CSE 460: VLSI Design

Lecture 9: Finite State Machines (part 2)

Example 2: A Simple Input Pattern (‘11’ Overlapping Sequence) Detection Circuit (Mealy Type)

Clock cycle:	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
w:	0	1	0	1	1	0	1	1	1	0	1
z:	0	0	0	0	1	0	0	1	1	0	0

Figure 6.22 Sequences of input and output signals.

Steps->

- ➢ State diagram
- ➢ State table
- ➢ State assigned table
- ➢ K-map
- ➢ Circuit

Example 2: A Simple Input Pattern (‘11’ Overlapping Sequence) Detection Circuit (Mealy Type)

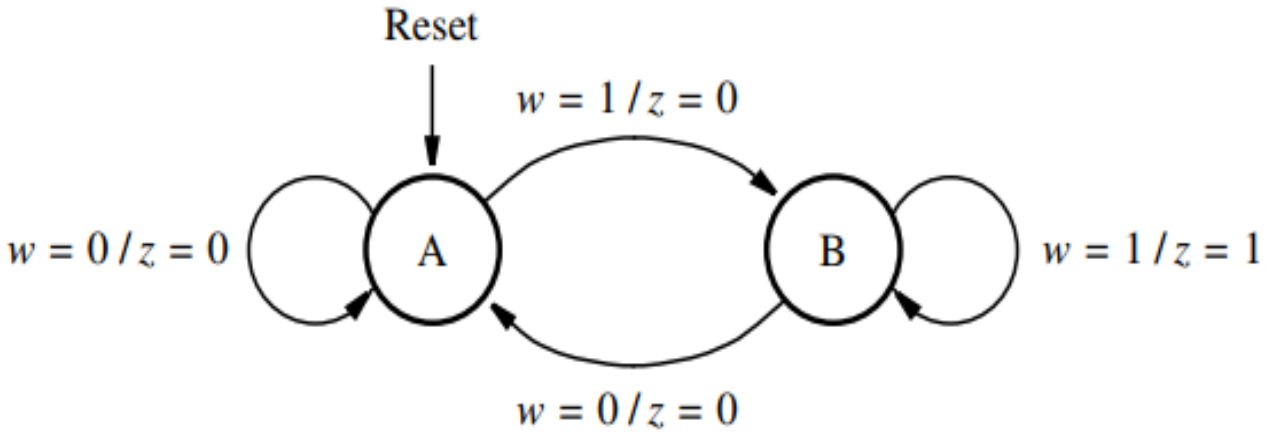


Figure 6.23 State diagram of an FSM that realizes the task in Figure 6.22.

Steps->

- ➢ **State diagram**
- ➢ State table
- ➢ State assigned table
- ➢ K-map
- ➢ Circuit

Example 2: A Simple Input Pattern (‘11’ Overlapping Sequence) Detection Circuit (Mealy Type)

Present state	Next state		Output z	
	$w = 0$	$w = 1$	$w = 0$	$w = 1$
A	A	B	0	0
B	A	B	0	1

Figure 6.24 State table for the FSM in Figure 6.23.

Steps->

- ➢ State diagram
- ➢ **State table**
- ➢ **State assigned table**
- ➢ K-map
- ➢ Circuit

	Present state	Next state		Output	
		$w = 0$	$w = 1$	$w = 0$	$w = 1$
	y	Y	Y	z	z
A	0	0	1	0	0
B	1	0	1	0	1

Figure 6.25 State-assigned table for the FSM in Figure 6.24.

Note that,

- ➢ Y = f(w,y)
- ➢ z = f(w,y)

Example 2: A Simple Input Pattern (‘11’ Overlapping Sequence) Detection Circuit (Mealy Type)

Y =>

y \ w	0	1
0	0	1
1	0	1

Y = w

z =>

y \ w	0	1
0	0	0
1	0	1

z = wy

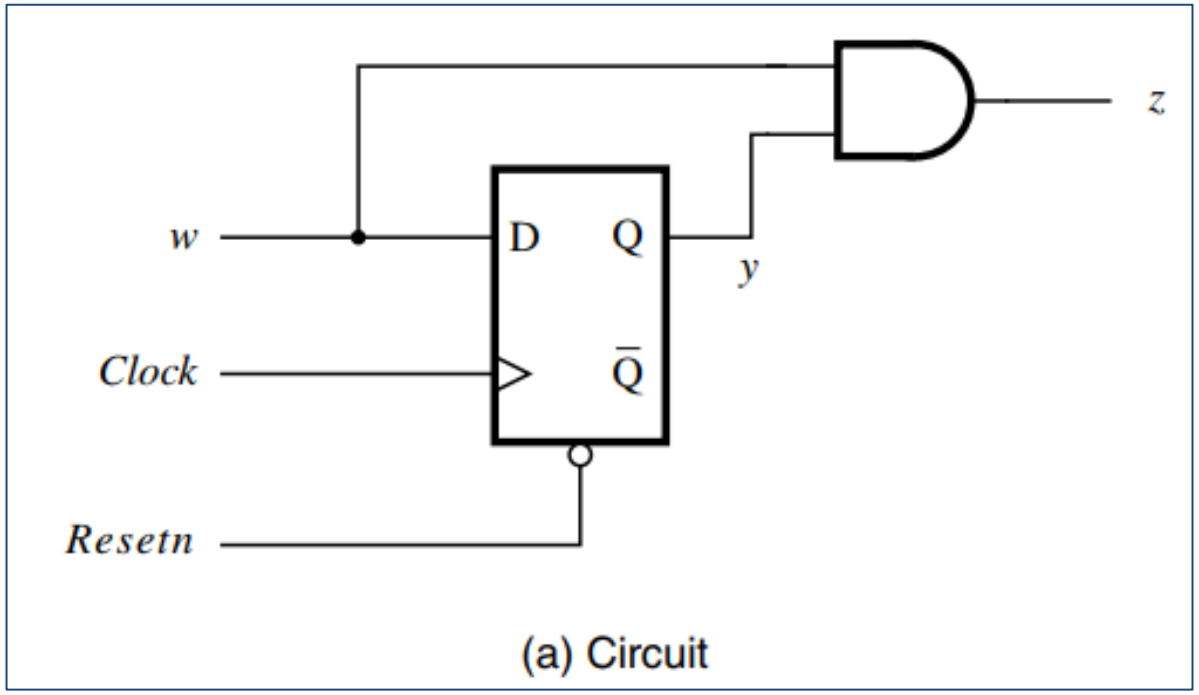
Steps->

- ➢ State diagram
- ➢ State table
- ➢ State assigned table
- ➢ **K-map** (Y and z)
- ➢ Circuit

Note that,

- ➢ Y = f(w,y)
- ➢ z = f(w,y)

Example 2: A Simple Input Pattern (‘11’ Overlapping Sequence) Detection Circuit (Mealy Type)



(a) Circuit

Steps->

- ➢ State diagram
- ➢ State table
- ➢ State assigned table
- ➢ K-map
- ➢ **Circuit**

Encoding Schemes (State Assignment)

Consider a state assigned table below:

	Present state	Next state		Output
		w=0	*w=1*	
	$y_2 y_1$	$Y_2 Y_1$	$Y_2 Y_1$	z
A	00	00	01	0
B	01	00	10	0
C	10	00	10	1
	11	*dd*	*dd*	*d*

Schemes->

- **Binary encoding**
- Gray encoding
- One-hot encoding

Encoding Schemes (State Assignment)

Consider another state assigned table below:

	Present state	Next state		Output
		w=0	*w=1*	
	$y_2 y_1$	$Y_2 Y_1$	$Y_2 Y_1$	z
A	00	00	01	0
B	01	00	11	0
C	11	00	11	1
	10	*dd*	*dd*	*d*

Schemes->

- Binary encoding
- **Gray encoding**
- One-hot encoding

Decimal Number	4 bit Binary Number ABCD	4 bit Gray Code $G_1G_2G_3G_4$
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	0 1 0 0
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1
10	1 0 1 0	1 1 1 1
11	1 0 1 1	1 1 1 0
12	1 1 0 0	1 0 1 0
13	1 1 0 1	1 0 1 1
14	1 1 1 0	1 0 0 1
15	1 1 1 1	1 0 0 0

Encoding Schemes (State Assignment)

Consider another state assigned table below:

	Present state	Next state		Output
		$w = 0$	$w = 1$	z
	$y_3 y_2 y_1$	$Y_3 Y_2 Y_1$	$Y_3 Y_2 Y_1$	
A	0 0 1	0 0 1	0 1 0	0
B	0 1 0	0 0 1	1 0 0	0
C	1 0 0	0 0 1	1 0 0	1

Schemes->

- ➢ Binary encoding
- ➢ Gray encoding
- ➢ **One-hot encoding**

How many flipflops are required? **Ans: 3**