DIGITAL SYSTEM DESIGN USING VERILOG

COURSE CODE: 19EC5DCDSV
(3 CREDITS)
MODULE-5A

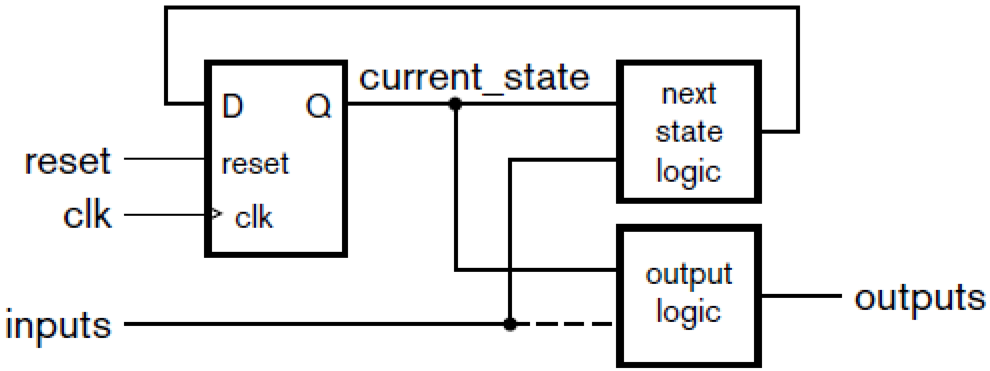
Faculty In-Charge: Dr. Dinesha P

drdinesh-ece@dayanandasagar.edu

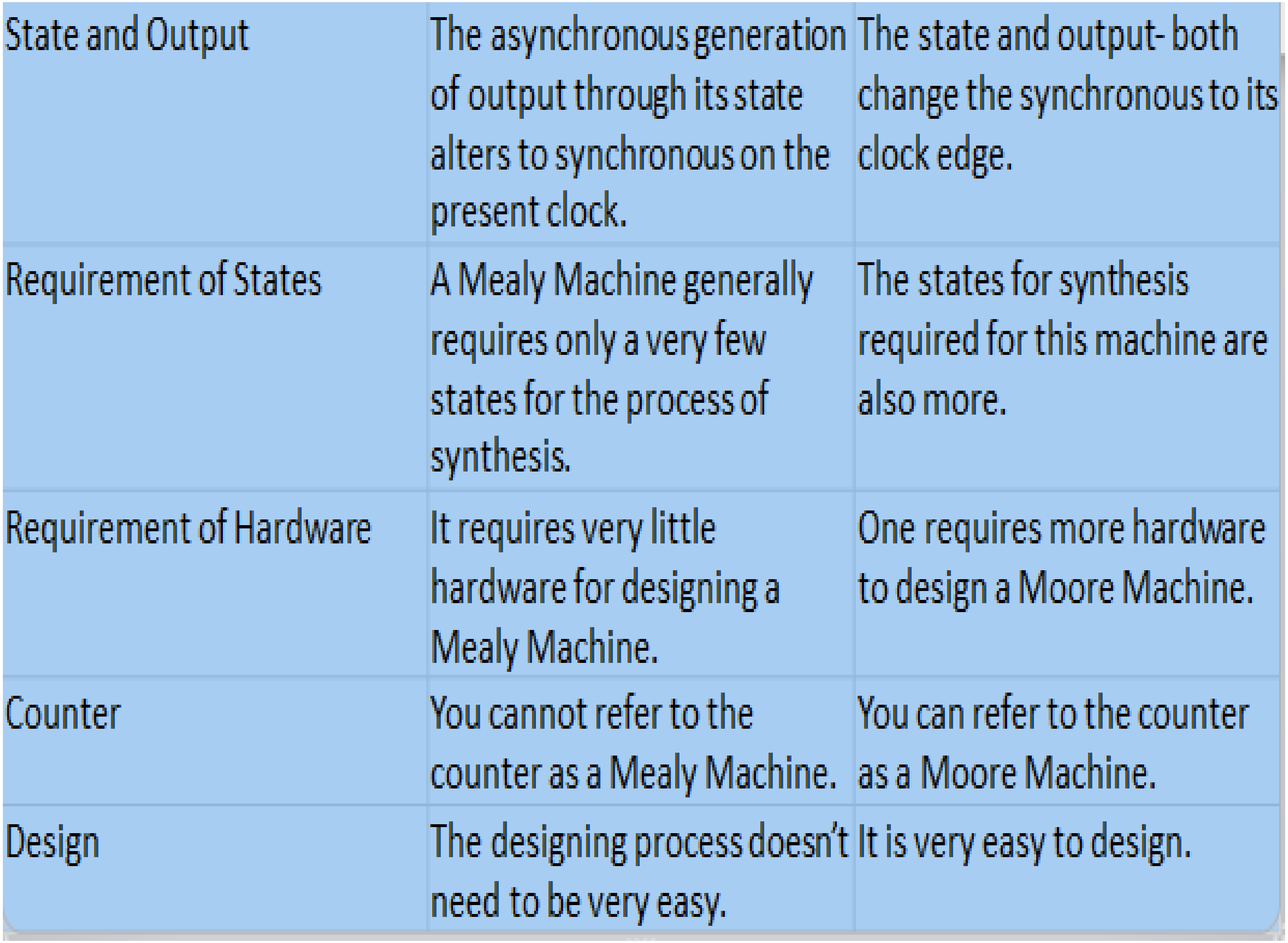
Synchronous sequential circuits:

Finite-state machine (FSM): FSM is a type of synchronous sequential circuits. A finite-state machine is defined by a set of inputs, a set of outputs, a set of states, a transition function that governs transitions between states, and an output function.

- The machine is called “finite-state” because the set of states is finite in size.
- The finite-state machine has a current state in a given clock cycle. The transition function determines the next state for the next clock cycle based on the current state and, possibly, the values of inputs in the given clock cycle. The output function determines the values of the outputs in a given clock cycle based on the current state and, possibly, the values of inputs in the given clock cycle.



parameters	Mealy Machine	Moore Machine
Definition	A Mealy Machine changes its output on the basis of its present state and current input.	A Moore Machine’s output depends only on the current state. It does not depend on the current input.
Output	Mealy Machine places its output on the transition.	Moore machine also places its output on the transition.
States	It has comparatively fewer or the same states as that of the Moore machine.	It has comparatively fewer or the same states as that of the Mealy machine.
Value of the Output Function	When the input logic is done in the present state, then the value of the output function becomes a function of transitions and changes.	Whenever a change occurs in the state, the output function’s value becomes the function of its current state along with the changes at the edges of the clock.
Reaction to the Inputs	A Mealy machine reacts comparatively faster to inputs than the Moore machine. Generally, it reacts in the very same clock cycle.	In a Moore Machine, one requires more logic for decoding the output. As a result, it leads to more delays in the circuit. Generally, these react after one clock cycle.



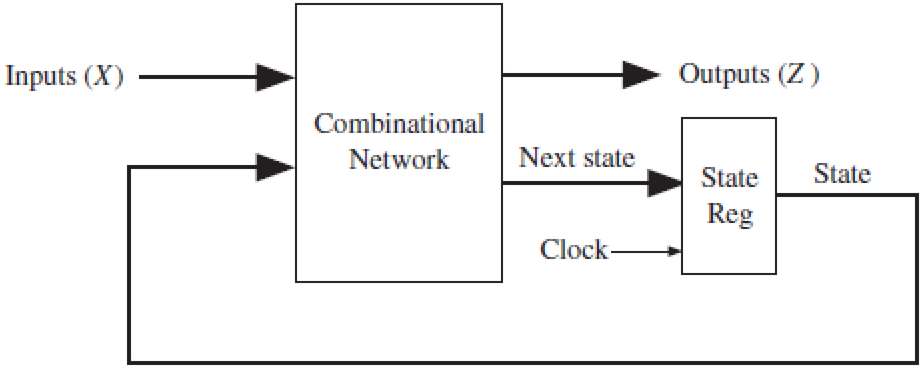
State and Output	The asynchronous generation of output through its state alters to synchronous on the present clock.	The state and output- both change the synchronous to its clock edge.
Requirement of States	A Mealy Machine generally requires only a very few states for the process of synthesis.	The states for synthesis required for this machine are also more.
Requirement of Hardware	It requires very little hardware for designing a Mealy Machine.	One requires more hardware to design a Moore Machine.
Counter	You cannot refer to the counter as a Mealy Machine.	You can refer to the counter as a Moore Machine.
Design	The designing process doesn't need to be very easy.	It is very easy to design.

- There are two types of sequential circuits:

  Mealy sequential circuits

  Moore sequential circuits

- In a Mealy circuit, the outputs depend on both the present state and the present inputs.
- In a Moore circuit, the outputs depend only on the present state.
- A general model of a Mealy sequential circuit consists of a combinational circuit, which generates the outputs and the next state, and a state register, which holds the present state as shown in figure.

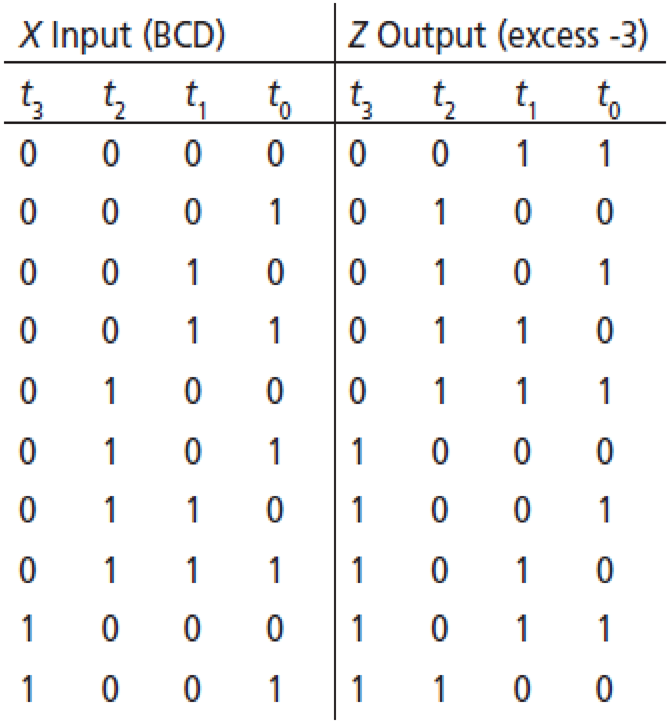


Mealy Machine Design:

Example: BCD to Ex-3 code convereter

Design a serial code converter that converts an 8-4-2-1 binary-coded-decimal (BCD) digit to an excess-3-coded decimal digit. The input *(X) will arrive serially with* the least significant bit first. The outputs will be generated serially as well.

- Table lists the desired inputs and outputs at times *t_0, t_1, t_2, and t_3. After* receiving four inputs, the circuit should reset to its initial state, ready to receive another BC



X Input (BCD)				*Z* Output (excess -3)			
t_3	t_2	t_1	t_0	t_3	t_2	t_1	t_0
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

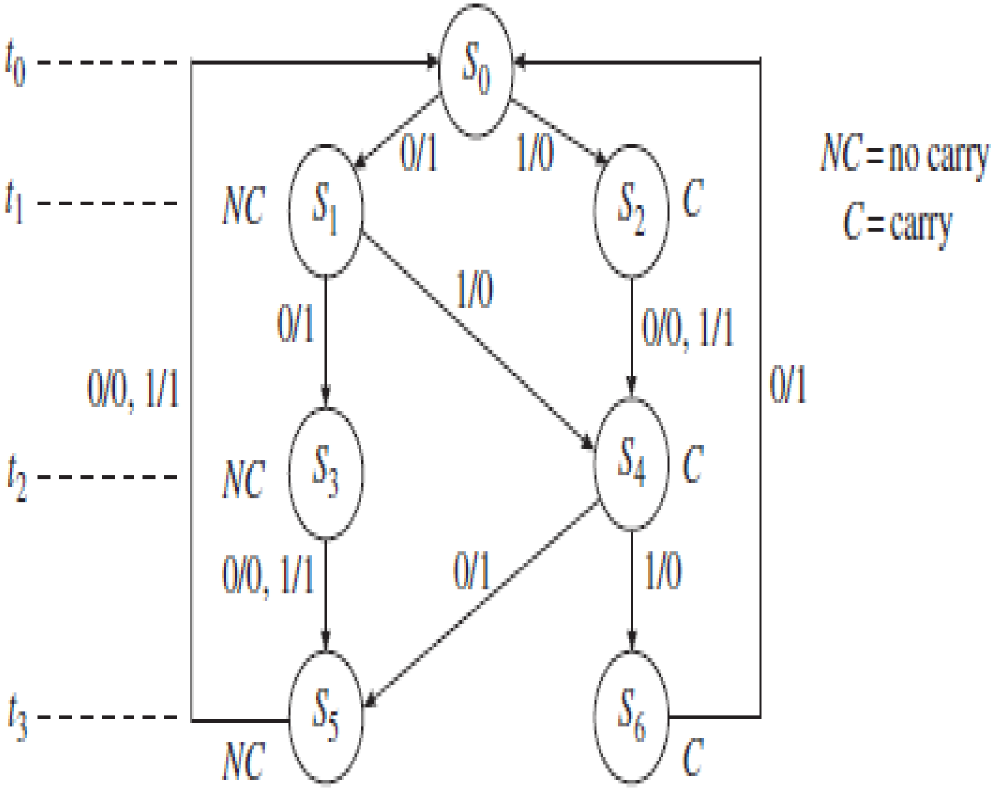
State graph:

We know that, Ex-3 obtained by adding 3 to each BCD number

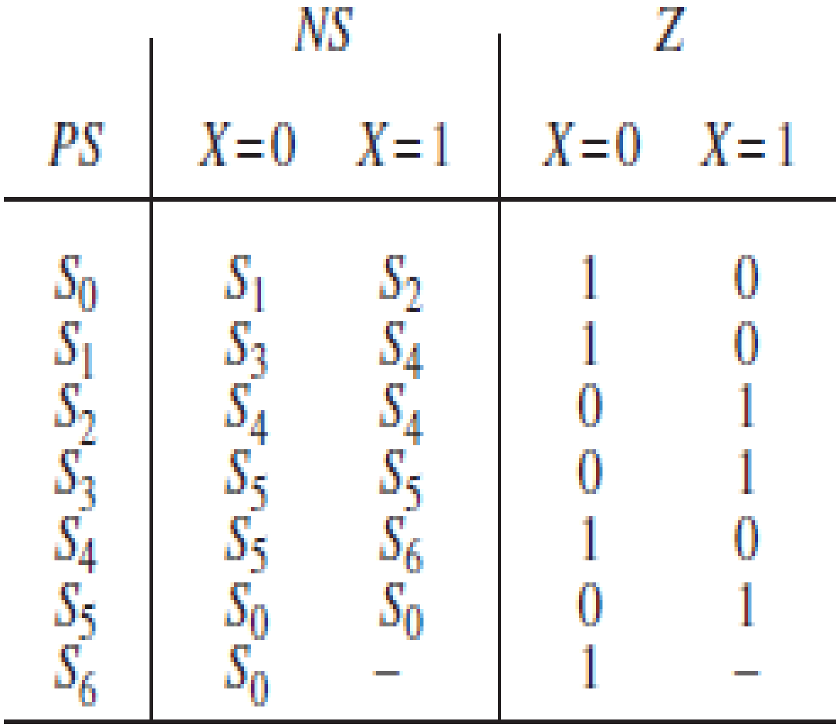
x=	0100	0101
	0011	0011
z=	0111	1000

- ○ Assume start state is S_0. The first bit arrives, and one needs to add 1 to this bit, as it is the least significant bit (LSB) of 0011, the number to be added to the BCD digit to obtain the excess-3 code.
- At t_0, we add 1 to the LSB, so if X = 0, Z = 1 (no carry), and if X = 1, Z = 0 (carry = 1). Let us use S_1 to indicate no carry after the first addition, and S_2 to indicate a carry of 1 after the addition to the LSB.
- At t_1, we add 1 to the next bit, so if there is no carry from the first addition (state S_1), X =0 gives Z = 0 + 1 + 0 = 1 and no carry (state S3), and X = 1 gives Z = 1 + 1 +0 =0 and a carry (state S_4). If there is a carry from the first addition (state S_2), then X = 0 gives Z = 0 + 1 + 1 = 0 and a carry (S_4), and X 5 1 gives Z = 1 + 1 + 1 = 1 and a carry (S_4).

- *At t_2, 0 is added to X, and transitions to S_5 (no carry) and S_6 are determined in a similar manner. At t_3, 0 is again added to X, and the circuit resets to S_0.*



(a) Mealy state graph



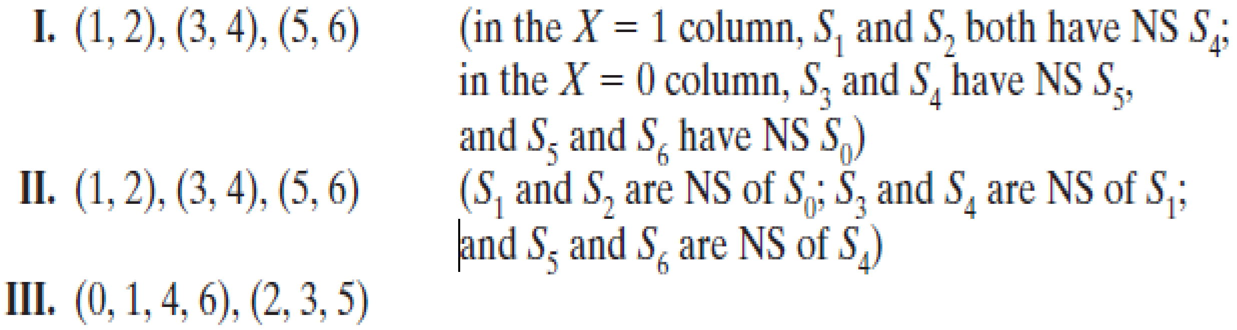
	NS		Z	
PS	X = 0	X = 1	X = 0	X = 1
S_0	S_1	S_2	1	0
S_1	S_3	S_4	1	0
S_2	S_4	S_4	0	1
S_3	S_5	S_5	0	1
S_4	S_5	S_6	1	0
S_5	S_0	S_0	0	1
S_6	S_0	–	1	–

- The next step is to make a state assignment that relates the flip-flop states to the states in the table.
- Simple way is a straight binary state assignment. But here using an optimal assignment.
- In order to reduce the amount of logic required, we will make a state assignment using the following guidelines,

  I. States that have the same next state (NS) for a given input should be given adjacent assignments (look at the columns of the state table).

  II. States that are the next states of the same state should be given adjacent assignments (look at the rows).

  III. States that have the same output for a given input should be given adjacent assignments.
- The guidelines indicate that the following states should be given adjacent assignments:

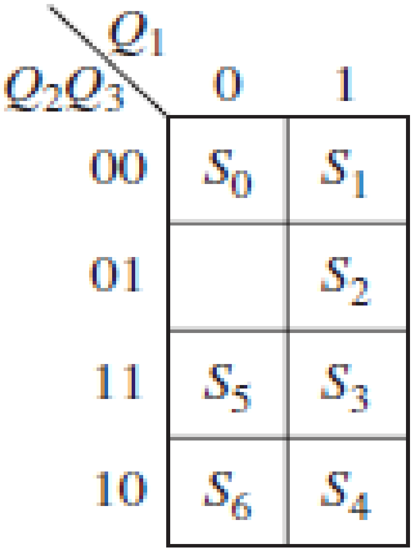


I. (1, 2), (3, 4), (5, 6) (in the $X = 1$ column, S_1 and S_2 both have NS S_4; in the $X = 0$ column, S_3 and S_4 have NS S_5, and S_5 and S_6 have NS S_0)

II. (1, 2), (3, 4), (5, 6) (S_1 and S_2 are NS of S_0; S_3 and S_4 are NS of S_1; and S_5 and S_6 are NS of S_4)

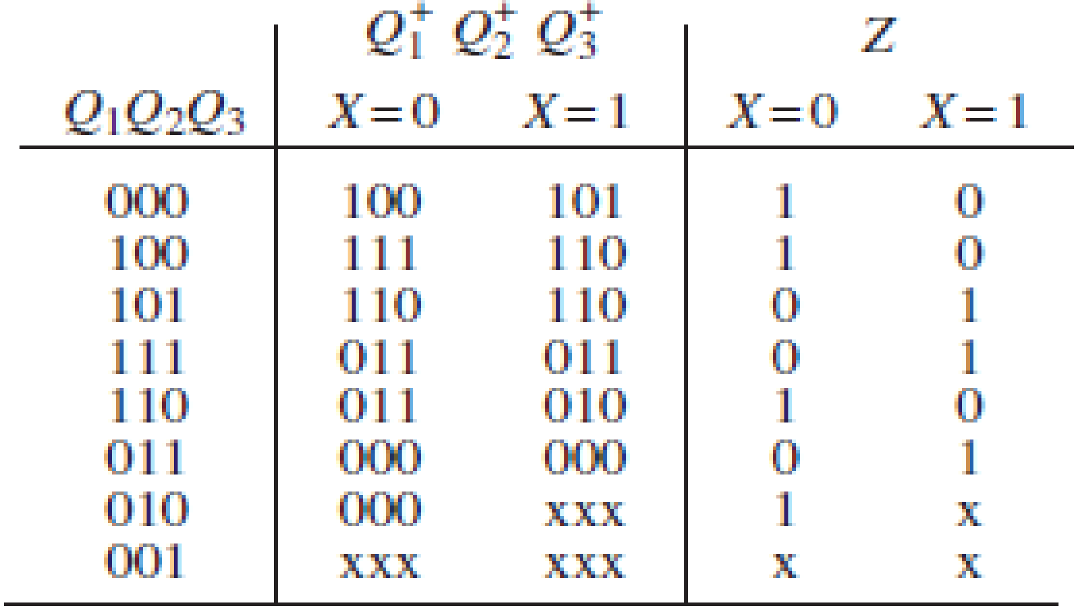
III. (0, 1, 4, 6), (2, 3, 5)

- Figure gives an assignment map, which satisfies the guidelines, and the corresponding transition table.



Q_2Q_3 \ Q_1	0	1
00	S_0	S_1
01		S_2
11	S_5	S_3
10	S_6	S_4

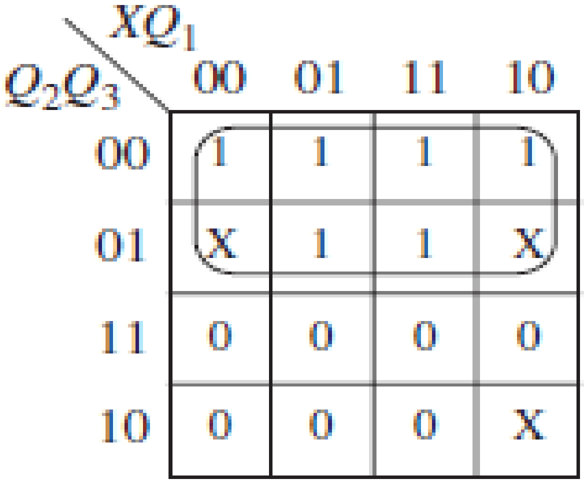
(a) Assignment map



	$Q_1^+ Q_2^+ Q_3^+$		Z	
$Q_1Q_2Q_3$	$X=0$	$X=1$	$X=0$	$X=1$
000	100	101	1	0
100	111	110	1	0
101	110	110	0	1
111	011	011	0	1
110	011	010	1	0
011	000	000	0	1
010	000	xxx	1	x
001	xxx	xxx	x	x

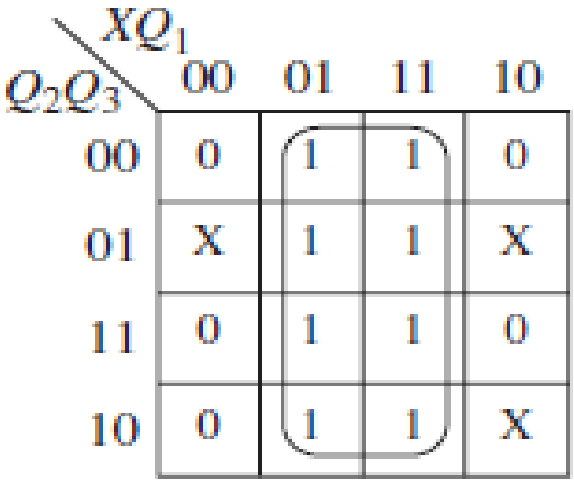
(b) Transition table

○ K-maps



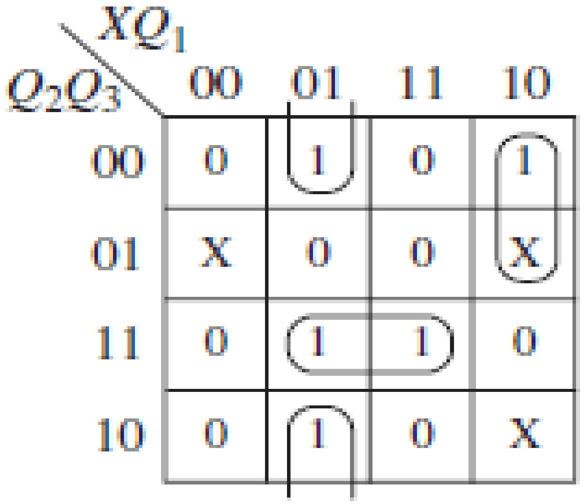
Q_2Q_3 \ XQ_1	00	01	11	10
00	1	1	1	1
01	X	1	1	X
11	0	0	0	0
10	0	0	0	X

$D_1 = Q_1^+ = Q_2'$



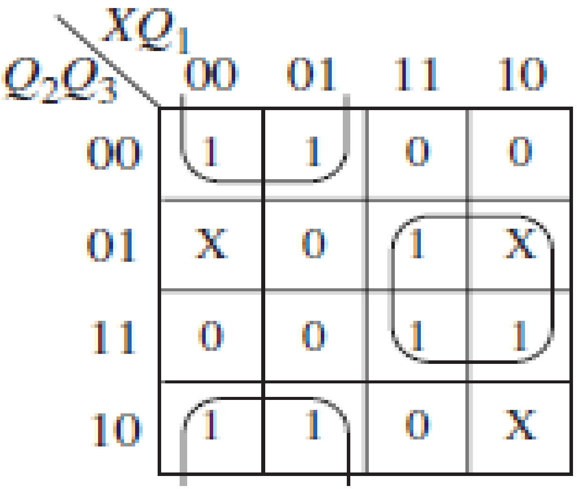
Q_2Q_3 \ XQ_1	00	01	11	10
00	0	1	1	0
01	X	1	1	X
11	0	1	1	0
10	0	1	1	X

$D_2 = Q_2^+ = Q_1$



Q_2Q_3 \ XQ_1	00	01	11	10
00	0	1	0	1
01	X	0	0	X
11	0	1	1	0
10	0	1	0	X

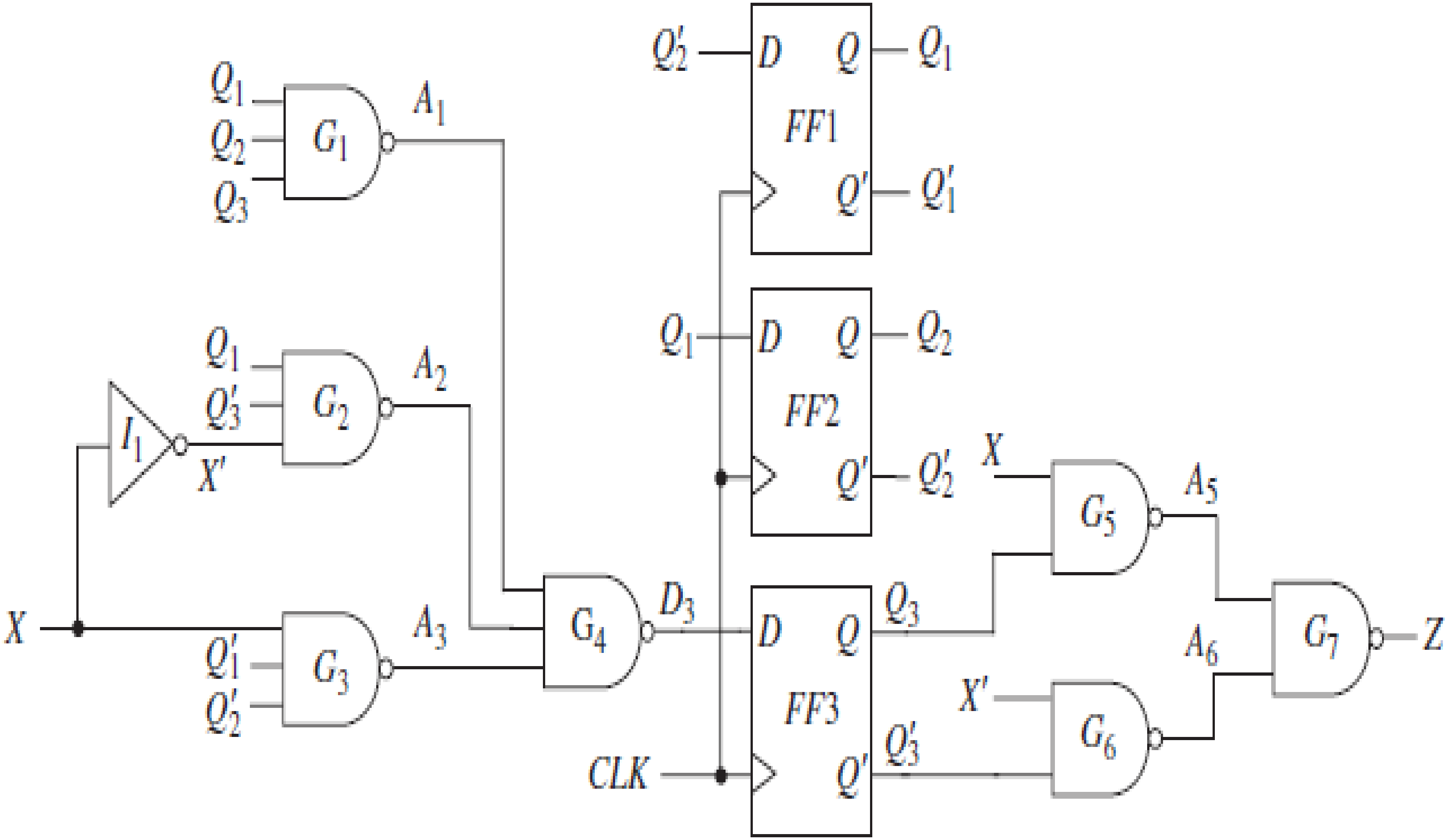
$D_3 = Q_3^+ = Q_1Q_2Q_3 + X'Q_1Q_3' + XQ_1'Q_2'$



Q_2Q_3 \ XQ_1	00	01	11	10
00	1	1	0	0
01	X	0	1	X
11	0	0	1	1
10	1	1	0	X

$Z = X'Q_3' + XQ_3$

Realization of Code Converter using D flip-flop:


Q1
Q2
Q3
G1
A1
Q1
Q3'
I1
X'
G2
A2
X
Q1'
Q2'
G3
A3
G4
D3
Q2'
D
Q
Q1
FF1
Q'
Q1'
Q1
D
Q
Q2
FF2
Q'
Q2'
D
Q
Q3
FF3
Q'
Q3'
CLK
X
G5
A5
A6
X'
G6
G7
Z