DSLab. 13 Design of Synchronous Sequential Logic

Lab. 13 Design of Synchronous Sequential Logic

- Design and Verify the following circuits using Verilog HDL
  - Design the sequence detector with D flip-flop in Fig. 5.27
  - Design the sequence detector with T flip-flop
- Verilog
  - Behavioral level modeling
  - Dataflow modeling
  - Structural level (Gate-level) modeling
- Please write and upload the lab report (Lab13) -- Due on 2022/12/02 23:59

Example 1: State Diagram-Based HDL Model (Mealy) (1/3)

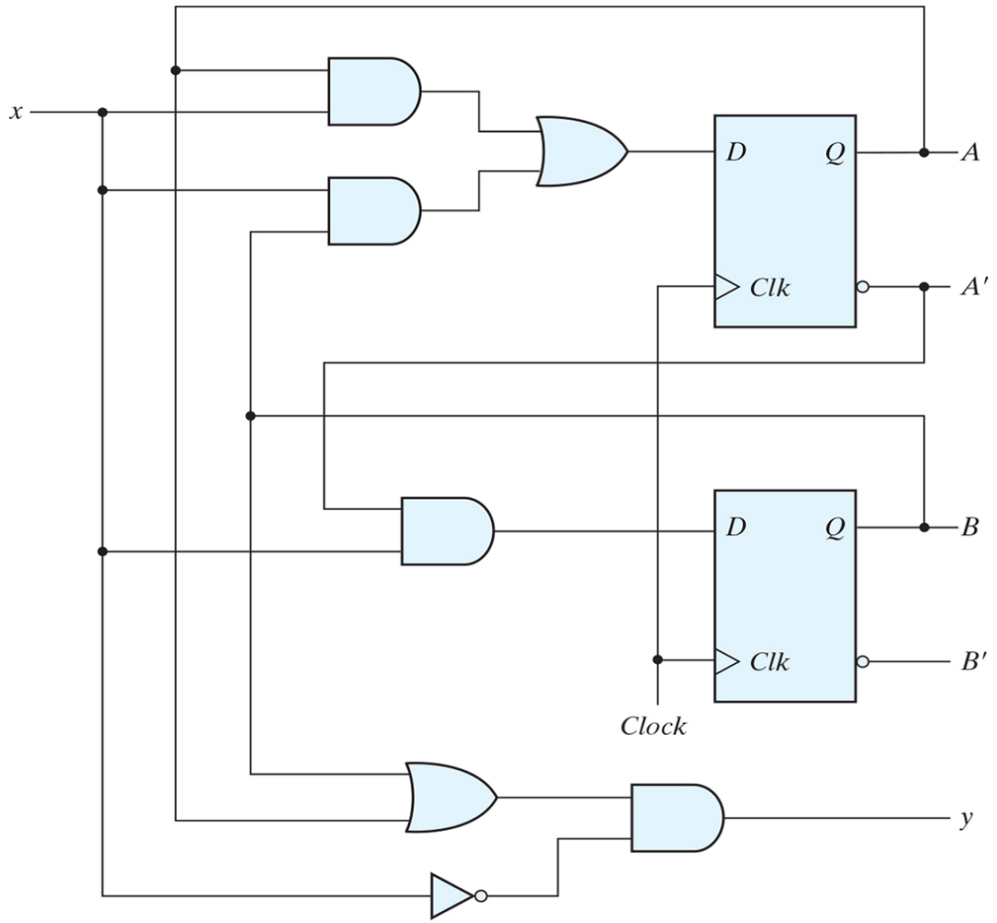


Fig. 5.15

Table 5.2
State Table for the Circuit of Fig. 5.15

Present State		Input	Next State		Output
A	**B**	**x**	**A**	**B**	**y**
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

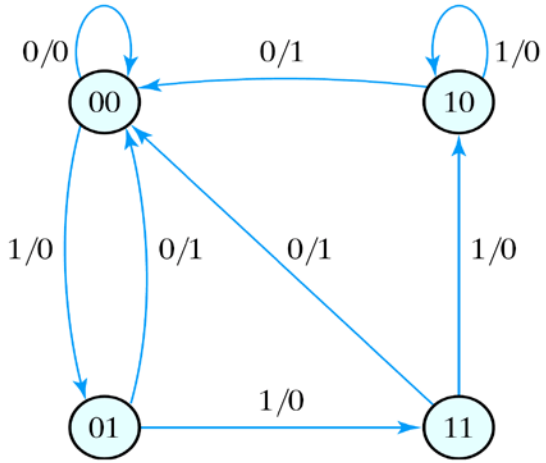


Fig. 5.16

Example 1: State Diagram-Based HDL Model (Mealy) (2/3)

Behavioral model (State diagram)

```
module Mealy_Zero_Detector (
  output reg y_out,
  inputx_in, clock, reset
);
  reg [1: 0]    state, next_state;
  parameter  S0 = 2'b00, S1 = 2'b01, S2 = 2'b10, S3 = 2'b11;

  always @ (posedge clock, negedge reset)  // state transition
    if (reset == 0) state <= S0;
    else state <= next_state;

  always @ (state, x_in) // Form the next state
    case (state)
      S0: if (x_in)   next_state = S1; else next_state = S0;
      S1: if (x_in)   next_state = S3; else next_state = S0;
      S2: if (~x_in) next_state = S0; else next_state = S2;
      S3: if (x_in)   next_state = S2; else next_state = S0;
    endcase

  always @ (state, x_in) // Form the output
    case (state)
      S0: y_out = 0;
      S1, S2, S3: y_out = ~x_in;
    endcase
endmodule
```

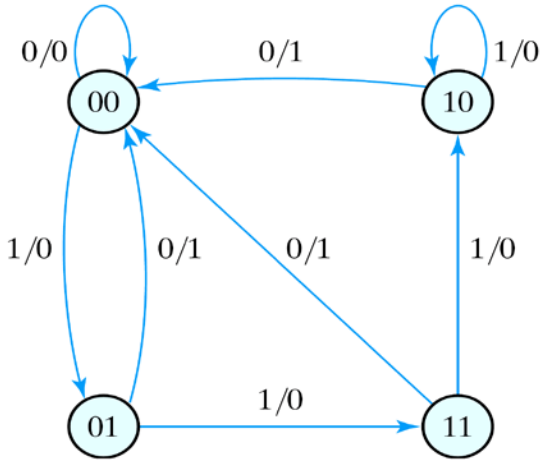


Fig. 5.16

Table 5.2
State Table for the Circuit of Fig. 5.15

Present State		Input	Next State		Output
A	*B*	*x*	*A*	*B*	*y*
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

Example 1: State Diagram-Based HDL Model (Mealy) (3/3)

```
module t_Mealy_Zero_Detector;
  wire  t_y_out;
  reg       t_x_in, t_clock, t_reset;

   Mealy_Zero_Detector M0 (t_y_out, t_x_in, t_clock, t_reset);

   initial #200 $finish;
   initial begin t_clock = 0; forever #5 t_clock = ~t_clock; end

   initial fork
         t_reset = 0;
     #2 t_reset = 1;
     #87 t_reset = 0;
     #89 t_reset = 1;
     #10 t_x_in = 1;
     #30 t_x_in = 0;
     #40 t_x_in = 1;
     #50 t_x_in = 0;
     #52 t_x_in = 1;
     #54 t_x_in = 0;
     #70 t_x_in = 1;
     #80 t_x_in = 0;
     #90 t_x_in = 1;
     #100 t_x_in = 0;
     #120 t_x_in = 1;
     #160 t_x_in = 0;
     #170 t_x_in = 1;
   join

endmodule
```

Example 2: State Diagram-Based HDL Model (Moore) (1/4)

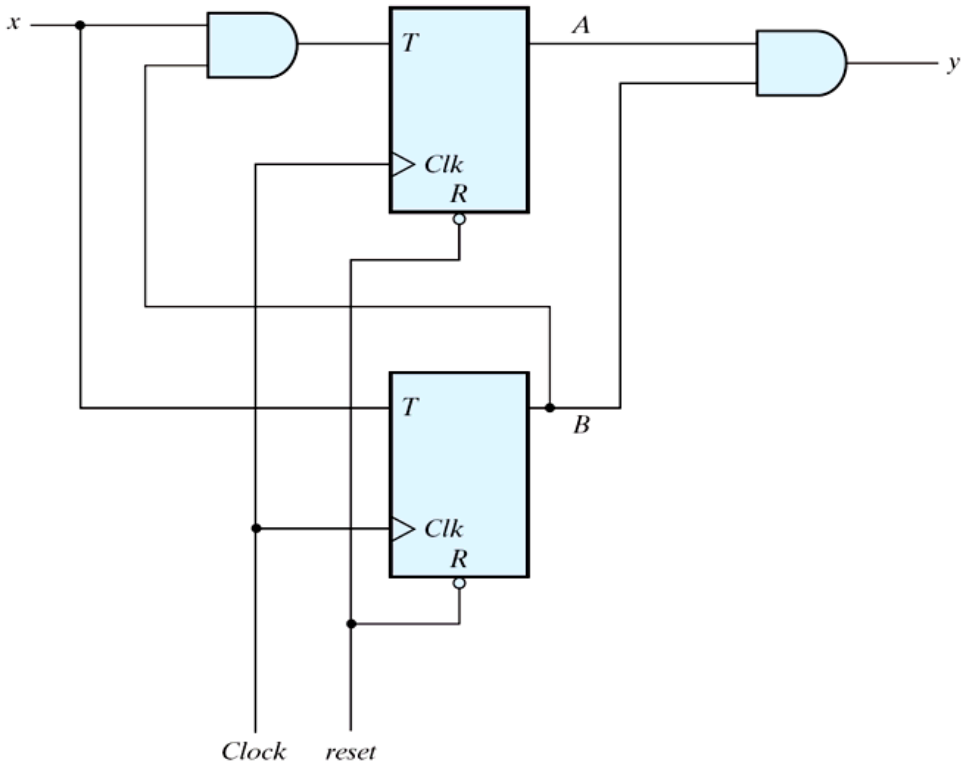
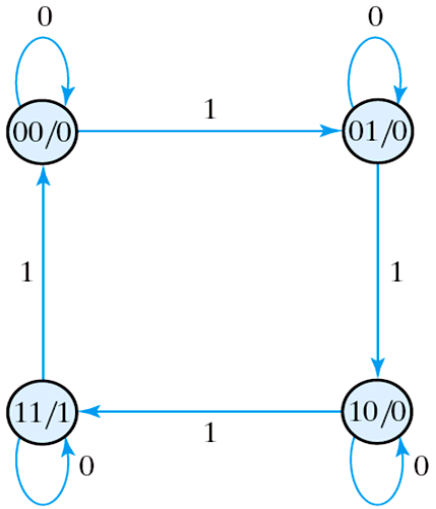


Fig. 20

Present State		Input	Next State		Output
A	B	x	A	B	y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	1



Example 2: State Diagram-Based HDL Model (Moore) (2/4)

Structural model (Fig. 20)

```
module Moore_Model_STR_Fig_5_20 (
   output   y_out, A, B,
   input    x_in, clock, reset
);
  wire      A, TB;

// Flip-flop input equations
    assign TA = x_in & B;
    assign TB = x_in;
//output equation
    assign y_out = A & B;
// Instantiate Toggle flip-flops
    Toggle_flip_flop_3 M_A (A, TA, clock, reset);
    Toggle_flip_flop_3 M_B (B, TB, clock, reset);

endmodule
```

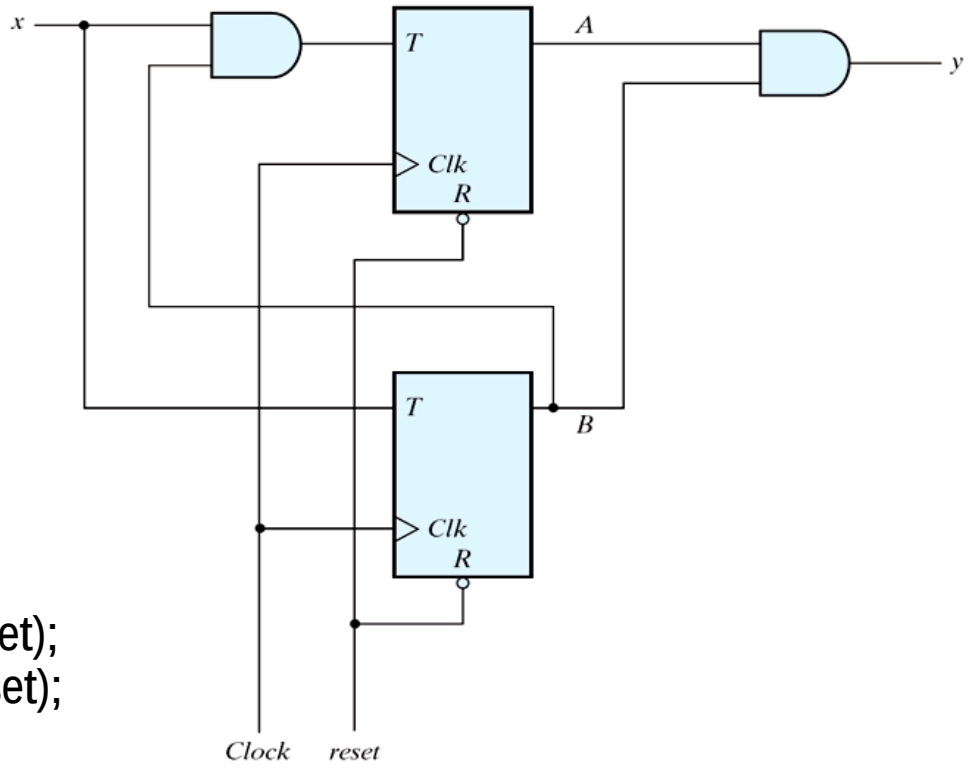


Fig. 20

Example 2: State Diagram-Based HDL Model (Moore) (3/4)

Behavioral model (State diagram)

```
module Moore_Model_Fig_5_20 (
  output   y_out,
  input    x_in, clock, reset
);
  reg [1: 0]   state;
  parameter  S0 = 2'b00, S1 = 2'b01, S2 = 2'b10, S3 = 2'b11;

  always @ (posedge clock, negedge reset)
    if (reset == 0) state <= S0;    // Initialize to state S0
    else case (state)
      S0: if (x_in)  state <= S1; else state <= S0;
      S1: if (x_in)  state <= S2; else state <= S1;
      S2: if (x_in)  state <= S3; else state <= S2;
      S3: if (x_in)  state <= S0; else state <= S3;
    endcase

  assign y_out = (state == S3); // Output of flip-flops
endmodule
```

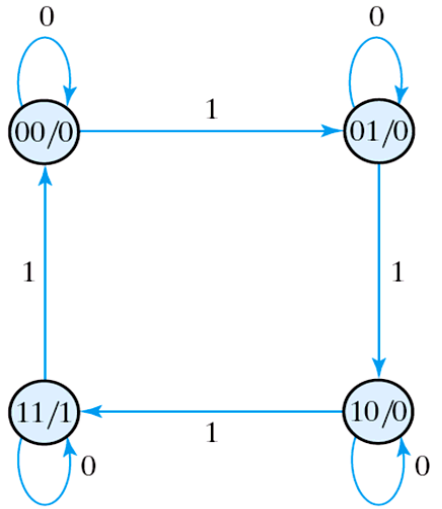


Fig. 5.20

Present State		Input	Next State		Output
A	B	x	A	B	y
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	1

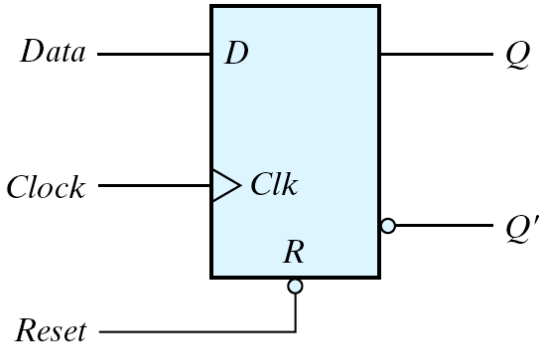
Example 2: State Diagram-Based HDL Model (Moore) (4/4)

```
module t_Moore_Fig_5_20;
  wire       t_y_out_2, t_y_out_1;
  reg        t_x_in, t_clock, t_reset;

  Moore_Model_Fig_5_20  M1  (t_y_out_1, t_x_in, t_clock, t_reset);
  Moore_Model_STR_Fig_5_20  M2  (t_y_out_2, A, B, t_x_in, t_clock, t_reset);

  initial #200 $finish;
  initial begin
        t_reset = 0;
        t_clock = 0;
    #5 t_reset = 1;
    repeat (16)
      #5 t_clock = ~t_clock;
  end
  initial begin
          t_x_in = 0;
    #15 t_x_in = 1;
    repeat (8)
      #10 t_x_in = ~t_x_in;
  end
endmodule
```

Example 3: D Flip-Flop with asynchronous reset

Behavioral Description

```
module D_flip_flop_AR_b (Q, Q_b, D, Clk, rst);
   output            Q, Q_b;
   input  D, Clk, rst;
   reg    Q;

   assign  Q_b = ~Q;
   always @ (posedge Clk, negedge rst)
      if (rst == 0) Q <= 1'b0;
      else Q <= D;
endmodule
```



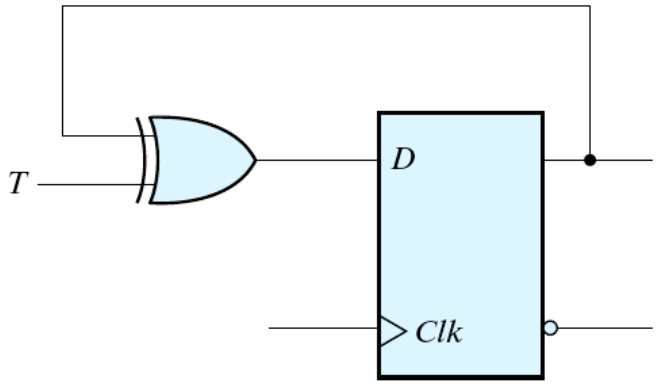
(b) Graphic symbol

R	Clk	D	Q	Q'
0	X	X	0	1
1	↑	0	0	1
1	↑	1	1	0

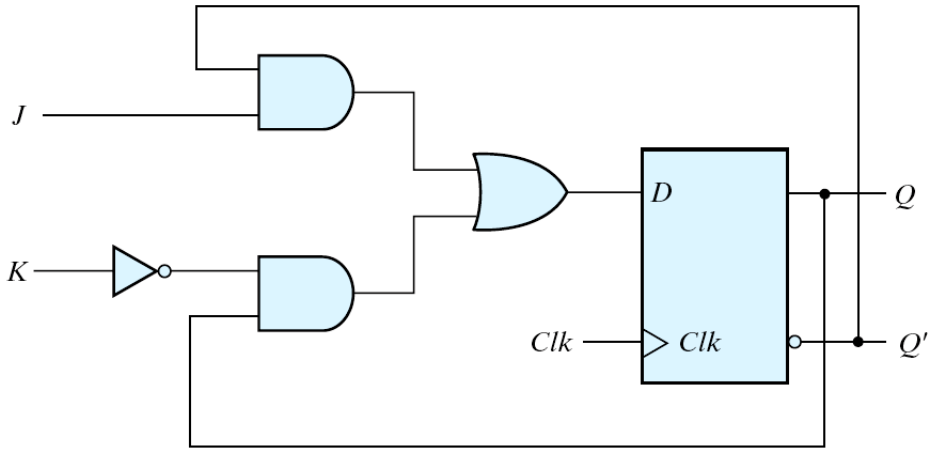
(b) Function table

Characteristic tables and Characteristic equations

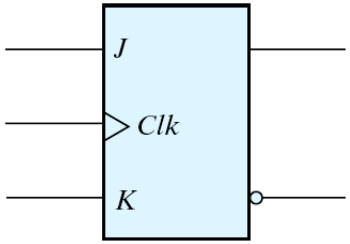
- Characteristic equations
  - D flip-flop
    - $Q(t+1) = D$
  - JK flip-flop
    - $Q(t+1) = JQ'+K'Q$
  - T flop-flop
    - $Q(t+1) = T \oplus Q = TQ'+T'Q$



(b) From *D* flip-flop



(a) Circuit diagram



(b) Graphic symbol

Exercise 1: Design of Sequence Detector (D-FF) (1/2)

- Design a sequence detector to detect a sequence of three or more consecutive 1's in a string of bits coming through an input line
  - The state diagram and state table of the sequence detector are shown in Fig. 5.27 and Table 5.11
  - Design the sequence detector with D flip-flops
  - Write the Verilog HDL description of the state diagram (i.e., Behavioral model)
  - Write the Verilog HDL description of the logic circuit diagram (i.e., a Structural model)
  - Write an Verilog HDL stimulus with a sequence of inputs: 0111110110. Verify that the responses are the same for both descriptions (first reset all flip-flops to 0).

Exercise 1: Design of Sequence Detector (D-FF) (2/2)

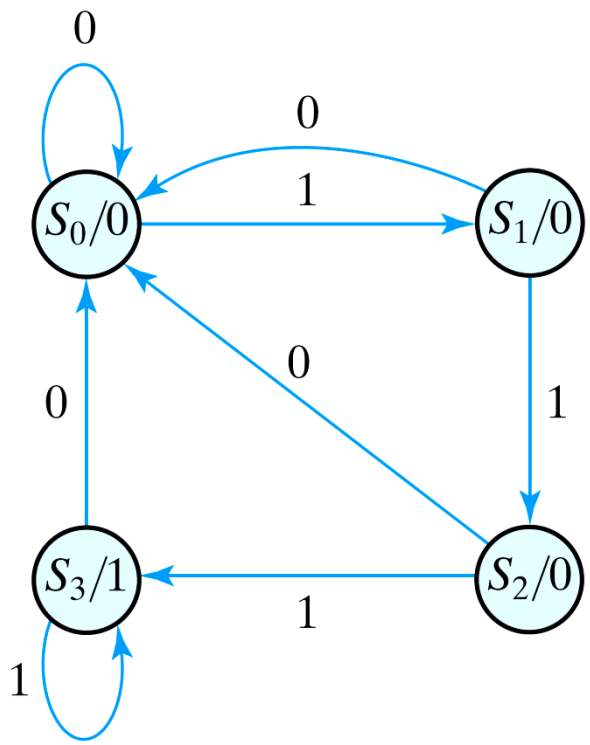


Fig. 5.27 State diagram for sequence detector

Table 5.11
State Table for Sequence Detector

Present State		Input	Next State		Output
A	*B*	*x*	*A*	*B*	*y*
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	0
0	1	1	1	0	0
1	0	0	0	0	0
1	0	1	1	1	0
1	1	0	0	0	1
1	1	1	1	1	1

Exercise 2: Design of Sequence Detector (T-FF)

- Design a sequence detector to detect a sequence of three or more consecutive 1's in a string of bits coming through an input line
  - The state diagram and state table of the sequence detector are shown in Fig. 5.27 and Table 5.11
  - Design the sequence detector with T flip-flops
  - Write the Verilog HDL description of the state diagram (i.e., Behavioral model)
  - Write the Verilog HDL description of the logic circuit diagram (i.e., a Structural model)
  - Write an Verilog HDL stimulus with a sequence of inputs: 0111110110. Verify that the responses are the same for both descriptions (first reset all flip-flops to 0).