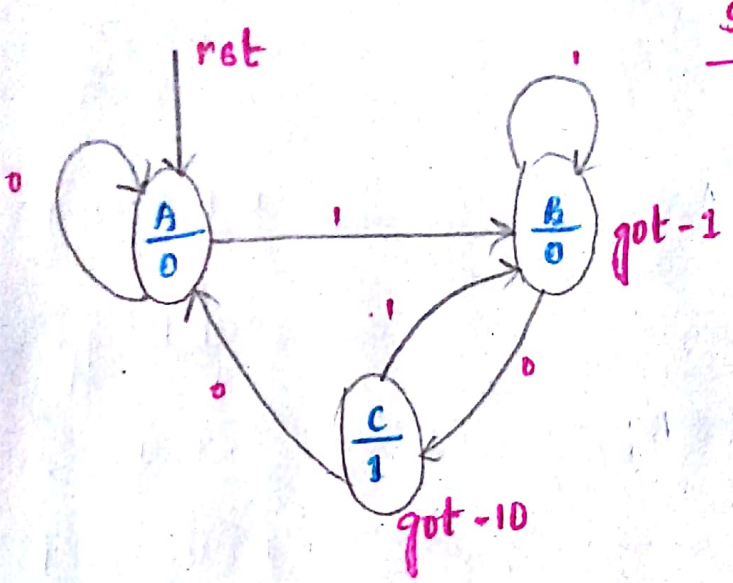
Seq -10 :



* moore fsm state diagram *

Verilog code:

```
module moore_10(x, clk, rst, y);
input x, clk, rst;
output y;
reg [1:0] state;
parameter A = 2'b00,
          B = 2'b01,
          C = 2'b10;

always @ (posedge clk, posedge rst)
if (rst)
    state <= A;
    else
    case(state)

A: if (x)
    state <= B;
```

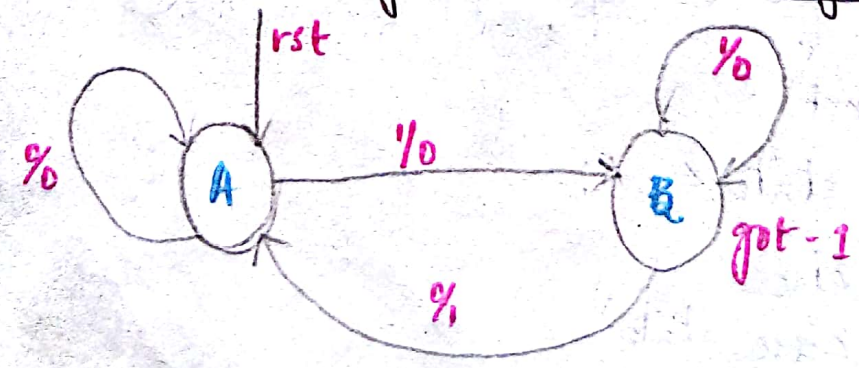
```
else
  state <= A;

B: if (x)
     state <= B;
else
     state <= C;

C: if (x)
  state <= B;
else
     state <= A;

default : state <= A;
endcase
assign y = (state == C);
endmodule
```

* Mealy FSM state-diagram: *



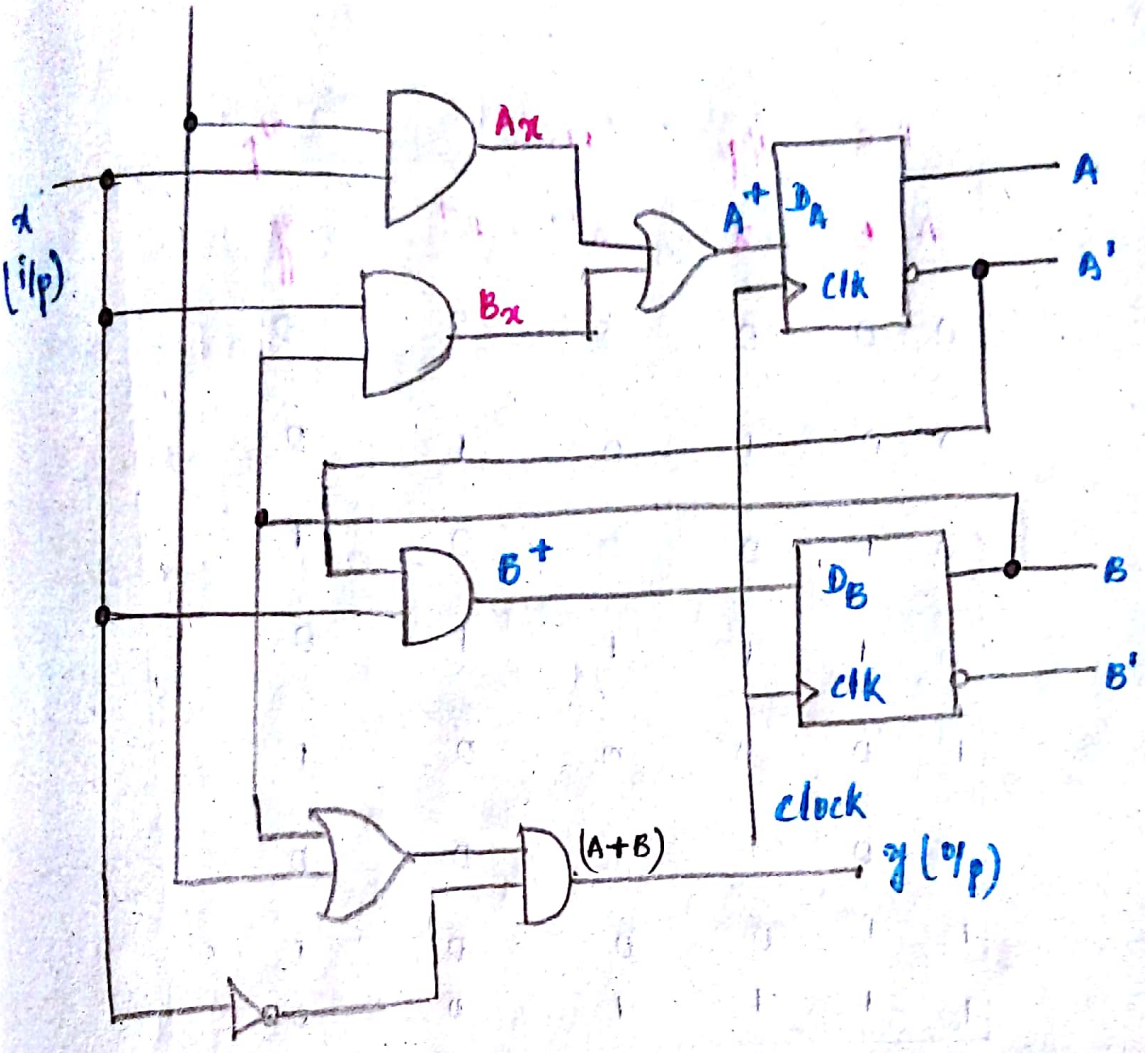
verilog code :

```
module mealy_10(x, clk, rst, y);
input x, clk, rst;
output y;
reg y;
reg state;
parameter A = 1'b0,
          B = 1'b1;
always @ (posedge clk, posedge rst)
if (rst)
    state <= A;
else
  case (state)
A: if (x)
    begin
     state <= B;
     y <= 0;
     end
else
    begin
    state <= A;
```

```
        y <= 0;
    end
B : if (x)
        begin
            state <= B;
            y <= 0;
        end
    else
        begin
            state <= A;
            y <= 1;
        end
    endcase
endmodule
```

* Analysis of clocked seq. circuit *
(Synchronous seq circuit):

D-F/F :



where,

A, B ⟶ P.S

A^+, B^+ ⟶ N.S

④ state eqⁿ : and o/p eqⁿˢ :

Flip-Flop eqⁿˢ

$D_A = A^+ = Ax + Bx$ ⟶ state eqⁿˢ

$D_B = B^+ = \bar{A}x$ ⟶ for D-F/F, $Q_{n+1} = D$ (char. eqⁿ)

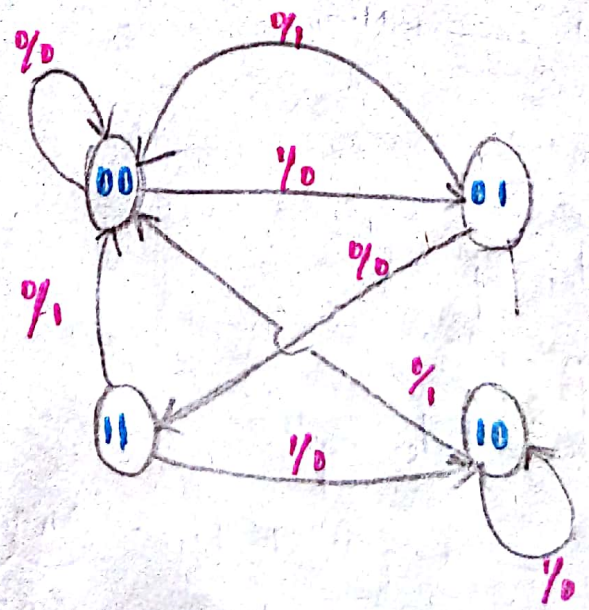
$y(o/p) = [A+B]\bar{x}$ ⟶ o/p eqⁿ

$A^+ = D_A = Ax + Bx$; $B^+ = D_B = \bar{A}x$

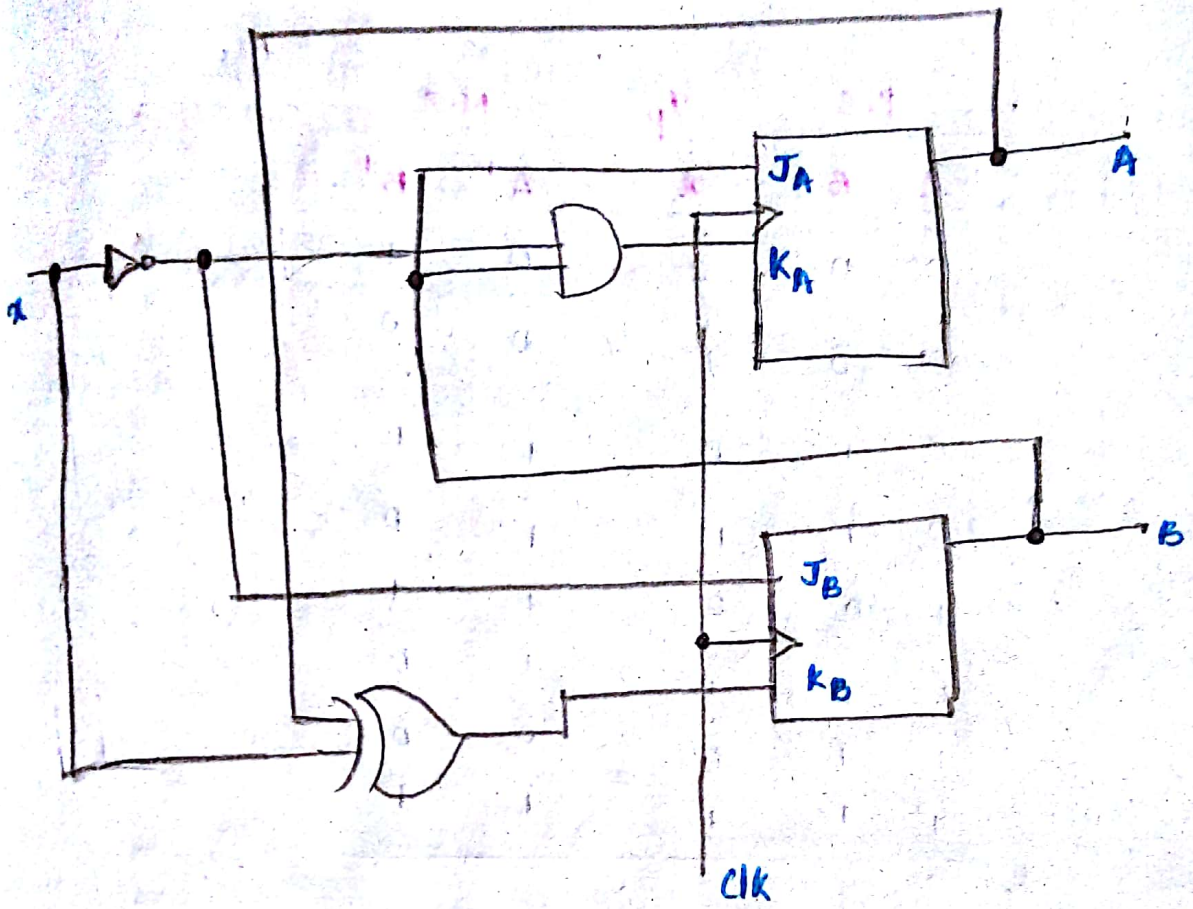
✳ state table :

P.S		i/p	N.S		o/p
A	B	x	A^+	B^+	y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

✳ state diagram :



* Analysis with JK - F/F :



(3) Flip-Flop eqn's :

$$J_A = B, \quad K_A = B\bar{x}$$

$$J_B = \bar{x}, \quad K_B = A \oplus x$$

(4) state eqn's and op eqn's :

$(\because Q_{n+1} = J\bar{Q}_n + \bar{K}Q_n)$

$$A^+ = J_A\bar{A} + \bar{K}_A A = B\bar{A} + (\overline{Bx})A = \bar{A}B + A\bar{B} + Ax$$

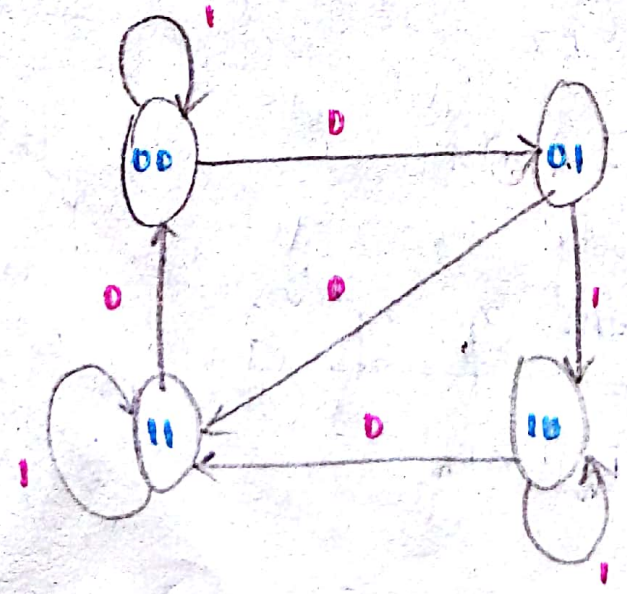
$$B^+ = J_B\bar{B} + \bar{K}_B B = \bar{x}\bar{B} + \left(\overline{A \oplus x}\right)B$$

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

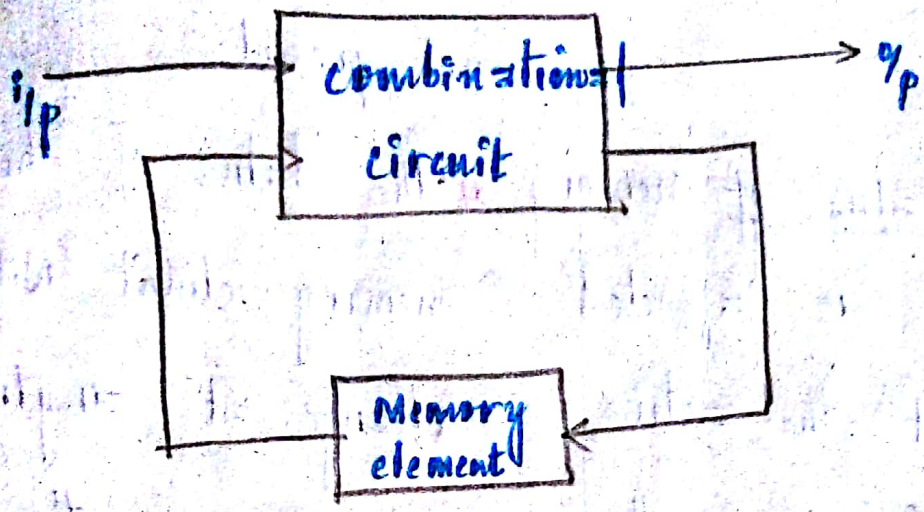
* state table :

P.S		i/p	N.S	
A	B	x	A^+	B^+
0	0	0	0	1
0	0	1	0	0
0	1	0	1	1
0	1	1	1	0
1	0	0	1	1
1	0	1	1	0
1	1	0	0	0
1	1	1	1	1

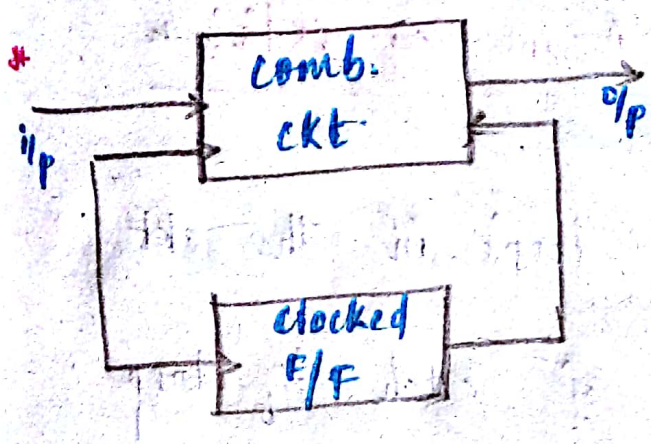
* state - diagram:



Seq. ckt. Block. dia:

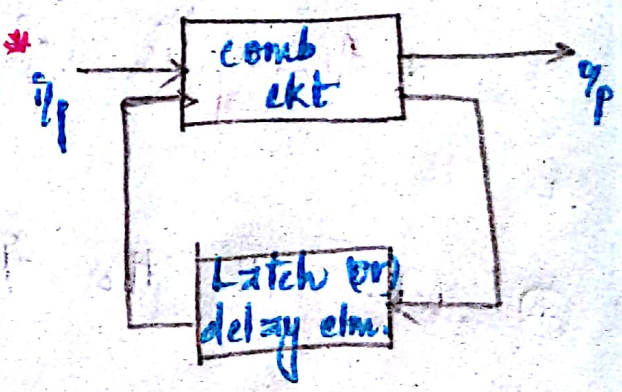


Syn: Seq ckt.



- These ckt's are easy to design
- A clocked F/F act's as a memory element.

Asyn seq. ckt



- These ckt's are difficult to design
- Lath (or) time delay element is used as a memory elmt.

* These are slow.	* These are fast as the clock is not present.
* The status of memory element is affected only at the active edge of clk, if the i/p is changed.	* The status of the memory elmnt will change at anytime as soon as the i/p is changed.

* Analysis of Asyn Seq ckt's: *

(*) Det. all the f/b loop's in the ckt

(*) Designate the o/p of each f/b loop with var. (Y_i) and it's corresponding i/p with (y_i) for $i = 1 \ldots k$ where,

(k) ⟶ no. of f/b loop's in the ckt.

* Derive the Boolean fun of all (Y's) as a fun of ext. i/p's and (y's).

* plot each y - fun in a map using y - var. for the row's and the ext. i/p for column's.

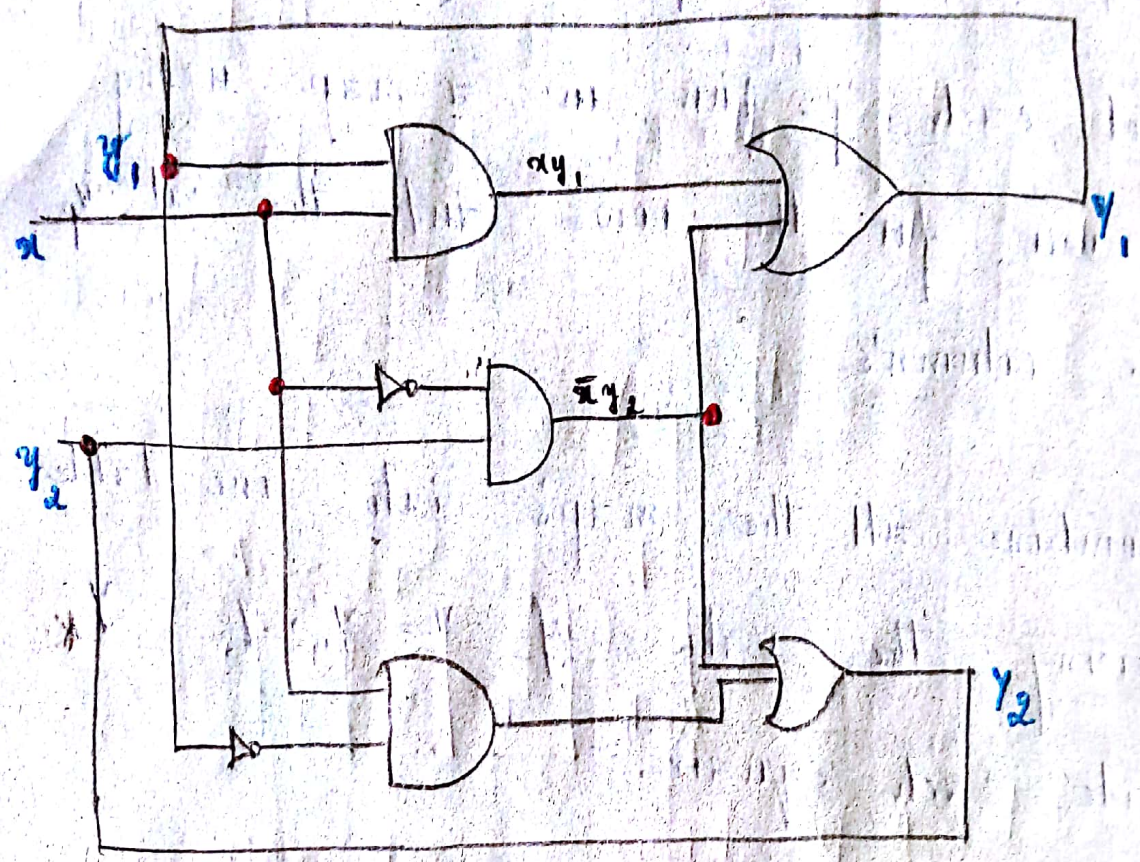
* Combine all the map's into one table showing the value of $Y = Y_1, Y_2 \cdots Y_K$ inside each square.

* Circle those value's of (Y) in each square that are = the value of $y = y_1, y_2, \cdots y_K$ in the same row

* Draw the flow - table.

Example of an Asyn. seq circuit :

-①



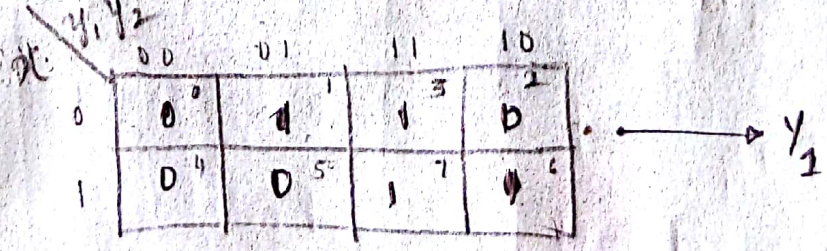
soln :

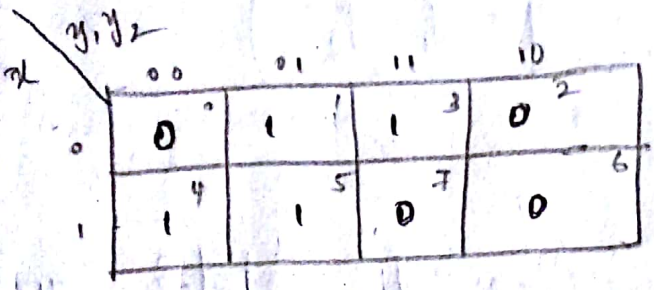
* There are 2 feed back loops.

* $Y_1 = xy_1 + \bar{x}y_2$.

  $Y_2 = x\bar{y}_1 + \bar{x}y_2$

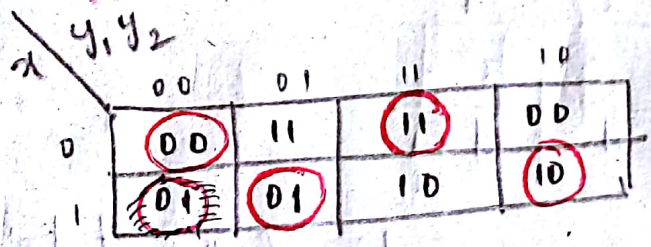
* k-map :





Transition table :-

- combining (Y_1 and Y_2) map's, we get the tr. table.



(or)

- tr. table is obtained by combining the binary values in the corresponding map's of (Y's).

- Tr. table indicate's the value's of ...

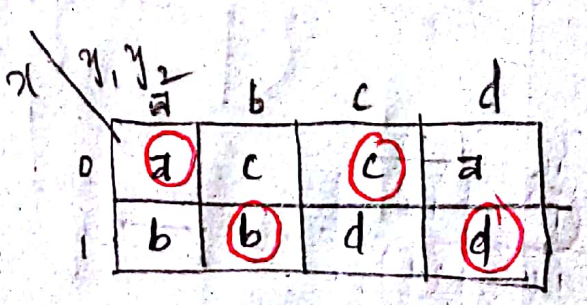
$Y = Y_1, Y_2$ inside each square.

- For a state to be stable, sec var must match the exitation table var (i.e., the value's of (Y) must be same

as that of --- $y = y_1, y_2 \cdots$)

* Flow table:

- In a tn. table, if the state's are named by letter symbol's, then, it is called as a flow table.



x \ y_1y_2	a	b	c	d
0	a	c	c	a
1	b	b	d	d

- The flow table consist's of 4 - state's with one i/p

* primitive flow-table:

In a flow-table, if each row consist's of only one stable state, then it is called as a primitive flow-table.

Example: ②

	00	01	11	10
a	(a), 0	(a), 0	(a), 0	b, 0
b	a, 0	a, 0	(b), 1	(b), 0

→ 2 state's with 2 i/p's and 1 o/p

sol^n :

	00	01	11	10
a	(a)	(a)	(a)	b
b	a	a	(b)	(b)

	00	01	11	10
a	0	0	0	0
b	0	0	1	0

y \ x_1x_2	00	01	11	10
0	0	0	0	1
1	0	0	1	1

map for y :

y \ x_1x_2	00	01	11	10
0	0	0	0	0
1	0	0	1	0

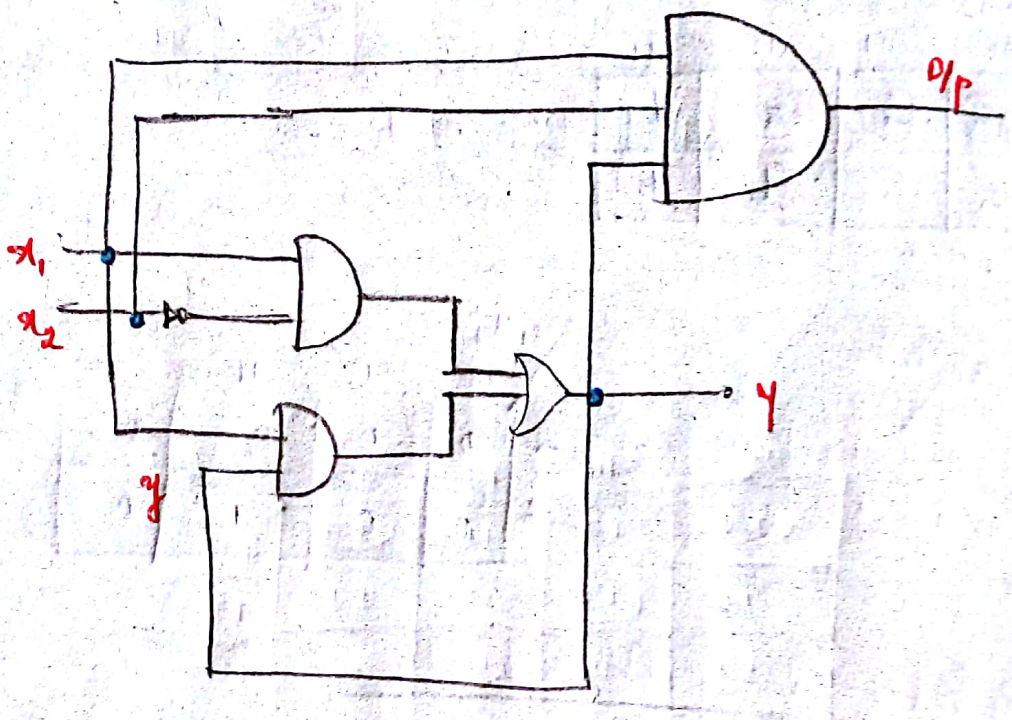
map for o/p :

0 → a

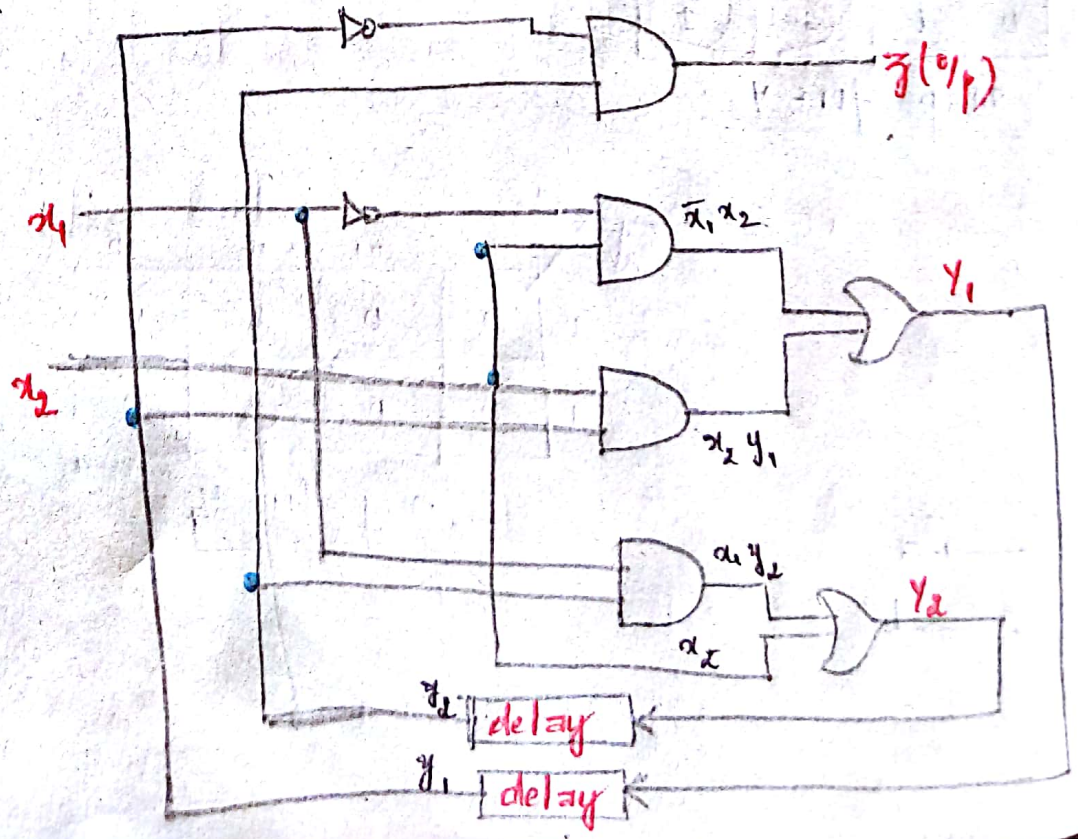
1 → b

$Y = x_1 y + x_1 \bar{x}_2$

output = $y\, x_1\, x_2$



Example -③ :



(*) Analyze the above asyn - seq ckt and obtain the S. table and timing dia:

sol"

* no. of feed back's = 2

* $Y_1 = \bar{x}_1 x_2 + x_2 y_1$

$Y_2 = x_1 y_2 + x_2$

$z = \bar{y}_1 y_2$

* k-map :

y_1y_2 \ x_1x_2	00	01	11	10
00	0	1	0	0
01	0	1	0	0
11	0	1	1	0
10	0	1	1	0

map - for (Y_1)

y_1y_2 \ x_1x_2	00	01	11	10
00	0	1	1	0
01	0	1	1	1
11	0	1	1	1
10	0	1	1	0

map for - (Y_2)

y_1y_2 \ x_1x_2	00	01	11	10
00	1	0	0	0
01	1	1	1	1
11	0	0	0	0
10	0	0	0	0

map for $z(y_p)$

* Tr. table :

y_1y_2 \ x_1x_2	00	01	11	10
00	**00**	11	01	**00**
01	00	11	**01**	01
11	00	**11**	**11**	01
10	00	11	11	00

bold state's are stable - state's.

* Flow-table :

00 ⟶ a
01 ⟶ b
11 ⟶ c
01 ⟶ d.

y_1y_2 \ x_1x_2	00	01	11	10
a	(a)	c	b	(a)
b	a	c	(b)	(b)
c	a	(c)	(c)	b
d	a	c	c	a

* **state table :**

P.S	N.S				O/P
	00	01	11	10	
a	a	c	b	a	0
b	a	c	b	b	1
c	a	c	c	b	0
d	a	c	c	a	0