CS204: 數位系統設計

Synchronous Sequential Logic

Outline of Chapter 5

- 5.1 Introduction
- 5.2 Sequential Circuits
- 5.3 Storage Element: Latches
- 5.4 Storage Element: Flip-Flops
- 5.5 Analysis of Clocked Sequential Circuits
- **5.6 Synthesizable HDL Models of Sequential Circuits**
- 5.7 State Reduction and Assignment
- 5.8 Design Procedure

Mealy and Moore models

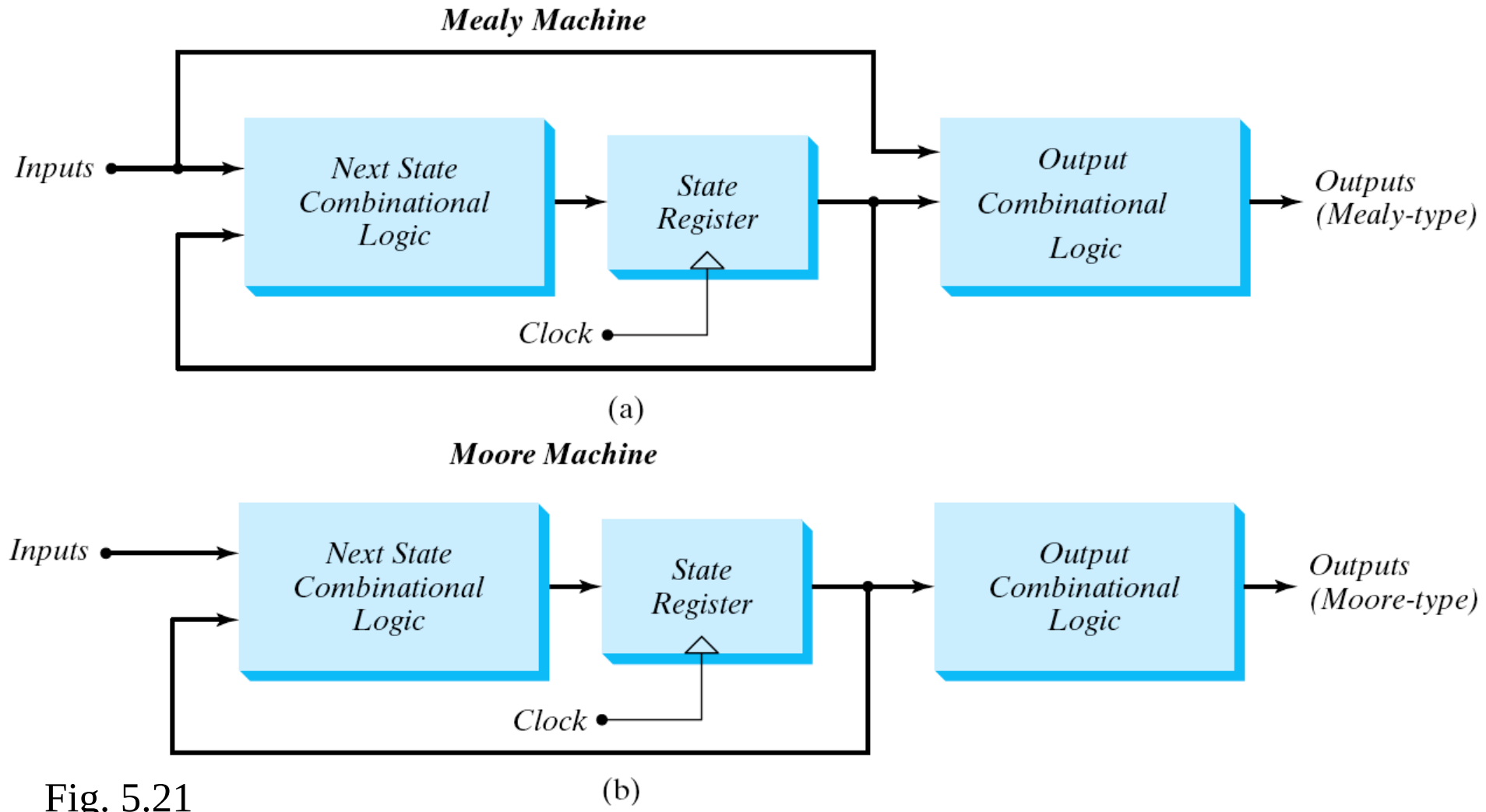

Fig. 5.21
Block diagram of Mealy and Moore state machine

5-7 Synthesizable HDL Models of Sequential Circuits

Behavioral Modeling

Example: Two ways to provide free-running clock

```
initial
  begin
    clock = 1'b0;
    repeat (30)
      #10 clock = ~clock;
  end
```

```
initial
  begin
    clock = 1'b0;
  end

initial 300 $finish;
always #10 clock = ~clock;
```

Example: Another way to describe free-running clock

```
initial begin clock = 0; forever #10 clock = ~clock; end
```

Behavioral Modeling

always statement

```
always @ (event control expression) begin
    // Procedural assignment statements that execute when the condition is met
end
```

Examples:

```
always @ (A or B or C)
```

```
always @(posedge clock or negedge reset)        // Verilog 1995
```

```
always @(posedge clock, negedge reset)          // Verilog 2001, 2005
```

Two procedural blocking assignments:

```
B = A
C = B + 1
```

Two nonblocking assignments:

```
B <= A
C <= B + 1
```

Flip-Flops and Latches

- HDL Example 5.1

HDL Example 5.1

```
// Description of D latch (See Fig. 5.6)
module D_latch (Q, D, enable);
  output Q;
  input  D, enable;
  reg    Q;
  always @ (enable or D)
    if (enable) Q <= D;                    // Same as: if (enable == 1)
  endmodule

// Alternative syntax (Verilog 2001, 2005)
module D_latch (output reg Q, input enable, D);
  always @ (enable, D)
    if (enable) Q <= D;                    // No action if enable not asserted
endmodule
```

Flip-Flops and Latches

- HDL Example 5.2

HDL Example 5.2

```
// D flip-flop without reset
module D_FF (Q, D, Clk);
  output Q;
  input  D, Clk;
  reg    Q;
  always @ (posedge Clk)
    Q <= D;
endmodule

// D flip-flop with asynchronous reset (V2001, V2005)
module DFF (output reg Q, input D, Clk, rst);
  always @ (posedge Clk, negedge rst)
  if (~rst) Q <= 1'b0;    // Same as: if (rst == 0)
  else Q <= D;
endmodule
```

Characteristic Equation

$Q(t + 1) = Q \oplus T$ → For a *T* flip-flop

$Q(t + 1) = JQ' + K'Q$ → For a *JK* flip-flop

- HDL Example 5.3

HDL Example 5.3

```
// T flip-flop from D flip-flop and gates
module TFF (Q, T, Clk, rst);
  output Q;
  input  T, Clk, rst;
  wire   DT;
  assign DT = Q ^ T ;                    // Continuous assignment
// Instantiate the D flip-flop
  DFF TF1 (Q, DT, Clk, rst);
endmodule
```

HDL Example 5-3 (Continued)

```
// JK flip-flop from D flip-flop and gates (V2001, 2005)
module JKFF (output reg Q, input J, K, Clk, rst);
  wire JK;
  assign JK = (J & ~Q) | (~K & Q);
// Instantiate D flip-flop
  DFF JK1 (Q, J, K, Clk, rst);
endmodule

// D flip-flop (V2001, V2005)
module DFF (output reg Q, input D, Clk, rst);
  always @ (posedge Clk, negedge rst)
    if (~rst) Q <= 1'b0;
    else Q <= D;
endmodule
```

HDL Example 5-4

- Functional description of JK flip-flop

HDL Example 5.4

```
// Functional description of JK flip-flop (V2001, 2005)
module  JK_FF (input J, K, Clk, output reg Q, output Q_b);
  assign Q_b = ~ Q ;
  always @ (posedge Clk)
    case ({J,K})
      2'b00: Q <= Q;
      2'b01: Q <= 1'b0;
      2'b10: Q <= 1'b1;
      2'b11: Q <= ~Q;
    endcase
endmodule
```

State Diagram

- HDL Example 5.5: Mealy HDL model

HDL Example 5.5

```
// Mealy FSM zero detector (See Fig. 5.16)                    Verilog 2001, 2005 syntax
module Mealy_Zero_Detector (
  output reg y_out,
  input      x_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)     Verilog 2001, 2005 syntax
   if (reset == 0) state <= S0;
   else state <= next_state;
```

HDL Example 5-5 (Continued)

```
 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

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
```

HDL Example 5-5 (Continued)

```
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 = 1;
  #70 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
```

Mealy_Zero_Detector

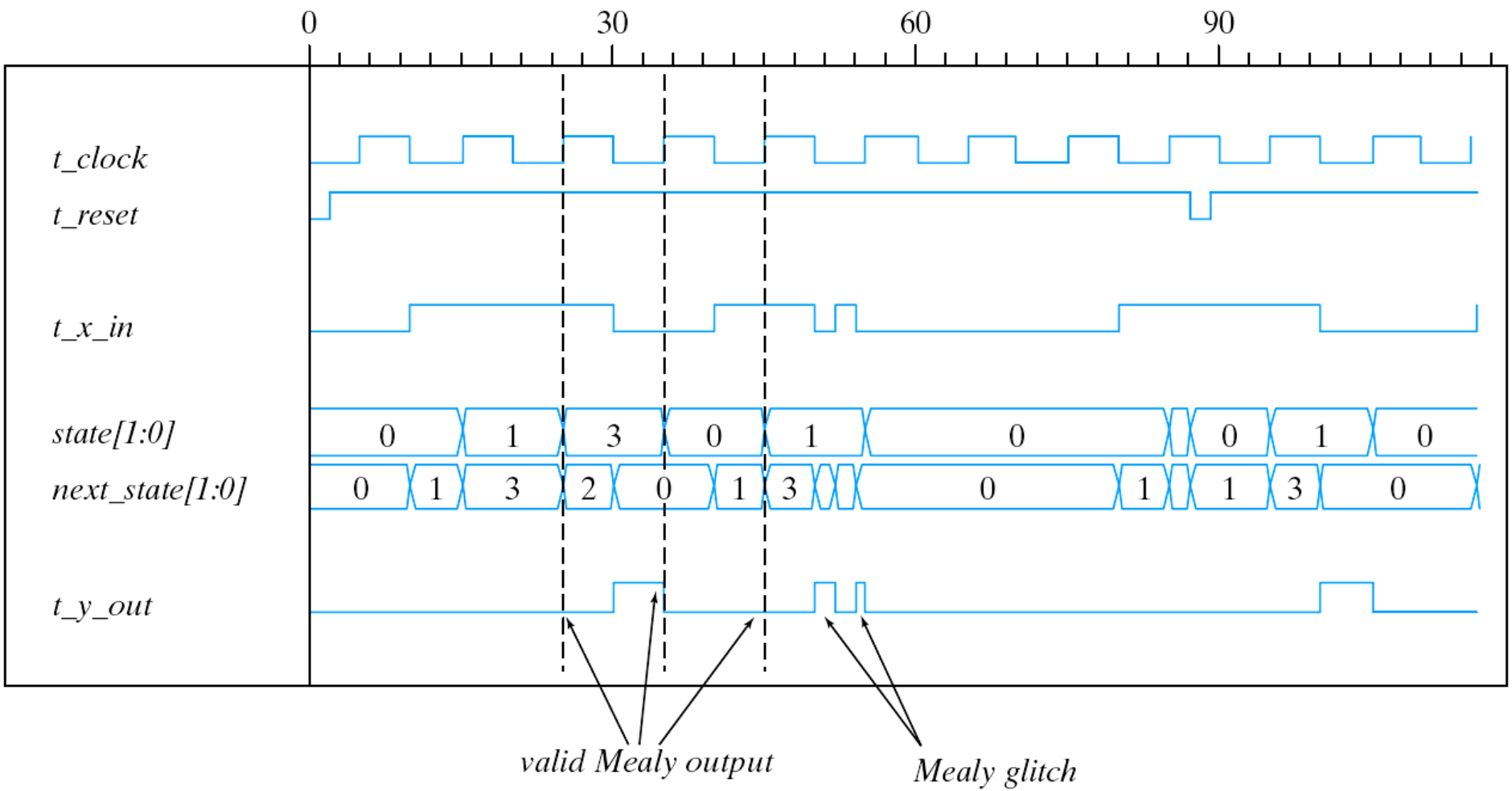

Fig. 5.22 Simulation output of *Mealy_Zero_Detector*

HDL Example 5-6: Moore Model FSM

HDL Example 5.6

```
// Moore model FSM (see Fig. 5.19)                    Verilog 2001, 2005 syntax
module Moore_Model_Fig_5_19 (
  output [1: 0]          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 <= S3;
      S2:    if (~x_in)  state <= S3; else state <= S2;
      S3:    if (~x_in)  state <= S0; else state <= S3;
    endcase

  assign y_out = state;      // Output of flip-flops

endmodule
```

Simulation Output of HDL Example 5-6

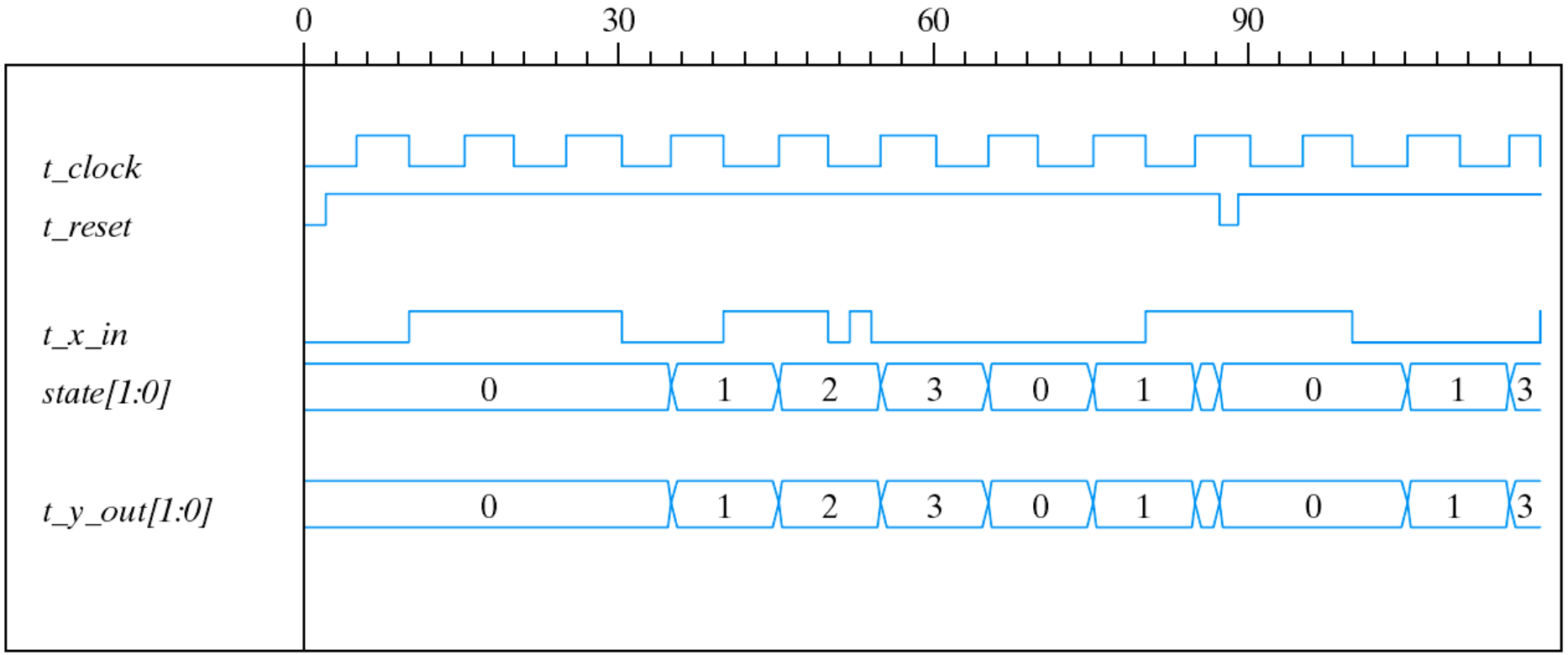

Fig. 5.23 Simulation output of HDL Example 5.6

Structural Description of Clocked Sequential Circuits

- HDL Example 5.7: State-diagram-based model

HDL Example 5.7

```
// State-diagram-based model (V2001, 2005)
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;
```

HDL Example 5-7 (Continued)

```
      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

// structural model
module Moore_Model_STR_Fig_5_20 (
  output   y_out, A, B,
  input    x_in, clock, reset
);
  wire     TA, TB;

// Flip-flop input equations
  assign TA = x_in & B;
  assign TB = x_in;
// Output equation
  assign y_out = A & B;
```

HDL Example 5-7 (Continued)

```
// 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

module Toggle_flip_flop (Q, T, CLK, RST_b);
  output   Q;
  input    T, CLK, RST_b;
  reg      Q;

  always @ (posedge CLK, negedge RST_b)
    if (RST_b == 0) Q <= 1'b0;
    else if (T) Q <= ~Q;
endmodule

// Alternative model using characteristic equation
//  module Toggle_flip_flop (Q, T, CLK, RST_b);
//  output  Q;
//  input   T, CLK, RST_b;
//  reg     Q;
```

HDL Example 5-7 (Continued)

```
//  always @ (posedge CLK, negedge RST)
//    if (RST_b == 0) Q <= 1'b0;
//    else  Q <= Q ^ T;
//  endmodule

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);
```

HDL Example 5-7 (Continued)

```
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
```

Simulation Output of HDL Example 5-7

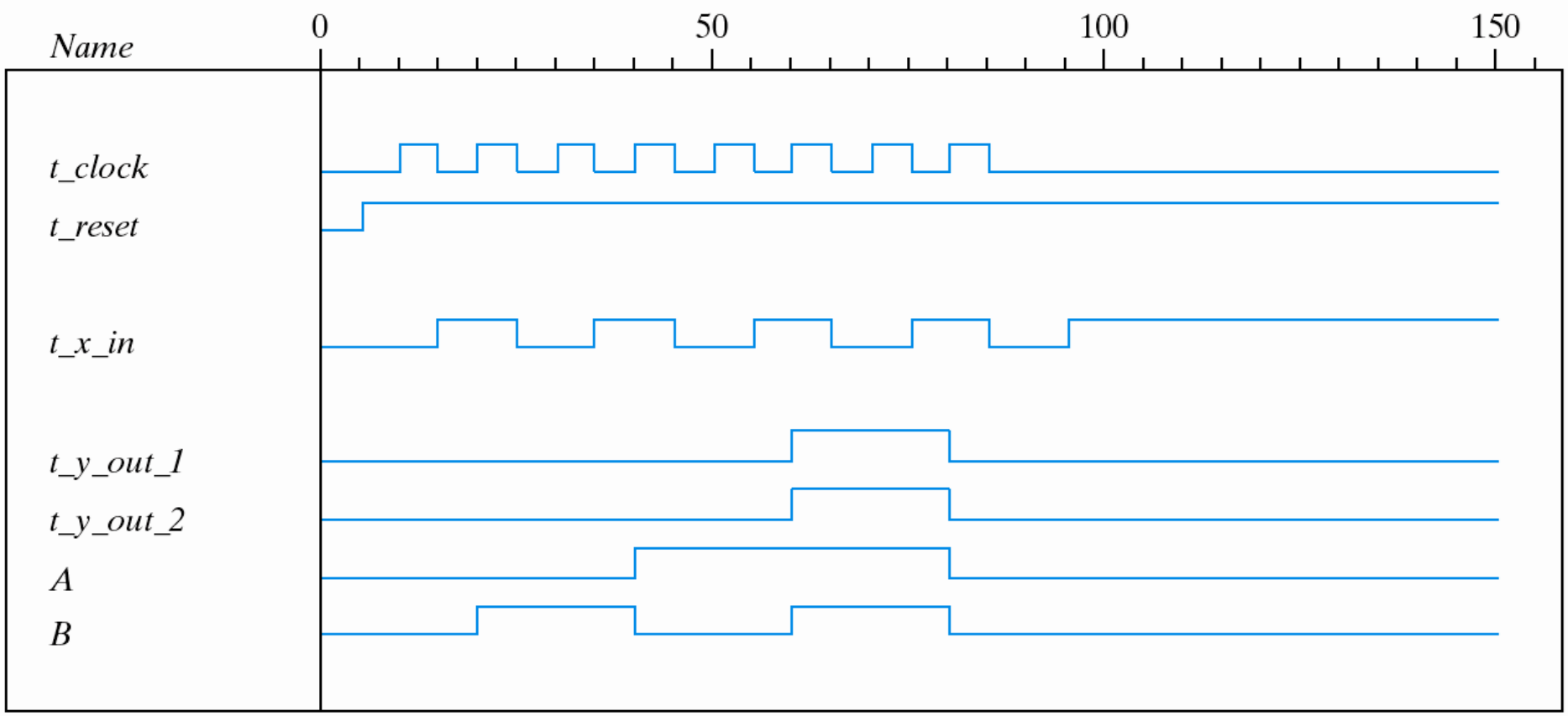

Fig. 5.24 Simulation output of HDL Example 5.7