State Reduction & Assignment

(Lecture-31)

State Reduction & Assignment

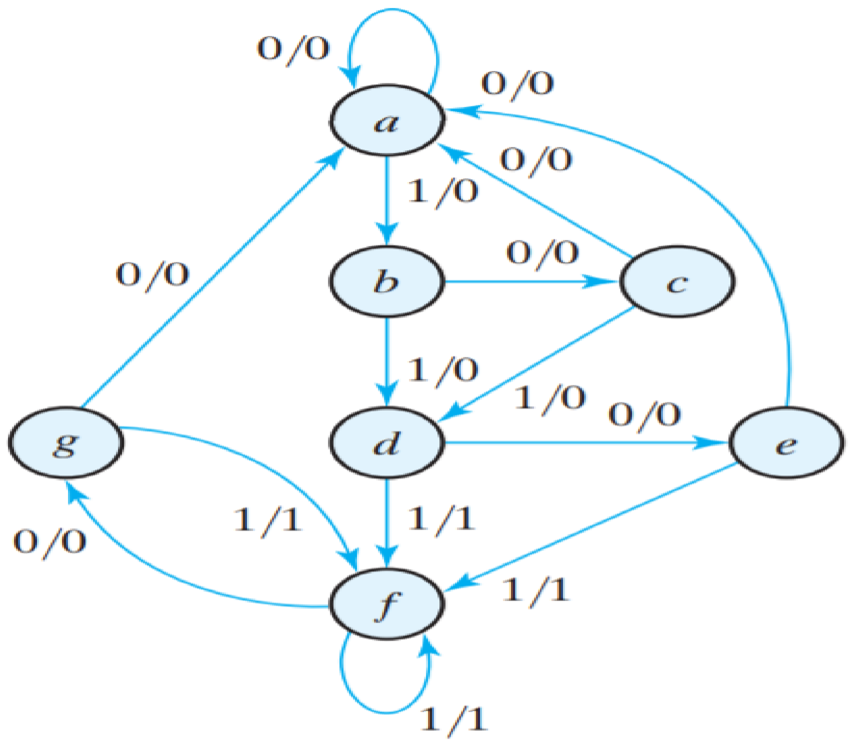
- The analysis of sequential circuits starts from a circuit diagram and culminates in a state table or diagram.
- The design (synthesis) of a sequential circuit starts from a set of specifications and culminates in a logic diagram.
- Two sequential circuits may exhibit the same input–output behaviour, but have a different number of internal states in their state diagram.
- State reduction and assignment discusses certain properties of sequential circuits that may simplify a design by reducing the number of gates and flip-flops it uses.In general, reducing the number of flipflops reduces the cost of a circuit.

State Reduction

- The reduction in the number of flip-flops in a sequential circuit is referred to as the state-reduction problem.
- State-reduction algorithms are concerned with procedures for reducing the number of states in a state table, while keeping the external input–output requirements unchanged.
- Since m flip-flops produce 2m states, a reduction in the number of states may (or may not) result in a reduction in the number of flip-flops.
- An unpredictable effect in reducing the number of flip-flops is that sometimes the equivalent circuit (with fewer flip-flops) may require more combinational gates to realize its next state and output logic.

State-reduction procedure example

- We start with a sequential circuit whose specification is given in the state diagram of Fig. below .



State-reduction procedure example

- In our example, only the input–output sequences are important; the internal states are used merely to provide the required sequences. For that reason, the states marked inside the circles are denoted by letter symbols instead of their binary values
- This is in contrast to a binary counter, wherein the binary value sequence of the states themselves is taken as the outputs.
- There are an infinite number of input sequences that may be applied to the circuit; each results in a unique output sequence.

Explanation of Example

- As an example, consider the input sequence 01010110100 starting from the initial state a . Each input of 0 or 1 produces an output of 0 or 1 and causes the circuit to go to the next state.
- From the state diagram, we obtain the output and state sequence for the given input sequence as follows: With the circuit in initial state a, an input of 0 produces an output of 0 and the circuit remains in state a . With present state a and an input of 1, the output is 0 and the next state is b . With present state b and an input of 0, the output is 0 and the next state is c . Continuing this process, we find the complete sequence to be as follows:

Explanation of Example

state	***a***	***a***	***b***	***c***	***d***	***e***	***f***	***f***	***g***	***f***	***g***	***a***
input	0	1	0	1	0	1	1	0	1	0	0	
output	0	0	0	0	0	1	1	0	1	0	0	

- In each column, we have the present state, input value, and output value.
- The next state is written on top of the next column.
- It is important to realize that in this circuit the states themselves are of secondary importance, because we are interested only in output sequences caused by input sequences.
- The problem of state reduction is to find ways of reducing the number of states in a sequential circuit without altering the input–output relationships.

Reduction of number of states

1. First, we need the state table; it is more convenient to apply procedures for state reduction with the use of a table rather than a diagram.
2. The state table of the circuit is listed in Table 5.6 and is obtained directly from the state diagram.
3. The following algorithm for the state reduction of a completely specified state table is given here without proof: “Two states are said to be equivalent if, for each member of the set of inputs, they give exactly the same output and send the circuit either to the same state or to an equivalent state.” When two states are equivalent, one of them can be removed without altering the input–output relationships.
4. Now apply this algorithm to Table 5.6 . Going through the state table, we look for two present states that go to the same next state and have the same output for both input combinations.

Reduction of number of states

5. States e and g are two such states: They both go to states a and f and have outputs of 0 and 1 for x = 0 and x = 1, respectively. Therefore, states g and e are equivalent, and one of these states can be removed.
6. The procedure of removing a state and replacing it by its equivalent is demonstrated in Table 5.7 .
7. The row with present state g is removed, and state g is replaced by state e each time it occurs in the columns headed "Next State."
8. Present state f now has next states e and f and outputs 0 and 1 for x = 0 and x = 1, respectively . The same next states and outputs appear in the row with present state d . Therefore, states f and d are equivalent, and state f can be removed and replaced by d .

Reduction of number of states

State Table

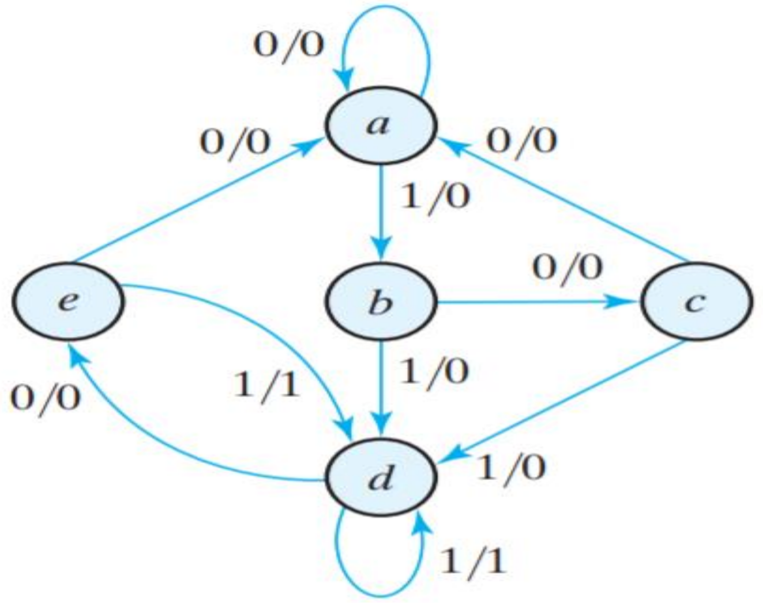
	Next State		Output	
Present State	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	*a*	*b*	0	0
b	*c*	*d*	0	0
c	*a*	*d*	0	0
d	*e*	*f*	0	1
e	*a*	*f*	0	1
f	*g*	*f*	0	1
g	*a*	*f*	0	1

Reducing the State Table

	Next State		Output	
Present State	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	*a*	*b*	0	0
b	*c*	*d*	0	0
c	*a*	*d*	0	0
d	*e*	*f*	0	1
e	*a*	*f*	0	1
f	*e*	*f*	0	1

Reduced State Table

	Next State		Output	
Present State	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	*a*	*b*	0	0
b	*c*	*d*	0	0
c	*a*	*d*	0	0
d	*e*	*d*	0	1
e	*a*	*d*	0	1



The state diagram for the reduced table consists of only five states

State Reduction

- The sequential circuit of this example was reduced from seven to five states.
- In general, reducing the number of states in a state table may result in a circuit with less equipment.
- However, the fact that a state table has been reduced to fewer states does not guarantee a saving in the number of flip-flops or the number of gates.
- In actual practice designers may skip this step because target devices are rich in resources.

State Assignment

- In order to design a sequential circuit with physical components, it is necessary to assign unique coded binary values to the states.
- For a circuit with m states, the codes must contain n bits, where $2^n \geq m$.
- For example, with three bits, it is possible to assign codes to eight states, denoted by binary numbers 000 through 111.
- If the state table is used, we must assign binary values to seven states; the remaining state is unused.
- If the reduced state table is used, only five states need binary assignment, and we are left with three unused states.

State Assignment

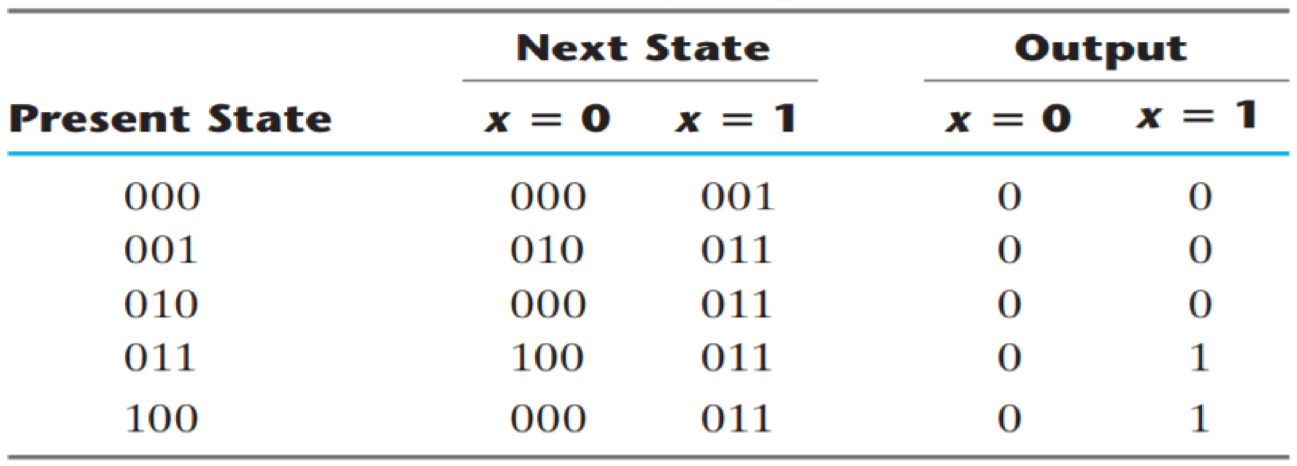
- Unused states are treated as don’t-care conditions during the design.
- Since don’t-care conditions usually help in obtaining a simpler circuit, it is more likely but not certain that the circuit with five states will require fewer combinational gates than the one with seven states.
- The simplest way to code five states is to use the first five integers in binary counting order, as shown in the first assignment of Table below .

Three Possible Binary State Assignments

State	Assignment 1, Binary	Assignment 2, Gray Code	Assignment 3, One-Hot
a	000	000	00001
b	001	001	00010
c	010	011	00100
d	011	010	01000
e	100	110	10000

State Assignment

- The reduced state table with binary assignment 1 substituted for the letter symbols of the states.



Reduced State Table with Binary Assignment 1

	Next State		Output	
Present State	$x = 0$	$x = 1$	$x = 0$	$x = 1$
000	000	001	0	0
001	010	011	0	0
010	000	011	0	0
011	100	011	0	1
100	000	011	0	1

State Assignment

- A different assignment will result in a state table with different binary values for the states.
- The binary form of the state table is used to derive the next state and output-forming combinational logic part of the sequential circuit.
- The complexity of the combinational circuit depends on the binary state assignment chosen.
- Sometimes, the name transition table is used for a state table with a binary assignment. This convention distinguishes it from a state table with symbolic names for the states.

Design Procedure

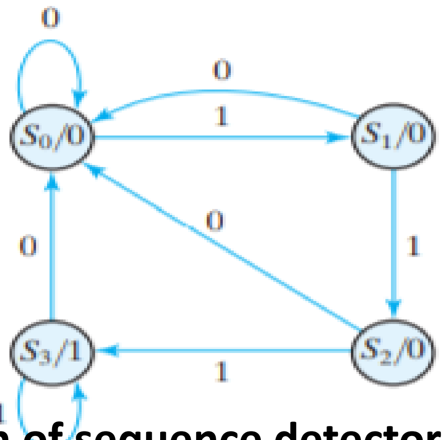
- From the word description and specifications of the desired operation, derive a state diagram for the circuit.
- Reduce the number of states if necessary.
- Assign binary values to the states.
- Obtain the binary-coded state table.
- Choose the type of flip-flops to be used.
- Derive the simplified flip-flop input equations and output equations.
- Draw the logic diagram.

Design Procedure

- The word specification of the circuit behaviour usually assumes that the reader is familiar with digital logic terminology.
- It is necessary that the designer use intuition and experience to arrive at the correct interpretation of the circuit specifications, because word descriptions may be incomplete and inexact.
- Once such a specification has been set down and the state diagram obtained, it is possible to use known synthesis procedures to complete the design.
- Although there are formal procedures for state reduction and assignment (steps 2 and 3), they are seldom used by experienced designers. Steps 4 through 7 in the design can be implemented by exact algorithms and therefore can be automated.
- The part of the design that follows a well-defined procedure is referred to as synthesis .

Design Procedure

- It is derived by starting with state S0, the reset state. If the input is 0, the circuit stays in S0, but if the input is 1, it goes to state S1 to indicate that a 1 was detected.
- If the next input is 1, the change is to state S2 to indicate the arrival of two consecutive 1's, but if the input is 0, the state goes back to S0. The third consecutive 1 sends the circuit to state S3.
- If more 1's are detected, the circuit stays in S3. Any 0 input sends the circuit back to S0. In this way, the circuit stays in S3 as long as there are three or more consecutive 1's received.
- This is a Moore model sequential circuit, since the output is 1 when the circuit is in state S3 and is 0 otherwise.



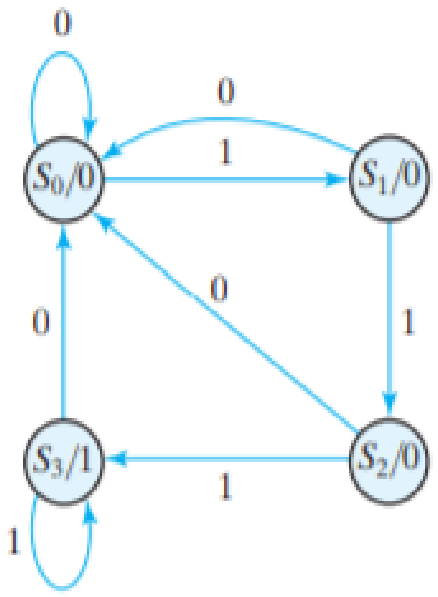
State diagram of sequence detector

Synthesis Using D Flip-Flops

- To design the circuit by hand, we need to assign binary codes to the states and list the state table.

State Table for Sequence Detector

Present State		Input	Next State		Output
A	B	x	A	B	y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	0	0
1	0	0	0	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	1	1	1



- The table is derived from the state diagram with a sequential binary assignment.

Synthesis Using D Flip-Flops

- We choose two D flip-flops to represent the four states, and we label their outputs A and B . There is one input x and one output y .
- The characteristic equation of the D flip-flop is Q(t + 1) = DQ, which means that the next-state values in the state table specify the D input condition for the flip-flop.
- The flip-flop input equations can be obtained directly from the next-state columns of A and B and expressed in sum-of-minterms form as $A(t+1) = D_A(A, B, x) = \sum(3,5,7)$

$$B(t+1) = D_B(A, B, x) = \sum(1,5,7)$$

$$y(A, B, x) = \sum(6,7)$$

where A and B are the present-state values of flip-flops A and B, x is the input, and $D_A$and D_B are the input equations. The minterms for output y are obtained from the output column in the state table

Synthesis Using D Flip-Flops

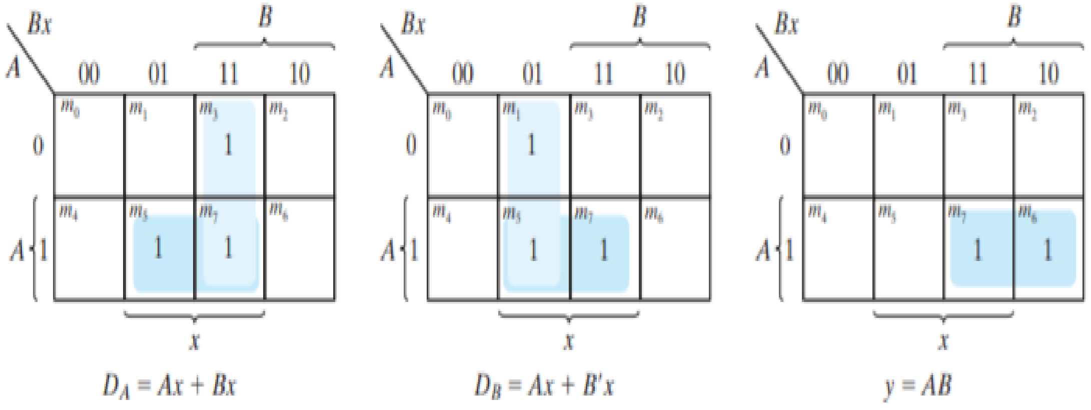
- The Boolean equations are simplified by means of the maps plotted .

The simplified equations are

$$D_A = Ax + Bx$$

$$D_B = Ax + B'x$$

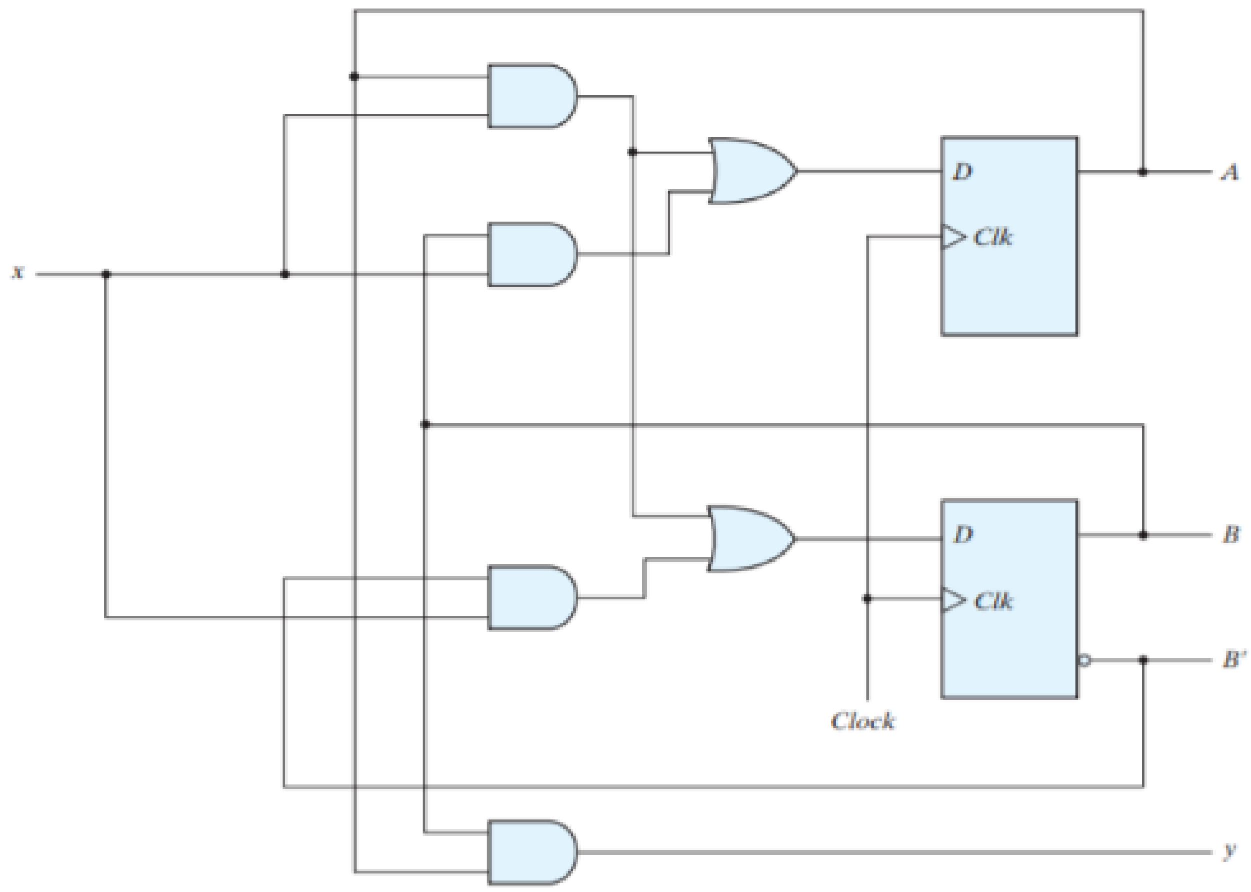
Y=AB



K maps for sequence detector

- The advantage of designing with D flip-flops is that the Boolean equations describing the inputs to the flip-flops can be obtained directly from the state table.

Synthesis Using D Flip-Flops



The schematic of the sequential circuit

Synthesis Using D Flip-Flops

- The design of a sequential circuit with flip-flops other than the D type is complicated by the fact that the input equations for the circuit must be derived indirectly from the state table.
- When D -type flip-flops are employed, the input equations are obtained directly from the next state. This is not the case for the JK and T types of flip-flops.
- In order to determine the input equations for these flip-flops, it is necessary to derive a functional relationship between the state table and the input equations.
- The flip-flop characteristic tables provide the value of the next state when the inputs and the present state are known.
- These tables are useful for analysing sequential circuits and for defining the operation of the flip-flops.

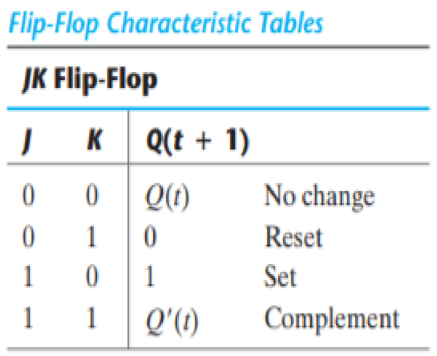
Synthesis Using D Flip-Flops

- During the design process, we usually know the transition from the present state to the next state and wish to find the flip-flop input conditions that will cause the required transition.

- For this reason, we need a table that lists the required inputs for a given change of state. Such a table is called an excitation table .

Synthesis Using D Flip-Flops

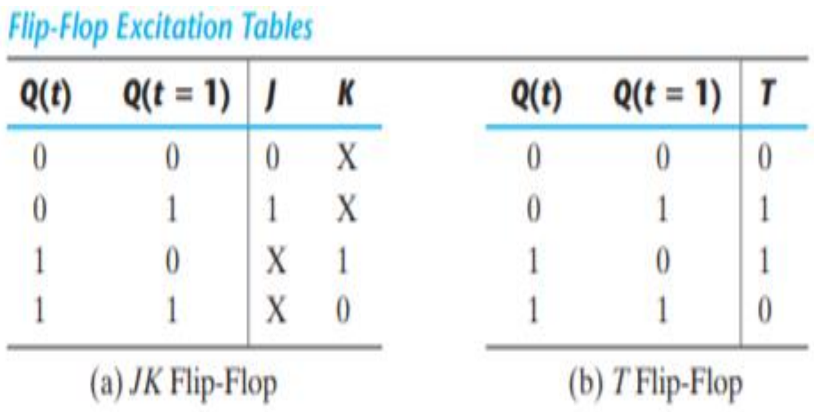
- The excitation tables for the two flip-flops (JK and T).



Flip-Flop Characteristic Tables

JK Flip-Flop

J	K	$Q(t+1)$	
0	0	$Q(t)$	No change
0	1	0	Reset
1	0	1	Set
1	1	$Q'(t)$	Complement



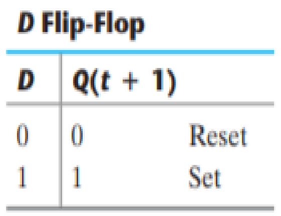
Flip-Flop Excitation Tables

$Q(t)$	$Q(t = 1)$	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

(a) *JK* Flip-Flop

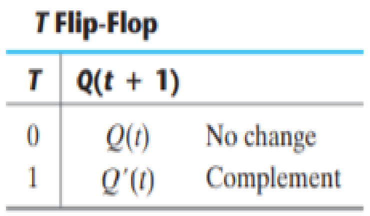
$Q(t)$	$Q(t = 1)$	T
0	0	0
0	1	1
1	0	1
1	1	0

(b) *T* Flip-Flop



D Flip-Flop

D	$Q(t+1)$	
0	0	Reset
1	1	Set



T Flip-Flop

T	$Q(t+1)$	
0	$Q(t)$	No change
1	$Q'(t)$	Complement

- Each table has a column for the present state Q (t), a column for the next state Q(t + 1), and a column for each input to show how the required transition is achieved.

Synthesis Using D Flip-Flops

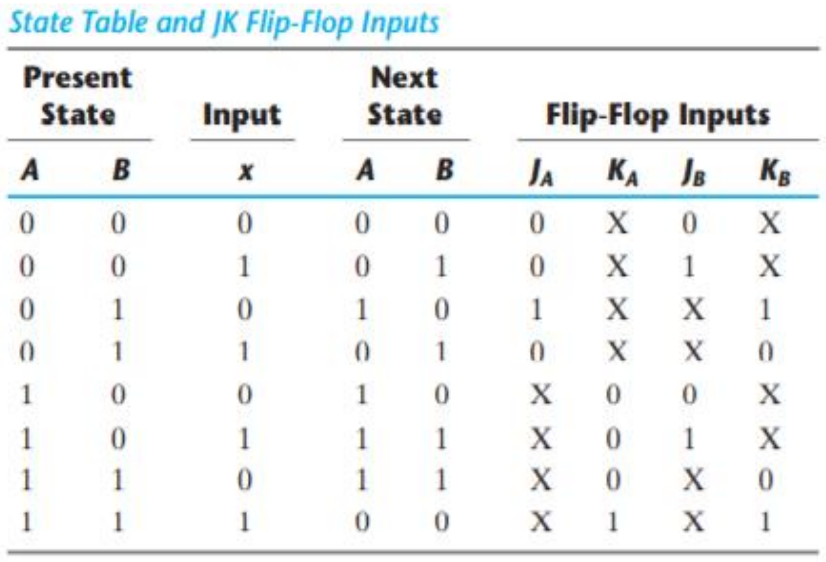
- There are four possible transitions from the present state to the next state. The required input conditions for each of the four transitions are derived from the information available in the characteristic table.
- The symbol X in the tables represents a don’t-care condition, which means that it does not matter whether the input is 1 or 0.

Synthesis Using JK Flip-Flops

- The manual synthesis procedure for sequential circuits with JK flip-flops is the same as with D flip-flops, except that the input equations must be evaluated from the present state to the next-state transition derived from the excitation table.
- In addition to having columns for the present state, input, and next state, as in a conventional state table, the table shows the flip-flop input conditions from which the input equations are derived. The flip-flop inputs are derived from the state table in conjunction with the excitation table for the JK flip-flop.

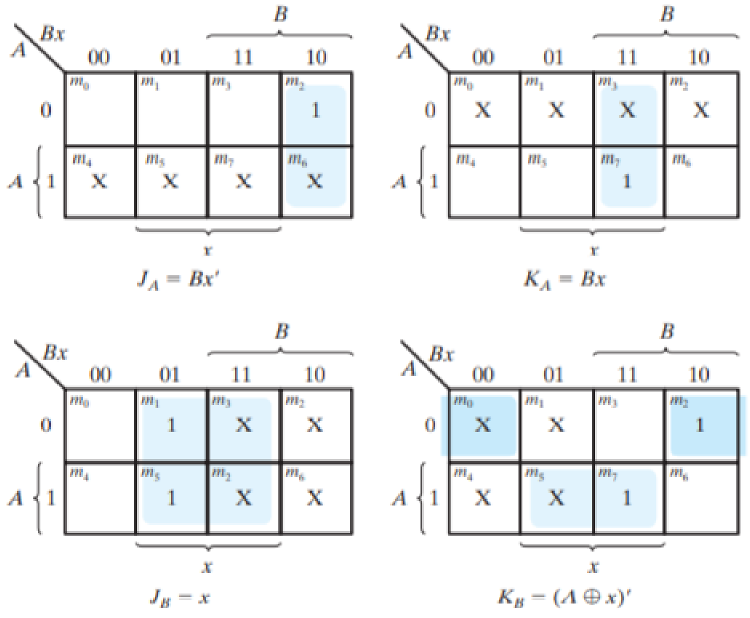
Synthesis Using JK Flip-Flops

- Synthesis the sequential circuit



State Table and JK Flip-Flop Inputs

Present State		Input	Next State		Flip-Flop Inputs			
A	**B**	**x**	**A**	**B**	J_A	K_A	J_B	K_B
0	0	0	0	0	0	X	0	X
0	0	1	0	1	0	X	1	X
0	1	0	1	0	1	X	X	1
0	1	1	0	1	0	X	X	0
1	0	0	1	0	X	0	0	X
1	0	1	1	1	X	0	1	X
1	1	0	1	1	X	0	X	0
1	1	1	0	0	X	1	X	1



$J_A = Bx'$

$K_A = Bx$

$J_B = x$

$K_B = (A \oplus x)'$

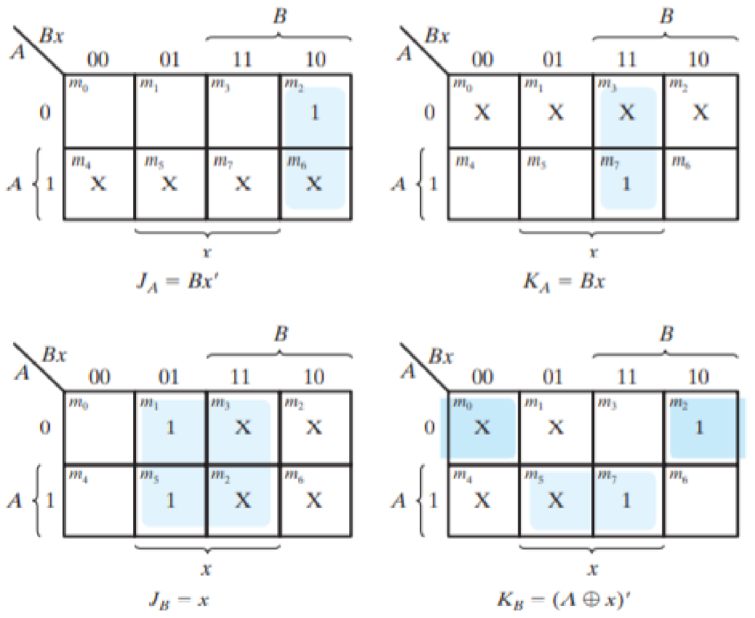
Maps for j and k input equations

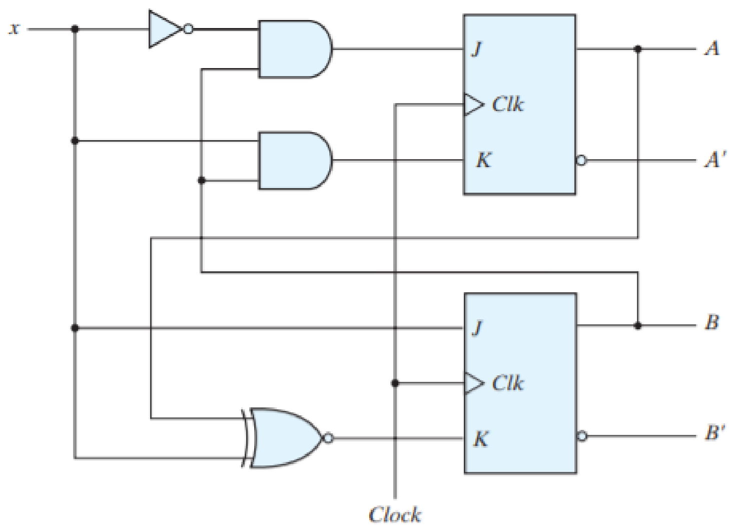
Advantage of using JK -type flip-flops

- The fact that there are so many don’t-care entries indicates that the combinational circuit for the input equations is likely to be simpler, because don’t-care minterms usually help in obtaining simpler expressions.
- If there are unused states in the state table, there will be additional don’t-care conditions in the map. Nonetheless, D-type flip-flops are more amenable to an automated design flow.

Advantage of using JK -type flip-flops

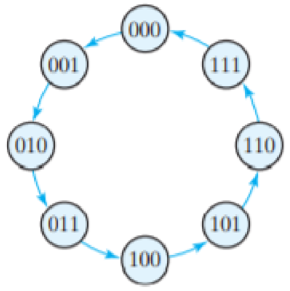
- The four input equations for the pair of JK flip-flops are listed under the maps of Fig. in LHS.



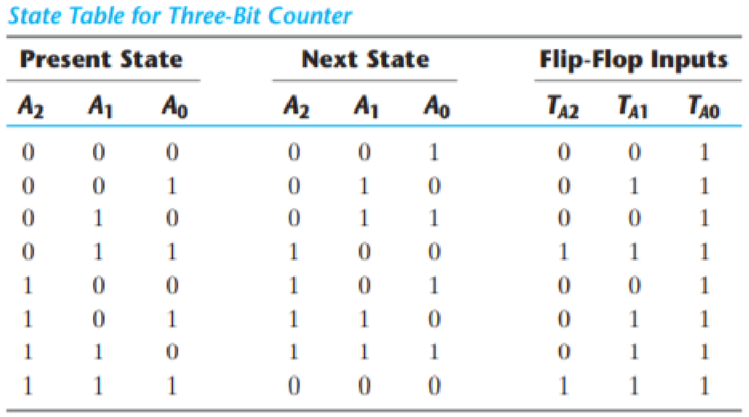


- The logic diagram (schematic) of the sequential circuit is drawn in RHS

Synthesis using T flip-flops

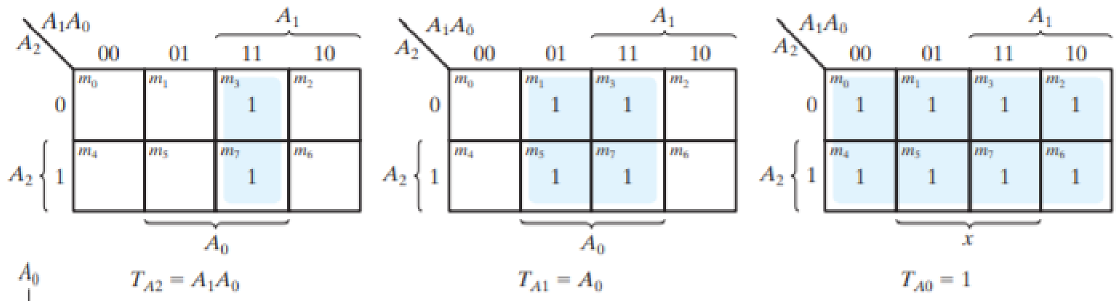


State Diagram of 3 bit binary counter

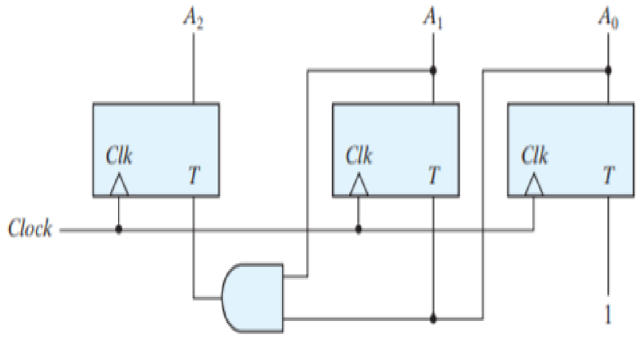


State Table for Three-Bit Counter

Present State			Next State			Flip-Flop Inputs		
A_2	A_1	A_0	A_2	A_1	A_0	T_{A2}	T_{A1}	T_{A0}
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	1	1
1	1	1	0	0	0	1	1	1



Maps for 3 bit binary counter



Logic Diagram of 3 bit binary counter

Assignment

1. A sequential circuit has two *JK* flip-flops *A* and *B* and one input *x*. The circuit is described by the following flip-flop input equations:

$$J_A = x \quad K_A = B$$
$$J_B = x \quad K_B = A'$$

(a) Derive the state equations $A(t+1)$ and $B(t+1)$ by substituting the input equations for the *J* and *K* variables.
(b) Draw the state diagram of the circuit.

2. A sequential circuit has two *JK* flip-flops *A* and *B*, two inputs *x* and *y*, and one output *z*. The flip-flop input equations and circuit output equation are

$$J_A = Bx + B'y' \qquad K_A = B'xy'$$
$$J_B = A'x \qquad K_B = A + xy'$$
$$z = Ax'y' + Bx'y'$$

(a) Draw the logic diagram of the circuit.
(b) Tabulate the state table.
(c) Derive the state equations for *A* and *B*.

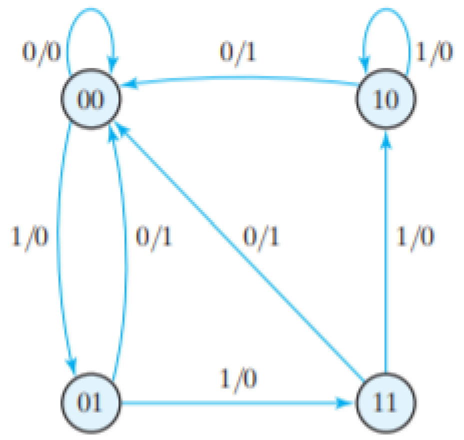
Assignment

3. For the circuit described by the state diagram of Fig.

   (a)* Determine the state transitions and output sequence that will be generated when an input sequence of 010110111011110 is applied to the circuit and it is initially in the state 00.

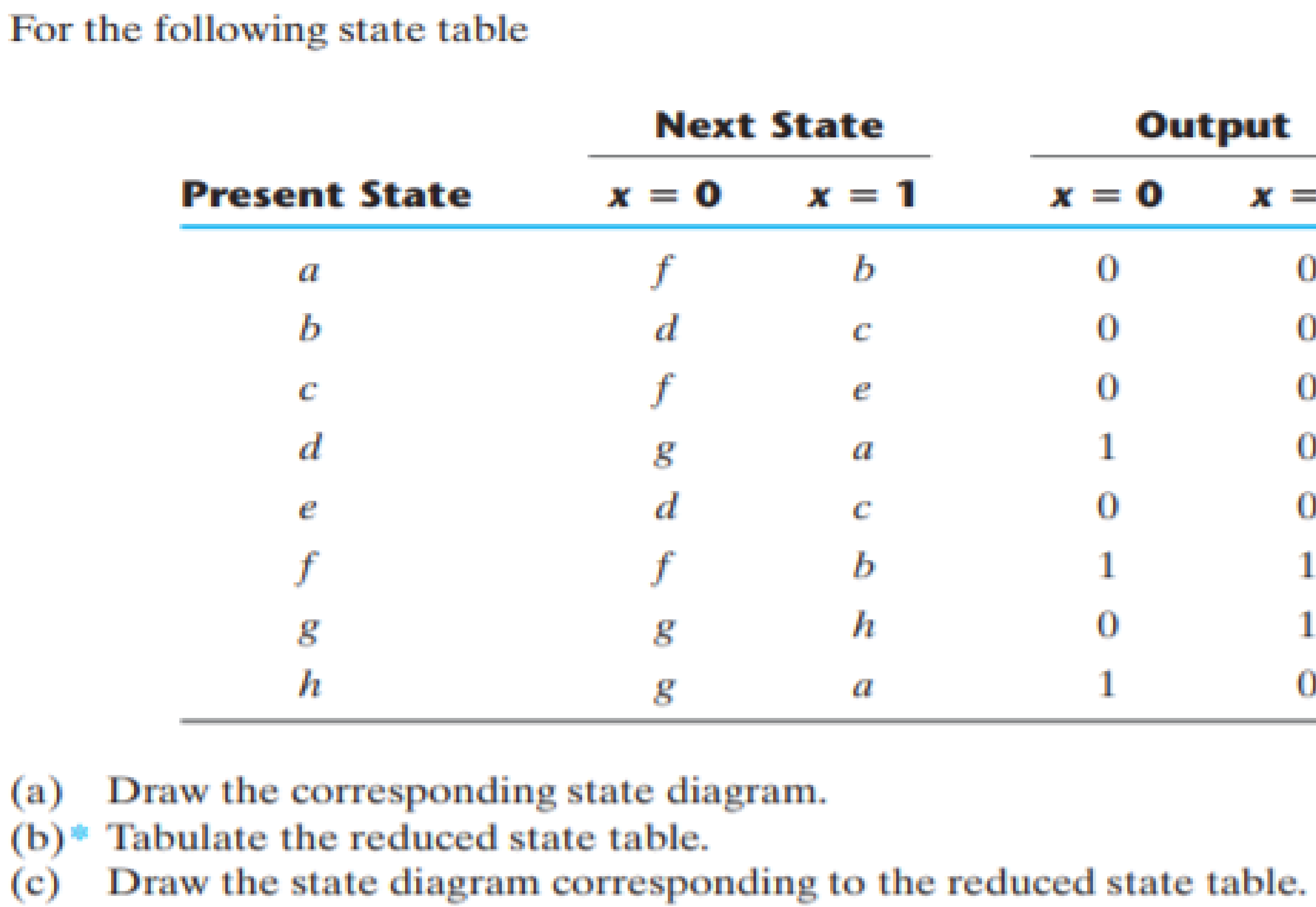
   (b) Find all of the equivalent states in Fig. 5.16 and draw a simpler, but equivalent, state diagram.

   (c) Using *D* flip-flops, design the equivalent machine (including its logic diagram) described by the state diagram in (b).



Assignment

3.



For the following state table

	Next State		Output	
Present State	$x = 0$	$x = 1$	$x = 0$	$x = 1$
a	f	b	0	0
b	d	c	0	0
c	f	e	0	0
d	g	a	1	0
e	d	c	0	0
f	f	b	1	1
g	g	h	0	1
h	g	a	1	0

(a) Draw the corresponding state diagram.
(b)* Tabulate the reduced state table.
(c) Draw the state diagram corresponding to the reduced state table.

Assignment

4. Starting from state *a*, and the input sequence 01110010011, determine the output sequence for

(a) The state table of the previous problem.

(b) The reduced state table from the previous problem. Show that the same output sequence is obtained for both.