Digital System Design with HDL (I)

Lecture 11

Dr. Ming Xu

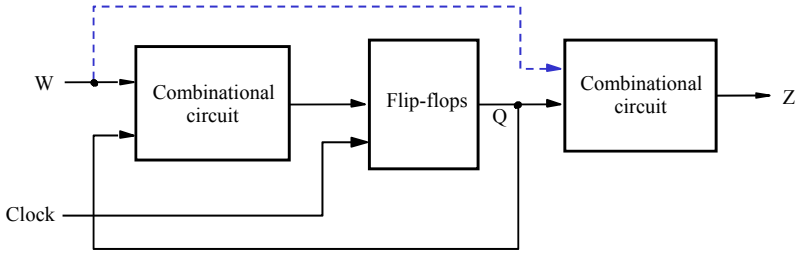
Dept of Electrical & Electronic Engineering

XJTLU

In This Session

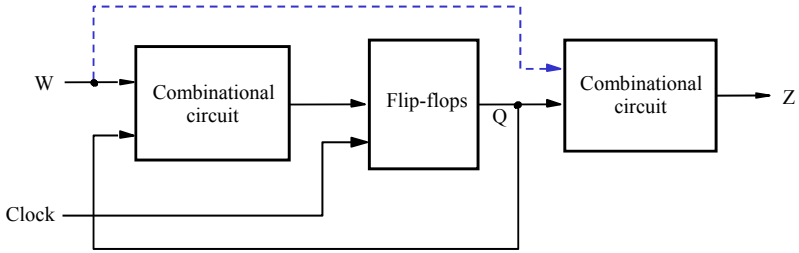
- Moore-Type FSMs
- Mealy-Type FSMs
- Verilog Code for FSMs

General Form of A Sequential Circuit.



- Sequential circuits are called **finite state machines (FSM)**.
- Combinational circuit 1 has inputs from the input W and the state Q of the flip-flops.
- The output Z always depends on the state Q of the flip-flops. It may also depend on the input W.

General Form of A Sequential Circuit.



- The sequential circuits whose outputs depend only on the state of the circuit are of **Moore type**.
- Those whose outputs depend on both the state and the inputs are of **Mealy type**.

Moore State Model.

We wish to design such a circuit:

- The circuit has one input w and one output z.
- All changes in the circuit occur on the positive edge of a clock signal.
- The output z is 1 if during the past two clock cycles w was 1. Otherwise z is 0.

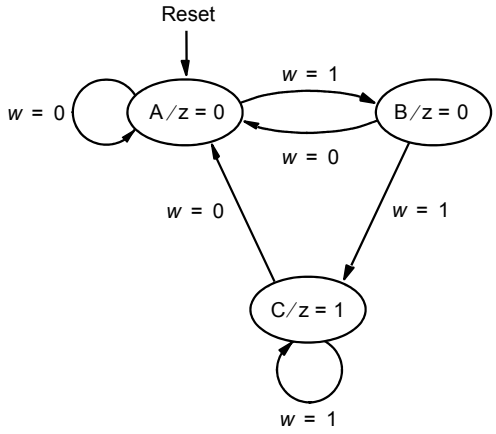
Clockcycle:	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
w:	0	1	0	1	1	0	1	1	1	0	1
z:	0	0	0	0	0	1	0	0	1	1	0

Sequences of input and output signals.

Moore State Model.

State Diagram

- State A: w is 0 during past 1 clock cycle.
- State B: w has been 1 for just 1 clock cycle.
- State C: w has been 1 for 2 clock cycles.



Moore State Model.

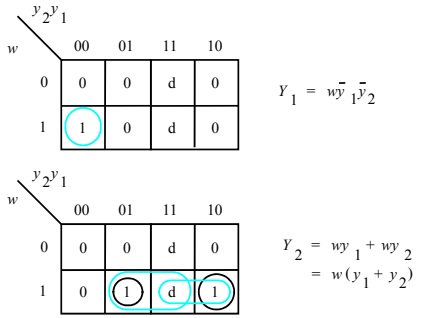
State Table

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

	Present state y_2y_1	Next state $w=0$ Y_2Y_1	Next state $w=1$ Y_2Y_1	Output z
A	00	00	01	0
B	01	00	10	0
C	10	00	10	1
	11	*dd*	*dd*	*d*

Moore State Model.

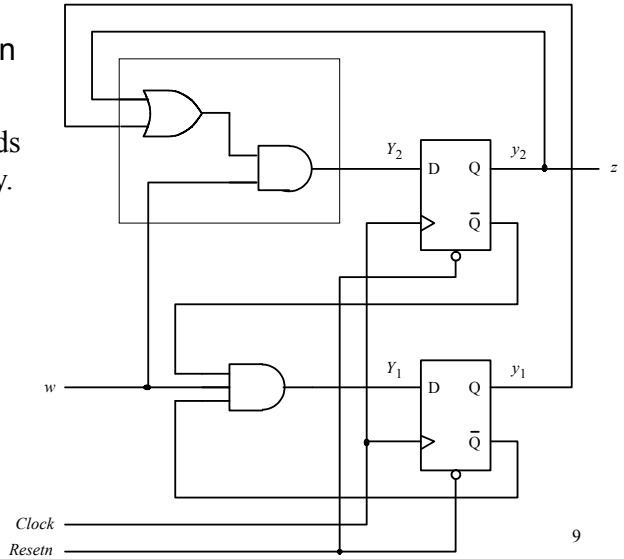
Next-State and Output Expressions



Moore State Model.

Implementation

Output z depends on the state only.



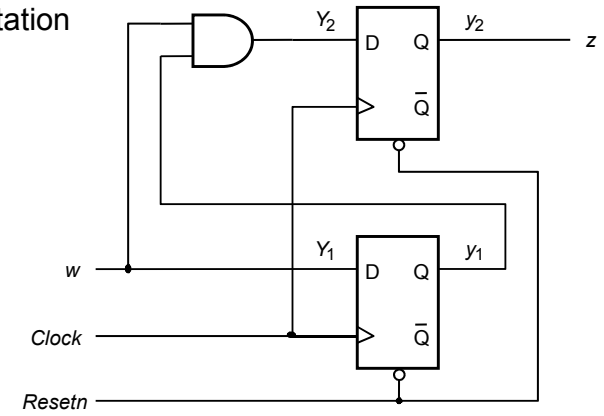
Moore State Model.

Alternative State Assignment

	Present state y_2y_1	Next state $w = 0$ Y_2Y_1	Next state $w = 1$ Y_2Y_1	Output z
A	00	00	01	0
B	01	00	11	0
C	11	00	11	1
	10	*dd*	*dd*	*d*

Moore State Model.

Improved Implementation



Mealy State Model.

We wish to design such a circuit:

- The circuit has one input w and one output z.
- The output z is 1 in the clock cycle when the second occurrence of w =1 is detected. Otherwise z is 0.

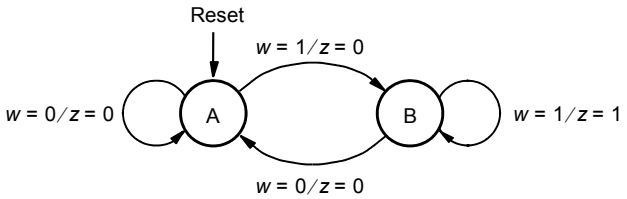
Clock cycle:	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
w:	0	1	0	1	1	0	1	1	1	0	1
z:	0	0	0	0	1	0	0	1	1	0	0

Sequences of input and output signals.

Mealy State Model.

State Diagram

- State A: w is 0, producing an output z = 0 .
- State B: w is 1.
- If w = 1 for two consecutive clock cycles, the machine remains in state B and produce an output z = 1.



Mealy State Model.

State Table

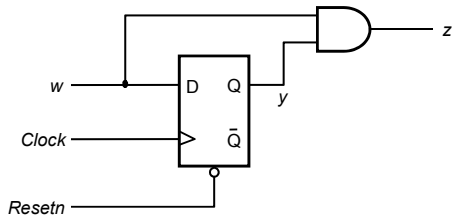
Present state	Next state		Output z	
	$w = 0$	$w = 1$	$w = 0$	$w = 1$
A	A	B	0	0
B	A	B	0	1

	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

Mealy State Model.

Implementation

$$Y = D = w$$
$$z = wy$$

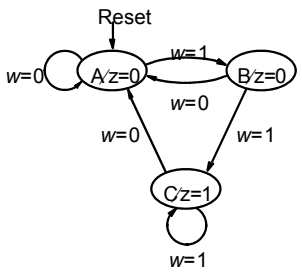


Output z depends on both the state and the input.

Verilog Code for the Moore FSM.

```
module simple (Clock, Resetn, w, z);
    input Clock, Resetn, w;
    output z;
    reg [2:1] y, Y;
    parameter [2:1] A = 2'b00, B = 2'b01, C =
2'b10;

    // Define the next state combinational
circuit
    always @(w, y)
        case (y)
            A: if (w) Y = B;
               else   Y = A;
            B: if (w) Y = C;
               else   Y = A;
            C: if (w) Y = C;
               else   Y = A;
            default:  Y = 2'bxx;
        endcase
```

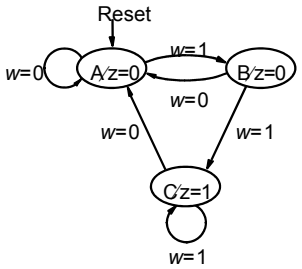



```
    // Define the sequential block
    always @(negedge Resetn, posedge Clock)
        if (Resetn = = 0)        y <= A;
        else                     y <= Y;

    // Define output
    assign z = (y = = C);

endmodule
```

Verilog Code for the Moore FSM.




```
module simple (Clock, Resetn, w, z);
    input Clock, Resetn, w;
    output z;
    reg [2:1] y;
    parameter [2:1] A = 2'b00, B = 2'b01, C = 2'b10;

    // Define the sequential block
    always @(negedge Resetn, posedge Clock)
        if (Resetn = = 0)        y <= A;
        else
            case (y)
                A: if (w) y <= B;
                   else   y <= A;
                B: if (w) y <= C;
                   else   y <= A;
                C: if (w) y <= C;
                   else   y <= A;
                default:  y <= 2'bxx;
            endcase

    // Define output
    assign z = (y = = C);

endmodule
```

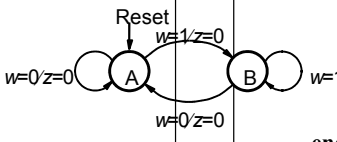

Verolig Code for the Mealy FSM.

```
module mealy (Clock, Resetn, w, z);
    input Clock, Resetn, w;
    output reg z;
    reg y, Y;
    parameter A = 1'b0, B = 1'b1;

    always @(w, y)
        case (y)
            A:  if (w)
                begin
                    z = 0;
                    Y = B;
                end
                else
                begin
                    z = 0;
                    Y = A;
                end
            B:  if (w)
                begin
                    z = 1;
                    Y = B;
                end
                else
                begin
                    z = 0;
                    Y = A;
                end
        endcase

    // Define the sequential block
    always @(negedge Resetn, posedge
Clock)
        if (Resetn = = 0) y <= A;
        else      y <= Y;

endmodule
```

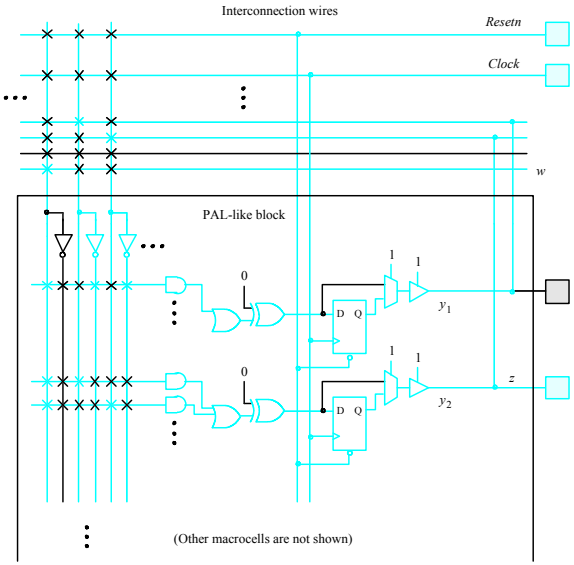


Verilog Code for the FSM.

State Assignment

- State assignments are specified by a **parameter** statement in Verilog code.
- Verilog compilers can recognize the code for an FSM.
- They can optimize the implementation by looking for a better state assignment based on the cost of implementation.
- The user can either allow the compiler to optimize the state assignment or suppress it.

Verilog Code for the FSM.



- The code for the Moore one is synthesised in a CPLD.
- The used parts are highlighted in blue.

$$Y_1 = w\bar{y}_1\bar{y}_2$$

$$Y_2 = wy_1 + wy_2 = w(y_1 + y_2)$$

$$z = y_2$$