draw the state diagrams for it. then from the state diagram we can design the circuit. It is possible to implement sequence detector using both types of sequential machines: Mealy machine and Moore machine.

Sequence detector using mealy model (non-overlapping / overlapping) :-

Example for sequence detector :-

* Design circuit to detect 110 from given input data stream.

→ If the given input sequence is

X = 0 1 1 1 0 1 1 0 1 0 1 1 0

Y = 0 0 0 0 1 0 0 1 0 0 0 0 1

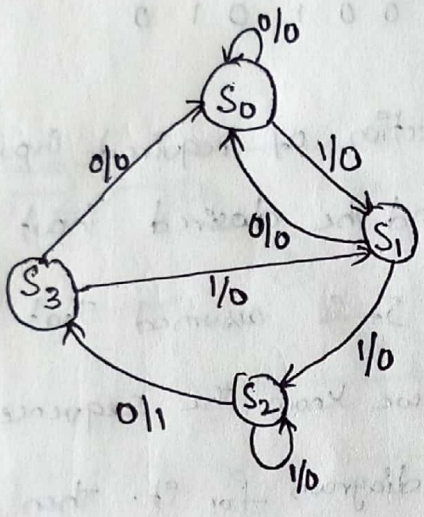
$S_0 = 0$ [Reset]

$S_1 = 1$

$S_2 = 1\ 1$

$S_3 = 1\ 1\ 0$

State diagram:-



State Table:-

	Q_A	Q_B
S_0	0	0
S_1	0	1
S_2	1	0
S_3	1	1

State Table:-

present state		input	D_A D_B Next state		
Q_A	Q_B	x	Q_A^+	Q_B^+	Y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	0	0
1	0	0	1	1	1
1	0	1	1	0	0
1	1	0	0	0	0
1	1	1	0	1	0

K- map simplification:-

x \ $Q_A Q_B$	00	01	11	10
0	0	0	0	1
1	0	1	0	1

for D_A

$$D_A = Q_A \bar{Q}_B + x \bar{Q}_A Q_B$$

x \ $Q_A Q_B$	00	01	11	10
0	0	0	0	1
1	1	0	1	0

for D_B

$$D_B = x \bar{Q}_A \bar{Q}_B + x Q_A Q_B + \bar{x} Q_A \bar{Q}_B$$

x \ $Q_A Q_B$	00	01	11	10
0	0	0	0	1
1	0	0	0	0

for output

$$Y = Q_A \bar{Q}_B \bar{x}$$

Logic diagram:-