COE328
Digital Systems

Lecture 16

Dr. Shazzat Hossain

Example 1

8.2 Derive a circuit that realizes the FSM defined by the state-assigned table in Figure P8.1 using JK flip-flops.

Present state	Next state $w=0$	Next state $w=1$	Output z
A y_2y_1 = 00	00	11	0
B = 11	11	00	0
C = 10	01	00	0
D = 01	10	01	1

state-assigned Table

Present state	Flip-flop inputs $w=0$		$w=1$		Output z
y_2y_1	J_2K_2	J_1K_1	J_2K_2	J_1K_1	
00	1d	0d	1d	1d	0
01	0d	0d	0d	d1	0
10	d0	1d	d1	0d	0
11	d0	d1	d1	d0	1

state-excitation table

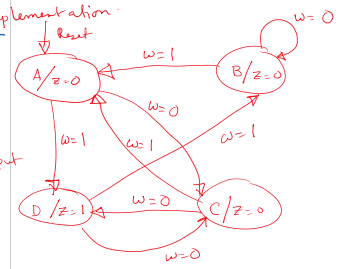
Present State	Next State D-FF $w=0$	Next State D-FF $w=1$	J-K FF	T FF
A 0	0	1	0d	0
B 1	1	0	d1	1
C 2	0	0	0d	1
D 3	1	0	d1	1

you should have this

1. State-diagram
2. State-Table
3. State-excitation table — state-assigned Table same → D FF
4. logic function
5. Implementation

Reset

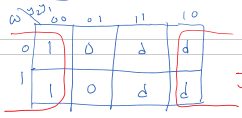
$J_2 = \bar{y}_1$
$K_2 = w$
$J_1 = \bar{w}y_2 + w\bar{y}_2$
$K_1 = \bar{J}_1 = \bar{w}y_2 + w\bar{y}_2$
$z = y_1y_2$



Present state	Next state $w=0$	Next state $w=1$	Output
A	C	D	0
B	B	A	0
C	D	A	0
D	C	B	1

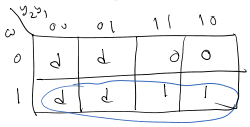
$Z = y_2y_1$

For J_2



$J_2 = \bar{y}_1$

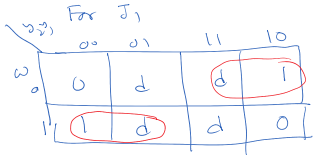
For K_2



$K_2 = w$

Example 1: Implement

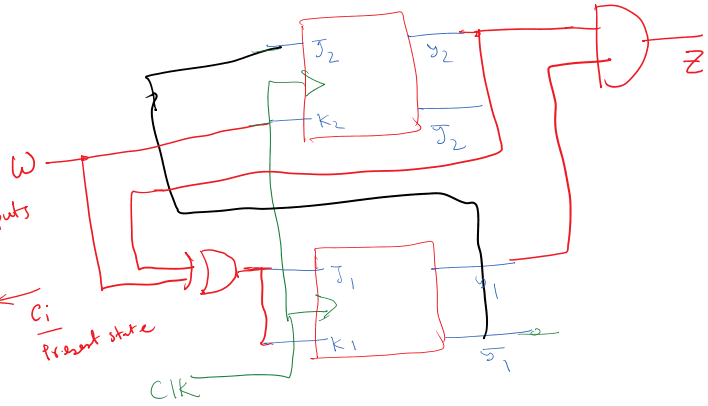
For J_1



$J_1 = \bar{w}y_2 + w\bar{y}_2 = w \oplus y_2$

$J_2 = \bar{y}_1 \qquad K_2 = w$

$J_1 = K_1 = \bar{w}y_2 + w\bar{y}_2 = w \oplus y_2$



Mealy-Type FSM for Serial Adder

8.5.1 Mealy-Type FSM for Serial Adder

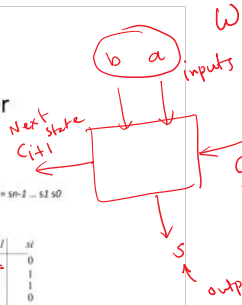
Mealy model

A = $a_{n-1} \ldots a_1\ a_0$ and B = $b_{n-1} \ldots b_1\ b_0$ are two assigned numbers with sum of S = $s_{n-1} \ldots s_1\ s_0$ and the states **G**: carry in = 0, **H**: carry in = 1

$A = a_4\ a_3\ a_2\ a_1\ a_0$ 01100
$B = b_4\ b_3\ b_2\ b_1\ b_0$ 01010 +
1000 + ← Carries
$S = s_4\ s_3\ s_2\ s_1\ s_0$ 00110

c_i	a	b	c_{i+1}	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

A ← G
B ← H



a/s 00/0 01/1 Reset 1

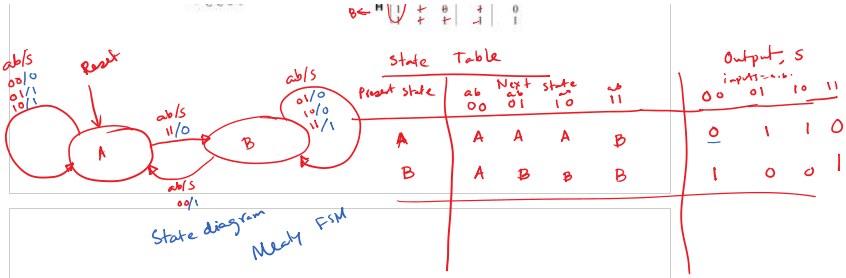
State Table

a/s 0/0 1/1

Present state	a/b 00	01	10	11
G	G/0	G/1	H/0	G/1
H	H/0	H/1	H/1	H/0

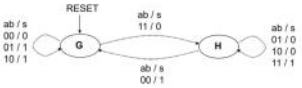
Output S

inputs a/b	00	01	10	11
	0	1	1	0



Serial Adder

	ci	a	b	$ci+1$	si
G	0	0	0	0	0
	0	0	1	0	1
	0	1	0	0	1
	0	1	1	1	0
H	1	0	0	0	1
	1	0	1	1	0
	1	1	0	1	0
	1	1	1	1	1



Present state	Next state *ab* 00	01	10	11	Output *s* 00	01	10	11
G	G	G	G	H	0	1	1	0
H	G	H	H	H	1	0	0	1

State table

Serial Adder

A → 0
B → 1

Present state	Next state *ab* 00	01	10	11	Output *s* 00	01	10	11
G	G	G	G	H	0	1	1	0
H	G	H	H	H	1	0	0	1

State table

Present state y	Next state Y 00	01	10	11	Output s 00	01	10	11
0	0	0	0	1	0	1	1	0
1	0	1	1	1	1	0	0	1

State assigned table

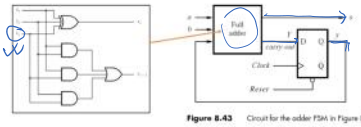
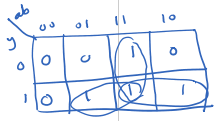
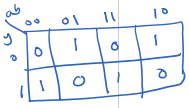


Figure 8.43 Circuit for the adder FSM in Figure 8.39.



$Y = ab + yb + ay$



$S = a \oplus b \oplus y$

Example 2: Mixed Flip-Flops

$w(\bar{y}_1 + y_2)$

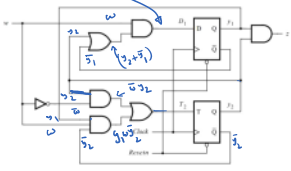


Figure 8.84 Circuit for Example 8.10.

Present state y_2y_1	Flip-flop inputs $w=0$ T_2D_1	$w=1$ T_2D_1	Output z
0 0	0 0	0 1	0
0 1	0 0	1 0	0
1 0	0 0	1 1	0
1 1	1 0	0 1	1

Figure 8.85 The excitation table for the circuit in Figure 8.84.

Logic Functions

$D_1 = w(\bar{y}_1 + y_2)$
$T_2 = \bar{w}y_2 + wy_1\bar{y}_2$
$z = y_1y_2$ ← States

Moore FSM

$y_2 = 0$
$y_1 = 0$
$w = 0$

$D_1 = \bar{w}\cdot(\bar{y}_1 + y_2)$ $\quad D_1 = \bar{y}_1 + y_2$

$T_2 =$

$T_2 = \bar{w}\,y_2 + w\,y_1\bar{y}_2$
$= \bar{0}\cdot y_2 + 0$
$T_2 = y_2$ if $w=0$

Example 2

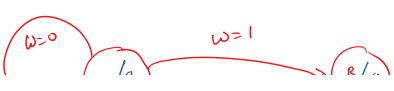
Present state y_2y_1	Next State $w=0$ Y_2Y_1	$w=1$ Y_2Y_1	Output z
0 0	0 0	0 1	0
0 1	0 0	1 0	0
1 0	0 0	1 1	0
1 1	0 0	1 1	1

A
B
C

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

(b) State table

y	Y	T
0	0	0
0	1	1
1	0	1
1	1	0



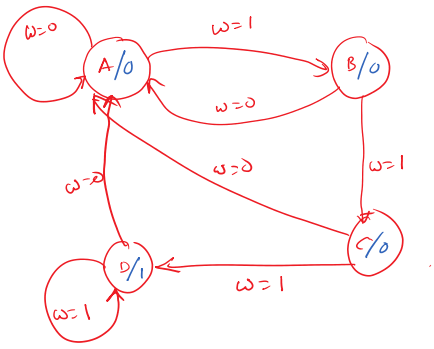
y_2y_1	Y_2Y_1	Y_2Y_1	z
00	00	01	0
01	00	10	0
10	00	11	0
11	00	11	1

(a) State-assigned table

A	A	B	0
B	A	C	0
C	A	D	0
D	A	D	1

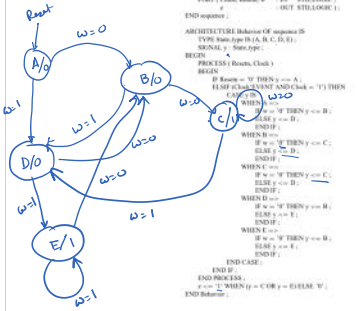
(b) State table

Figure 8.81 Tables for the circuit in Figure 8.80.



Example 3

VHDL code




```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;

ENTITY sequence IS
    PORT ( Clock, Resetn, w    : IN    STD_LOGIC ;
           z                   : OUT   STD_LOGIC ) ;
END sequence ;

ARCHITECTURE Behavior OF sequence IS
    TYPE State_type IS (A, B, C, D, E) ;
    SIGNAL y : State_type ;
BEGIN
    PROCESS ( Resetn, Clock )
    BEGIN
        IF Resetn = '0' THEN y <= A ;
        ELSIF (Clock'EVENT AND Clock = '1') THEN
            CASE y IS
                WHEN A =>
                    IF w = '0' THEN y <= B ;
                    ELSE y <= A ;
                    END IF ;
                WHEN B =>
                    IF w = '0' THEN y <= C ;
                    ELSE y <= D ;
                    END IF ;
                WHEN C =>
                    IF w = '0' THEN y <= C ;
                    ELSE y <= D ;
                    END IF ;
                WHEN D =>
                    IF w = '0' THEN y <= B ;
                    ELSE y <= E ;
                    END IF ;
                WHEN E =>
                    IF w = '0' THEN y <= B ;
                    ELSE y <= E ;
                    END IF ;
            END CASE ;
        END IF ;
    END PROCESS ;
    z <= '1' WHEN (y = C OR y = E) ELSE '0' ;
END Behavior ;
```

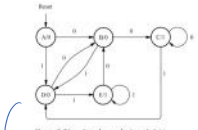


Figure 8.91 State diagram for Example 8.11.

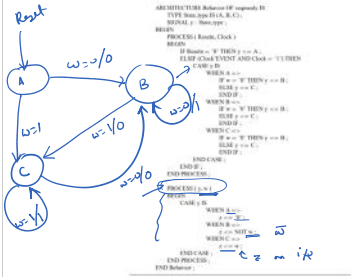
Present state	Next state $w=0$	Next state $w=1$	Output z
A	B	D	0
B	C	D	0
C	C	D	1
D	B	E	0
E	B	E	1

Figure 8.92 State table for the FSM in Figure 8.91.

State-assigned table

Example 4

Find the state diagram and State Table




```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;

ENTITY sequence IS
    PORT ( Clock, Resetn, w    : IN    STD_LOGIC ;
           z                   : OUT   STD_LOGIC ) ;
END sequence ;

ARCHITECTURE Behavior OF sequence IS
    TYPE State_type IS (A, B, C) ;
    SIGNAL y : State_type ;
BEGIN
    PROCESS ( Resetn, Clock )
    BEGIN
        IF Resetn = '0' THEN y <= A ;
        ELSIF (Clock'EVENT AND Clock = '1') THEN
            CASE y IS
                WHEN A =>
                    IF w = '0' THEN y <= B ;
                    ELSE y <= C ;
                    END IF ;
                WHEN B =>
                    IF w = '0' THEN y <= B ;
                    ELSE y <= C ;
                    END IF ;
                WHEN C =>
                    IF w = '0' THEN y <= B ;
                    ELSE y <= C ;
                    END IF ;
            END CASE ;
        END IF ;
    END PROCESS ;
    z <= '1' WHEN y = C ELSE '0' ;
END Behavior ;
```

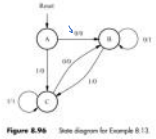


Figure 8.96 State diagram for Example 8.13.

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

Figure 8.97 State table for the FSM in Figure 8.96.