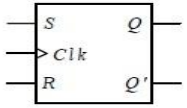
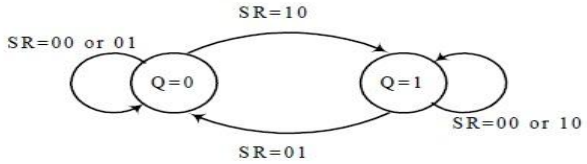
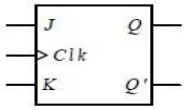
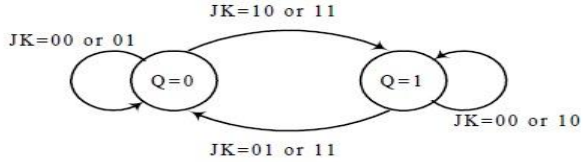
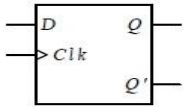
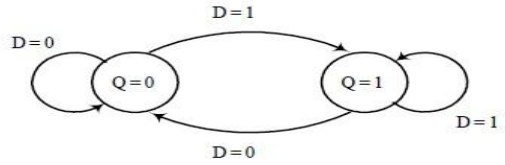
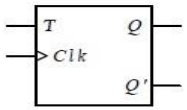
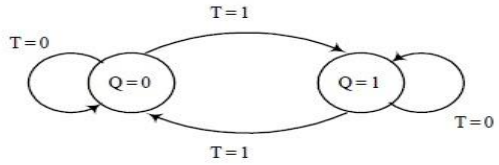
Course Name: Digital Hardware Design
Course Code: 17B1NEC741

Finite State Machine-3

Dr. Arti Noor
Dean, Academic Affairs
Electronics and Communication Engineering,
Jaypee Institute of Information Technology, Noida

Sequential Storage Units

Name / Symbol	Characteristic (Truth) Table	State Diagram / Characteristic Equations	Excitation Table
SR 	$S\ R\ Q\ Q_{next}$ 0 0 0 0 0 0 1 1 0 1 0 0 0 1 1 0 1 0 0 1 1 0 1 1 1 1 0 × 1 1 1 ×	 $Q_{next} = S + R'Q$ $SR = 0$	$Q\ Q_{next}\ S\ R$ 0 0 0 × 0 1 1 0 1 0 0 1 1 1 × 0
JK 	$J\ K\ Q\ Q_{next}$ 0 0 0 0 0 0 1 1 0 1 0 0 0 1 1 0 1 0 0 1 1 0 1 1 1 1 0 1 1 1 1 0	 $Q_{next} = J'K'Q + JK' + JKQ'$ $= J'K'Q + JK'Q + JK'Q' + JKQ'$ $= K'Q(J'+J) + JQ'(K'+K)$ $= K'Q + JQ'$	$Q\ Q_{next}\ J\ K$ 0 0 0 × 0 1 1 × 1 0 × 1 1 1 × 0
D 	$D\ Q\ Q_{next}$ 0 × 0 1 × 1	 $Q_{next} = D$	$Q\ Q_{next}\ D$ 0 0 0 0 1 1 1 0 0 1 1 1
T 	$T\ Q\ Q_{next}$ 0 0 0 0 1 1 1 0 1 1 1 0	 $Q_{next} = TQ' + T'Q = T \oplus Q$	$Q\ Q_{next}\ T$ 0 0 0 0 1 1 1 0 1 1 1 0

FSM Design Procedure

1. Understand specifications
2. Derive state diagram
3. Create state table
4. Perform state minimization (if necessary)
5. Encode states (state assignment)
6. Create state-assigned table
7. Select type of Flip-Flop to use
8. Determine Flip-Flop input equations and FSM output equation(s)
9. Draw logic diagram

FSM Design Example 1 (Mealy)

1. Understand specifications

Design a FSM that detects a sequence of three or more consecutive ones on an input bit stream.

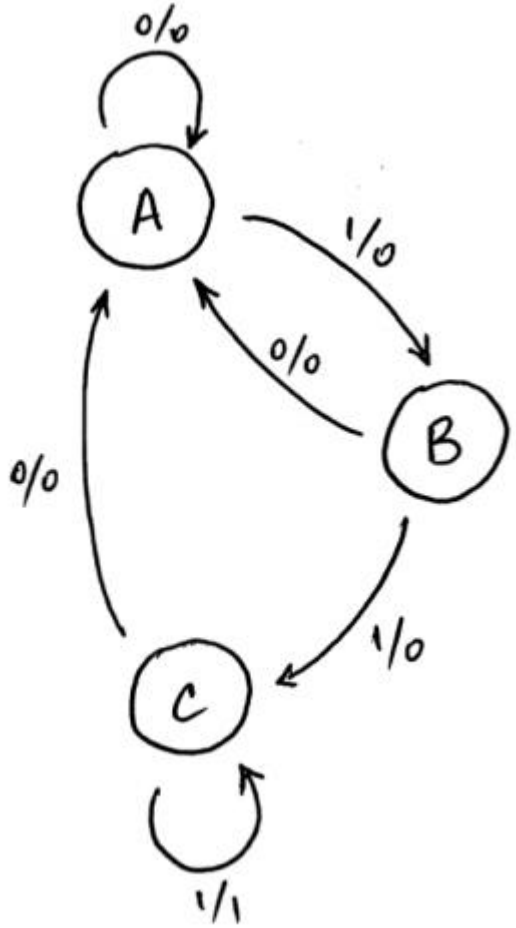
The FSM should output a 1 when the sequence is detected, and a 0 otherwise.

Input: 0 1 1 1 0 1 0 1 1 0 1 1 1 0 1 …

Output: 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0

FSM Design Example 1 (Mealy)

2. Derive state diagram



FSM Design Example 1 (Mealy)

3. Create state table
4. Perform state minimization (if necessary)
5. Encode states (state assignment)

Present State			Next State						Output	
			w = 0			w = 1			w = 0	w = 1
	Q_A	Q_B		Q_A^+	Q_B^+		Q_A^+	Q_B^+	z	z
A	0	0	A	0	0	B	0	1	0	0
B	0	1	A	0	0	C	1	0	0	0
C	1	0	A	0	0	C	1	0	0	1
D	1	1		d	d		d	d	d	d

Using Binary Encoding for the State Assignment

Next state is a function of the present state and the input

Output is a function of the present state and the input (Mealy Machine)

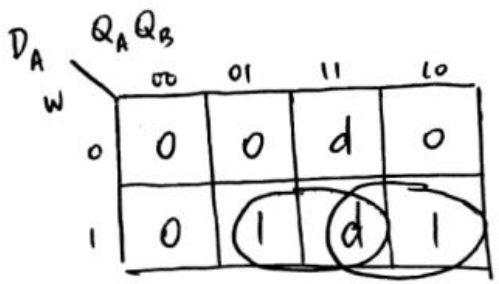
FSM Design Example 1 (Mealy)

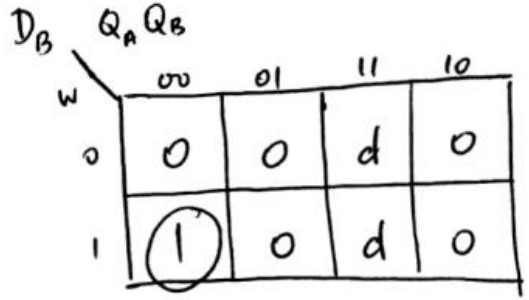
6. Create state-assigned table
7. Select type of Flip-Flop to use

Present State			Next State				FF Inputs			
			w = 0		w = 1		w = 0		w = 1	
	Q_A	Q_B	Q_A^+	Q_B^+	Q_A^+	Q_B^+	D_A	D_B	D_A	D_B
A	0	0	0	0	0	1	0	0	0	1
B	0	1	0	0	1	0	0	0	1	0
C	1	0	0	0	1	0	0	0	1	0
D	1	1	d	d	d	d	d	d	d	d

FSM Design Example 1 (Mealy)

8. Determine Flip-Flop input equations and FSM output equation(s)

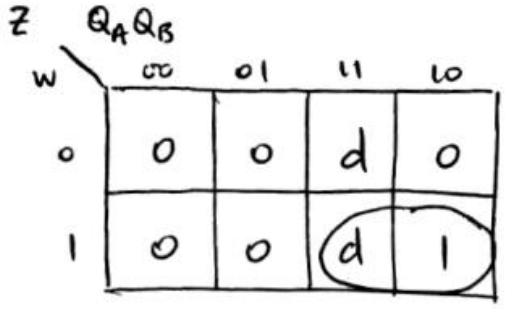




$$D_A = W \cdot Q_B + W \cdot Q_A$$

$$D_A = W \cdot (Q_A + Q_B)$$

$$D_B = W \cdot \overline{Q_A} \cdot \overline{Q_B}$$



$$Z = W \cdot Q_A$$

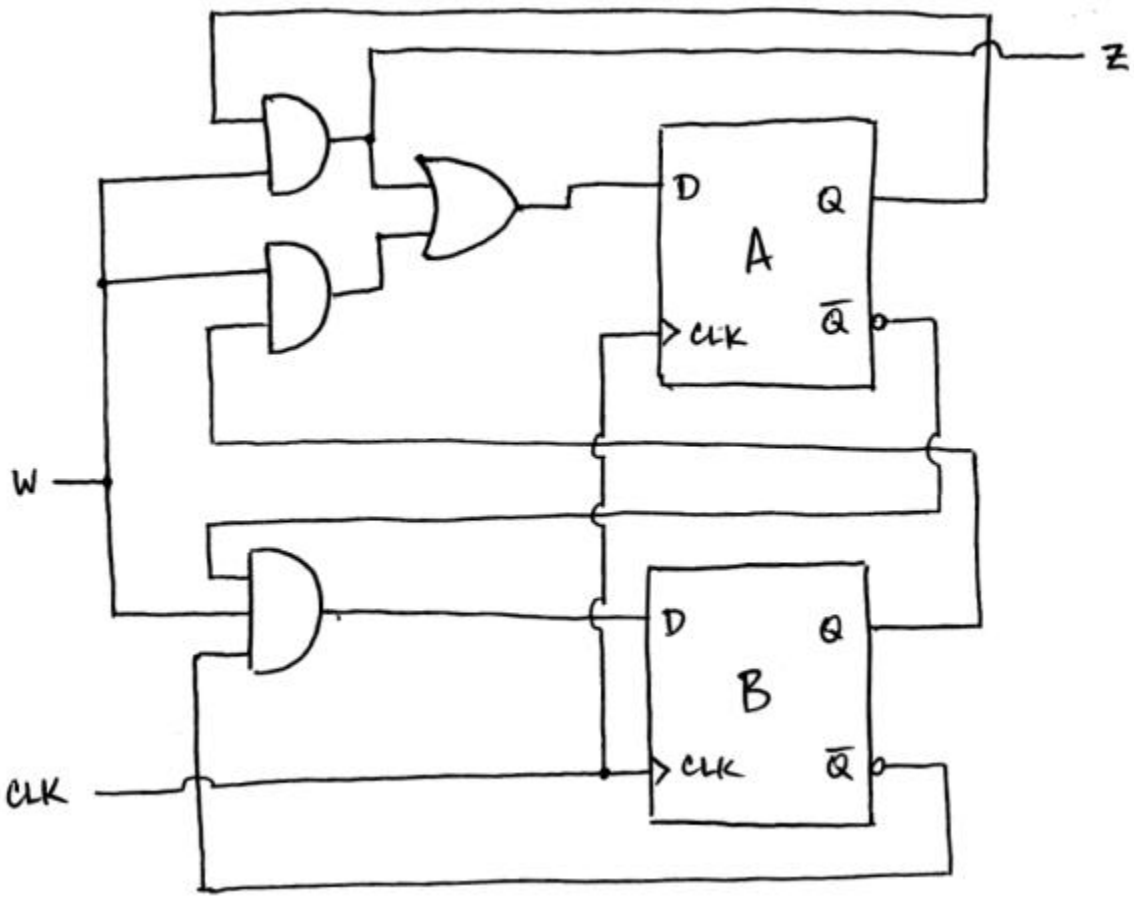
FSM Design Example 1 (Mealy)

9. Draw logic diagram

$D_A=W(Q_A+Q_B)$

$\mathbf{D_B=W \cdot Q_A' \cdot Q_B'}$

$\mathbf{Z=W.Q_A}$

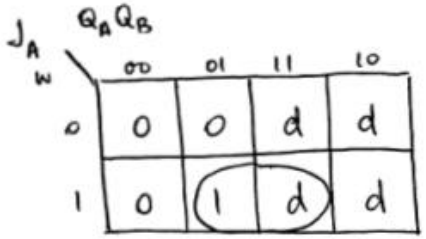


FSM Design Example 1 (Mealy)

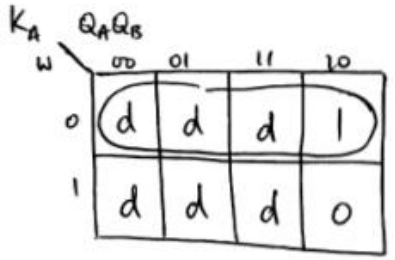
Q	Q^+	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Present State			Next State				FF Inputs							
			w = 0		w = 1		w = 0				w = 1			
	Q_A	Q_B	Q_A^+	Q_B^+	Q_A^+	Q_B^+	J_A	K_A	J_B	K_B	J_A	K_A	J_B	K_B
A	0	0	0	0	0	1	0	d	0	d	0	d	1	d
B	0	1	0	0	1	0	0	d	d	1	1	d	d	1
C	1	0	0	0	1	0	d	1	0	d	d	0	0	d
D	1	1	d	d	d	d	d	d	d	d	d	d	d	d

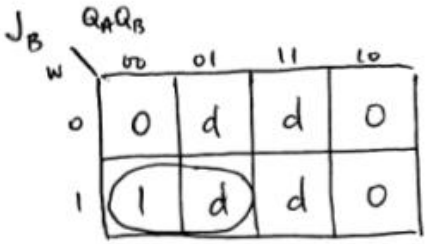
FSM Design Example 1 (Mealy)



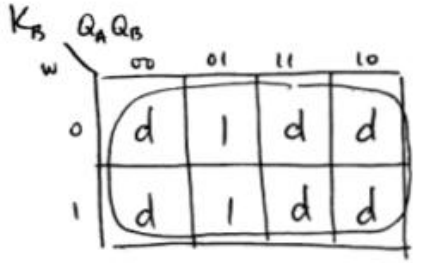
$J_A = w \cdot Q_B$



$K_A = \bar{w}$

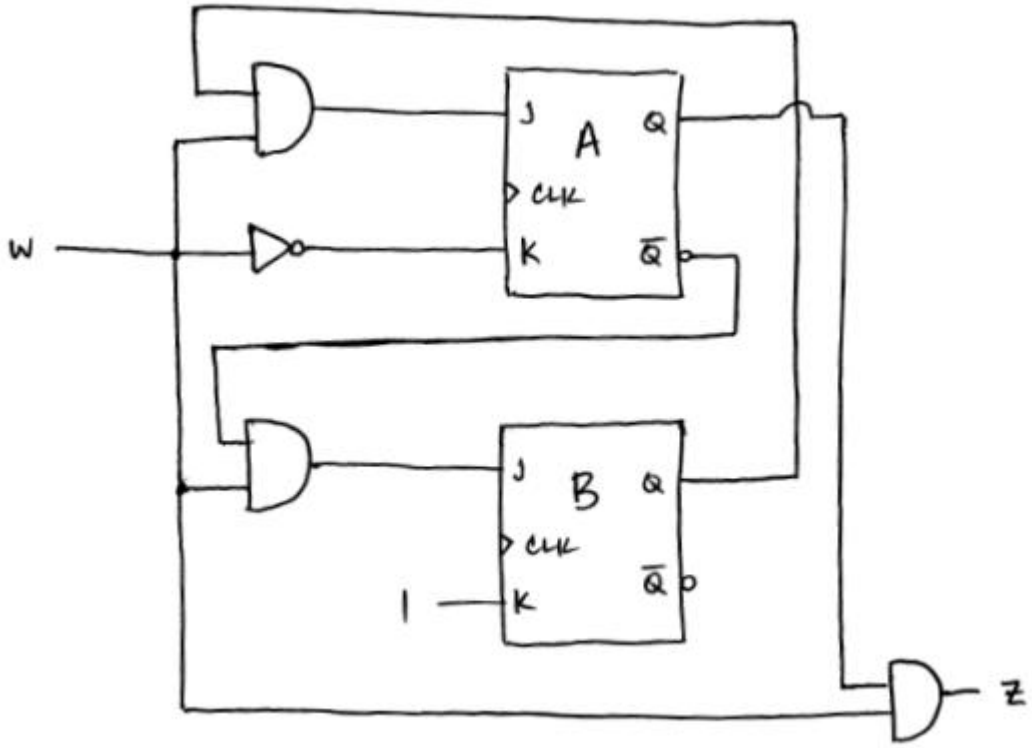


$J_B = w \cdot \bar{Q}_A$



$K_B = 1$

FSM Design Example 1 (Mealy)



FSM Design Example 2 (Mealy)

Design a Finite State Machine (FSM) that meets the following specifications:

1. The circuit has one input, w, and one output, z.

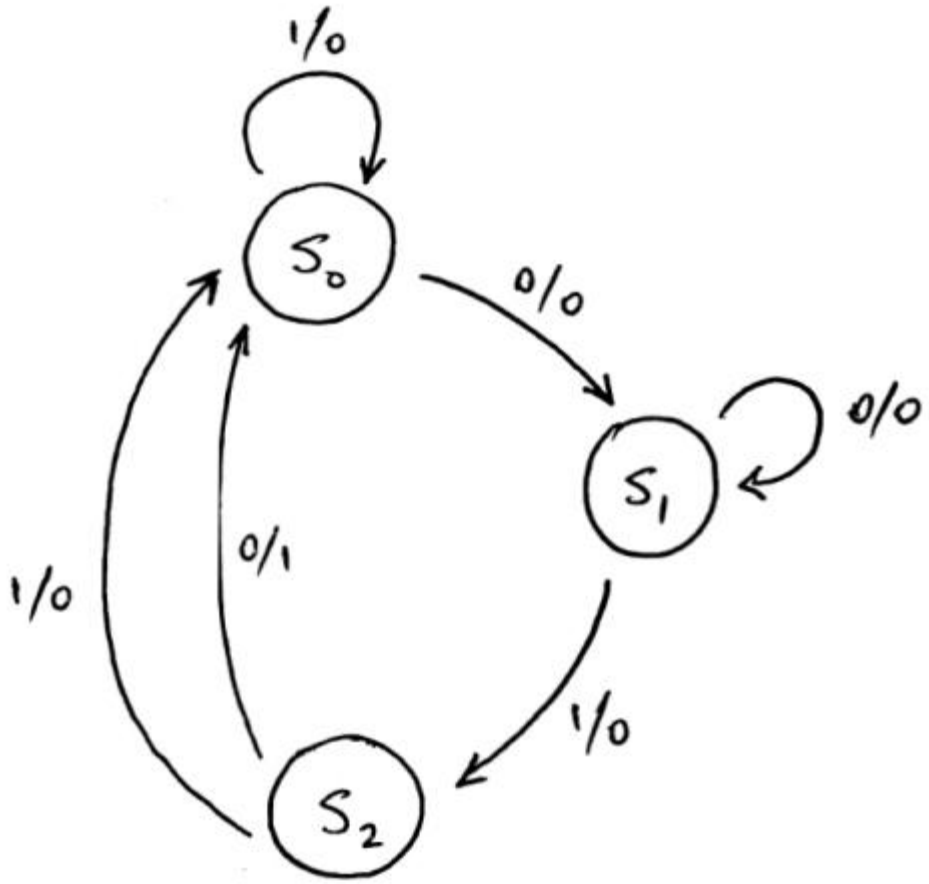
2. All changes in the circuit occur on the positive edge of the clock.

3. The output z is equal to 1 if the pattern 010 is detected on the input w. Otherwise, the value of z is equal to 0. Overlapping sequences should **not** be detected.

Input (w): 0 0 0 0 1 0 1 0 0 1 0 0 1 0 1 ...

Output (z): 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 ...

FSM Design Example 2 (Mealy)



FSM Design Example 3 (Mealy)

Design a Finite State Machine (FSM) that meets the following specifications:

1. The circuit has one input, w, and one output, z.

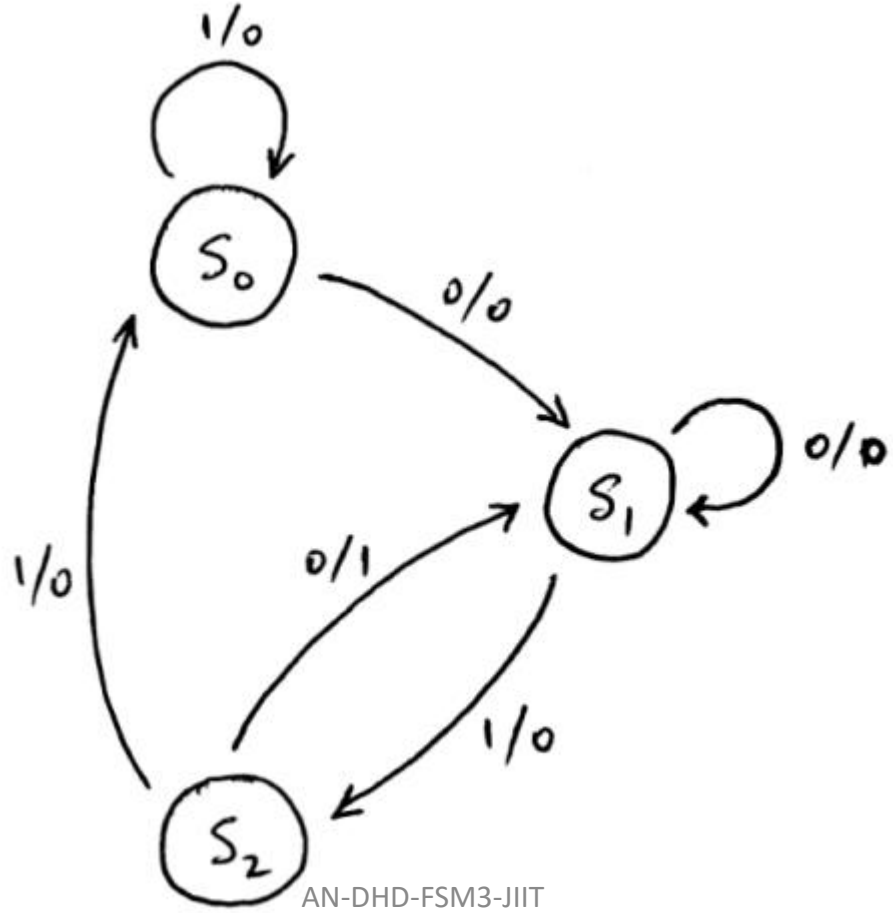
2. All changes in the circuit occur on the positive edge of the clock.

3. The output z is equal to 1 if the pattern 010 is detected on the input w. Otherwise, the value of z is equal to 0. Overlapping sequences **should** be detected.

Input (w): 0 0 0 0 1 0 1 0 0 1 0 0 1 0 1 ...

Output (z): 0 0 0 0 0 1 0 1 0 0 1 0 0 1 0 ...

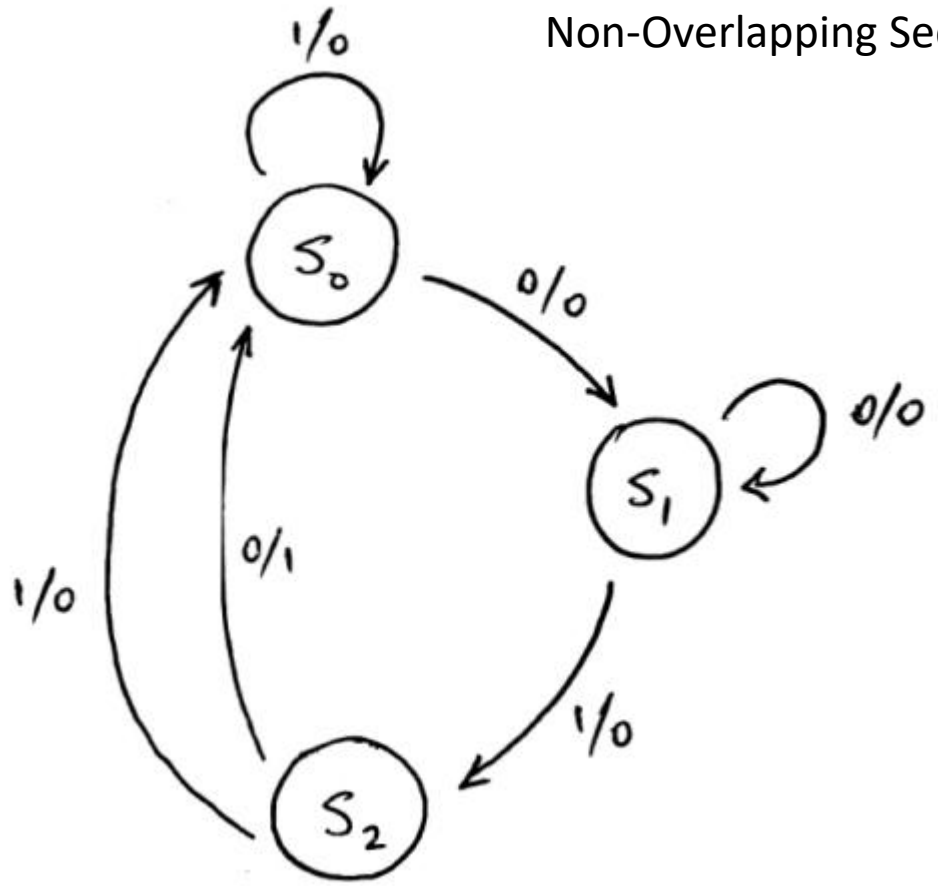
FSM Design Example 3 (Mealy)



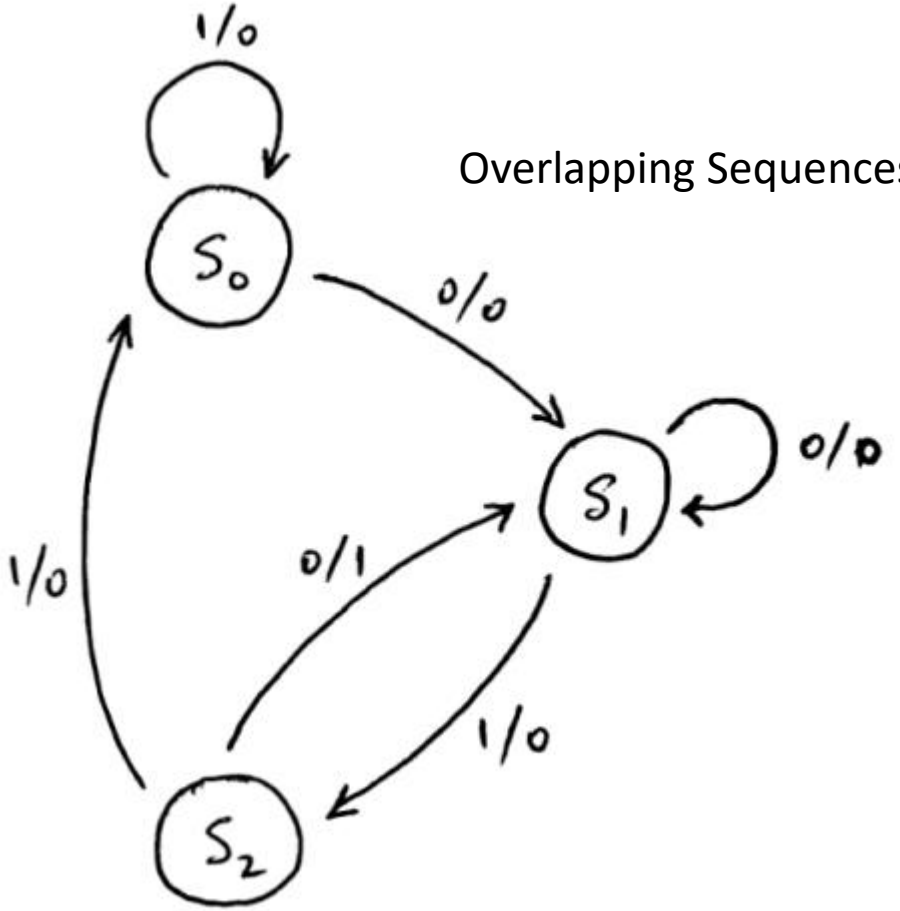
FSM Design Example 3 (Mealy)

Comparison:

Non-Overlapping Sequences

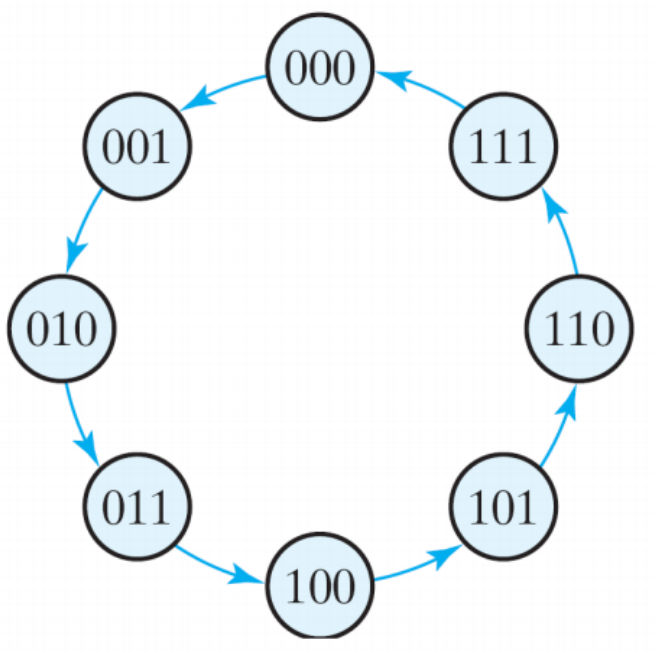


Overlapping Sequences



FSM Design Example-1

Design a 3-bit counter



FSM Design Example

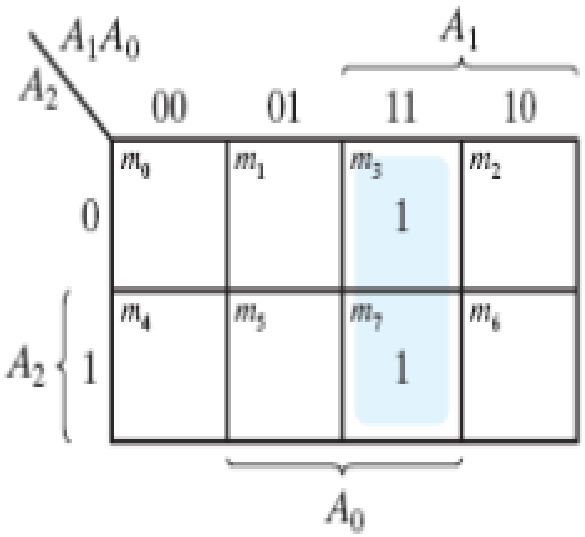
Present State			Next State		
A_2	A_1	A_0	A_2	A_1	A_0
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

FSM Design Example

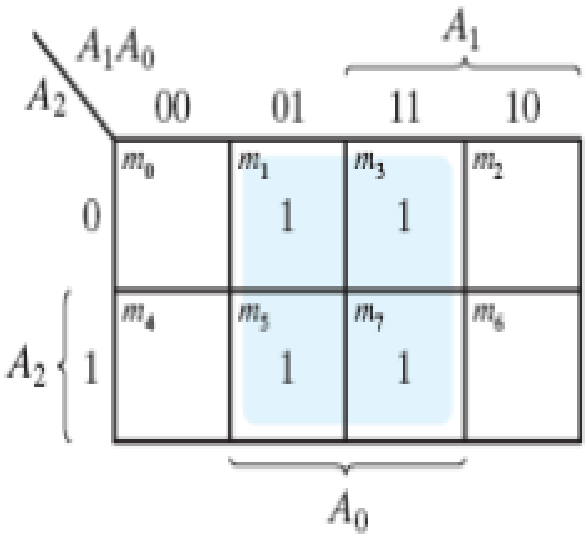
Present State			Next State			FF-Inputs		
A2	A1	A0	A2+	A1+	A0+	TA2	TA1	TA0
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	0	1
1	1	1	0	0	0	1	1	1

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

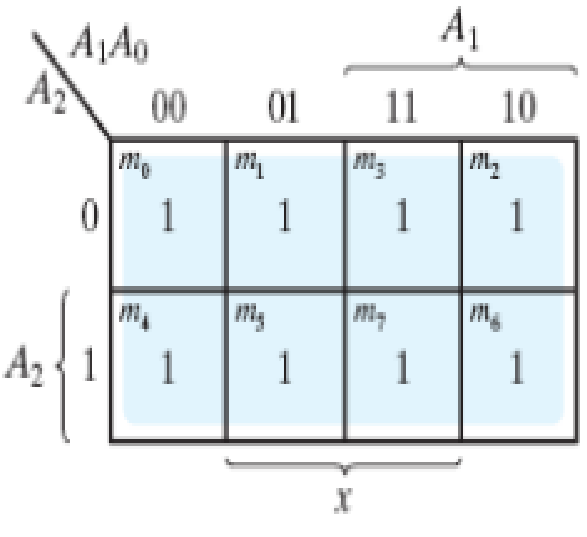
FSM Design Example



$T_{A2} = A_1A_0$



$T_{A1} = A_0$



$T_{A0} = 1$

FSM Design Example

TA0=1

TA1=A0

TA2=A0. A1