Unit – VI

Objectives:

- To understand and implement finite state machines

Syllabus:

Finite State Machines: Analysis of clocked sequential circuits, state diagrams, state tables, reduction of state tables and state assignment, design procedures. Realization of circuits using various flip-flops. Meelay to Moore conversion and vice-versa.

Outcomes:

Students will be able to

- Understand Capabilities and limitations of finite state machine.
- Demonstrate the types of finite state machine.
- Reduce state tables using Partition technique.
- Realize finite state machine using sequential circuits.
- Convert Mealy to Moore FSM and vice-versa.

Learning Material

Introduction

There are two types of digital circuits, namely combinational and sequential circuits. In combinational circuits, output at any instant of time is entirely dependent on the input present at that time. A sequential circuit can be regarded as a collection of memory elements and combinational circuits.

A sequential circuit whose behaviour depends upon the sequence in which the inputs areapplied, is called Asynchronous Sequential Circuit. In these circuits, outputs are affectedwhenever a change in inputs is detected.

A **synchronous sequential** circuit may be defined as a sequential circuit, whose state can be affected only at discrete instants of time. The synchronization is achieved by using a timing device, termed as System Clock Generator, which generates a periodic train of clock pulses. All the state variables in sequential circuits are binary in nature. Therefore, total possible states for the sequential circuit having state variables '*n*' is *2n*. Even for larger values of '*n*', the number of possible state is finite. Therefore, sequential circuits are referred to as finite state machines (FSM).

General Model of FSM

A typical sequential system is composed of

- Inputs
- Internal state of the system stored in the memory elements
- Outputs
- Next state decoder
- Output decoder

The model for a general sequential circuit is shown in Fig. 6.1. The current state/present state of the circuit is stored in the memory element. The memory can be any device capable of storing information to specify the state of the system. Both the next state and output are functions of the input and current state Next state = G (Input, current state), Output = F (Input, current state).

The next state of the system is determined by the present state (current state) and by the inputs. The function of the next state decoder is to decode the external inputs and the current state of the system (stored in memory) and to generate at its output a code called **next state variable**.When

the next state variables stored in the memory, they became the present state variables. This process is called a state change.

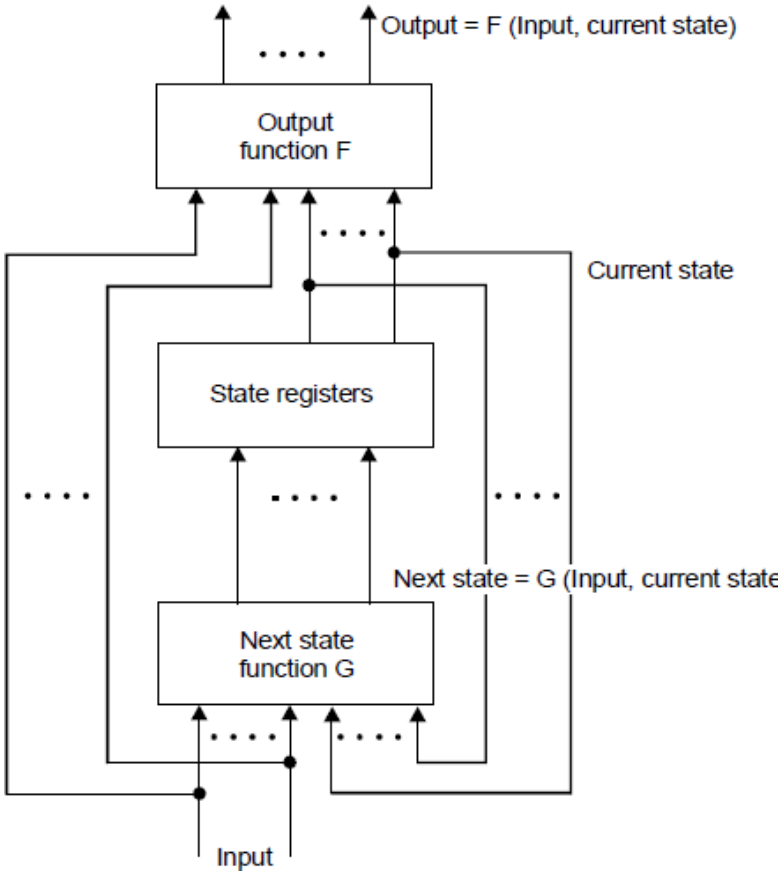


Fig 6.1. General model of FSM

CAPABILITIES AND LIMITATIONS OF FINITE STATE MACHINES

- With ail n-state machine, we can generate a periodic sequence of u states or less than n states,
- Certain infinite sequences cannot be produced by a finite state machine.
- The finite state machine has limited memory and due to limited memory it cannot produce certain outputs.

CLASSIFICATION OF FSM (MEALY & MOORE MODELS)

FSM are classified into two types

(i) Mealy FSM

(ii) Moore FSM

The different models can be derived from the general model of FSM shown in Fig. 6.1.

The Mealy FSM named after G.H. Mealy, a pioneer in this field. The basic property of Mealy FSM is that the output is a function of the present input conditions and the current state of FSM.

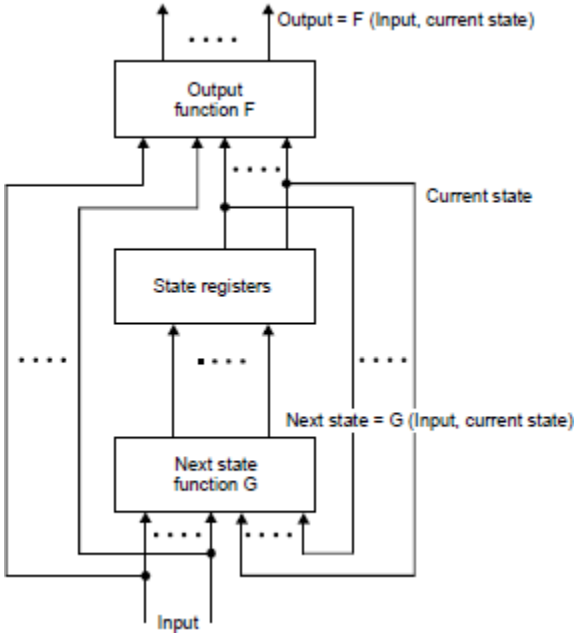


Fig.6.2. Mealy FSM

The Moore FSM, named after another pioneer E.F. Moore. In Moore FSMs, the output is associated only with the current state (present state). The block diagram is shown in Fig. 6.3

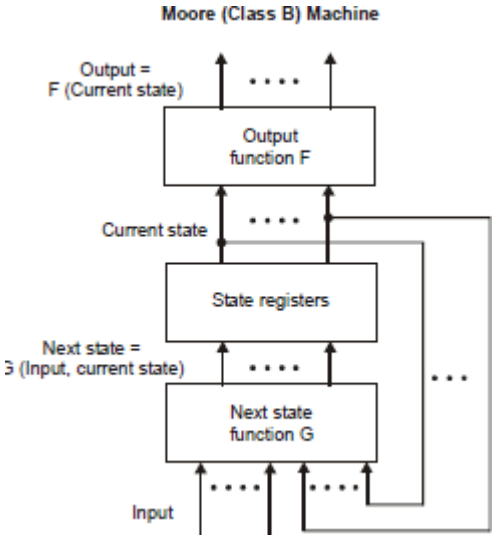


Fig.6.3. Moore FSM

Comparison of Mealy and Moore machine

Moore Machine	Mealy Machine
Its output is a function of present state only y(t)=g{x(t)}	Its output is a function of present state as well as present state input y(t)=g{x1(t),x2(t)}
Input changes do not affect the output	Input changes may affect the output of the circuit
It requires more number of states for implementing same function	It requires less number of states for implementing same function

Applications of FSM:

FSM can be designed to control processes of *digital* nature (discrete inerrable with but different from the PID controllers which are used to control processes of *analog* nature (continuous in both time and variable values) described by differential equations. Fig. 6.4 shows FSM used as a controller.

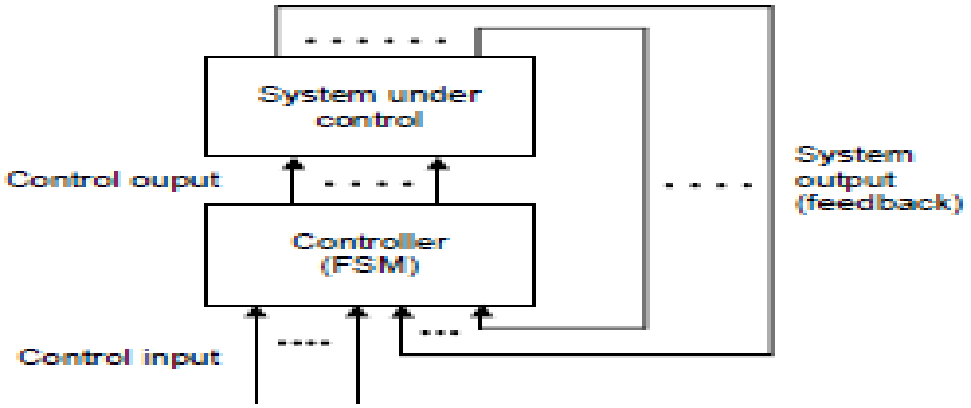


Fig.6.4. Finite State FSM (FSM) used as a Controller

State Tables and State Diagram

In the general model for sequential circuits, the effect of all previous inputs on the outputs is represented by a state of the circuit. Thus, the output of the circuit at any time depends upon its current state and the input. These also determine the next state of the circuit. The relationship that exists among the inputs, outputs, present states and next states can be specified by either the **state table** or the **state diagram**.

State Diagram

In addition to graphical symbols, tables or equations, flip-flops can also be represented

graphically by a state diagram. In this diagram, a state is represented by a circle, and the transition between states is indicated by directed lines (or arcs) connecting the circles. An example of a state diagram is shown in Figure 6.5 below.

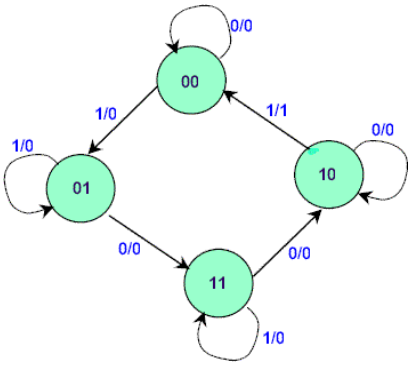


Figure 6.5: State Diagram

The binary number inside each circle identifies the state the circle represents. The directed lines are labelled with two binary numbers separated by a slash (/). The input value that causes the state transition is labelled first. The number after the slash symbol / gives the value of the output. For example, the directed line from state 00 to 01 is labelled 1/0, meaning that, if the sequential circuit is in a present state and the input is 1, then the next state is 01 and the output is 0. If it is in a present state 00 and the input is 0, it will remain in that state. A directed line connecting a circle with itself indicates that no change of state occurs. The state diagram provides exactly the same information as the state table and is obtained directly from the state table.

State Table

The state table representation of a sequential circuit consists of three sections labelled *present state*, *next state* and *output*. The present state designates the state of flip-flops before the occurrence of a clock pulse. The next state shows the states of flip-flops after the clock pulse, and the output section lists the value of the output variables during the present state.

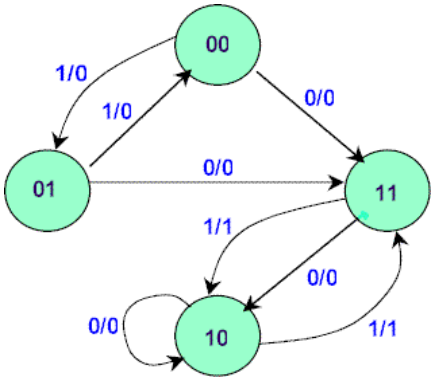


Figure 6.7 State Diagram

Table 1. State table for the state diagram in figure 6.7

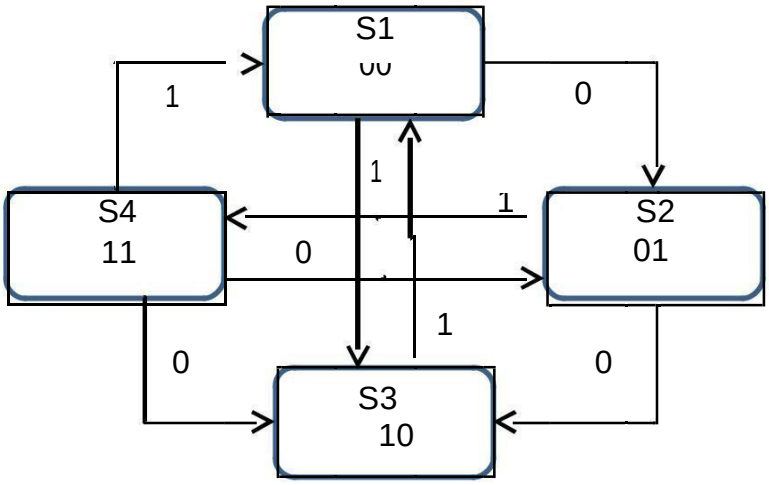
Present State Q1Q2	Next State x = 0	Next State x = 1	Output x = 0	Output x = 1
0 0	1 1	0 1	0	0
0 1	1 1	0 0	0	0
1 0	1 0	1 1	0	1
1 1	1 0	1 0	0	1

State Assignment

Assigning codes to different states is known as state assignment. State assignment is also known as state encoding. The different types of state assignment are given below.

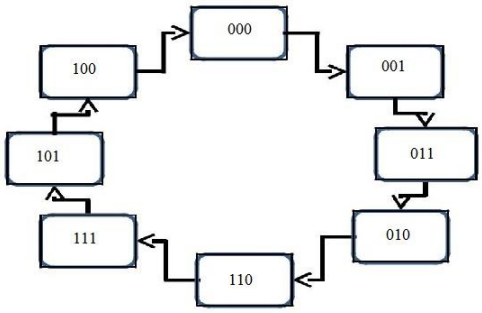
Binary code:

The Binary Method for assigning states is a common method to use. It counts up in binary starting from 0 and going up assigning each state the next number. It will use Log_2 bits to assign the states. Binary is a common method to use because it is easy to think of what the bit code will be for each state. Students often learn binary coding as their first encoding sequence because of this reason. However, it is very inefficient. There are a minimum number of bits used to encode the machine, but the encoding equations derived for each bit are often large and complex.



Gray Code:

Gray code was named after Frank Gray. In gray code, each successive state differs from the previous state by only one bit. With only one bit changing from each state to the next, power consumption is reduced from binary code where multiple states can change at the same time. In addition, This allows for a minimum number of bits used and an also a small equation for each bit. However, the encoding only works if the machine transfers from one state to the next in order. If the states do not travel in order, then the gray code does not work very well.

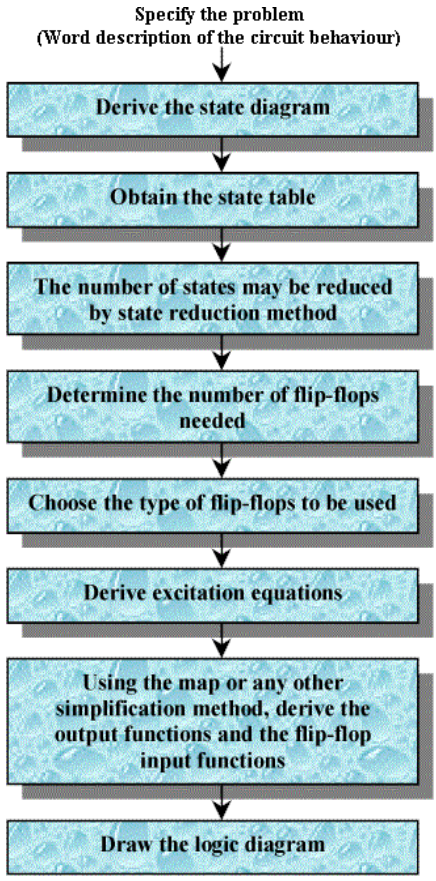


Design of Sequential Circuits

The design of a synchronous sequential circuit starts from a set of specifications and culminates in a logic diagram or a list of Boolean functions from which a logic diagram can be obtained. In contrast to a combinational logic, which is fully specified by a truth table, a sequential circuit requires a state table for its specification. The first step in the design of sequential circuits is to obtain a state table or an equivalence representation, such as a state diagram.

A synchronous sequential circuit is made up of flip-flops and combinational gates. The design of the circuit consists of choosing the flip-flops and then finding the combinational structure which, together with the flip-flops, produces a circuit that fulfils the required specifications. The number of flip-flops is determined from the number of states needed in the circuit.

The recommended steps for the design of sequential circuits are set out below.



Design of Sequential Circuits

Example 1:

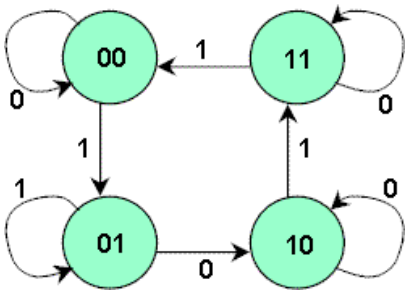


Figure 6.7. State diagram

From the state diagram, we can generate the state table shown in Table 3. Note that there is no output section for this circuit. Two flip-flops are needed to represent the four states and are designated Q0Q1. The input variable is labeled x.

Table 3. State table.

Present State Q0 Q1	Next State x = 0	Next State x = 1
0 0	0 0	0 1
0 1	1 0	0 1
1 0	1 0	1 1
1 1	1 1	0 0

We shall now derive the excitation table and the combinational structure. The table is now arranged in a different form shown in Table 5, where the present state and input variables are arranged in the form of a truth table. Remember, the excitable for the JK flip-flop was derive in Table 4.

Table 4. Excitation table for JK flip-flop

Output Transitions Q ➔ Q(next)	Flip-flop inputs J K
0 ➔ 0	0 X
0 ➔ 1	1 X
1 ➔ 0	X 1
1 ➔ 1	X 0

Table 5. Excitation table of the circuit

Present State Q0 Q1	Next State Q0 Q1	Input x	Flip-flop Inputs J0K0	Flip-flop Inputs J1K1
0 0	0 0	0	0 X	0 X
0 0	0 1	1	0 X	1 X
0 1	1 0	0	1 X	X 1
0 1	0 1	1	0 X	X 0
1 0	1 0	0	X 0	0 X
1 0	1 1	1	X 0	1 X
1 1	1 1	0	X 0	X 0
1 1	0 0	1	X 1	X 1

In the first row of Table 5, we have a transition for flip-flop Q0 from 0 in the present state to 0 in the next state. In Table 4 we find that a transition of states from 0 to 0 requires that input J = 0 and input K = X. So 0 and X are copied in the first row under J0 and K0 respectively. Since the first row also shows a transition for the flip-flop Q1 from 0 in the present state to 0 in the next state, 0 and X are copied in the first row under J1 and K1. This process is continued for each row of the table and for each flip-flop, with the input conditions as specified in Table 4.

The simplified Boolean functions for the combinational circuit can now be derived. The input variables are Q0, Q1, and x; the output are the variables J0, K0, J1 and K1. The information from the truth table is plotted on the Karnaugh maps shown in Figure 6.8.

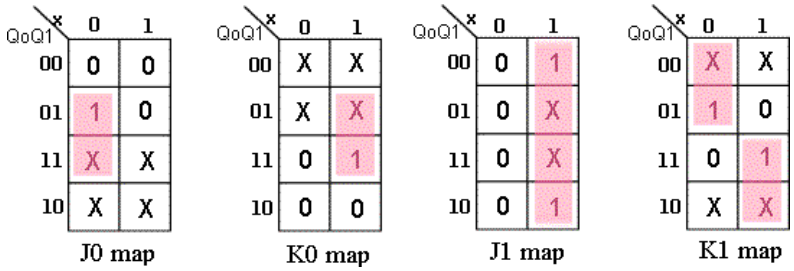


Figure 6.8 Karnaugh Maps

The flip-flop input functions are derived:

J0 = Q1*x' **K0 = Q1*x**

J1 = x **K1 = Q0'*x' + Q0*x = Q0Θ x**

Note: the symbol Θ is exclusive-NOR.
The logic diagram is drawn in Figure 6.9.

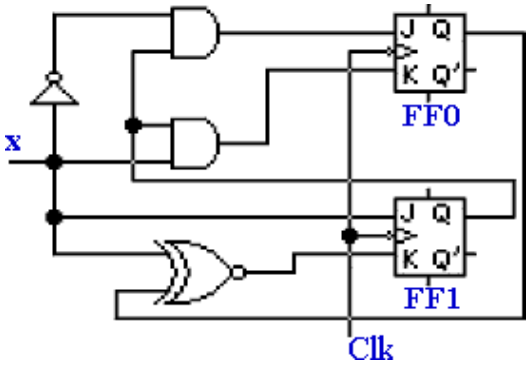


Figure 6.9. Logic diagram of the sequential circuit.

Example2 : Design a sequential circuit whose state tables are specified in Table 6, using D flip-flops.

Table 6. State table of a sequential circuit.

Present State Q0 Q1	Next State x = 0	Next State x = 1	Output x = 0	Output x = 1
0 0	0 0	0 1	0	0
0 1	0 0	1 0	0	0
1 0	1 1	1 0	0	0
1 1	0 0	0 1	0	1

Table 7. Excitation table for a D flip-flop.

Output Transitions Q→Q(next)	Flip-flop inputs D
0 → 0	0
0 → 1	1
1 → 0	0
1 → 1	1

Next step is to derive the excitation table for the design circuit, which is shown in Table 8. The output of the circuit is labeled Z.

Table 8. Excitation table

Present State Q0 Q1	Next State Q0 Q1	Input x	Flip-flop Inputs D0	D1	Output Z
0 0	0 0	0	0	0	0
0 0	0 1	1	0	1	0
0 1	0 0	0	0	0	0
0 1	1 0	1	1	0	0
1 0	1 1	0	1	1	0
1 0	1 0	1	1	0	0
1 1	0 0	0	0	0	0
1 1	0 1	1	0	1	1

Now plot the flip-flop inputs and output functions on the Karnaugh map to derive the Boolean expressions, which is shown in Figure 6.10.

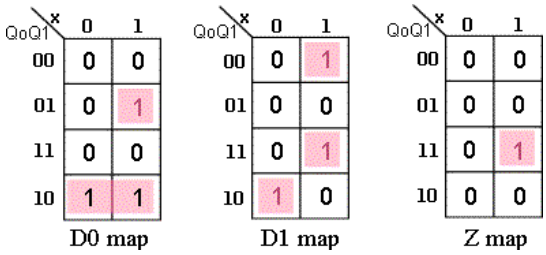


Figure 6.10. Karnaugh maps

The simplified Boolean expressions are:

D0 = Q0*Q1' + Q0'*Q1*x
D1 = Q0'*Q1'*x + Q0*Q1*x + Q0*Q1'*x'
Z = Q0*Q1*x

Finally, the logic gates diagram is shown in figure 6.11.

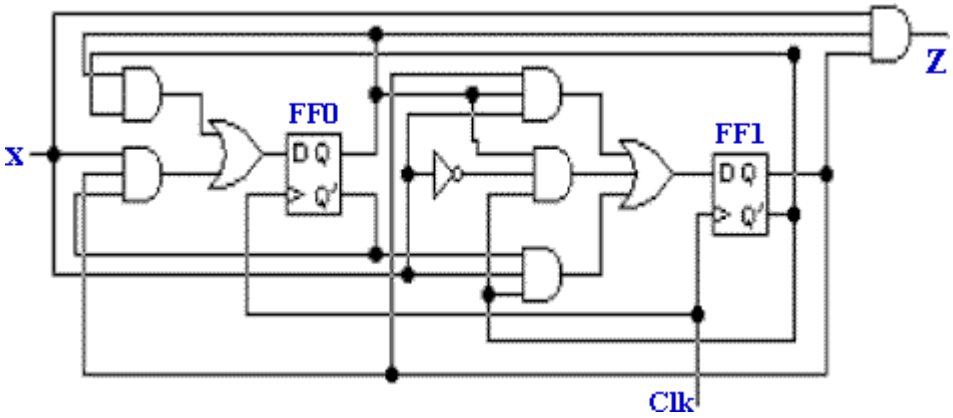


Figure 6.11. Logic diagram of the sequential circuit.

State Reduction

The three main methods of state reduction include: row matching, implication charts, and successive partitioning. Row matching, which is the easiest of the three, works well for state transition tables which have an obvious next state and output equivalences for each of the present states? This method will generally not give the most simplified state machine available, but its ease of use and consistently fair results is a good reason to pursue the method. The implication chart uses a graphical grid to help find any implications or equivalences and is a great systematic approach to reducing state machines.

Successive partitioning is almost a cross between row matching and implication chart where both a graphical table and equivalent matching is used. Each of these methods will, in most cases, reduce a state machine into a smaller number of states. Keep in mind that one method may result in a simpler state machine than another. Experience will help in understanding and determining the best method to use in any particular situation.

Partitioning Method

Partitioning Method has the following three steps:

- The first partition P1 is formed by placing two or more states in the same block of P1. If and only if their output is identical for each input. For the example on the next page we have P1 = (ABC)(DE), and hence the states within each block are 1-equivalent. This step guarantees that each block in P1 satisfies condition 1.

- Successive partitions P_K , K = 2, 3, 4..., l, are derived by placing two or more states in the same block of P_K if and only if for each input value their next states all lie in a single block of P_{K-1}. This iterative procedure is suggested by condition 2 for equivalent states.

- When $P_{K+1} = P_K$, which means a partition repeats, the states in each block of P_K that are K-equivalent are (K+1)-equivalent, (K+2)-equivalent and so on. The last partition P_K is said to be an equivalence partition. Condition 2 for equivalent states is now satisfied by P_K.

Example:

State	Next State	
	X=0	X=1
A	C/1	B/0
B	C/1	E/0
C	B/1	E/0
D	D/0	B/1
E	E/0	A/1

State Partitioning Table

Partition Next States	Partition Blocks	Action
P_0	(ABCDE)	
Output for X = 0	111 00	Separate (ABC) and (DE)
Output for X = 1	000 11	Separate (ABC) and (DE)
P_1	(ABC) (DE)	
NS for X = 0	C C B D E	Separate (A) and (BC)
NS for X = 1	B E E B A	
P_2	(A) (BC) (DE)	
NS for X = 0	C C B D E	
NS for X = 1	B E E B A	Separate (D) and (E)
P_3	(A) (BC) (D) (E)	
NS for X = 0	C C B D E	
NS for X = 1	B E E B A	
$P_4 = P_3$	(A) (BC) (D) (E)	

Partition P2 is obtained by examining each block of P1.

- In the first block of P1 the next states for A, B and C with X = 0 all lie in the same block of P1. For X = 1 the next state of A lies in a different block of P1 than the next states of B and C. Therefore the block (ABC) contained in P1 is split into the blocks (A) (BC) in P2.
- In the second block of P1 the next states for D and E lie in the same block of P1 for both X values. Hence D and E will remain in the same block of P2.
- P2 = (A) (BC) (DE), the states within each block are 2-equivalent.

Partition P3 is obtained by examining each block of P2.

- The next states of B and C lie in the same block of P2 for each input and hence the block (BC) remains in P3.
- The next states for D and E with x = 1 now lie in different blocks of P2 and hence these two states must be separated into different blocks of P3.
- P3 = (A) (BC) (D) (E), the states B and C are 3-equivalent.

Moore to Mealy FSM conversion

- Take a blank Mealy Machine transition table format.
- Copy all the Moore Machine transition states into this table format.
- Check the present states and their corresponding outputs in the Moore Machine state table; if for a state Q_i output is m, copy it into the output columns of the Mealy Machine state table wherever Q_i appears in the next state.
- After repeating the above process for other states, the final mealy machine transition table is obtained.

Example:

Suppose the moore machine transition table is:

Present State	Next State		Output
	a = 0	a = 1	
-> q0	q3	q1	1
q1	q0	q3	0
q2	q2	q2	0
q3	q1	q0	1

Convert this transition table into mealy machine.

First of all take the mealy machine transition table format, i.e.,

Present State	Next State			
	a = 0		a = 1	
	State	Output	State	Output

Next step is to copy Moore machine transition table states into mealy machine transition table format

Present State	Next State			
	a = 0		a = 1	
	State	Output	State	Output
-> q0	q3		q1	
q1	q0		q3	
q2	q2		q2	
q3	q1		q0	

Now in the Moore machine, the output of the q0 is 1. so make the output of q0 in the mealy machine next state column of the above table is 1. same process is repeated for q1, q2 and q3.

Present State	Next State			
	a = 0		a = 1	
	State	Output	State	Output
-> q0	q3		q1	
q1	q0	1	q3	
q2	q2		q2	
q3	q1		q0	1

After repeating the above process for q1,q2 and q3 states, the final mealy machine transition table is:

Present State	Next State			
	a = 0		a = 1	
	State	Output	State	Output
-> q0	q3	1	q1	0
q1	q0	1	q3	1
q2	q2	0	q2	0
q3	q1	0	q0	1

Mealy to Moore FSM conversion

1. Determine the number of different output associated with qi in the next state column.
2. Split qi into different states according to different output associated with it. for ex. suppose in the next state column of the above sample transition table of mealy machine, the output associated with q1 is "0" in the first next state column and "1" in the second next state column. so we split q1 into q10 and q11 states. similarly check others and split them.

Example:

Suppose the moore machine transition table is:

Present State	Next State			
	a = 0		a = 1	
	State	Output	State	Output
-> q0	q3	0	q1	1
q1	q0	1	q3	0
q2	q2	1	q2	0
q3	q1	0	q0	1

Convert this transition table into mealy machine.

Solution:

After applying the conversion steps, we get two states (q1 and q2) that are associated with different outputs (0 and 1). so we split both states into q10 , q11 and q20, q21.
Now the transition table becomes

Present State	Next State			
	a = 0		a = 1	
	State	Output	State	Output
-> q0	q3	0	q11	1
q10	q0	1	q3	0
q11	q0	1	q3	0
q20	q21	1	q20	0
q21	q21	1	q20	0
q3	q10	0	q0	1

- Whole row of q1 is copied to q10 , q11 and whole row of q2 is copied to q20 and q21 of the sample transition table of mealy machine.
- The outputs of the next state columns of q1 and q2 are depend on the previous output. For ex. in the first row, q1 becomes q11 because the out of q1 is 1. in the fourth row, q2 becomes q21 because the output of the q2 is 1. and in the subsequent column q2 becomes q20 because the output of q2 in that column was 0. and so on.

- Now in moore machine format, we copied all the states and common output because in moore machine, the outputs of the next state are common.

Present State	Next State		Output
	a = 0	a = 1	
-> q0	q3	q11	1
q10	q0	q3	0
q11	q0	q3	1
q20	q21	q20	0
q21	q21	q20	1
q3	q10	q0	0

This table is Moore machine table corresponding to the sample mealy machine.

Assignment-Cum-Tutorial Questions

A. Questions testing the remembering / understanding level of students

I) Objective Questions

1. The sequential machine is another name of_____.
2. The sequential circuit in which the output depends only on the present state of the flip-flop is called a_____circuit.
3. The sequential circuit in which the output depends only on the present state of the flip-flop as well as on the present input is called a_____circuit.
4. State diagram can also be called as_____.
5. The next state part of the state table is called the_____table.
6. The process of assigning the states of a physical device tothe states of a sequential machine is known as_____.
7. _____states are states whose functions can be accomplished by other states.
8. A_____FSM has more number of states.
9. If the outputs of two states are different after P-state transitions they are said to be----------.

II) Descriptive Questions

1. What are the Moore and Mealy machines? Compare them.
2. What are the capabilities and limitations of Finite state machines?
3. What does the word state diagram represent and explain with example?
4. Compare state diagram and state table.
4. Explain the reduction of state tables using Partition technique
5. Give various types of state assignment
6. Explain the steps involved in designing FSMs
7. Describe the process of converting Moore to Mealy FSM with an example.
8. Discuss the steps involved to convert a Mealy FSM to Moore FSM.

B. Question testing the ability of students in applying the concepts.

I) Objective Questions

1) In a sequential circuit design, state reduction is done for designing the circuit with
A. a minimum number of flip flops B. a minimum number of gates
C. a minimum number of gates and memory elements D. none of the above

2) A sequential circuit with 10 states will have
A. 10 flip flops B. 5 flip flops C. 4 flip flops D. 0 flip flops

3) In general, a sequential logic circuit consists of
A. only flip flops B. only gates
C. flip flops and combinational logic circuits D. only combinational logic circuits

4). Reduction of flip-flops in a sequential circuit is referred to as
A.Reduction B. State reduction C. Next state D. Both a and b

5). M flip-flops produces
A. 2^m-1 states B. 2^1 states C. 2^m+1 states D. 2^m states

6) Two states are said to be equal if they have exactly same
A. Inputs B. Next state C. Output D. Both a and b

7). A State which has no outgoing arcs is called a_____state.

8) _____outputs provide additional flexibility in state reduction.

9) The design of a clocked sequential requires
A. the state reduction B. the state assignment
C. the design of the next state decoder D. all of the above

II) Descriptive Questions

1. A clocked sequential circuit with single input x and single output z produces an output z=1 whenever the input x complete the sequence 1011 and overlapping is allowed.
   (a) Obtain the state diagram
   (b) Obtain its minimum state table and design the circuit with D flip flops.

2. A clocked sequential circuit with single input x and single output z produces an output z=1 whenever the input x complete the sequence 1011 and overlapping not allowed.
   (c) Obtain the state diagram
   (d) Obtain its minimum state table and design the circuit with D flip flops.

3. Realize the 4-bit serial binary adder using D flip flops.

4. For the state tables of the machines given below, find the equivalence partition and corresponding reduced machine in standard form

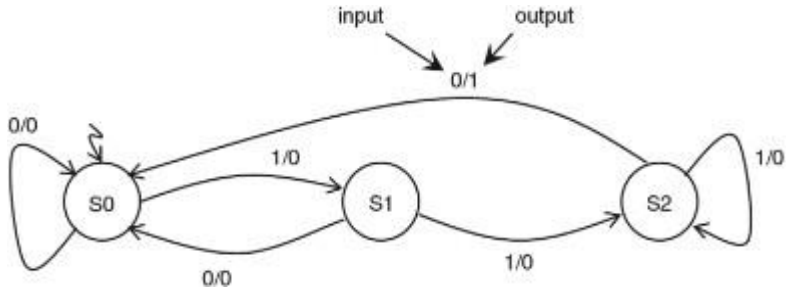
PS	NS, Z	
	X = 0	X = 1
A	D, 0	H, 1
B	F, 1	C, 1
C	D, 0	F, 1
D	C, 0	E, 1
E	C, 1	D, 1
F	D, 1	D, 1

PS	NS, Z	
	X = 0	X = 1
A	B, 1	H, 1
B	F, 1	D, 1
C	D, 0	E, 1
D	C, 1	F, 1
E	D, 1	C, 1
F	C, 1	C, 1
G	C, 1	D, 1
H	C, 0	A, 1

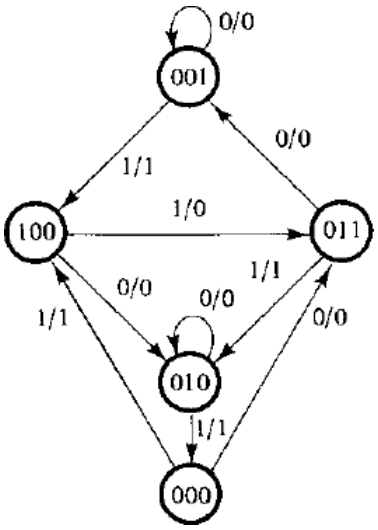
5. Design a sequential circuit with minimum hardware to produce the output when the input data stream contains 101 pattern. (consider overlapping pattern is not allowed)

6. Design a sequential circuit with minimum hardware to produce the output when the input data stream contains 1010 pattern. (consider overlapping pattern is allowed)

7. Convert the following Mealy FSM to Moore FSM.



8. For the state diagram given below obtain the following:
   (Assume your own inputs and outputs according to the state diagram)
   i) State table
   ii) Next state and output functions
   iii) Sequential circuit using logic gates and T-flip-flops.



C. Questions testing the analyzing / evaluating ability of students

1. Design a finite state machine (FSM) for a counter that counts through the 3-bit prime numbers downwards. Assume the counter starts with initial prime value set to 010 as its first 3 bit prime number. You need to provide the state transition table and the state transition diagram. Assume that the state is stored in three D-FFs.
2. Design a 2-input 2-output synchronous sequential circuit which produces an output Z=1, whenever any of the following input sequence-1101,1011, or 1001- occurs. The circuit resets to the initial state after a 1 output is generated.

3) Construct Moore machine whose output is 1 if the last five inputs were 11010 using JK flip-flops.

4) Simplify the state table given in the figure below.

PS	inputs, xy			
	xy=00	01	10	11
A	A,0	A,0	B,1	C,0
B	A,0	B,0	D,0	F,1
C	C,0	B,0	B,1	A,0
D	D,0	C,0	E,1	C,0
E	A,0	E,0	B,1	C,0
F	E,0	E,0	F,0	F,0
	NS, Z			