Unit6-Design of Sequential Circuit

- Design problem normally starts with a word description of input output relation and ends with a circuit diagram having sequential and combinatorial logic elements.
- Sequential logic circuit design refers to synchronous clock-triggered circuit because of its design and implementation advantages.

Steps involved in design of sequential circuit

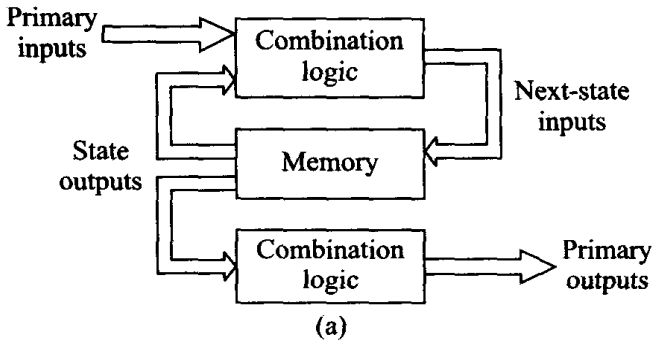
1. Input output relation
2. State transition diagram or Algorithmic State Machine (ASM)chart
3. State synthesis table
    a. **Flip Flop based implementation-** excitation tables are used to generate design equations through Karnaugh Map
    b. **Read Only Memory (ROM) based implementation**- excitation tables are not required but flip-flops are used as delay elements
4. Circuit diagram is developed from these design equations.

State machine design

There are **two different approaches of state machine design** called Moore model and Mealy model.

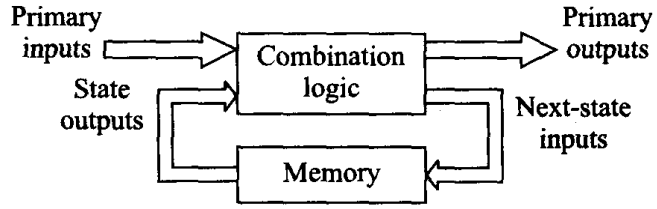
I.INPUT OUTPUT RELATION

In **Moore model** circuit outputs, also called primary outputs are generated **solely from secondary outputs or memory values.**



(a)

In **Mealy model** circuit inputs, also known as **primary inputs combine with memory elements** to generate circuit output.



(b)

Difference between moore and mealy model

Moore model	Mealy model
In **Moore model** circuit outputs, also called primary outputs are generated **solely from secondary outputs or memory values.**	In **Mealy model** circuit inputs, also known as **primary inputs combine with memory elements** to generate circuit output.
The output depends only on present state and not on input	The output is derived from present state as well as input
it requires less number of states and thereby less hardware to solve any problem	it requires more number of states and thereby more hardware to solve any problem
the output is generated one clock cycle earlier.	the output is generated one clock cycle after
The output remains stable over entire clock period and changes only when	The glitches occurs

there occurs a state change at clock trigger based on input available at that time.	

CONVERSION FROM ONE MODEL TO OTHER

- Conversion from one model to other can be done through state diagram representation.
- Depending on application requirements we choose one of these two models or a mixed model where a part of the circuit follows Mealy model and the other Moore model.

Example:-

CONVERSION FROM MEALY TO MOORE MODEL

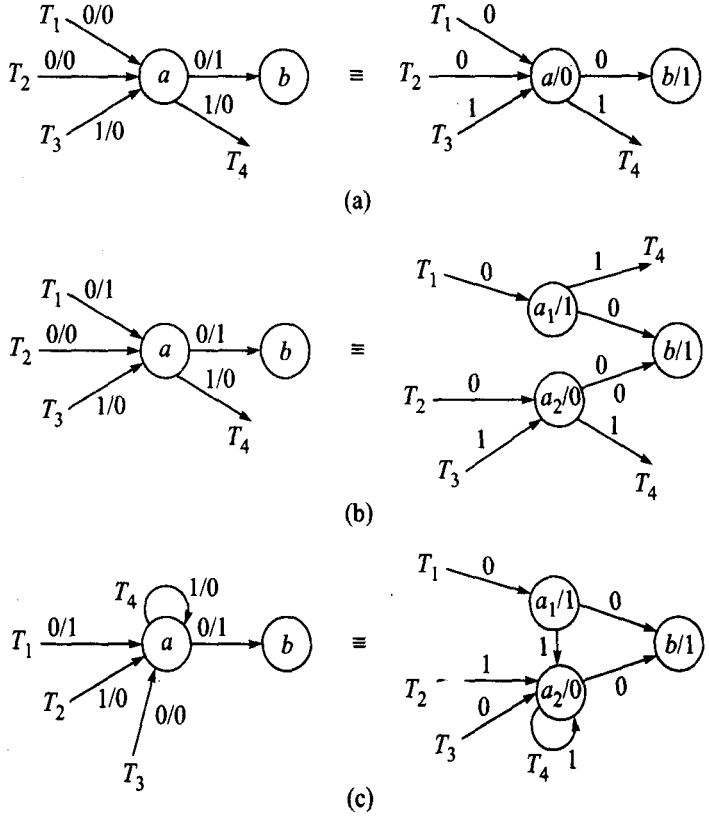


Fig. 11.3 *Conversion between Mealy and Moore model.*

- Conversion between Mealy and Moore models can take place as shown in Fig. 11.3 where, T1, T2, T3 represent paths leading to state a.
- The path T4 leads from state a when input is 1.
- If input is 0, state a leads to state b and there are no other paths reaching b.
- The rule of conversion is as follows. If all the transitions in a Mealy model to a particular state are associated with only one type of output then in corresponding Moore model that output becomes state output (Fig. 11 .3a).
- If there is more than one output in Mealy model we need as **many intermediate** state variables, as shown in Fig. 11 .3b. In Fig. 11 .3c it is shown how to treat transitions that loop within a particular state. The reverse of this is applied in converting Moore model to Mealy model.

DESIGN OF SYNCHRONOUS SEQUENTIAL CIRCUIT

EXAMPLE:-

SEQUENCE DETECTOR PROBLEM

The Problem Design a sequence detector that receives binary data stream at its input, X and signals when a combination **'011'** arrives at the input by making its output, Y high which otherwise remains low. Consider, data is coming from **right** i.e. the first bit to be identified is **1, second 1** and **third 0** from the input sequence.

The first step in a sequential logic synthesis problem is to convert this **word description** to **State transition diagram or Algorithm State Machine (ASM) Chart.**

MOORE MODEL

1) Input output relation

In **Moore model** circuit outputs, also called primary outputs are generated **solely from secondary outputs or memory value**

2) State transition diagram

- Since, the output is generated only from the state variables ,the detector circuit be at state **a** when initialized where **none of the bit** in input sequence is properly detected or the starting point of detection.
- Then if **1st bit** is detected properly the circuit should be at a different state say, **b**
- Similarly, we need two more states say, **c and d** to represent detection of **2nd and 3rd** bit in proper order. When the detector circuit is at state d, output Y is asserted and kept high as long as circuit remains in state d signaling sequence detection. For other states detector output, Y = 0.

II STATE TRANSITION DIAGRAM

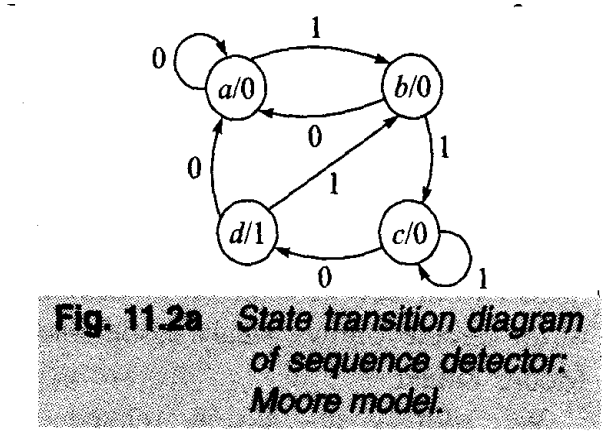


Fig. 11.2a *State transition diagram of sequence detector: Moore model.*

- **Each state and output is defined within a circle** in state transition diagram in the format **s/V**
- where **s** represents a **symbol or memory values** identified with a state and **V** represents the **output of the circuit.**
- An arrow sign marks state transition following an input value 0 or I that is written along the path.
- X represents the binary data input from which sequence '011' is to be detected.

Figure 11 .2a shows the state transition diagram following Moore model.

- Initialized with a (if input x=0 - remain in same state a ,if input x=1 go to next state b **first bit(1)** is detected and output y=0)
- State b (if input x=0 - go to initial state a ,if input x=1 go to next state c **second bit(1)** is detected and output y=0)
- State c (if input x=0 - go to final state d **third bit (0)** is detected, if input x=1 remain in same state c and output y=0)
- State d (if input x=0 - go to initial state a ,if input x=1 go to state b and repeat the sequence ie, '011011' and output y=1)

Mealy Model

1) Input output relation

- Since, the output can be derived using **state as well as input** , three different states for 3- bit sequence detector circuit following Mealy model.
- The three states say, a, b, c represents none, 1st bit and 2' bit detection.
- When the circuit is at state c if the input is as per the pattern the output is generated in state c itself with proper logic combination of input.

State Transition Diagram: Mealy Model

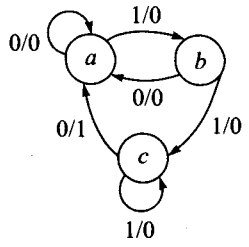


Fig. 11.2b *State transition diagram of sequence detector: Mealy model.*

Figure 11 .2b shows state transition diagram of the given problem following Mealy model.

- Initialized with a (if input x=0 - remain in same state a ,if input x=1 go to next state b **first bit(1)** is detected and output y=0)
- State b (if input x=0 - go to initial state a ,if input x=1 go to next state c **second bit(1)** is detected and output y=0)
- State c (if input x=0 - go to state a **third bit (0)** is detected and output y=1, if input x=1 remain in same state c and output y=0)

Note :-The difference with Moore model where output is generated one clock cycle later in state d and also requires one additional state.

III STATE SYNTHESIS TABLE

The next step in design process is to develop state synthesis table, also called **circuit excitation table or simply state table** from state transition diagram.

For **m** number of memory elements , 2^m number of different states in a circuit.

- State assignment
- No of memory elements
- Decide about filp flops (JK,D) ,normally prefer JK flip-flop as it has maximum number of don't care states in its excitation table and that leads to simpler design equations

STATE ASSIGNMENT FOR SEQUENCE DETECTOR

- Each state a binary combination of memory values.
- For both Moore and Mealy models require minimum two flip-flops (say A and B) to define their states (4 for Moore and 3 for Mealy)

The state assignment be as follows.

$a: B = 0, \quad A = 0 \qquad b: B = 0, \quad A = 1 \qquad c: B = 1, \quad A = 0 \qquad d: B = 1, \quad A = 1$

Note that Mealy model does not use state d. Assignment can be done in any order, e.g. we can make a: B = 0, A = 1 and b: B = 0, A = 0

NO OF MEMORY ELEMENTS

2 number of memory elements

DECIDE ABOUT FILP FLOPS

JK flip flop

Moore Model

State synthesis table obtained from state transition diagram of Moore model (Fig. 11 .2a) and excitation table of JK flip-flop is shown in Table 11.1.

JK flip flop excitation table11.1

Present state	Next state	J	k
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

- It has eight rows as for each of the four possible states there can be two different types of inputs.
- The table is prepared as follows. When the circuit is at state 00, i.e. a and receives X = 0 it remains at state 00 and output in this state Y = 0. Since both B and A flip-flop makes 0->0 transition both the flip-flops should have input **Ox** from excitation table.

Table 11.1 State Synthesis Table for Moore Model

Present State B_n	A_n	*Present Input* X_n	*Next State* B_{n+1}	A_{n+1}	*Output* Y_n	J_B	K_B	J_A	K_A
0	0	0	0	0	0	0	×	0	×
0	0	1	0	1	0	0	×	1	×
0	1	0	0	0	0	0	×	×	1
0	1	1	1	0	0	1	×	×	1
1	0	0	1	1	0	×	0	1	×
1	0	1	1	0	0	×	0	0	×
1	1	0	0	0	1	×	1	×	1
1	1	1	0	1	1	×	1	×	0

Mealy Model

- Since, Mealy Model requires three states for this problem we have six rows in state synthesis table
- In each state there can be two different types of input X = 0 or X = 1.

Table 11.2 represents state synthesis table for Mealy model. The method remains the same as Moore model but we use stat transition diagram (Fig. 11 .2b) corresponding to Mealy model from Section 11.2.

Table 11.2 State Synthesis Table for Mealy Model

Present State B_n	Present State A_n	Present Input X_n	Next State B_{n+1}	Next State A_{n+1}	Present Output Y_n	J_B	K_B	J_A	K_A
0	0	0	0	0	0	0	×	0	×
0	0	1	0	1	0	0	×	1	×
0	1	0	0	0	0	0	×	×	1
0	1	1	1	0	0	1	×	×	1
1	0	0	0	0	1	×	1	0	×
1	0	1	1	0	0	×	0	0	×

IV.DESIGN EQUATIONS AND CIRCUIT DIAGRAM

- In design equation we express flip-flop inputs as a function of present state,i.e**. memory values** (here, B and A) and **present input** (here, X). This ensures proper transfer of the circuit to next state.
- The design equations also give **output (here, Y)** equation in terms of **state variables or memory elements in Moore model** and **state variables together with input in Mealy model**.
- Use Karnaugh map technique to get a simplified form of these relations.

Moore Model

- Figure 11 .4a presents **Kamaugh map** developed from **state synthesis Table** 11.1 and also shows corresponding **design equations.**
- Figure 11 .4b shows the sequence detector circuit diagram developed from these equations.

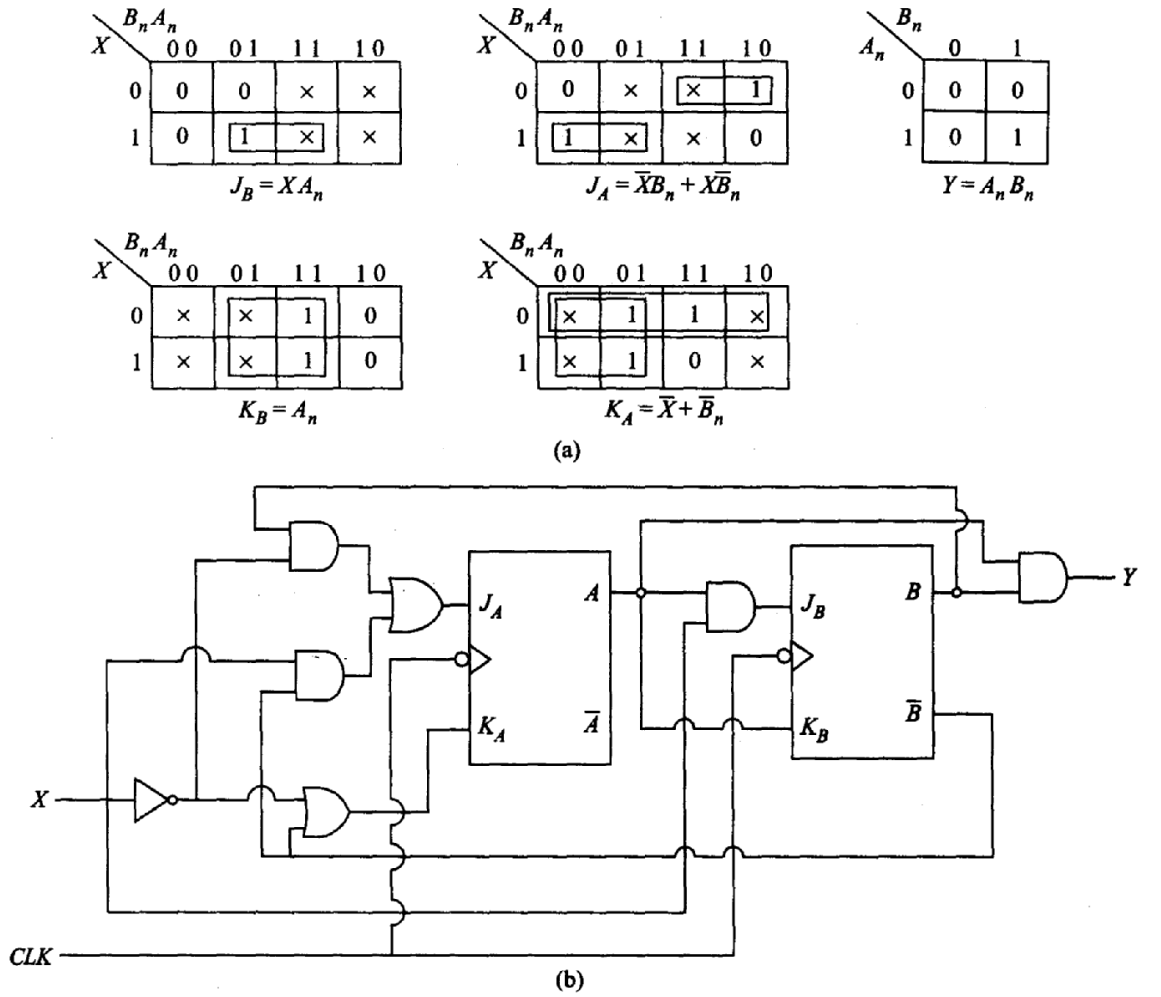


Fig. 11.4 *(a) Design equations for Moore model. (b) Circuit diagram following Moore model.*

Mealy Model

Using state synthesis table corresponding to Mealy model (Table 11.2) we can fill six positions in each Karnaugh map (Fig. 1 1.5a). Locations B,,A,X = 110 and B,A,,K = 111 are filled with don't care(x) conditions as such a combination never occur in the detector circuit if properly initialized.

The design equations are obtained from these Karnaugh maps from which circuit diagram is drawn as shown in Fig. 11 .5b. In this circuit, output directly **uses input information.**

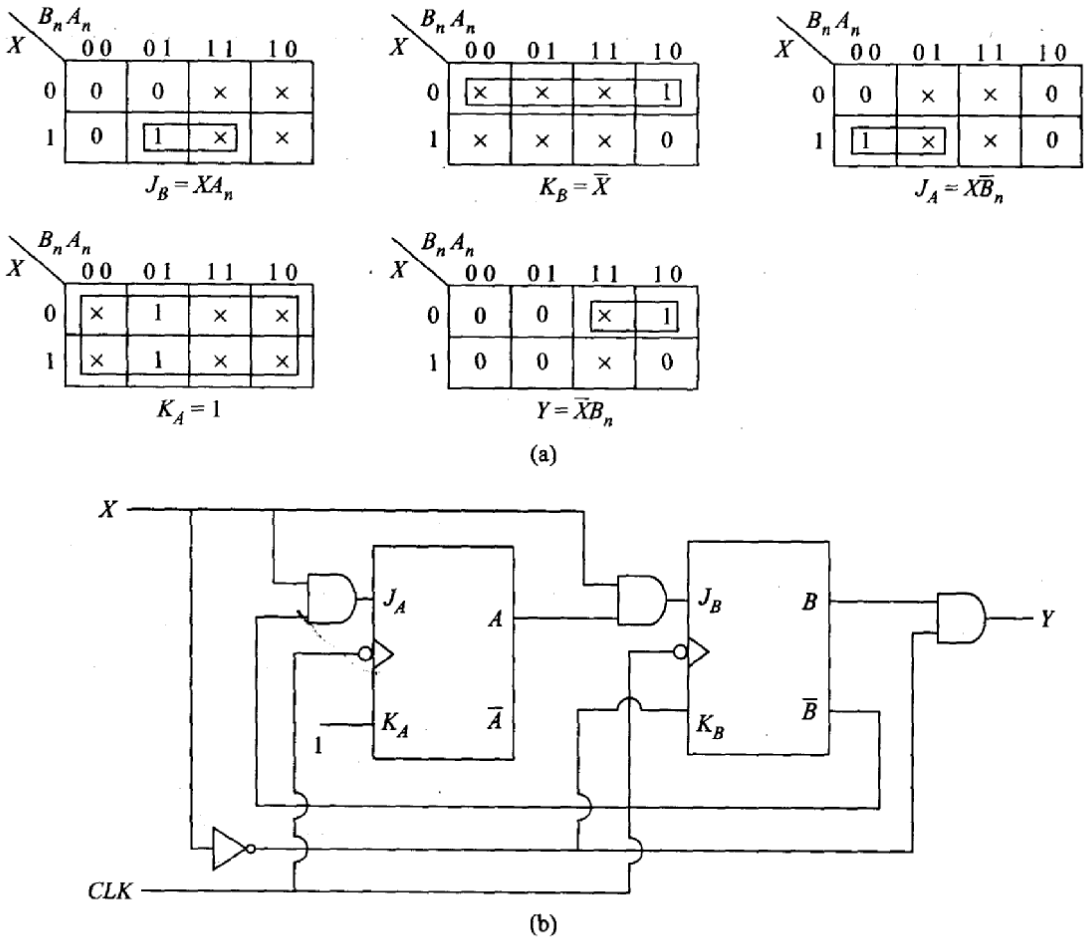


Fig. 11.5 *(a) Design equations for Mealy models. (b) Circuit diagram following Mealy model.*

IMPLEMENTATION USING READ ONLY MEMORY

Solution to sequential logic problem using Read Only Memory (ROM) which though called memory is a combinatorial circuit.

- The output of ROM is immediately available when a particular location in memory is addressed
- ROM that has as many memory locations as the number of rows in a ***state synthesis table***.
- Each row is uniquely identified by present state and present input.
- This present state and input combination through an address decoder points to memory spaces in ROM. In each location of ROM we store the next state value of the circuit.
- Bank of Delay flip-flops, the number is same as the number of state variables or memory elements. Next state information for each state variable that is stored in ROM is fed to these D flip-flop inputs.
- At clock trigger they appear at the output of the flip-flop.

- The circuit has advanced by one clock cycle these ***D flip-flop outputs serve as present state information and fed to ROM address decoder***.
- Together with **present input** they point to another location in memory that has next state information.
- ROM being a combinatorial circuit these next state values are immediately available to D flip-flop inputs and the cycle goes on.

The final output is – generate from **state variables in Moore model**

The final output is – generate from **state variables and uses direct input in Mealy model**

Moore model for sequence detector ("011") using ROM

8x2 ROM as there are 8 rows in state synthesis Tables 11.1 and 11.2.

The circuit diagram is shown in Fig. 11.6.

The 3 to 8 address decoder is fed by B, A and X.

The output of decoder is 000, 001, 010.. in same order as they appear in state Table 11.1.

Example:-

when BAX = 000 next state is 00 from state table and we store 00 in ROM corresponding to decoder output 000 and output is generated following logic equation Y = AB.

Function:-

The D flip-flops are initially cleared, i.e. BA = 00.

(a) If X = 0, the first location in ROM corresponding to BAX = 000 is selected and ROM output = 00 and at clock trigger next state remains at BA = 00. Thus 00 state remains at 00 for X = 0.

If X = 1, then BAX = 001 location in ROM is selected which stores 01, i.e. the circuit (D flip-flops) goes to BA = 01 state with clock trigger.

(b) For BA = 01, if X = 0 then BAX = 010 location in ROM is selected which stores 00 that means with NT of clock the circuit goes to state 00 or initial state.
If BA = 01 and X = 1 then BAX = 011 location of ROM is selected which stores 10. Thus next state becomes 10 with NT.

(c) For BA = 10, if X = 0 then BAX =. 100 location is selected which stores 11 and next state becomes 11.
If when 100 location in ROM is selected in ROM the pattern '011' has arrived in proper order. Stored ROM data is immediately available in the **same clock cycle** and generate circuit output from this signaling detection.

- Compared to flip flop implementation here sequence detection signal comes one cycle earlier.

- No need to remember flip-flop excitation table or simplify design equation, which gives different circuit for different problem.
- For all problems circuit remains same only the content of ROM changes.
- The output logic may also be different but an option to store the output as bit in the ROM
- Don't need any **output logic equation** to be realized by basic gates.

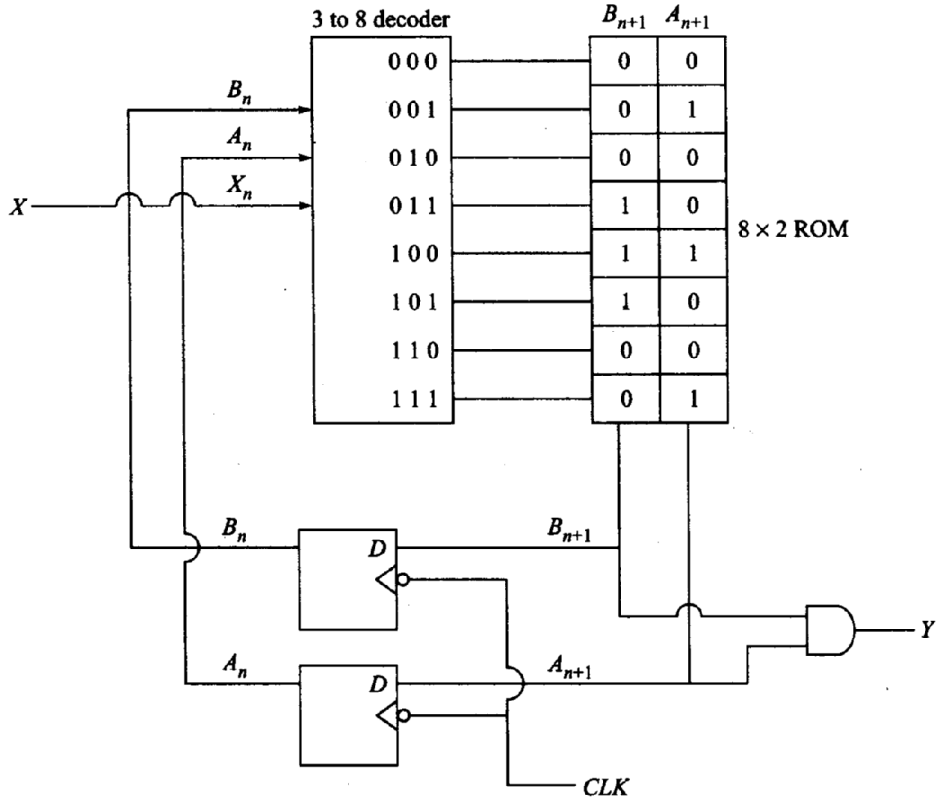


Fig. 11.6 ROM based implementation of sequence detector: Moore model.

Mealy Model

ROM based solution of Mealy model uses state synthesis table in the same way as Moore ROM locations are selected by present state and input as appears in state table and next state's fills corresponding ROM locations (Fig. 11.7).

- Delay flip-flop banks are used in the same way final output is generated from D flip-flop outputs (representing present state) and data input.
- In Moore model used ROM outputs directly to generate sequence detector output. 011 that ROM of size 6 x 2 is sufficient for Mealy model and last two locations of 8 x 2 ROM are not used.

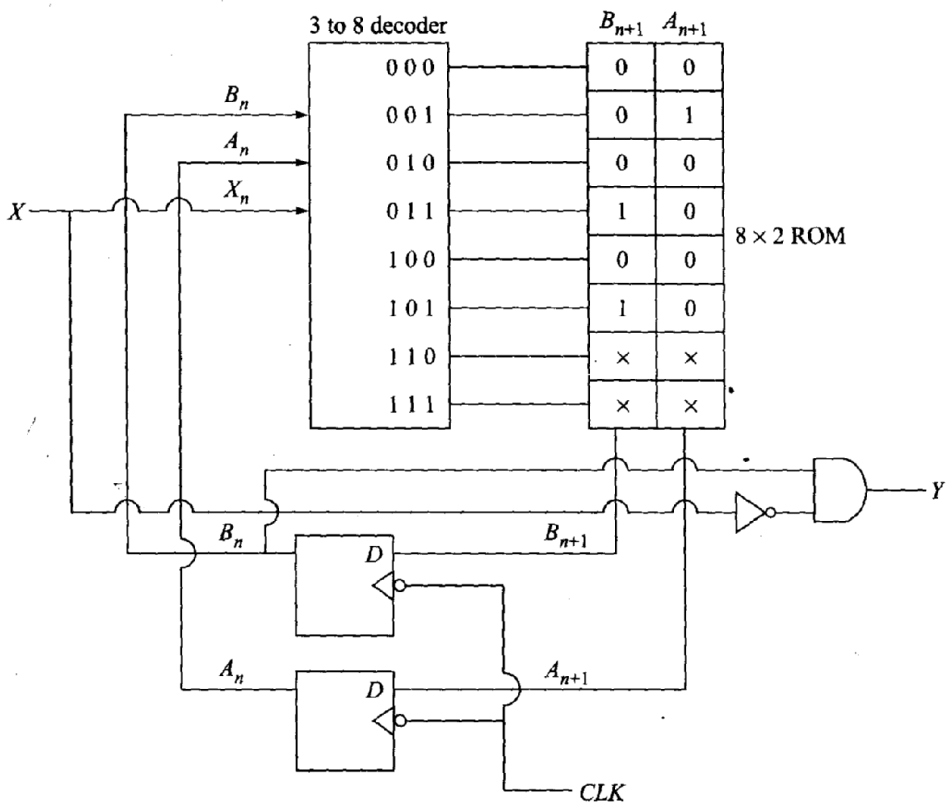


Fig. 11.7 *ROM based implementation of sequence detector: Mealy model.*

Give design equations for the synchronous sequential logic circuit that has two inputs X and Y. The output Z of this circuit is generated according to the timing diagram shown in Fig.11.8.

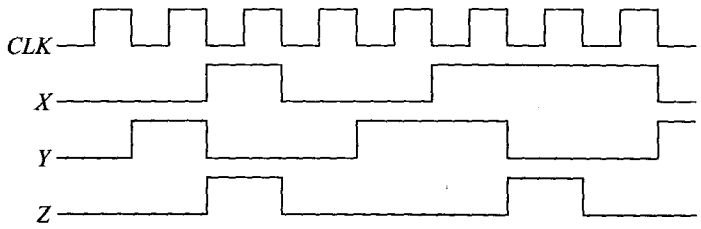
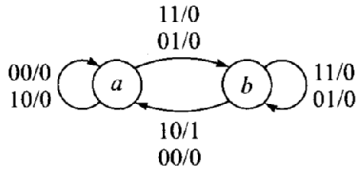


Fig. 11.8 *Timing diagram for Example 11.1.*

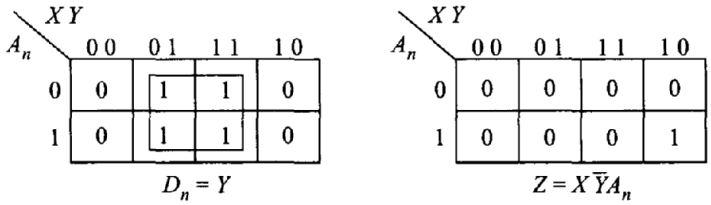
When Y goes from high to low and if at that time (Y low) X remains at logic high. Thus if we adopt a Mealy model, the circuit needs one memory element that remembers if previous state of Y was high for any X = 1, Y = 0 input. The state transition diagram, state synthesis table and design equations are shown in Figs. 11.9(a), (b) and (c) respective The design has been done with D flip-flop in which D input simply follows next state. Refer to excitation table of Fig. 8.34.



(a)

A_n	X	Y	A_{n+1}	D_n	Z
0	0	0	0	0	0
	0	1	1	1	0
	1	0	0	0	0
	1	1	1	1	0
1	0	0	0	0	0
	0	1	1	1	0
	1	0	0	0	1
	1	1	1	1	0

(b)



(c)

Fig. 11.9 *(a) State transition diagram. (b) State synthesis table. (c) Design equations for Example 11.1.*

ALGORITHMIC STATE MACHINE

Algorithmic State Machine (ASM) is a flow chart like representation (ASM Chart) of the a1gorithm a state machine performs.

Advantages:-

- Use for complex problem where number of inputs and states are higher the state diagram it requires more number of states it is difficult to represent using state transition diagram
- It handles implementation issues with greater ease offering better timing information.

- ASM chart, **square boxes** represents a **state.**
- If a state generates an **unconditional output** (Moore model) it can be specified within the **square box**.
- A **diamond shaped box** represents **decision with a question mark**. There are **two exit paths** of this decision box since the decision is binary in nature.
- For Mealy model, **oval shaped boxes** are used to **describe the output** that depends on present state as well as the present input.

- Circles are used to denote start, stop of the algorithm

ASM chart **for sequence detector problem** :-

Moore model:-

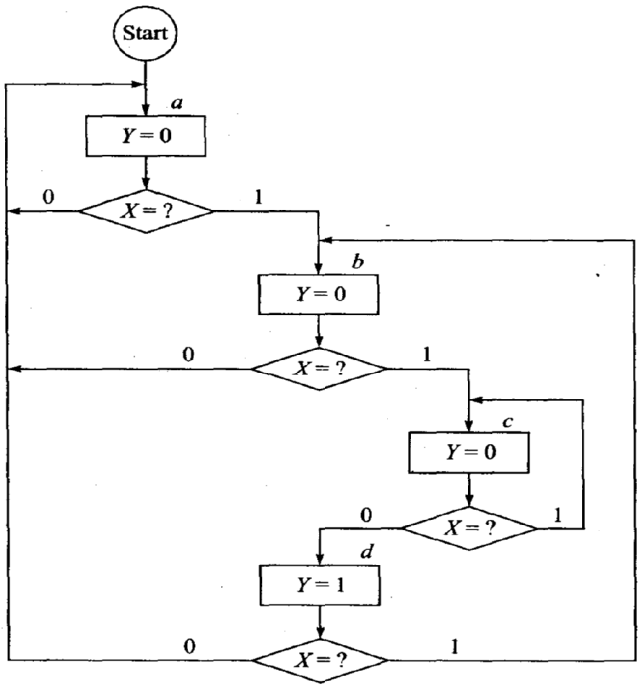


Fig. 11.13 *ASM chart of sequence detector problem: Moore model.*

Mealy model:-

ASM chart for Vending Machine Problem

- The task is to design a synchronous logic control unit of a vending machine
- It take only two types of coins of denomination 1 and 2 in any order.
- It delivers only one product that is priced Rs 3 On receiving Rs 3 the product is delivered by asserting an output X= 1 which otherwise remains 0
- If it gets Rs 4 then product is delivered by asserting X =0 and also a coin return mechanism is activated by output Y = 1 to return a Rs. I coin.

- There are two sensors to sense the denomination of the coins that give binary output as shown in the following table.

I	*J*	Coin
0	×	No Coin dropped
1	0	One Rupee
1	1	Two Rupees

ASM Chart (Mealy Model)

- The ASM chart is prepared following Mealy model and is shown in Fig. 11.10.

- The initial state when no coin is deposited is designated as **state a**. sensor output I = 0 indicates no coin is deposited. so x=0 and y=0 and product is delivered or coin returned
- Check for any coin inserted for every clock cycle

(a) If I = 1, the controller tests J.

If **J = 0** it goes to **state b** that represents **Re. 1** is received

if **J = 1,** goes to **state c** indicating Rs. 2 is received

(b)The controller remains at state b if no further coin is deposited found by checking I.

if **I = 1** and **J = 0**, the machine has received two Re. 1 coins (Rs 2)move to **state c**.

if **I = 1** and **J = 1** means a Rs. 2 coin is received so Rs=1 and Rs=2- totaling **Rs. 3** the cost of the product. the product is delivered by asserting **X = 1** and the circuit goes to initial state.

(c)At **state c**

if on testing I = 1 and J=0 Rs. 1 coin is received(2+1) then **x=1 and y=0**. the product is delivered by asserting **X = 1** and the circuit goes to initial state

if on testing I = 1 and J=1 Rs. 2 coin is received(2+2) then **x=1 and y=1**. the product is delivered by asserting **X = 1** and the circuit goes to initial state and return Rs 1

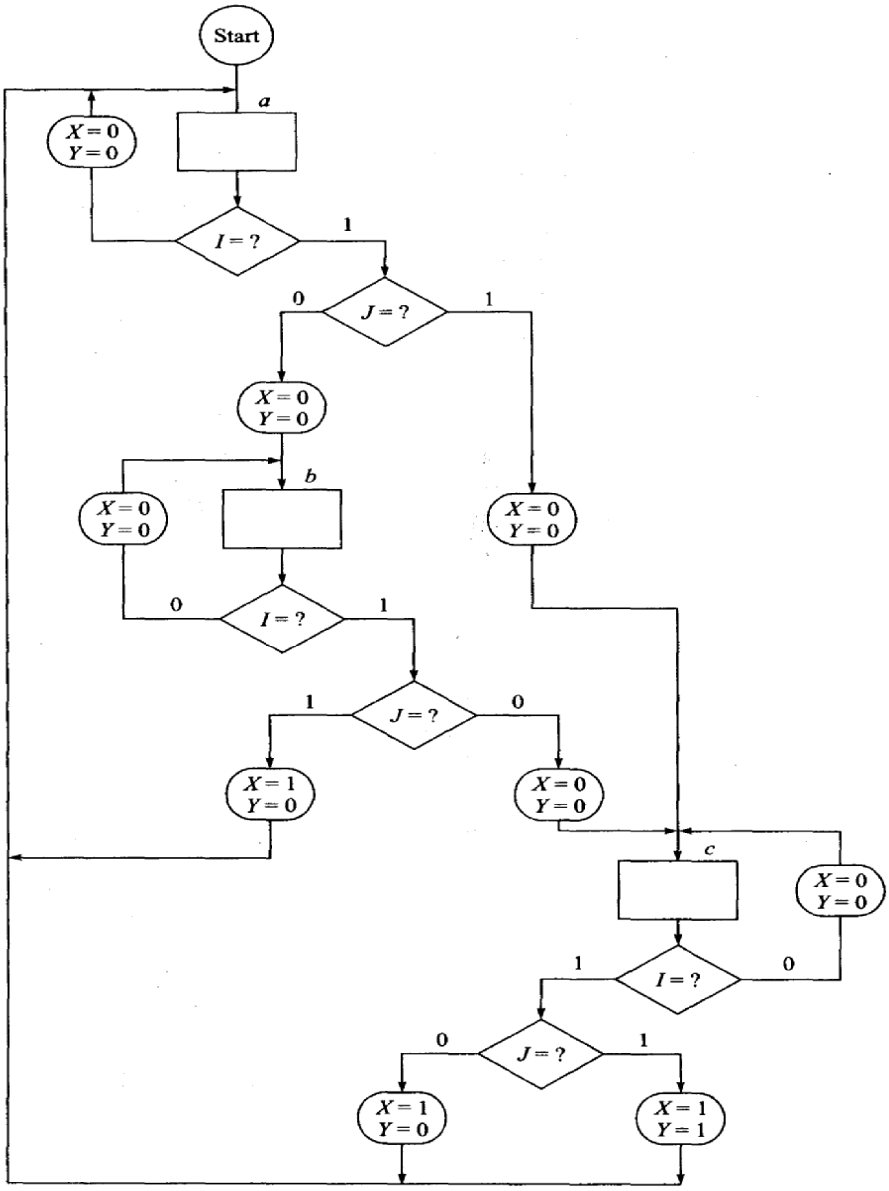


Fig. 11.10 *ASM chart for vending machine problem: Mealy model.*

State Assignment and State Synthesis Table

- Prepare state table from above ASM Chart.
- Use D flip-flop as the memory element corresponds to D input in a given state is easier than JK flip-flop, and has only one data input.
- There are three different states - two flip-flops (say, B and A) to represent them.
- Let BA =00 represent **state a**, BA = 01 **state b**, BA = 10 **state c.** State BA = 11 is not used in this problem.

- Table 11.4 shows the state table corresponding to ASM chart shown in Fig. 11.10 and also the D inputs corresponding to every state.

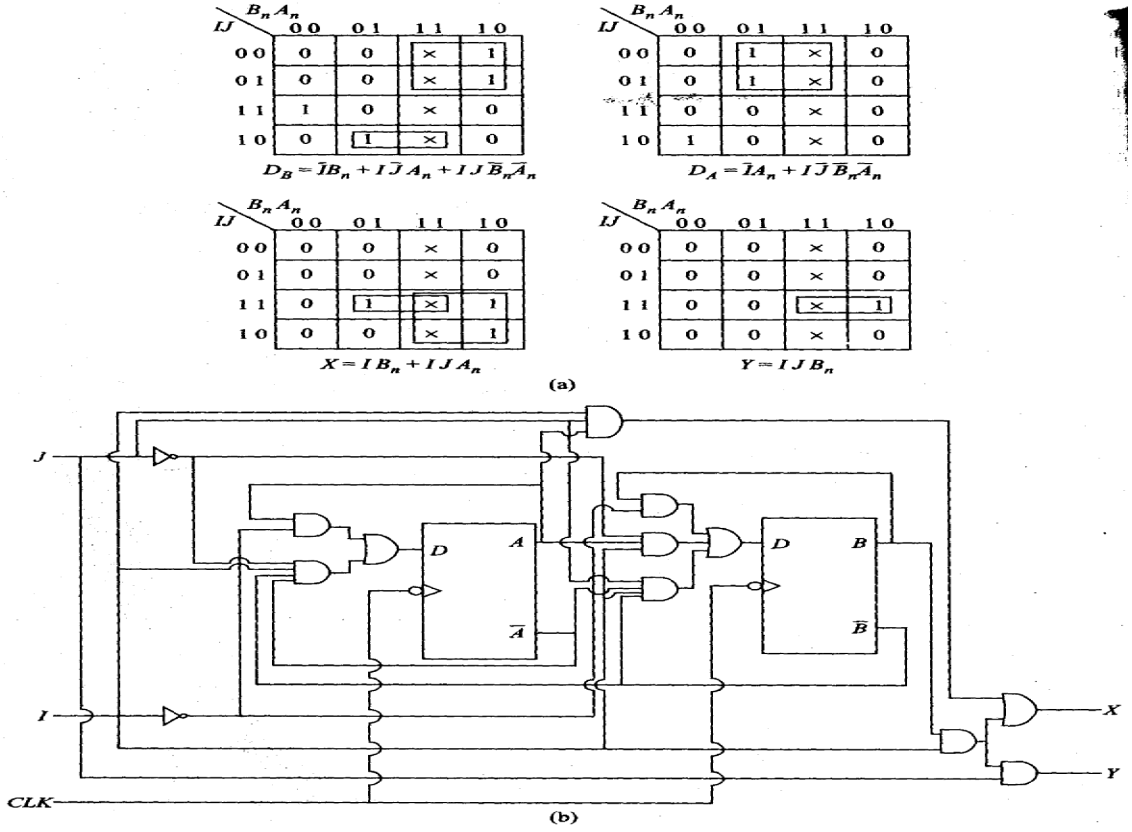
Table 11.4 State Synthesis Table for Vending Machine Problem: Mealy Model

Present State		Input		Next State		Output		D_B	D_A
B_n	A_n	I	J	B_{n+1}	A_{n+1}	X	Y		
0	0	0	0	0	0	0	0	0	0
		0	1	0	0	0	0	0	0
		1	0	0	1	0	0	0	1
		1	1	1	0	0	0	1	0
0	1	0	0	0	1	0	0	0	1
		0	1	0	1	0	0	0	1
		1	0	1	0	0	0	1	0
		1	1	0	0	1	0	0	0
1	0	0	0	1	0	0	0	1	0
		0	1	1	0	0	0	1	0
		1	0	0	0	1	0	0	0
		1	1	0	0	1	1	0	0

Design Equations from Karnaugh map and Circuit Diagram

Karnaugh maps for each flip-flop input and both the outputs arc shown in Fig. 11.1 la along with design equations. for BA = 11 don t care states in each map that helps in minimizing design equation.

The final digital controller circuit for the vending machine problem is shown in Fig. 11.11b.



Implementation using ROM based

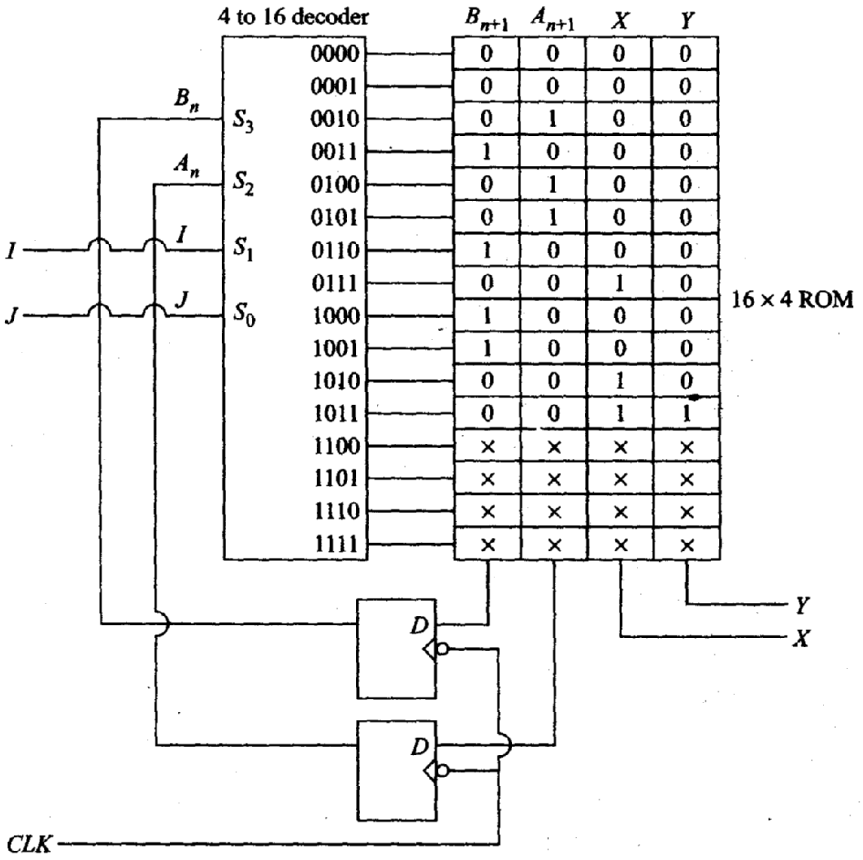


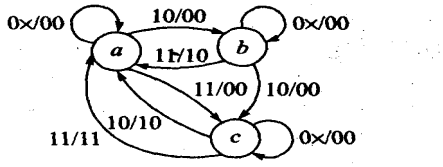
Fig. 11.12a *ROM implementation of vending machine problem.*

- The circuit is shown in Fig. 11.12a. The values stored in ROM is derived from state stable.
- Used higher ROM size adding two more bits in the memory location for each address
- but do not need any basic gate for output logic.
- Use logic gates as shown in sequence detector problem and reduce two columns corresponding to X and Y in ROM.
- **When BAIJ=1010 then x=1 and y=0 product delivered**
- **When BAIJ=1011 then x=1 and y=1 product delivered and Rs 1 returned**

Exercise

Draw the state transition diagram of the Mealy model for the vending machine problem.

Solution



g. 11.12b State transition diagram of vending machine problem.

STATE REDUCTION TECHNIQUE

While converting problem statement to state transition diagram or state table we may use more number of states than necessary. On removing redundant state which require less hardware to implement a circuit.

They are two methods

- Row elimination method
- Implication table method

Example.

State transition diagram drawn following a Mealy model is as shown in Fig. 11.14.

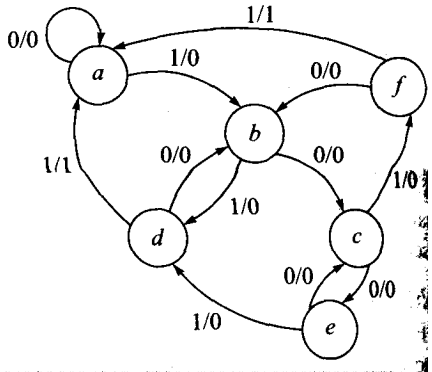


Fig. 11.14 A state transition diag

(a)Row Elimination Method

In this method

- Prepare a state table where at any given state the next state and present output(s) are written for each combination of input(s).
- In this example there are only two possible values of input X = 0 and X = 1.
- For 2 input circuits there will be 2^2 = 4 such combinations in this table.
- Two states are considered equivalent if they move to same or equivalent state for every input combination and also generate same output.

Fig. 11.1 5a shows the state table for Fig. 11.14

Procedure for reduction:-

- Identify states where next state and output are same ,if same eliminate one state and replace that state in next state by remain state.
- If state goes to next state is same as next state to state like bc=cb eliminate one row and replace that state in next state by remain state.

Present state	Next state $X=0$	Next state $X=1$	Present output $X=0$	Present output $X=1$
a	a	b	0	0
$\sqrt{b}$	c	d	0	0
c	e	f	0	0
d	b	a	0	1
$\sqrt{e}$	c	d	0	0
f	b	a	0	1

(a) Original table

Present state	Next state $X=0$	Next state $X=1$	Present output $X=0$	Present output $X=1$
a	a	b	0	0
b	c	d	0	0
c	b	f	0	0
$\sqrt{d}$	b	a	0	1
$\sqrt{f}$	b	a	0	1

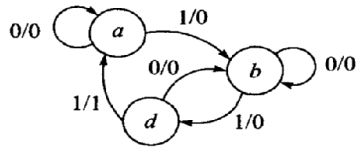
(b) After one row elimination

Present state	Next state $X=0$	Next state $X=1$	Present output $X=0$	Present output $X=1$
a	a	b	0	0
$\sqrt{b}$	c	d	0	0
$\sqrt{c}$	b	d	0	0
d	b	a	0	1

(c) After two row elimination

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

(d) Final reduced table after three row elimination



(e)

Fig. 11.15 *(a)–(d) Tables showing row elimination steps. (e) Reduced state diagram.*

- States b and e are equivalent as next state and output are same.

- Therefore, retain one(b) of these two and discard the other(e). wherever e appears we replace it by b and get table of Fig. 11. 15b.
- Reduced table shows state d and f are equivalent. retain **d** and eliminate row **f** from this table and replace f with d in rest of the rows and get Fig. 11. 15c.
- For states b and c except for next state at X = 0 the rest are same. Now b and c would have been equivalent if these next states are equivalent.
- For b, next state is c and for c, next state is b. so bc=cb and state b is retained and row c is eliminated in the same manner and shown in Fig. 11. 15d, which cannot be further reduced.
- The final reduced state table has three states reduced from six in the original state diagram and final reduced state diagram is shown in Fig. 11.15e.

Present state	Next state $X=0$	Next state $X=1$	Present output $X=0$	Present output $X=1$
a	a	b	0	0
$\sqrt{b}$	c	d	0	0
c	e	f	0	0
d	b	a	0	1
$\sqrt{e}$	c	d	0	0
f	b	a	0	1

(a) Original table

Present state	Next state $X=0$	Next state $X=1$	Present output $X=0$	Present output $X=1$
a	a	b	0	0
b	c	d	0	0
c	b	f	0	0
$\sqrt{d}$	b	a	0	1
$\sqrt{f}$	b	a	0	1

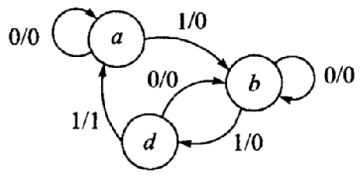
(b) After one row elimination

Present state	Next state $X=0$	Next state $X=1$	Present output $X=0$	Present output $X=1$
a	a	b	0	0
$\sqrt{b}$	c	d	0	0
$\sqrt{c}$	b	d	0	0
d	b	a	0	1

(c) After two row elimination

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

(d) Final reduced table after three row elimination



(e)

Fig. 11.15 *(a)–(d) Tables showing row elimination steps. (e) Reduced state diagram.*

Implication Table Method

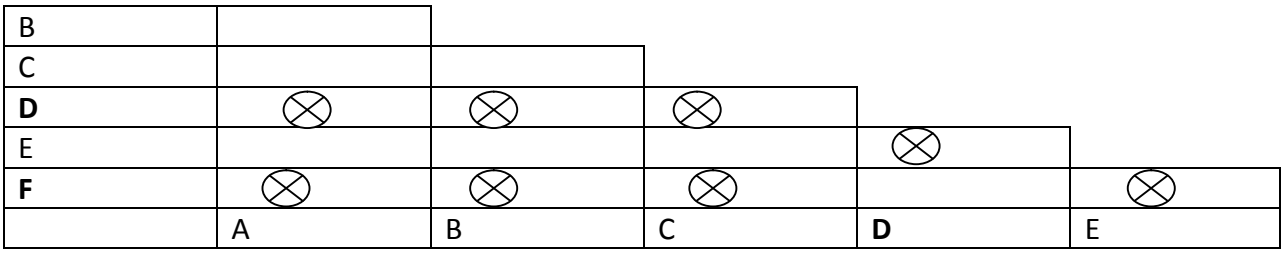
Procedure for reduction

- Identify final and non final state

- Construct table by using states leaving 1st state –vertical and using states leaving last state-horizontal
- Put X mark for one final and other non final state
- Construct state table for input
- Again identify next state for one final and other non final state for any input and put X mark for that state
- Then which left out take those states and left out states also separately

(a) Step1-D and F are final others-non final

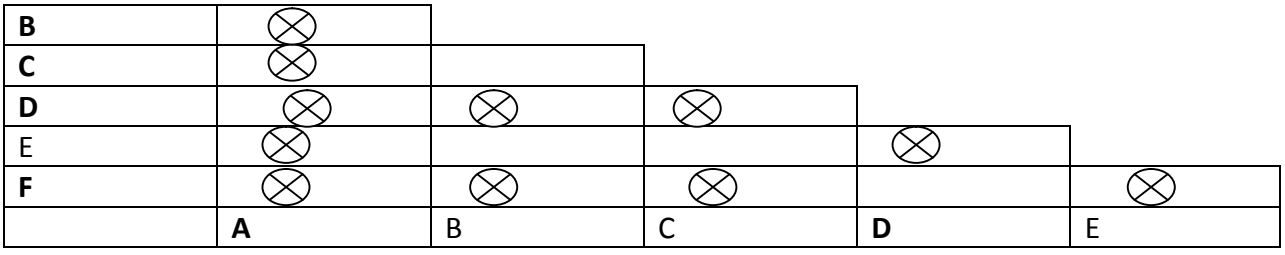
(b) Put x marks as shown below



(c) Construct state table which are left as shown below

STATE	INPUT	
	X=0	X=1
AE »	AC	**BD**
AC »	AE	**BF**
AB»	AC	**BD**
BE	CC	DD
BC	CE	DD
CE	EC	FD
DF	BB	AA

(d) put x marks for AE,AC AND AB



(d)Construct state table which are left as shown below

STATE	INPUT	
	X=0	X=1
BE	CC	DD
BC	CE	DD

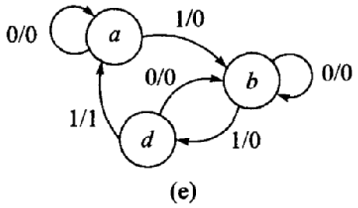
CE	EC	FD
DF	BB	AA

- E->C and E->C eliminate one row (**BE)**

(e)Construct state table which are left as shown below

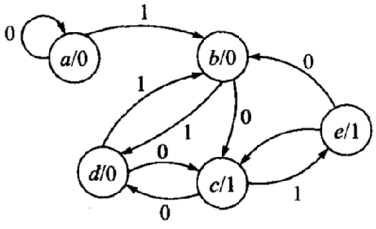
STATE	INPUT	
	X=0	X=1
BC	CE	DD
CE	EC	FD
DF	BB	AA

- (BC)(CE) (DF)
- **COMMON C** in BC and CE so (BCE) (DF) add remaining states in initial .
- **(BCE)= b, (DF)=d ,(A)=a**



(e)

2) Reduce state transition diagram (Moore Model) of Fig. 11. 17a by (i) row elimination method and (ii) implication table method



(a)

Present state	Next state $X=0$	Next state $X=1$	Present output
a	a	b	0
$\sqrt{b}$	c	d	0
c	d	e	1
$\sqrt{d}$	c	b	0
e	b	c	1

(b) Original table

Present state	Next state $X=0$	Next state $X=1$	Present output
a	a	b	0
b	c	b	0
$\sqrt{c}$	b	e	1
$\sqrt{e}$	b	c	1

(c) Table after elimination of one row

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

(d) Final reduced table after elimination of two rows

Fig. 11.17 *Reduction by row elimination method.*

Implication table method

(a) Step1-C and E are final others-non final

(b) Put x marks as shown below

B				
C	⊗	⊗		
D			⊗	
E	⊗	⊗		⊗
	A	B	**C**	D

(d) Construct state table which are left as shown below

STATE	INPUT	
	X=0	X=1
AD **»**	**AC**	BB
AB **»**	**AC**	BD
BD	CC	DB
CE	DB	EC

(d) put x marks for AD,AB

B	⊗			
C	⊗	⊗		
D	⊗		⊗	
E	⊗	⊗		⊗
	A	B	**C**	D

(d)Construct state table which are left as shown below

STATE	INPUT	
	X=0	X=1
BD	CC	DB
CE	DB	EC

- (BD)(CE) add remaining states in initial .
- **(BD)= b, (CE)=C ,(A)=a**

ASYNCHRONOUS SEQUENTIAL CIRCUIT

In synchronous sequential circuit (clock driven circuit)all the memory elements change their states together. since, state change can only take place at time t = nT, where T = Time period of clock signal and inverse of frequency and n is an integer. If the input changes in a manner that warrants change in the state, it cannot do that immediately and wait till the next clock trigger comes. ***Speed is the limitation***

To over come go for Asynchronous Sequential Circuit

Asynchronous Sequential Circuit, also called **Event Driven Circuit** does not have any clock to trigger change of state. State changes are triggered by change in input signal.

Asynchronous sequential circuit is a solution to this but design of such circuit is very complex and has several constraints to be taken care of, which is not required for synchronous circuit.

Problems are:-

- Oscillation of output
- Critical race
- Hazards

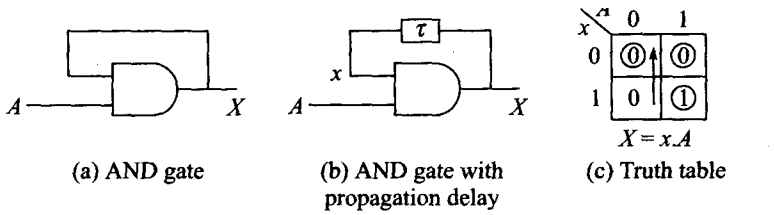
ANALYSIS OF ASYNCHRONOUS SEQUENTIAL CIRCUIT

Memory is the most important element in sequential logic circuit. In synchronous system we use clock driven flip-flops

This is done through feedback similar to basic latch portion of a flip-flop in asynchronous sequential circuit

AND Gate

The two input AND gate with output fed back as one input is shown in Fig. 11.1 9a. The circuit can be redrawn as shown in Fig. 11. 19b that includes the effect of propagation delay of the gate (say, x), the finite time after which a gate reacts to its input.



(a) AND gate (b) AND gate with propagation delay (c) Truth table

Fig. 11.19 *Two input AND gate with output feedback.*

- if X is current output obtained following logic relation and x is the feedback output we write, x = X (t—T).
- Figure. 11.1 9c shows the truth table is also called **state table** and each location in Karnaugh map a state of the asynchronous sequential circuit.
- In the truth table of given circuit and **encircled states indicate stable condition** of the circuit.

Example:-

(a)If A=1 and x=1 then X=1

After PD then

X=1

(b) If A changes from 1 to 0 and x=1

x A=10 and output X=0

x not equal to X (unstable state)

After PD then

x A=00 and output X=0

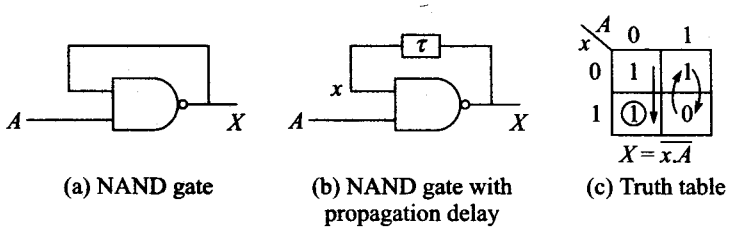
x equal to X (stable state)

For asynchronous sequential circuit. For any state, if x =X then the circuit is stable and if x not= X it is unstable.

NAND Gate

We extend the above observation to feedback NAND circuit shown in Fig. 11.20(a) and arrive at the Truth Table given in Fig. 11.20(c).

For A = 1 there is no stable state and x = X' for both x = 0 and x = 1. Thus there is **oscillation** between x = 0, A = I and x = 1, A = I state.



(a) NAND gate (b) NAND gate with propagation delay (c) Truth table

Fig. 11.20 *Two input NAND gate with output feedback.*

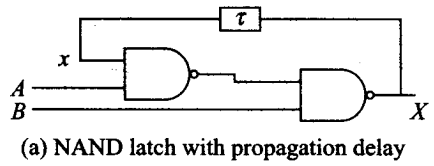
Two Input NAND Latch

In analysis of asynchronous sequential circuit there is an important constraint to be followed.

If **more than one input** feeding the circuit, at a time only **one input variable** can change The other input can change only when the circuit is stabilized following the previous input changed. The time required to stabilize the circuit is in the order of propagation delay of a gate

Similarly, if **two or more output variables** only **one output variable** can change at any time instant, as propagation delays in different paths are different.

While analyzing the NAND latch given in Fig.11.21a we shall keep this in mind.



(a) NAND latch with propagation delay

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

(b) Truth table

Fig. 11.21 *Two input NAND latch.*

The stable and unstable states are arrived at (Fig.11.21b)

if, X = x, the circuit is stable otherwise not. Stables states are encircled and arrows show the movements from transient states.

For each input combination the **circuit has at least one stable state**

Input AB change from 00 to 01: The circuit moves from xAB = 100, a stable position to xAB = 101 (Note, x takes a time T to react to a new set of input) which is unstable and then moves to xAB = 001, a stable state that has output 0. Therefore, a 00->01 transition in AB has output X making l->0 transition.

Input AB changes from 00 to 10: The circuit moves from xAB = 100, a stable position to xAB = 110, another stable state that has output 1. Therefore, a 00->10 transition in AB does not alter the value of output, X = 1.

AB cannot change from 00 to 11 as there will be a finite delay, however small it may be between A and B change. Thus, the transition path of AB is either 00->01-11 1 (then output changes as 1 —>0→0) or 00-> 10—>1 1 (output changes as 1 —>1 —>1) depending on which of A or B changes earlier.

Therefore, output is 0 or 1 depending on intermediate value and in asynchronous logic design such transitions are not allowed.

- Possible transitions of state (xAB) for input change and get transition Table 11.5.
- At AB = 11, there are two stable states xAB = 011 and xAB = 111
- Transition of AB, 01 —> 11 reaches xAB = 011 state while 10—> 11 reaches xAB = 111. so output of the circuit when AB = 11 depends on whether AB = 01 or 10 before AB becomes 11.
- Thus at AB = 11, the circuit generates output x = X from memory or it has latched the information of previous input combination.

Table 11.5 Transition Table of NAND Latch

Input AB	State(xAB) transition	Output X	Remark
00→01	100→101→001	1→0→0	At *AB* = 00,
00→10	100→110	1→1	stable *x* = 1,
01→00	001→000→100	0→1→1	At *AB* = 01,
01→11	001→011	0→0	stable *x* = 0,
10→00	110→100,	1→1	At *AB* = 10,
10→11	110→111	1→1	stable x = 1,
11→01	011→001, 111→101→001	0→0, 1→0→0	At *AB* = 11,
11→10	011→010→101, 111→110	0→1→1, 1→1	stable *x* = 0, 1.

Example:-

(1) Analyze the Mealy model asynchronous sequential circuit of Fig. 11.22 and show its stable state and corresponding outputs. (ii) Give the state diagram of this circuit.

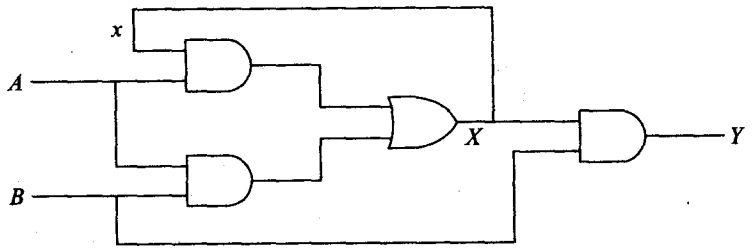


Fig. 11.22 *An asynchronous sequential circuit: Mealy model.*

Solution

- To analyze the circuit x = X(t—T) where T is the cumulative propagation delay from input side up to X. For all possible combinations of xAB .X and Y following logic relation shown in the circuit and prepare Karnaugh map of Fig. 1 I.23a. States where X = x are stable and encircled.
- Outputs corresponding to each state and input combination are shown beside.
- Since, there are two stable states x = 0 and x = I the state diagram can be drawn from Table 11.5 by considering all possible input combinations for each state as shown in Fig. l1.23b.
- The output is dependent on inputs as well as state and is shown along the transition path beside the input.

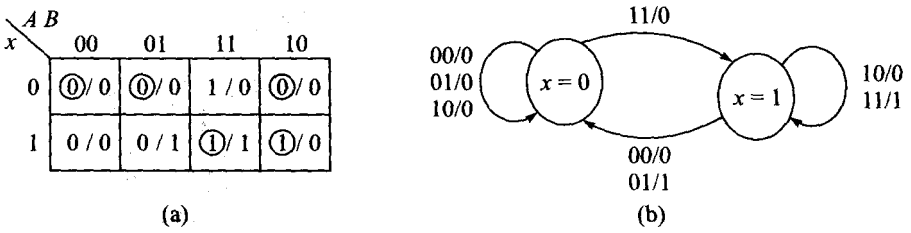


Fig. 11.23 *(a) Karnaugh map. (b) State diagram for asynchronous circuit shown in Fig. 11.*

PROBLEMS WITH ASYNCHRONOUS SEQUENTIAL CIRCUITS

Problems are:-

- Oscillation of output
- Critical race
- Hazards

can cause major problem unless they are addressed at design stage.

These problems are non-issues in synchronous circuit where external clock trigger arrives after all the inputs are stabilized.

EXERCISE

To find these problems in fallowing Truth Table shown in Fig. 11.24

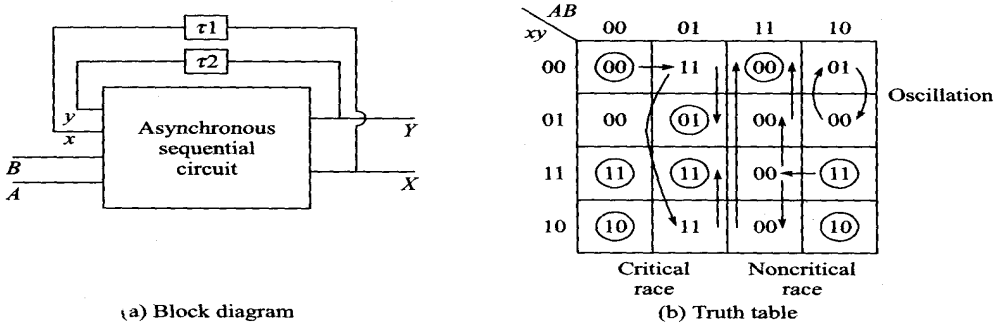


Fig. 11.24 *(a) Block diagram. (b) Truth table of a 2-input, 2-output circuit.*

Asynchronous circuit responds to all the transient values and problems like oscillation, critical race, hazards where the circuit has two external inputs A, B and two outputs X, Y. Both the outputs are fed back to the input side in the form of x and y but with different propagation delays.

Thus x, y cannot change simultaneously but with time delays T1 and T2 respectively and

x = X(t—T1) and y = Yt— T2).

Oscillation

(a)If input AB changes from 00 to 10 and xy=00

xyAB=0000 then XY=00

when AB changes from 00 to 10

xyAB=0001 then XY=01

xy not equal to XY

00!=01(unstable state)

After PD of Y(T2)

xyAB=0101 then XY=00

again

xy not equal to XY

01!=00(unstable state)

After PD of Y(T2)

xyAB=0001 then XY=01

again

xy not equal to XY

00!=01(unstable state)

This will repeat so oscillate between 00 and 01

In asynchronous sequential circuits for any given input, transitions between two unstable states like these are to be avoided to remove oscillation.

Critical Race

Race condition that could be a major problem in asynchronous sequential circuit. This occurs when an input change tries to modify more than one output.

In the truth table of Fig. 11 .24b, consider the stable state xyAB = 0000.

(a)If input AB changes from 00 to 01 and xy=00

xyAB=0000 then XY=00

when AB changes from 00 to 01

xyAB=0001 then XY=11

Asynchronous sequential circuit wil not change both variable at a time ie 00-> 11 or 11->00 because of propagation delay.

So T1 ->x and T2->y

(a)When T1<T2

Then x changes first then y

After PD of x(T1)

xyAB=1001 then XY=11

xy not equal to XY

00!=01(unstable state)

After PD of y(T2)

xyAB=1101 then XY=11

xy equal to XY **so final output is XY=11**

(b)When T2<T1

Then y changes first then x

After PD of y(T2)

xyAB=0101 then XY=01

xy equal to XY

so final output is XY=01

01=01(stable state)

Here circuit settle at two different values this situation is called **critical race condition** and is to be avoided in asynchronous sequential circuit.

NON CRITICAL RACE.

Race can be non-critical too, its presence does not pose any problem for the circuit behavior.

In the truth table, consider stable state xyAB = 1110.

If input AB changes to 11, the circuit goes to xyAB = 1111 where output XY = 00.

when AB changes from 10 to 11

xyAB=1101 then XY=00

Asynchronous sequential circuit wil not change both variable at a time ie 00-> 11 or 11->00 because of propagation delay.

So T1 ->x and T2->y

(a)When T1<T2

Then x changes first then y

After PD of x(T1)

xyAB=1011 then XY=00

xy not equal to XY

10!=00(unstable state)

After PD of y(T2)

xyAB=0011 then XY=00

xy equal to XY **so final output is XY=00**

(b)When T2<T1

Then y changes first then x

After PD of y(T2)

xyAB=1011 then XY=00

xy not equal to XY

10!=00(unstable state)

After PD of x(T1)

xyAB=0011 then XY=00

xy equal to XY

00=00(stable state)

so final output is XY=00

Since, the race condition **does not** lead to two different state it is termed as **non-critical race**.

Hazards

- Static and dynamic hazards causes malfunctioning of asynchronous sequential circuit. Situations like Y = A + A' or Y = AA 'are to be avoided for any input output combination with the help of hazard covers in truth table.
- There can be another problem called essential hazard. This occurs when change in input does not reach one part of the circuit while from other part one output fed back to the input side becomes available. Essential hazard is avoided by adding delay, may be in the form of additional gates that does not change the logic level, in the feedback path.
- This ensures effect of input change propagates to the all parts of the circuit and then only feed back output, generated from that input-change makes its presence felt.

Problem:-

In an asynchronous sequential circuit, the state variable outputs of X and Y are related with primary A and B and its own feedback x and y as shown in Kamaugh map of Fig. 11.25. Can the circuit face problem in its operation?

xy \ AB	00	01	11	10
00	11	(00)	11	(00)
01	(01)	11	11	(01)
11	10	11	(11)	10
10	(10)	(10)	11	11

Fig. 11.25 Kamaugh map for Example

Solution

Asynchronous circuit responds to all the transient values and problems like oscillation, critical race, hazards where the circuit has two external inputs A, B and two outputs X, Y. Both the outputs are fed back to the input side in the form of x and y but with different propagation delays.

Thus x, y cannot change simultaneously but with time delays T1 and T2 respectively and

x = X(t—T1) and y = Yt— T2).

Oscillation

(a)If input AB changes from 11 to 10 and xy=11

xyAB=1111 then XY=11

when AB changes from 11 to 10

xyAB=1110 then XY=10

xy not equal to XY

11!=10(unstable state)

After PD of Y(T2)

xyAB=1010 then XY=11

again

xy not equal to XY

10!=11(unstable state)

After PD of Y(T2)

xyAB=1110then XY=10

again

xy not equal to XY

11!=10(unstable state)

This will repeat so oscillate between 11 and 10

In asynchronous sequential circuits for any given input, transitions between two unstable states like these are to be avoided to remove oscillation.

Critical Race

Race condition that could be a major problem in asynchronous sequential circuit. This occurs when an input change tries to modify more than one output.

In the truth table of Fig. 11 .24b, consider the stable state xyAB = 0001.

(a)If input AB changes from 01 to 00 and xy=00

xyAB=0001 then XY=00

when AB changes from 01 to 00

xyAB=0000 then XY=11

Asynchronous sequential circuit wil not change both variable at a time ie 00-> 11 or 11->00 because of propagation delay.

So T1 ->x and T2->y

(a)When T1<T2

Then x changes first then y

After PD of x(T1)

xyAB=1000 then XY=10

xy equal to XY **so final output is XY=10**

(b)When T2<T1

Then y changes first then x

After PD of y(T2)

xyAB=0100 then XY=01

xy equal to XY

so final output is XY=01

01=01(stable state)

Here circuit settle at two different values this situation is called **critical race condition** and is to be avoided in asynchronous sequential circuit.

NON CRITICAL RACE.

Race can be non-critical too, its presence does not pose any problem for the circuit behavior.

In the truth table, consider stable state xyAB = 0001.

If input AB changes to 11, the circuit goes to xyAB = 0011 where output XY = 11.

when AB changes from 01 to 11

xyAB=0011 then XY=11

Asynchronous sequential circuit wil not change both variable at a time ie 00-> 11 or 11->00 because of propagation delay.

So T1 ->x and T2->y

(a)When T1<T2

Then x changes first then y

After PD of x(T1)

xyAB=1011 then XY=11

xy not equal to XY

10!=11(unstable state)

After PD of y(T2)

xyAB=1111 then XY=11

xy equal to XY **so final output is XY=11**

(b)When T2<T1

Then y changes first then x

After PD of y(T2)

xyAB=0111 then XY=11

xy not equal to XY

01!=11(unstable state)

After PD of x(T1)

xyAB=1111 then XY=11

xy equal to XY

11=11(stable state)

so final output is XY=11

Since, the race condition **does not** lead to two different state it is termed as **non-critical race**.

DESIGN OF ASYNCHRONOUS SEQUENTIAL CIRCUIT

Steps involve in design of asynchronous sequentail circuit are:-

1. Find relationship between input and output
2. Construct state transition diagram
3. Construct primitive table /primitive flow table/simple flow table
4. State reduction –either using row elimination method/implication table method
5. State assignment
6. Design input equation and circuit diagram

This makes the design of such circuit complex and cumbersome and if benefits like speed are not of critical importance, synchronous design is preferred to asynchronous design

Example:-

- A digital logic circuit is to be designed that has **two inputs A, B** and one output X.

I Relationship between input and output

- X goes high ifat A = 1, B makes a transition 1—> 0.
- X remains high as long as this A = 1, B = 0 are maintained.
- If any of A or B changes at this time output X goes low.
- It becomes high again when at A = 1,B goes from 1 to 0.

The timing diagram corresponding to this problem is shown in Fig. 11.26.

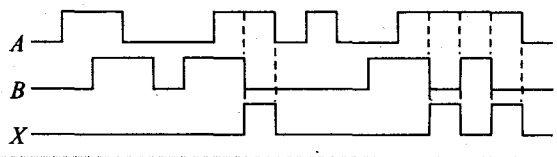


Fig. 11.26 *Timing diagram of the problem.*

II Construct state transition diagram

State transition diagram, using Moore model. The state symbol and output at that state is shown together within a circle in this diagram (Fig. 11.27).

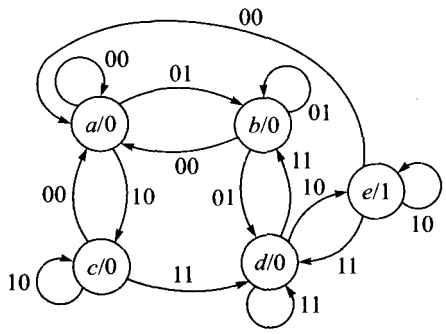


Fig. 11.27 *State transition diagram of the problem.*

- Let the initial state be considered as a when AB = 00 with output 0.
- As long as AB remains 00, the circuit remains at **a.**
- **A**s soon as one of A or B changes the circuit may immediately move to different states.
- A and B cannot change together, a constraint in asynchronous design.
- At state a, if input AB = 01, the circuit goes to **state b** and if input AB = 10 it goes to **c**
- (0O—>1 1 prohibited).
- Both b and c generate output 0 as they have not yet fulfilled the condition stated in the problem for assertion of output.
- The circuit remains at b for AB = 01. If AB changes to 11, the circuit moves to state d with output X = 0
- if AB becomes 00 the circuit goes back to **a**.
- Similarly the circuit stays **at c** if input stays at 10 but goes **to d receiving** 11 and to **a receiving 00**.
- At state d, the input AB = 11 and now if B changes from 1—>0 then condition for output X = 1 is fulfilled and next state e for AB = 10 shows output as 1.
- The circuit remains at state e as long as AB = 10. It goes back to state d if AB becomes 11 because if B again goes to 0 output should be high.
- At e, if AB changes to 00 the circuit goes to initial state a as AB becoming 10 following 00 will not assert output.

Construct primitive table /primitive flow table/simple flow table

Primitive Table

The next step is to form state table from state transition diagram.

In this table if all the rows representing a state has only one stable state for all possible input combinations and it is termed as **primitive table, or primitive flow table or simply flow table**. Primitive table prepared from state transition diagram is shown in table Fig. 11.28.

	AB 00	01	11	10	X
a	(a)	b	×1	c	0
b	a	(b)	d	×2	0
c	a	×3	d	(c)	0
d	×4	b	(d)	e	0
e	a	×5	d	(e)	1

Fig. 11.28 *Primitive table for the problem.*

Each row in this table has one don't care state. The don't care state in each row comes which asks for both the input variables to change to move from stable state, a condition not allowed in asynchronous sequential logic

IV State reduction –either using row elimination method/implication table method

- **State reduction** is always useful to check state redundancy before going for actual circuit design.
- Removing redundant states helps in generating the circuit in a simpler way and with less hardware. We **use implication table** for this example to remove redundant state, if there is any.

Implication table method,

B				
C				
D				
E	⊗	⊗	⊗	⊗
	A	B	C	D

(A) CE one final and one non final state .put x marks for AD and CD

State	input			
	xy=00	xy=01	xy=10	xy=11
AD	AX1	BB	**CE**	X1D
AC	AA	BX3	CC	X1D
AB	AA	BB	CX2	X1D
BD	AX4	BB	X2E	DD
BC	AA	BX3	X2C	DD
CD	AX4	X3B	**CE**	DD

B				
C				
D	⊗		⊗	
E	⊗	⊗	⊗	⊗
	A	B	C	D

(B)Replace x1=d ,x2=c and x3=b

State	input			
	xy=00	xy=01	xy=10	xy=11
AC	AA	BB	CC	DD
AB	AA	BB	CC	DD
BD	AX4	BB	CE	DD
BC	AA	BB	CC	DD

(AC)(AB) AND (BC)

Common A in (AC)(AB)->ABC

(ABC)(BC)

Common BC in (ABC)(BC) ->ABC

(ABC)=a (D)=d (E)=e

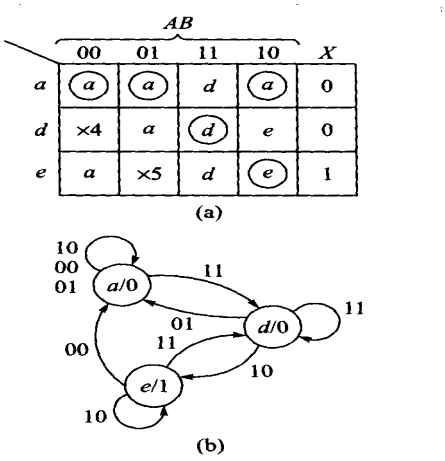


Fig. 11.30 *Reduced (a) State table. (b) State transition diagram.*

V State assignment

- This step in asynchronous sequential circuit design has to be done very carefully so that a valid state transition does not require two or more output variables to change simultaneously which may lead to racing problem.
- In this problem there are three states in the reduced state diagram which needs two variables to represent them. Figure 11.30 shows that we cannot avoid two variables changing together in one or more occasions for the reduced state transition diagram.
- If {a,d,e) is represented by (00,01,10) it occurs twice for d—>e and e—>d transitions in state transition diagram.
- A representation of (00,01,11) requires two variables to change only once when e—>a transition occurs.
- The solution to this may be found if the unused fourth combination of two variable representation is used as a dummy state, include between **e and a.**
- if one dummy state is not enough need to use a third variable to represent the states that will make 2— 3 = 5 dummy variable available for this purpose.
- Represent the states in this problem by two variables PQ in the following way

The modified state diagram and state table with dummy variable =10 included are shown in Fig. 11.31.

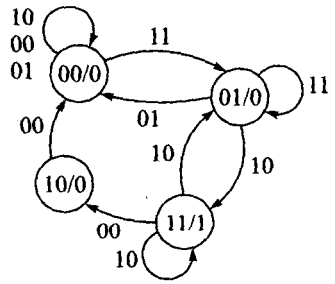


Fig. 11.31 *Modified state transition diagram.*

a:00 ,d:01,e:11 and dummy state =10

- Dummy state is an unstable state and before the input can change it goes to next stable **state a.**
- Represent state variables by P and Q, the corresponding feedback variables are represented by p and q respectively.

VI Design Equations and Circuit Diagram

Use Karnaugh map to get expression of state variables P and Q as a function of input A,B and feedback variables p and q. The equations derived from Karnaugh map are shown in Fig. 11.32. The equation of output X is generated from P and Q as use Moore model. The final circuit is developed from these equations and is shown in Fig. 11.33.

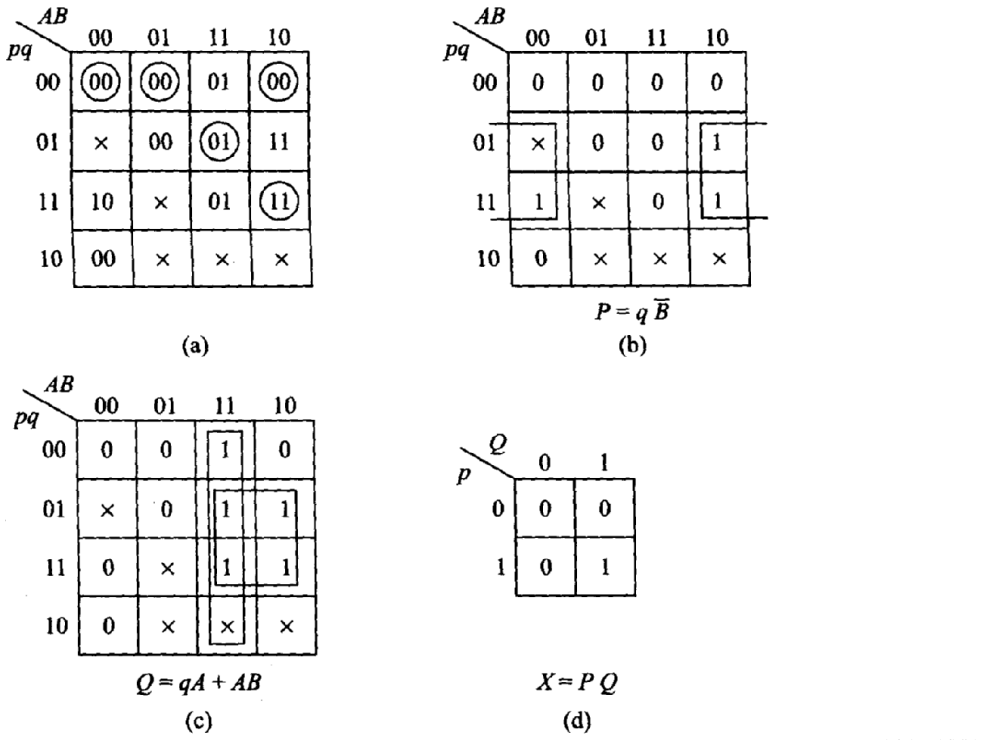


Fig. 11.32 *(a) Reduced state diagram from Fig. 11.27. (b)–(d) Karnaugh map and design equations.*

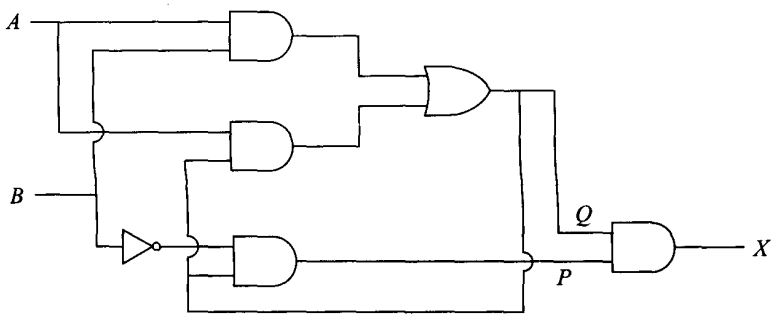


Fig. 11.33 Circuit diagram of asynchronous sequential logic for the problem in Fig. 11.23.

Problem:-

Design an asynchronous sequential logic circuit for state transition diagram shown in figure

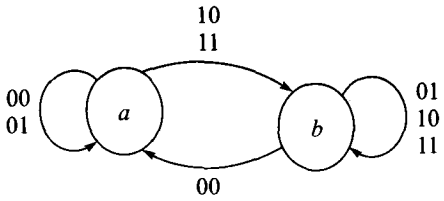


Fig. 11.34 *State transition diagram for Example 11.8.*

Solution

- Let the two input variables be termed A and B
- since the state transition diagram has two states
- then $\log_2 2)=1$ so one output feedback serving as memory (output variable be termed X and its feedback x..)
- If current state a as x = 0 and b as x = I then output X can be expressed as shown in Fig. ll.35b.
- The asynchronous sequential logic circuit drawn from design equation is shown in Fig. ll.35c.

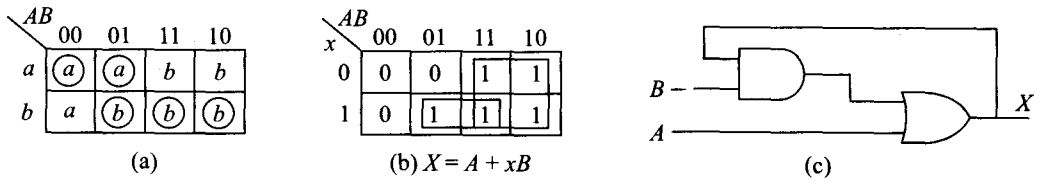


Fig. 11.35 *Solution for Example 11.8.*