```
==> d_flip_flop.v <==
module d_flip_flop (
    input clk,
    input rst,
    input data,
    output reg q
);
  always @(posedge clk or rst) begin
    q = (rst) ? ~rst : data;
  end
endmodule

==> t_flip_flop.v <==
module t_flip_flop (
    input clk,
    input rst,
    input t,
    output reg q
);
  always @(posedge clk or rst) begin
    q = (rst) ? ~rst : ((t) ? ~q : q);
  end
endmodule

==> jk_flip_flop.v <==
module jk_flip_flop (
    input j,
    input k,
    input clk,
    output reg q
);
  always @(posedge clk) begin
    case ({
      j, k
    })
      2'b00:  q = q;
      2'b01:  q = 1'b0;
      2'b10:  q = 1'b1;
      2'b11:  q = ~q;
      default: q = q;
    endcase
  end
endmodule

==> sr_flip_flop.v <==
module sr_flip_flop (
    input clk,
    input set,
    input rst,
    output reg q
);
  always @(posedge clk) begin
    case ({
      set, rst
    })
      2'b00: q = q;
      2'b01: q = 1'b0;
      2'b10: q = 1'b1;
      2'b11: q = 1'bx;
      default q = q;
    endcase
  end
endmodule

==> d_from_t.v <==
module d_from_t (

    input  data,
    input  clk,
    input  rst,
    output out
```

```verilog
 71 );
 72    wire med0;
 73    xor (med0, out, data);
 74    t_flip_flop U0 (
 75        .clk(clk),
 76        .rst(rst),
 77        .t  (med0),
 78        .q  (out)
 79    );
 80 endmodule
 81
 82 ==> flip_flop_tb.v <==
 83 module flip_flop_tb ();
 84    reg inp1, inp2, clk, rst;
 85    wire t_out, d_out, jk_out, sr_out, dft_out;
 86
 87    initial begin
 88      $monitor("Inputs=%b %b :: T=%b D=%b JK=%b SR=%b DfT=%b :: rst=%b", inp1, inp2, t_out, d_out,
 89               jk_out, sr_out, dft_out, rst);
 90      {inp1, inp2} = 2'b01;
 91      rst = 1'b1;
 92      clk = 1'b0;
 93
 94      #5 rst = 1'b0;
 95      #10{inp1, inp2} = 2'b00;
 96      #10{inp1, inp2} = 2'b01;
 97      #10{inp1, inp2} = 2'b10;
 98      #10{inp1, inp2} = 2'b11;
 99      #20 $finish(0);
100    end
101
102    initial begin
103      $dumpfile("flip_flops.vcd");
104      $dumpvars(0, flip_flop_tb);
105    end
106
107    always
108      forever begin
109        #5 clk = ~clk;
110      end
111
112    t_flip_flop U0 (
113        .clk(clk),
114        .rst(rst),
115        .t  (inp1),
116        .q  (t_out)
117    );
118
119    d_flip_flop U1 (
120        .clk(clk),
121        .rst(rst),
122        .data(inp1),
123        .q(d_out)
124    );
125
126    jk_flip_flop U2 (
127        .clk(clk),
128        .j  (inp1),
129        .k  (inp2),
130        .q  (jk_out)
131    );
132
133    sr_flip_flop U3 (
134        .clk(clk),
135        .set(inp1),
136        .rst(inp2),
137        .q  (sr_out)
138    );
139
140    d_from_t U4 (
141        .data(inp1),
```

```
142        .clk (clk),
143        .rst (rst),
144        .out (dft_out)
145    );
146 endmodule
147
148 ==> output.txt <==
149 VCD info: dumpfile flip_flops.vcd opened for output.
150 Inputs=0 1 :: T=0 D=0 JK=x SR=x DfT=0 :: rst=1
151 Inputs=0 1 :: T=0 D=0 JK=0 SR=0 DfT=0 :: rst=0
152 Inputs=0 0 :: T=0 D=0 JK=0 SR=0 DfT=0 :: rst=0
153 Inputs=0 1 :: T=0 D=0 JK=0 SR=0 DfT=0 :: rst=0
154 Inputs=1 0 :: T=1 D=1 JK=1 SR=1 DfT=1 :: rst=0
155 Inputs=1 1 :: T=0 D=1 JK=0 SR=x DfT=1 :: rst=0
156 Inputs=1 1 :: T=1 D=1 JK=1 SR=x DfT=1 :: rst=0
157 Inputs=1 1 :: T=0 D=1 JK=0 SR=x DfT=1 :: rst=0
```

Time

clk=1
inp1=1
inp2=0
rst=0
d_out=1
t_out=1
jk_out=1
sr_out=1
dft_out=1

8 sec   16 sec   24 sec   32 sec   40 sec   48 sec   56 sec