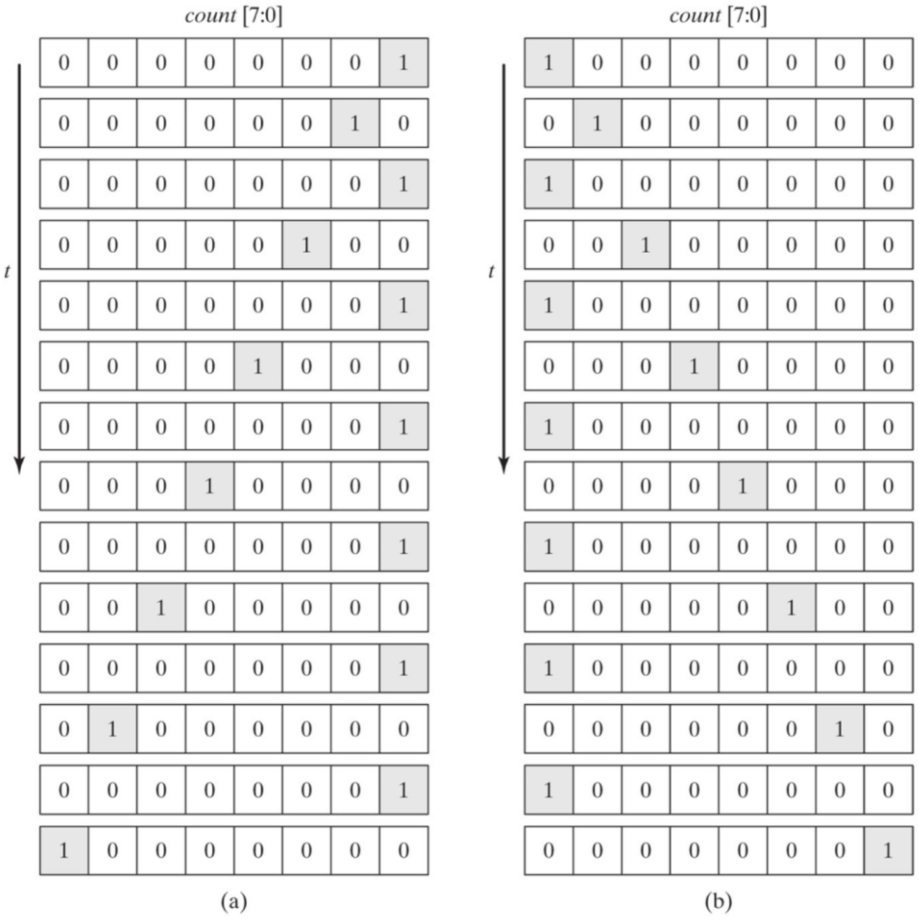
Assignment 4: Problem 2: 4Bit Jerky counter

Using a FSM, design a 4-bit jerky counter that follows the same pattern as the counter given in the book page 226. Test your counter and include the code and simulation results.

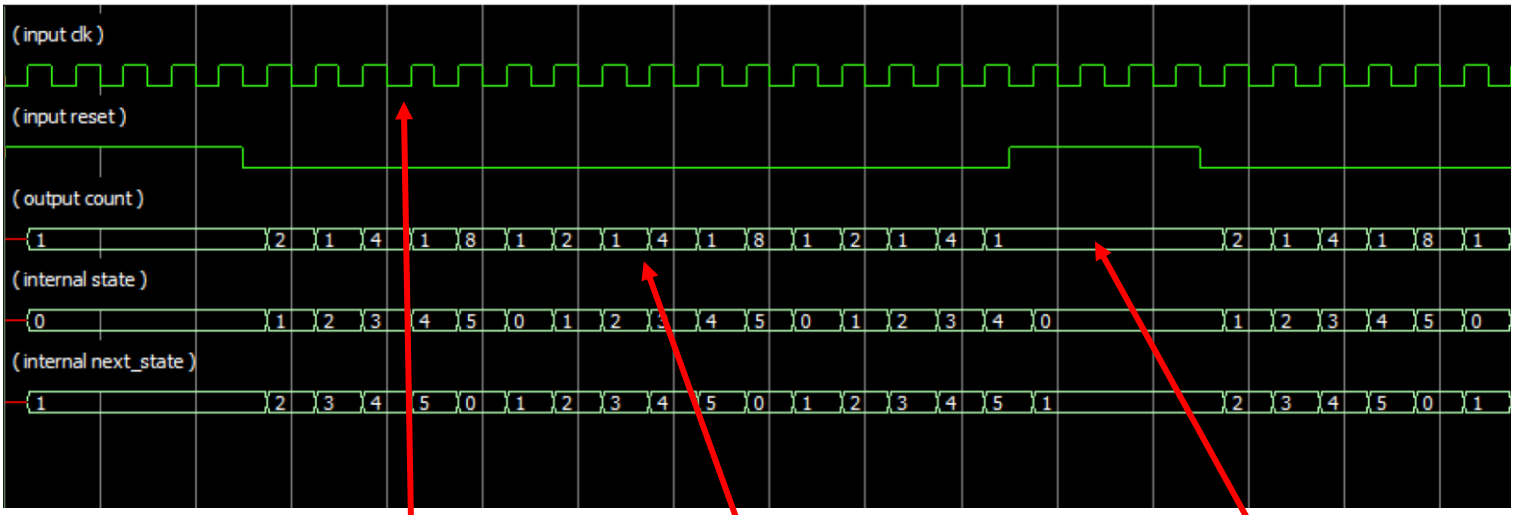


For the following solution, the (a) counter was followed. The only difference in the design for (b) will be the output assigned to the different states as shown in the comments within the code.

```
/* ---------------------------------------------------------------------------------------
   Name:           Nashwa Elaraby
   file:           Jerky_4Bit_counter
   testbench:      Jerky_4Bit_counter_tb
   Description:    Using a FSM, design a 4-bit jerky counter that follows the same
                   pattern as the counter given in the book page 226. Test your counter
                   and include the code and simulation results.
                   Moore FSM
   ---------------------------------------------------------------------------------------
                   (a)     0001            (b)     1000
                           0010                    0100
                           0001                    1000
                           0100                    0010
                           0001                    1000
                           1000                    0001
   ---------------------------------------------------------------------------------------*/

   module Jerky_4Bit_counter (clk, reset, count);

   input             clk;
   input             reset;
   output reg [3:0]  count;


   reg [2:0] state, next_state;

   parameter  S0 = 3'b000;
   parameter  S1 = 3'b001;
   parameter  S2 = 3'b010;
   parameter  S3 = 3'b011;
   parameter  S4 = 3'b100;
   parameter  S5 = 3'b101;
   parameter  S6 = 3'b110;

   always @ (posedge clk)
   if (reset)   state <= S0;
   else         state <= next_state;

   always @ *
      case  (state)
            S0 : begin count      = 4'b0001;      // count = 1000;
                       next_state = S1;       end
            S1 : begin count      = 4'b0010;      // count = 0100;
                       next_state = S2;       end
            S2 : begin count      = 4'b0001;      // count = 1000;
                       next_state = S3;       end
            S3 : begin count      = 4'b0100;      // count = 0010;
                       next_state = S4;       end
            S4 : begin count      = 4'b0001;      // count = 1000;
                       next_state = S5;       end
            S5 : begin count      = 4'b1000;      // count = 0001;
                       next_state = S0;       end
        default: begin count      = 4'b0001;      // count = 1000;
                       next_state = S0;       end
      endcase

    endmodule
```

```
module Jerky_4Bit_counter_tb ();

 reg        clk, reset;
 wire [3:0] count;
 wire [2:0] state;
 wire [2:0] next_state;

 Jerky_4Bit_counter UUT  (clk, reset, count);

 assign state      = UUT.state;
 assign next_state = UUT.next_state;

 initial
   begin
      clk = 1'b0;
      forever
      #5 clk = ~clk;
   end

  initial
   begin
        reset = 1'b1;
   #50  reset = 1'b0;
   #160 reset = 1'b1;
   #40  reset = 1'b0;
   end

endmodule
```

Internal probes to observe the state and the net state registers.



Moore FSM, the ouput is perfectly aligned with the change in the state at the positive clock edge of the clk.

Jerky counter behavior showing the powers of 2 as expected.

When the reset is active, the FSM counter stays at S0 state.