UNIT IV

ASYNCHRONOUS SEQUENTIAL CIRCUITS

Introduction

In asynchronous sequential circuits memory elements are either un clocked flip-flops or time delay elements. Therefore in asynchronous sequential circuits change in input signals can affect memory element at any instant of time. In synchronous circuits the designer has to consider the time delays involved to determine the maximum operating speed of the clock. In asynchronous circuits clock is absent and state change occurs according to delay times of the logic. Due to this asynchronous circuits are more difficult to design. However because of absence of clock asynchronous circuits are faster than synchronous circuits.

Types of asynchronous circuits

Q1a)Explain about asynchronous sequential circuits?
(or)
b)Describe the types of asynchronous circuits

It consists of a combinational circuit and delay elements connected to form feedback loops. There are n input variables, m output variables, and k internal states. The delay elements provide a short term memory for the sequential circuit. The present state and next state variables in asynchronous sequential circuits are called secondary and excitation variables respectively.

According to the input variables are to be considered there are two types of asynchronous circuits

- Fundamental mode circuits
- Pulse mode circuits

Fundamental mode circuits

It assumes that:

Input changes should be spaced in time by at least, the time needed for the circuit to settle into a stable state following an input change. The input variables should change only when the circuit is stable.

Only one input variable can change at a given instant
of time Inputs are levels and not pulses Delay lines are
used as memory elements

Pulse mode circuit

It assumes that:

- The input variables are pulses instead of levels
- The width of the pulses is long enough for the circuit to respond to the input

- The pulse width must not be so long that it is still present after the new state is reached Pulses should not occur simultaneously on two or more input lines
- Flip-flops are commonly used as a memory elements
  Memory element transitions are initiated only by input pulses
- Input variables are used only in the un complemented or the complemented forms but not both.

Analysis of pulse mode asynchronous circuits

In the Analysis of pulse mode asynchronous sequential circuits, circuits respond immediately to pulse on their inputs rather than waiting for clock signal, as in synchronous sequential circuits. The pulse mode circuits assume that pulses do not occur simultaneously on two or more input lines means that a circuit with n input lines has only n+ 1 input condition rather than 2^n as is the case for synchronous circuits. Hence the memory elements of the circuit respond only when an input pulse arrives.

Design of pulse mode circuit

The design of pulse mode circuits is similar to the design of synchronous circuits. However when designing pulse mode circuits remember that no clock pulse is present inputs occur on only one line at a time and only un complemented forms of input signals may be used.

The absence of a clock pulse indicates that latch or flip-flop triggering must be accomplished by utilizing the pulses on the input signals and therefore all circuit timing information must be obtained from the input pulses. Hence the input pulses not only provide input information but also assume the functions performed by the clock pulse in synchronous circuits.

The steps involved in the design of pulse mode asynchronous sequential circuits are

- Define states and draw a state diagram and/or state table of the circuit.
- Minimize the state table.
- Do state assignment.
- Choose the type of latch or flip-flop to be used and determine excitation equations.
- Construct excitation table for the circuit.
- Determine the output equation and the flip-flop input equation using k-map simplifications.
- Draw the logic diagram.

Analysis of fundamental mode sequential circuits:

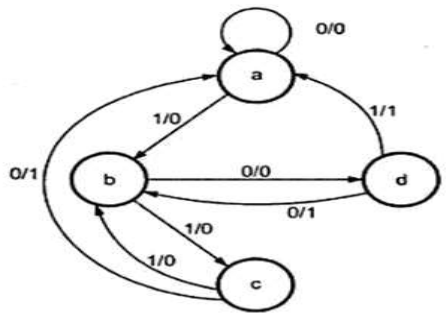
Fundamental mode asynchronous sequential circuit analysis requires careful attention because these circuits utilize unclocked memory and level inputs. The procedure to analyze these circuits is as follows:

- Determine the next state and output equation from the given sequential circuits
- Construct the state table
- Construct the transition table
- Construct output map

Q2a) Design a clocked sequential machine using T flip-flops for the following state diagram. Use state reduction if possible. Also use straight binary state assignment.

(or)

b)Design a synchronous sequential circuit using T flip-flops



Solution: State table for the given state diagram shown in table

Present State	Next state		Output (Z)	
	X = 0	X = 1	X = 0	X = 1
a	a	b	0	0
b	d	c	0	0
c	a	b	1	0
d	b	a	1	1

Even though states a and c are having same next states for input X=0 and X=1, as the outputs are not same state reduction is not possible.

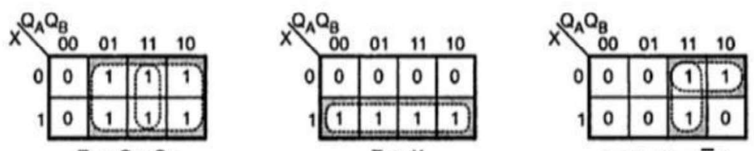
Using straight binary assignments as a=00, b=01, c=10 and d=11, the transition table(Excitation table) is shown in table.

Input	Present state		Next state		Flip-flop input		Output
X	Q_A	Q_B	Q_{A+1}	Q_{B+1}	T_A	T_B	Z
0	0	0	0	0	0	0	0
0	0	1	1	1	1	0	0
0	1	0	0	0	1	0	1
0	1	1	0	1	1	0	1
1	0	0	0	1	0	1	0
1	0	1	1	0	1	1	0
1	1	0	0	1	1	1	0
1	1	1	0	0	1	1	1

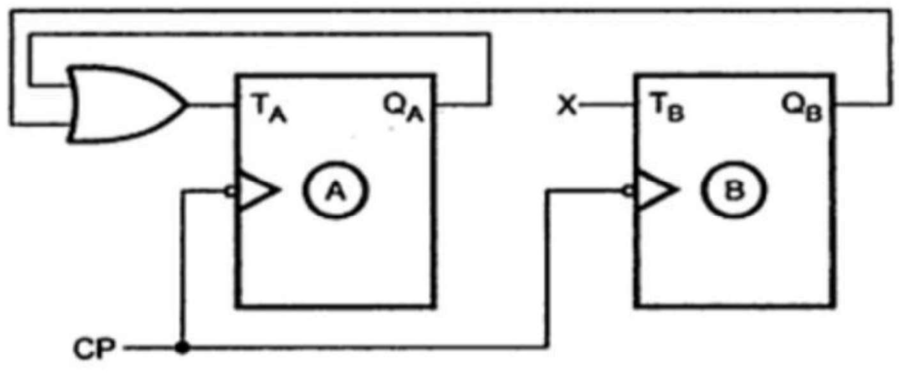
The flip-flop inputs and the circuit outputs are

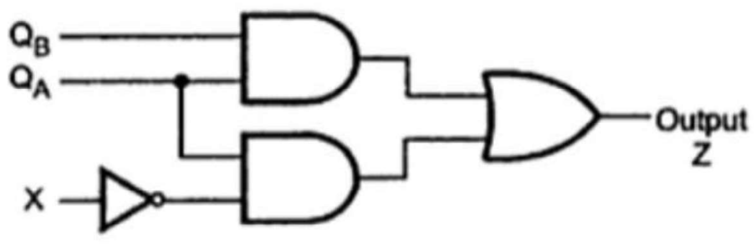
$$T_A = f(Q_A, Q_B, X)$$
$$T_B = f(Q_A, Q_B, X)$$
$$Z = f(Q_A, Q_B, X)$$

K-map Simplification



$T_A = Q_A + Q_B$ $T_B = X$ $Z = Q_A Q_B + \bar{X} Q_A$

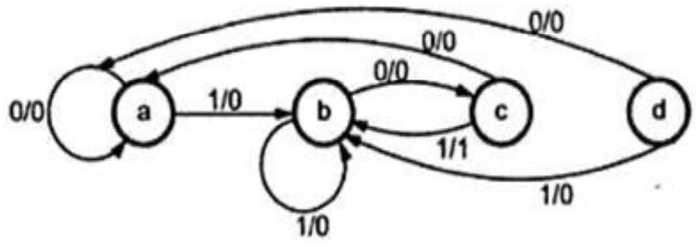




Q3a) Design a clocked sequential machine using JK flip-flops for the state diagram shown in figure. Use state reduction if possible. Make proper state assignment.

(or)

b) Design a clocked sequential machine using JK flip-flops



Step 1: Write the State table

State	Output	
	X = 0	X = 1
a	a, 0	b, 0
b	c, 0	b, 0
c	a, 0	b, 1
d	a, 0	b, 0

In the above table a, d have the same output hence we can reduce it to single.

Step 2: Reduced state table

State	Output	
	X = 0	X = 1
a	a, 0	b, 0
b	c, 0	b, 0
c	a, 0	b, 1

Now each state is assigned with binary values. Since there are three states, number of flip-flops required is two and 2 two binary numbers are assigned to the states as shown below:

a =00, b=01 and c=10

With above state assignment, the excitation table can be given as shown below

Step 3: write the excitation table for JK flipflop

Input	Present State		Next State		Flip-flop inputs				Output
X	Q_B	Q_A	Q_{B+1}	Q_{A+1}	J_B	K_B	J_A	K_A	Z
0	0	0	0	0	0	X	0	X	0
0	0	1	1	0	1	X	X	1	0
0	1	0	0	0	X	1	0	X	0
1	0	0	0	1	0	X	1	X	0
1	0	1	0	1	0	X	X	0	0
1	1	0	0	1	X	1	1	X	1

K-map Simplification

For J_B

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

$J_B = \overline{X}\,Q_A$

For K_B

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

$K_B = 1$

For J_A

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

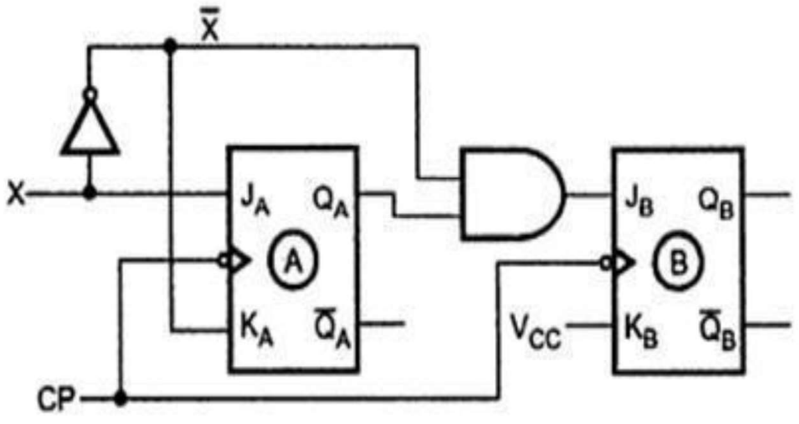
$J_A = X$

For K_A

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

$K_A = \overline{X}$

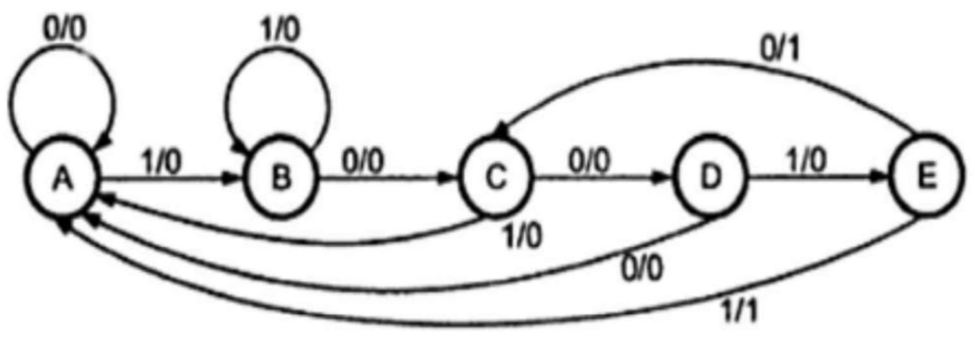
Step 4: Draw the Logic diagram



Q4a)Design the sequential circuit using D flip-flops for the state diagram given in figure

(or)

b)Design the circuit using D flipflop



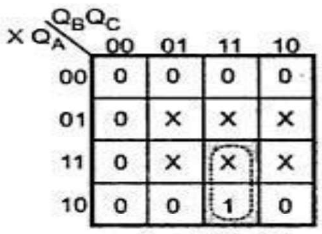
Solution : State assignments are A = 000, B = 001, C = 010, D = 011, E = 100. Three D flips-flops are required.

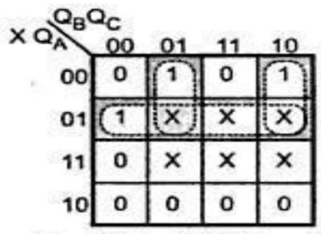
X	Present state			Next state			Output
	Q_A	Q_B	Q_C	Q_{A+1}	Q_{B+1}	Q_{C+1}	Y
0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0
0	0	1	0	0	1	1	0
0	0	1	1	0	0	0	0
0	1	0	0	0	1	0	1
1	0	0	0	0	0	1	0
1	0	0	1	0	0	1	0
1	0	1	0	0	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	0	0	0	1

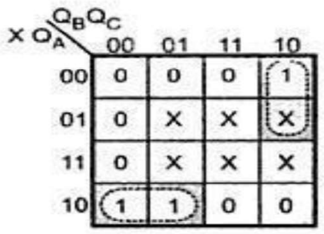
The flip-flop inputs D_A, D_B and D_C are not included in the excitation table as they equal to the next state.

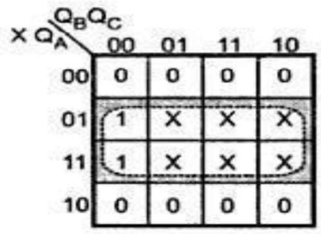
$$D_A = Q_{A+1}, \quad D_B = Q_{B+1}, \quad \text{and} \quad D_C = Q_{C+1}$$

Kmap simplifications

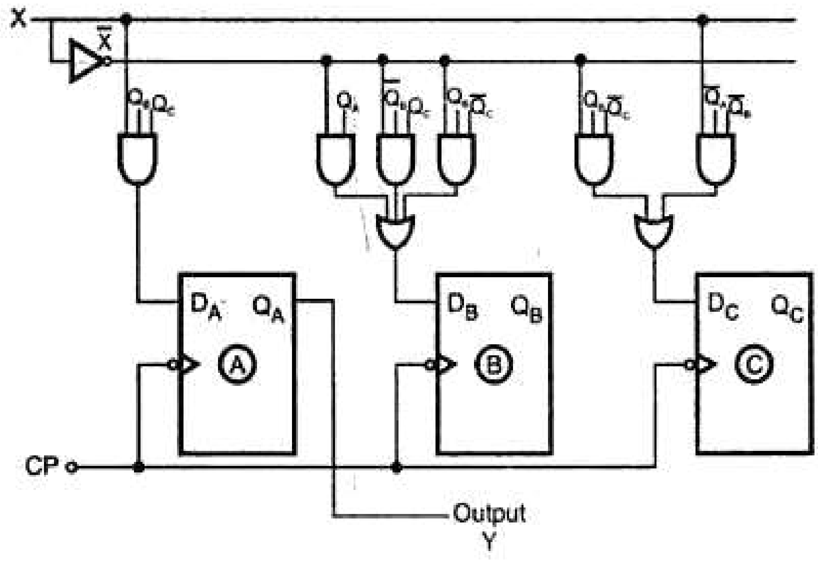

$\therefore D_A = X Q_B Q_C$


$\therefore D_B = \overline{X} Q_A + \overline{X}\,\overline{Q}_B Q_C + \overline{X} Q_B \overline{Q}_C$


$\therefore D_C = \overline{X} Q_B Q_C + X Q_A \overline{Q}_B$


$\therefore Y = Q_A$

The logic diagram is as shown below :

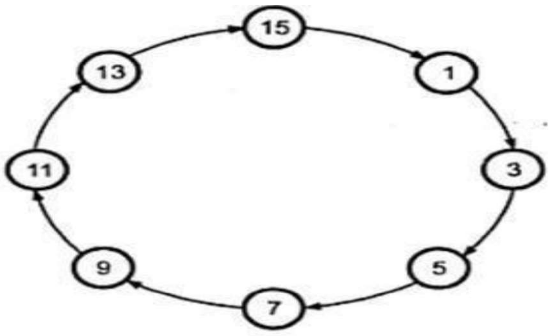


Q5a) Design and implement 4 bit binary counter (using D flip-flops) which counts all possible odd numbers only.

(or)

b)Implement the binary counter using D flip-flop

Solution: The state diagram for given problem is as shown in the figure



Excitation table

Present state				Next state			
Q_A	Q_B	Q_C	Q_D	Q_{A+}	Q_{B+}	Q_{C+}	Q_{D+}
0	0	0	0	×	×	×	×
0	0	0	1	0	0	1	1
0	0	1	0	×	×	×	×
0	0	1	1	0	1	0	1
0	1	0	0	×	×	×	×
0	1	0	1	0	1	1	1
0	1	1	0	×	×	×	×
0	1	1	1	1	0	0	1
1	0	0	0	×	×	×	×
1	0	0	1	1	0	1	1
1	0	1	0	×	×	×	×
1	0	1	1	1	1	0	1
1	1	0	0	×	×	×	×
1	1	0	1	1	1	1	1
1	1	1	0	×	×	×	×
1	1	1	1	0	0	0	1

K-map simplification

For D_A

$Q_AQ_B \backslash Q_CQ_D$	00	01	11	10
00	X	0	0	X
01	X	0	1	X
11	X	1	0	X
10	X	1	1	X

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

For D_B

$Q_AQ_B \backslash Q_CQ_D$	00	01	11	10
00	X	0	1	X
01	X	1	0	X
11	X	1	0	X
10	X	0	1	X

$$D_B = Q_B \bar{Q}_C + \bar{Q}_B Q_C$$

For D_C

$Q_AQ_B \backslash Q_CQ_D$	00	01	11	10
00	X	1	0	X
01	X	1	0	X
11	X	1	0	X
10	X	1	0	X

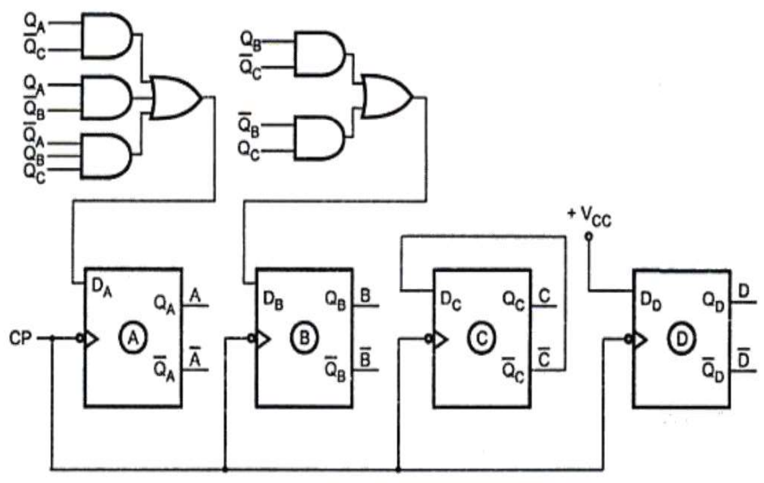
$$D_C = \bar{Q}_C$$

For D_D

$Q_AQ_B \backslash Q_CQ_D$	00	01	11	10
00	X	1	1	X
01	X	1	1	X
11	X	1	1	X
10	X	1	1	X

$$D_D = 1$$

Logic diagram



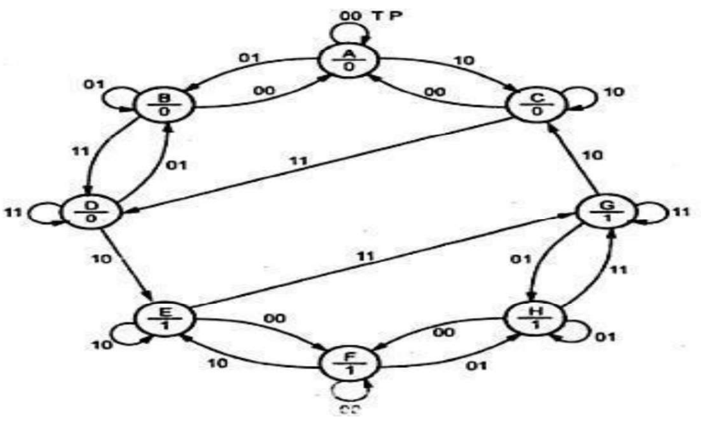
Q6a) Design a T flipflop from logic gates

(or)

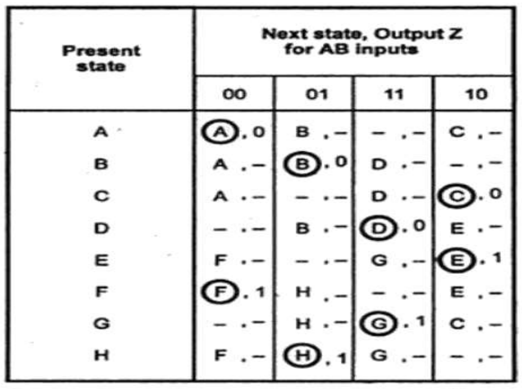
b)The T flipflop has one excitation input and one clock input. But here we use another input P that will function as a clock. The flipflop will change state if T=1 and when he clock P changes from 1 to 0. Under all other input conditions output Q will remain constant. We assume that T and P do not change simultaneously.

Solution:

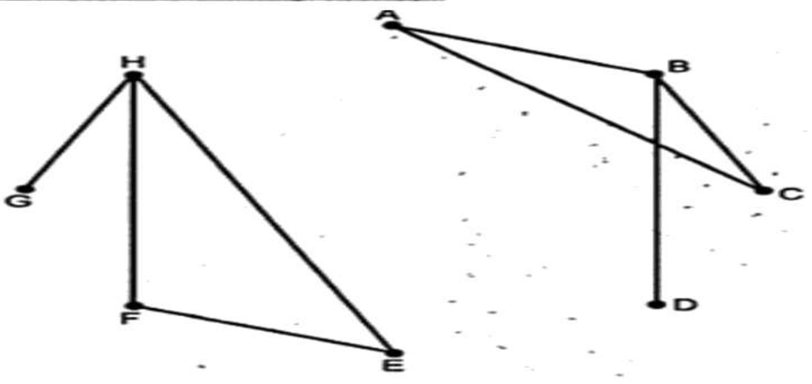
The state diagram for above problem state is as shown in figure



Draw the primitive flow table



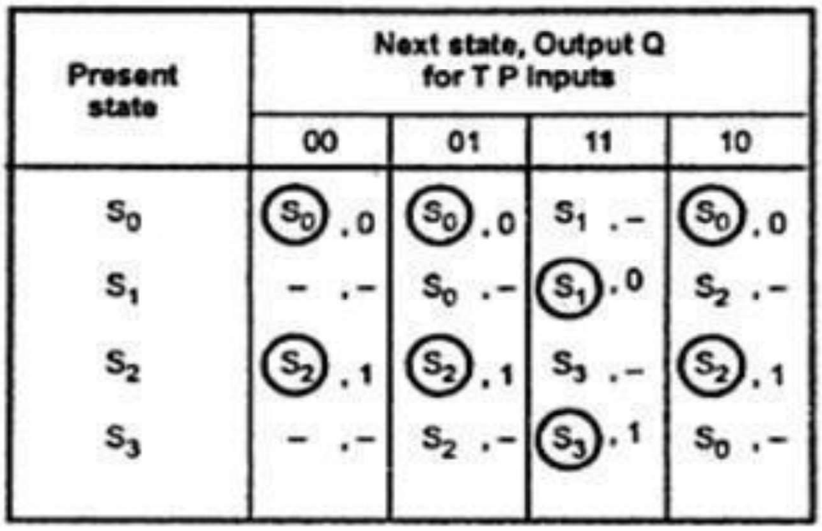
Present state	Next state, Output Z for AB inputs			
	00	01	11	10
A	Ⓐ, 0	B , –	– , –	C , –
B	A , –	Ⓑ, 0	D , –	– , –
C	A , –	– , –	D , –	Ⓒ, 0
D	– , –	B , –	Ⓓ, 0	E , –
E	F , –	– , –	G , –	Ⓔ, 1
F	Ⓕ, 1	H , –	– , –	E , –
G	– , –	H , –	Ⓖ, 1	C , –
H	F , –	Ⓗ, 1	G , –	– , –



The merger graph shown in figure gives the four compatible pairs as a set of maximal compatibles

(A, B, C)	$\rightarrow$	S_0
D	$\rightarrow$	S_1
(E, F, H)	$\rightarrow$	S_2
G	$\rightarrow$	S_3

This set of maximum compatibles covers all of the original states resulting in the reduced flow table as shown in the figure



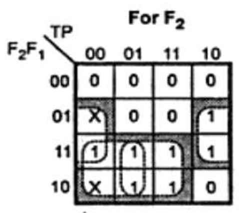
Present state	Next state, Output Q for T P Inputs			
	00	01	11	10
S_0	$\textcircled{S_0}$, 0	$\textcircled{S_0}$, 0	S_1 , –	$\textcircled{S_0}$, 0
S_1	– , –	S_0 , –	$\textcircled{S_1}$, 0	S_2 , –
S_2	$\textcircled{S_2}$, 1	$\textcircled{S_2}$, 1	S_3 , –	$\textcircled{S_2}$, 1
S_3	– , –	S_2 , –	$\textcircled{S_3}$, 1	S_0 , –

By making state assignment as S0=00, S1=01, S2=11 and S3=10 we can avoid all the races.

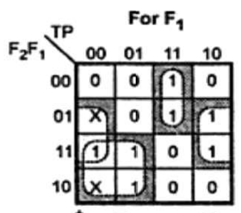
Substituting state assigned values to state we get transition table as shown in figure

Present state		Next state, Output Q for T P inputs			
		Next state			
F_2	F_1	00	01	11	10
0	0	00,0	00,0	01,–	00,0
0	1	–,–	00,–	01,0	11,–
1	1	11,1	11,1	10,–	11,1
1	0	–,–	11,–	10,1	00,–

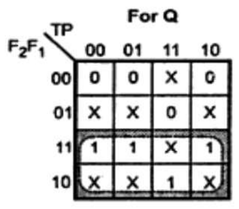
K-map simplification



$F_2^+ = F_1\bar{P} + F_2\bar{T} + F_2P$



$F_1^+ = \bar{F}_2TP + F_1\bar{P} + F_2\bar{T}$



$Q = F_2$

Design of fundamental mode sequential circuits

The steps involved in designing of asynchronous sequential circuit.

- Construction of a primitive flow table from the problem statement. An intermediate step may include the development of a state diagram.
- Primitive flow table is reduced by eliminating redundant states by using state reduction technique.
- State assignment is made.
- The primitive flow table is realized using appropriate logic elements.

Race Free State Assignment

Q7a) Explain in detail about Race Free State Assignment

(or)

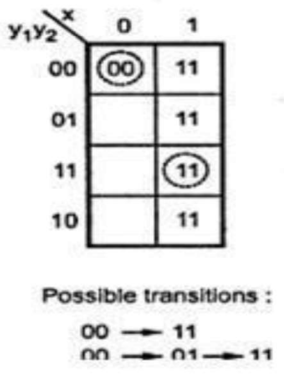
b) What is races and cycles ? Explain its types?

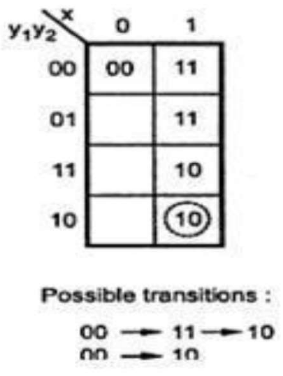
The state assignment step in asynchronous circuits is essentially the same as it is for synchronous circuits, except for one difference. In synchronous circuits, the state assignments are made with the objective of circuit reduction. In asynchronous circuits, the objective of state assignment is to avoid critical races.

Races and cycles

When two or more binary state variables change their values in response to change in an input variable, race condition occurs in an asynchronous sequential circuit. In case of unequal delays, a race condition may cause the state variables to change in an unpredictable manner. For example, if there is a change in two state variables due to change in input variable such that both change from 00 to 11. In this situation the difference in delays may cause the first variable to change faster than the second resulting the state variables to change in sequence from 00 to 10 and then to 11. On the other hand, if the second variables changes faster than the first, the state variables changes from 00 to 01 and then to 11. If the state that the circuit reaches does not depend on the order in which the state variable changes, the race condition is not harmful and then it is called **Noncritical Races**

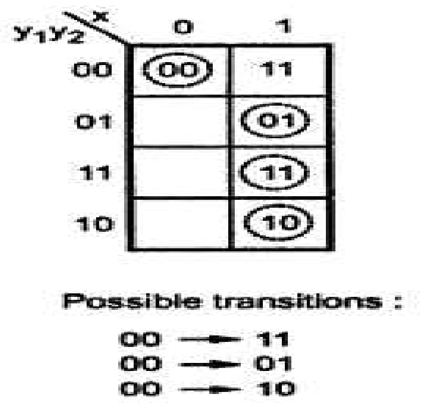
Fig. illustrates noncritical races. It shows transition tables in which X is a input variable and Y_1 Y_2 are the state variables. Consider a circuit is a stable state Y_1Y_2X =000 and there is a change in input from 0 to 1. With this change in the input there are three possibilities that the state variables may change. They can either change simultaneously from 00 to 11, or they may change in sequence from 00 to 01 and then to 11, or they may change in sequence from 00 to 10 and then to 11. In all cases final stable state is 11, which results in a noncritical race condition. In Fig. final stable state is Y_1Y_2X =101.





Critical Races

` Fig. illustrates critical race. Consider a circuit is in a stable Y_1Y_2X = 000 and there is a change in input from 0 to 1. If state variables change simultaneously, the final stable state is Y_1Y_2X = 111. If Y_2 changes to 1 before Y_1 because of unequal propagation delay, then the circuit goes to the stable state 011 and remain there. On the other hand, if Y_1 changes faster than Y_2, then the circuit goes to the stable state 101 and remain there. Hence, the race is critical because the circuit goes to different stable states depending on the order in which the state variables change.



Cycles

A cycle occurs when an asynchronous circuit makes a transition through a series of unstable states. When a state assignment is made so that it introduces cycles, care must be taken to ensure that each cycle terminates on a stable state. If a cycle does not contain a stable state, the circuit will go from one unstable state to another, until the inputs are changed. Obviously such a situation must always be avoided when designing asynchronous circuits.

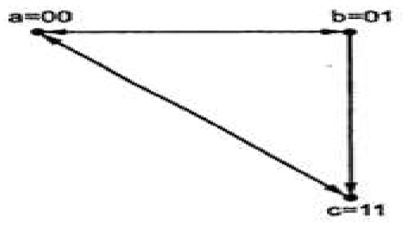
Two techniques are commonly used for making a critical-race free state assignment.

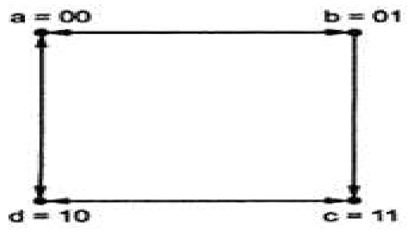
1. Shared row state assignment
2. One hot state assignment

Shared Row State Assignment

Races can be avoided by making a proper binary assignment to the state variables. Here, the state variables are assigned with binary numbers in such a way that only one state variable can change at any one time when a transition occurs. To accomplish this, it is necessary that states between which transition occur be given adjacent assignments. Two binary values are said to be adjacent if they differ in only one variable. For example, 110 and 111 are adjacent because they differ only in the third bit.

Fig. shows the transition diagram. The transition diagram shows that there is transition from state a to state b and transition from state a to state c. The state a is assigned binary value 00 and state c is assigned binary value 11. This assignment will cause a critical race during the transition from a to c because there are two changes in the binary table variables. A race free assignment can be obtained by introducing addition binary state say d with binary value 10, which is adjacent to both a and c. Fig. shows the modified transition diagram. As shown in the Fig shows that the transition from a to c will go through d. This causes the binary variables to change from 00 -> 10 - > 11 which satisfy the condition that only one binary variable changes during each state transition, thus avoiding the critical race.





This technique is called shared row state assignment because in this technique extra state, i.e. extra row is introduced in a flow table. This extra state is shared between two stable states.

HAZARDS

Q8a)Explain the types of hazards

(or)

b)Write detailed notes on hazards in combinational and sequential circuits

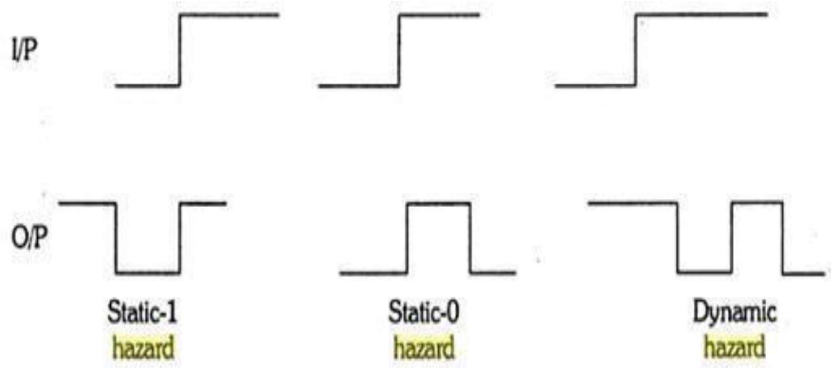
In analysis of combinational logic circuits, we ignore the circuit delay and predict only the steady state behaviour. That is, the circuit output is a function of its inputs under assumption that the inputs have been stable for a long time, relative to the delays of the elements of the circuit.

With the effect of the circuit delays, the transient behavior of the combinational circuit may vary from the predicted steady state analysis. With this, the circuit may produce a short pulse of a output, also called of ‘glitch’, at that time, when the steady state analysis predicts that there is no change in the output.
A ‘Hazard’ is a unwanted switching transient i.e. spike or glitch, that occurs due to unequal path or unequal propagation delay through a combinational circuit. There are two types of hazards, namely
(1) Static hazard
(2) Dynamic hazard. These two hazards occur in a combinational circuits,
Similarly,
(3)Essential hazard is a third type, occur in the sequential circuits.
The Fig. shows the Hazards.

Static hazard. Static hazard is a condition which results in a single momentary in correct output due to change in a input variable when the output is expected to remain in the same state. There are two types, static-1 hazard and static-0 hazard.

Static-1 hazard: If the output momentarily goes to state '0' when the output is expected to remain in state '1', as per the steady state analysis.

Static-0 hazard: If the output momentarily goes to state '1' when the output is expected to remain in state '0' as per the steady state analysis.

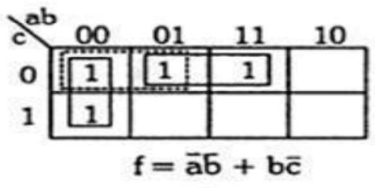


From below diagram, when abc = 000, the output f =1 due to 1 output from the AND Gate (A1). Now, if the input b is changes from 0 to 1, causes abc = 000 to 010, the output f should be in 1 due to 1 at the output of Lower AND gate (A2). Due to change in the value of input variable, the output f is supported to remain in HIGH state. But, due to unequal propagation delay, if the

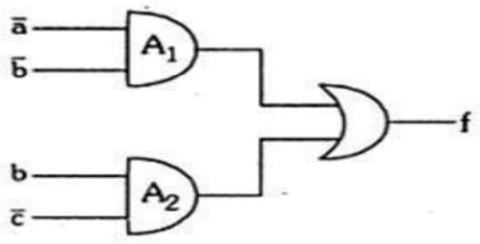
A_1 gate output changes to 0 shortly before A_2 gate output becomes _1', then during this short period, the output f = 0 momentarily. This situation is called static hazard.

A static hazard can be removed by covering the adjacent cells with a redundant grouping that overloops both grouping. For the above circuit, static hazard can be eliminated by covering adjacent cells (000, 010 shown in dotted lines). This gives a term $\bar{a}c$, which overloops both ab and bc groupings.

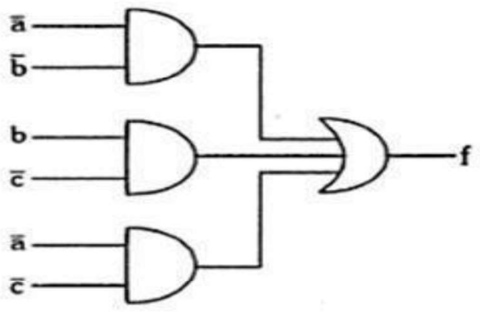
Now, if the input (abc) changes 000 to 010, the output will remain at _1' state, because the output of new AND gate is high for both input combinations. Here, there is a change in propagation delay, since the lower AND gate output is 1, the output of the circuit is 1 only, without any hazard.



(a) K-map for
$f = \Sigma(0, 1, 2, 6)$



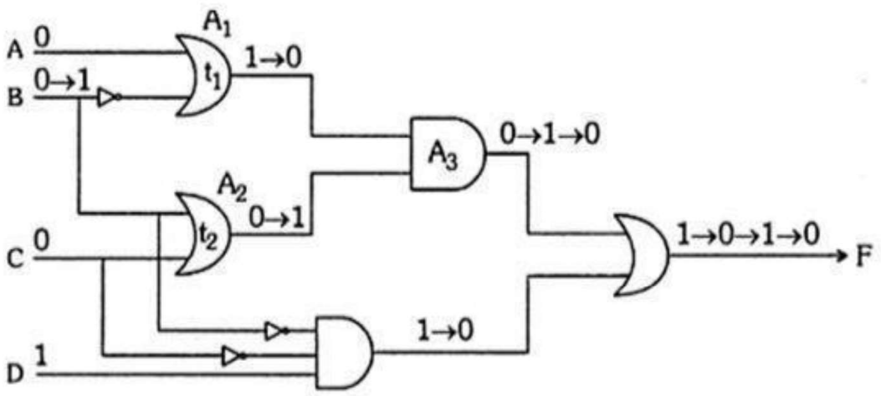
(b) Logic diagram
with static hazard



(c) Logic diagram
without static hazard

Dynamic hazard.

When the output is supported to change from 0 to 1 or from 1 to 0, the circuit may go three or more transients and produce more than one spike or glitch. Such multiple glitch situation is called of 'dynamic hazards'. Dynamic hazards can be eliminated by some method of used in static hazards. Here multiple output transitions can occur b if there are multiple paths with different delays from the changing inputs to the changing output.



That has three different paths from input A to the output F. One of the path goes through OR(A_1) gate with delay —t_1(slow) and other path is through OR gate (A2) with propagation delay 't_2 '(slower). If the input is (ABCD) = 0001 then the output is 1. If the change in the input (i.e. B is from 0 to 1), then the input is 0101, the output AND (A_3) gate is changes from 0 to 1 and then, 0, because the difference in propagation delay of two OR gate (A_1 and A_2). Finally the output is changed from 1 to 0, with the glitch change.

Note: (1)Dynamic hazards can be eliminated properly by designing a two-level AND-OR (or) OR-AND logic circuits.

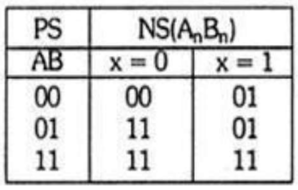
(2)The variables and its complements are not connected to the some first-level gate.

Essential Hazards:

The essential hazard is a type of hazard that exists only in asynchronous sequential circuits with two or more feedback. Generally, it occur in toggling type circuits. It is an operational error generally caused by an excessive delay to a feedback variable in response to an input change, ledge a transition into an unwanted state. This type of hazard cannot be eliminated by adding any redundant gates, but they can be eliminated by adjusting the amount of delay in the affected path. So, the core must be taken while designing the feedback path delay, so that the delay is long enough compared to the delay of the other signals that originate from the input terminals.

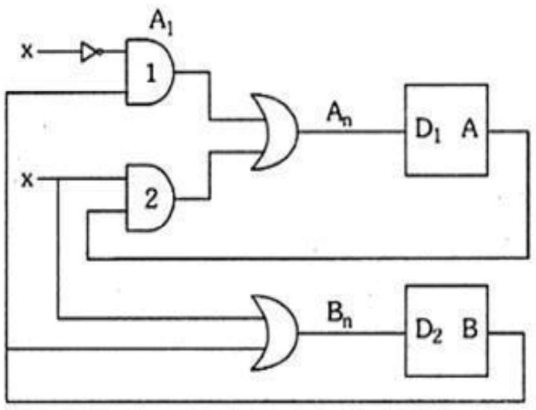
Elimination of Essential Hazards.

Consider the figure PS (present state) and NS(next state) table and its corresponding sequential circuit with DFF.



PS	NS(A_nB_n)	
AB	x = 0	x = 1
00	00	01
01	11	01
11	11	11

(a)

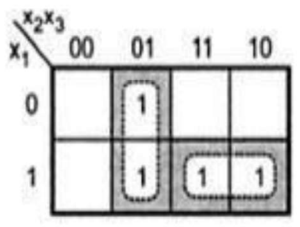


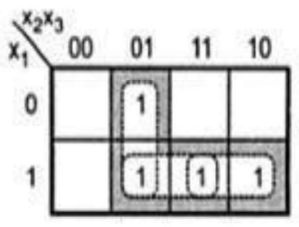
(b)

In the above circuit, the A_n depend on **x** and B_n depends on x. Assume, the delay associated with NOT gate is very large compared to the combined propagation delay of FFS and Gates. If the PS of the circuit is 00, where x changes from ‘0 to 1’, then B_n becomes 1 and there by A becomes 1. This B is feedback to the input of the flight FF through AND gate 1. If the change in input is not propagated through NOT gate, then An becomes 1 and there by A also becomes 1. Hence the NS = A_nB_n = 11 instead of 01 of given in the table. This is called of ‘Essential Hazard’, which occuring due to the result of the input change and B is changing faster. This essential hazard can be eliminated by (1) Reducing the delay of NOT gate or (2) Increasing the delay in the feedback path of B to AND gate 1.

Eliminating a Hazard

The hazard exists because of the change of input results in a different product terms covering two midterms or different sum terms covering two maxterms. Whenever the circuit move from one product term to another or move one sum term to another, there is a possibility of a momentary interval when neither term is equal to 1, giving rise to an undesirable 0 output. Hazards can be eliminated by enclosing two minterms or maxterms in question. For Example, it the circuit has minterms $X_1X_2 + X_2X_3$, then these two minterms must be enclosed by introducing another minterm X_1X_3.

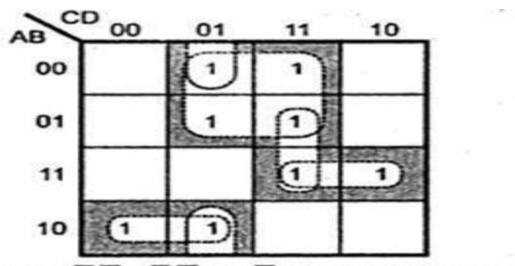

(a) $Y = x_1x_2 + \bar{x}_2x_3$


(b) $Y = x_1x_2 + \bar{x}_2x_3 + x_1x_3$

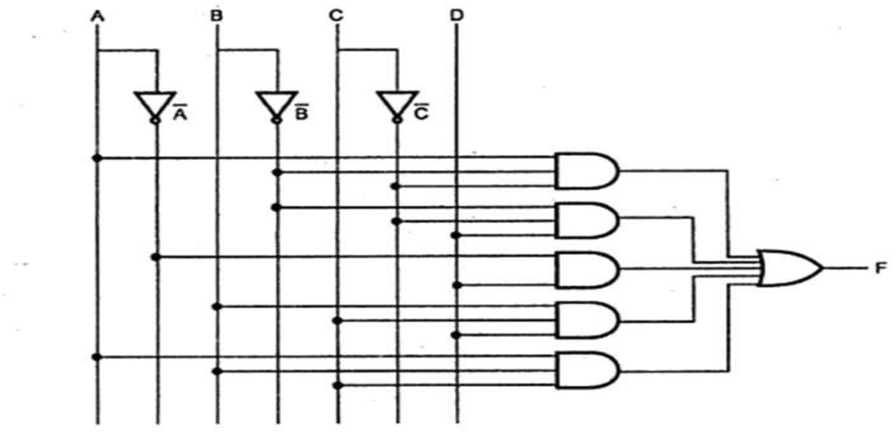
Q9a) Implement the switching function F=∑(1,3,5,7,8,9,14,15) by a static hazard free two level AND-OR gate network.

(or)

b)Implement the boolean function F=∑(1,3,5,7,8,9,14,15) by a static hazard free circuits


$F = A\bar{B}\bar{C} + \bar{B}\bar{C}D + \bar{A}D + BCD + ABC$

AND-OR Network



2 MARKS

1. **What is fundamental mode sequential circuit?**
   (or)
   Define fundamental mode sequential circuit
   a. input variables changes if the circuit is stable
   b. inputs are levels, not pulses
   c. only one input can change at a given time

2. **What is pulse mode circuit?**
   (or)
   Define pulse mode circuit
   a. inputs are pulses
   b. width of pulses are long for circuit to respond to the input
   c. pulse width must not be so long that it is still present after the new state is reached

3. **What is non critical race?**
   (or)
   Define non critical race
   final stable state does not depend on the order in which the state variable changes
   race condition is not harmful

4. **What is critical race?**
   (or)
   Define critical race
   final stable state depends on the order in which the state variable change
   race condition is harmful