Implicit FSM in a Testbench Program
(example 7.1 by Donald Thomas)

Top level and Testbench for FSM Testbench example

```
//////////////////////////////
// DThomas_example7_1.sv -  Example from section 7.3 of D. Thomas
// SystemVerilog book
//
// This example shows how to use an implicit state machine for testing
// a FSM
//////////////////////////////
module top;  // used in testbench chapter
   logic x, z, clk, r_l;

   FSMbehavior dut(.*);
   tBench tb(.*);

   initial begin: I
      $monitor($time, " Current State = %s", dut.state.name);

      ck = 0;
      r_l = 0;
      r_l <= #1 1;

      forever #5 ck = ~ck;
   end: I
endmodule: top

program tBench (
   input   logic    ck,
   output  logic    x
);
   initial begin: J // the implicit FSM
      x <= 1;
      @(posedge ck); // resumes at time 5, no change to x
      @(posedge ck); // resumes at time 15, no change to x
      @(posedge ck); // resumes at 25, x changes
      x <= 0;
      @(posedge ck); // resumes at 35
      #1 $stop;
   end: J
endprogram: tBench
```

FSM Behavioral model for FSM Testbench example

```
module FSMbehavior (input logic x, ck, r_l, output logic z);
   enum {A, B, C} state;

   always_ff @(posedge ck, negedge r_l) begin
      if (~r_l) begin
         state <= A;
      end
      else begin
         case (state)
            A:         state <= (x) ? B : A;
            B:         state <= (x) ? C : A;
            C:         state <= (x) ? C : B;
            default:state <= A;
         endcase
      end
   end // always

   assign z = (state == C) ? ~x : 1'b1;
endmodule: FSMbehavior;
```