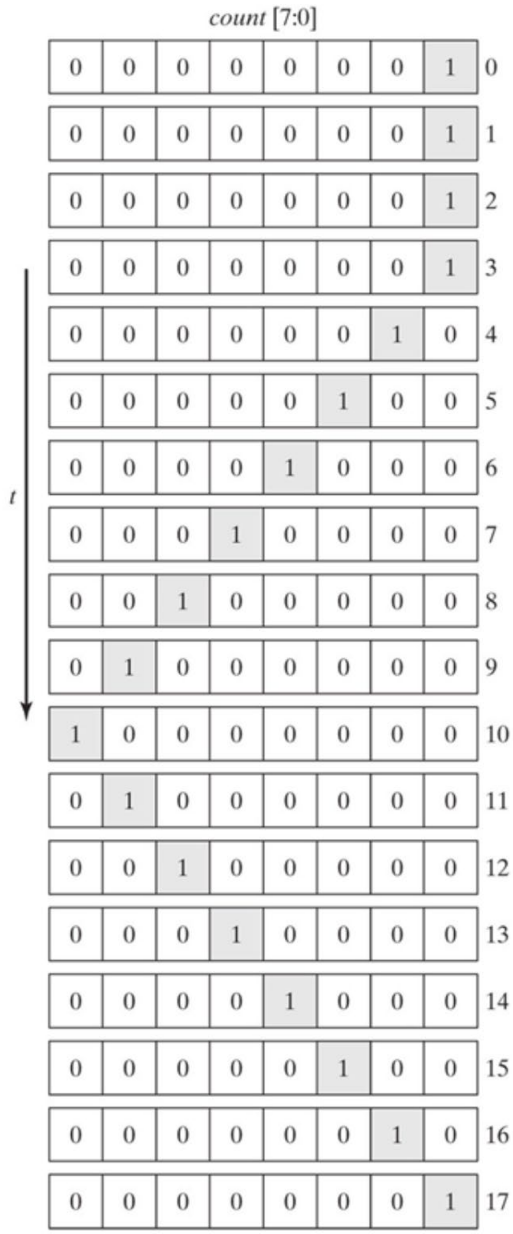
Lesson 7: DataPath controllers – Assignment 2 – Special Counter

Write and verify a Verilog model for a counter with 18-step pattern cyclically until the counter is interrupted by a reset. This counter can be designed using an 18-state FSM! We would like to design it using a DataPath Controller structure. The DataPath may include counters, comparators, and shifter submodules combined in one DataPath top module. The controller will handle control single-bit signals to orchestrate the counter operation. The controller will have a reduced number of states, and the design may have a parametrized word size of the count as well as the number of cycles it keeps the 1 value (0000_0001).

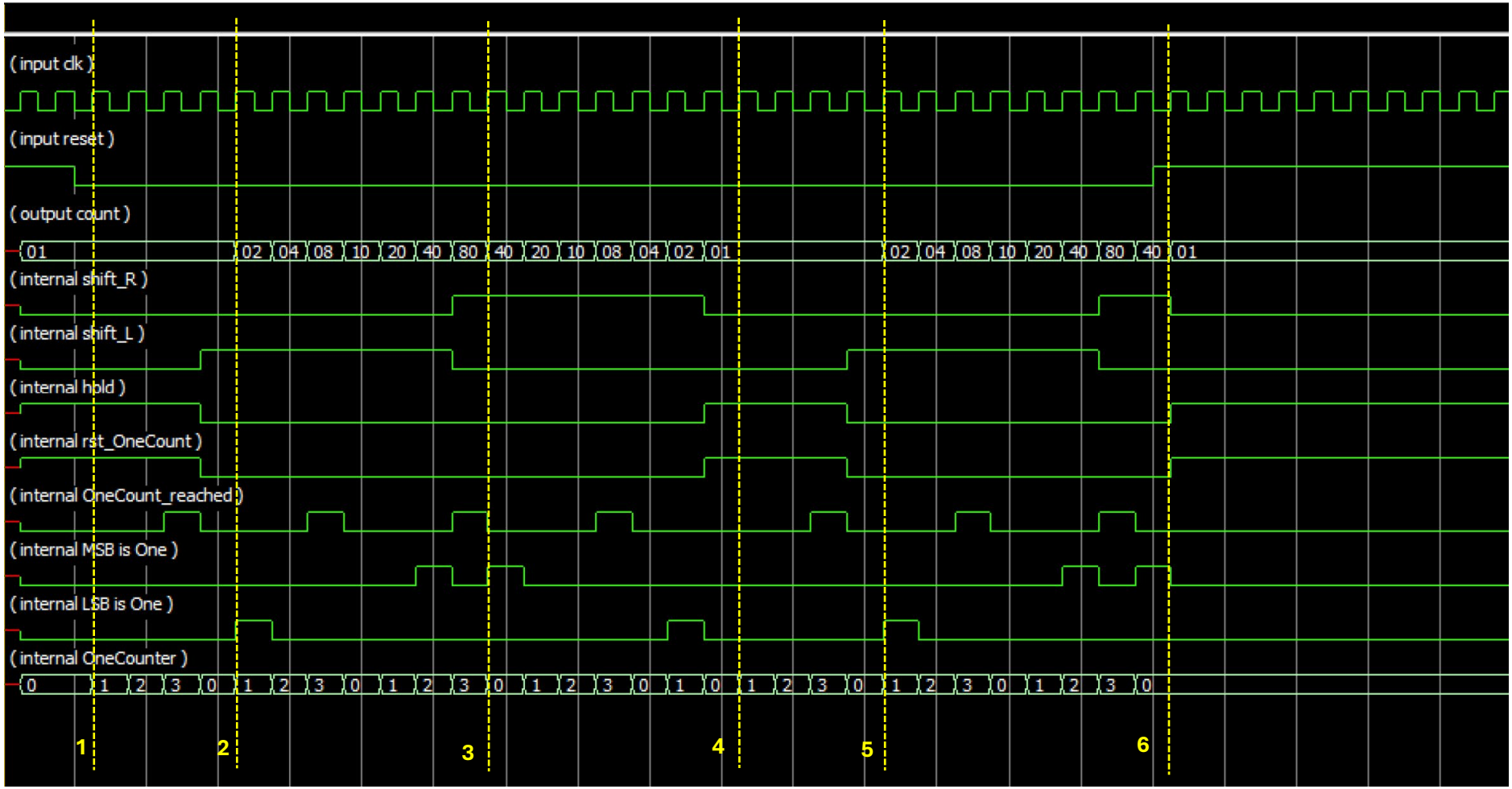
Complete the design and test it. Comment on the comparison between a FSM design and the DataPath Controller structure implemented, in terms of state reductions and the extendibility of the design.



```
/*-------------------------------------------------------------------------------------------
                        Lesson 7  DataPath Controller
  -------------------------------------------------------------------------------------------
  Write and verify a Verilog model for a counter with 18-step pattern cyclically until the
  counter is interrupted by a reset. This counter can be designed using an 18-state FSM!
  We would like to design it using a DataPath Controller structure. The DataPath may include
  counters, comparators, and shifter submodules combined in one DataPath top module.
  The controller will handle control single-bit signals to orchestrate the counter operation.
  The controller will have a reduced number of states, and the design may have a parametrized
  word size of the count as well as the number of cycles it keeps the 1 value (0000_0001).
  Complete the design and test it. Comment on the comparison between a FSM design and the
  DataPath Controller structure implemented, in terms of state reductions and the
  extendibility of the design.
  -------------------------------------------------------------------------------------------*/

  module Special_Counter (clk, reset, count);

  parameter word_size = 8;

  input                         clk, reset;
  output   [word_size-1:0]      count;
  wire                          shift_R, shift_L, hold;                 // from controller to DataPath
  wire                          OneCount_rst;
  wire                          OneCount_reached, MSBisONE, LSBisONE;   // from DataPath to controler

  Counter_DataPath    M1 (count, clk, reset, shift_R, shift_L, hold, OneCount_rst,
                          OneCount_reached, MSBisONE, LSBisONE);
  Counter_Controler   M2 (clk, reset, shift_R, shift_L, hold, OneCount_rst,
                          OneCount_reached, MSBisONE, LSBisONE);

  endmodule


  //-------------------------------------------------------------------------------------------
```

```
//-------------------------------------------------------------------------------------------

module Counter_DataPath (count, clk, reset,
                         shift_R, shift_L, hold, OneCount_rst,
                         OneCount_reached, MSBisONE, LSBisONE);

parameter word_size      = 8;
parameter hold_oneCount  = 3;
parameter internal_OneCount_size   =$clog2(hold_oneCount);     // 2

output  reg   [word_size -1 :0]    count;
input                              clk, reset;
input                              shift_R, shift_L, hold, OneCount_rst;
output                             OneCount_reached, MSBisONE, LSBisONE;

reg           [internal_OneCount_size -1 : 0]   internal_OneCount;

// output flags
assign OneCount_reached = (internal_OneCount  == hold_oneCount);
assign MSBisONE         = (count[word_size-2] == 1);          // allow for one clock cycle delay
assign LSBisONE         = (count[1] == 1);                    // between DataPath & Controller

always @ (posedge clk)
       if (reset | OneCount_rst)
          internal_OneCount <= 0;
       else
          internal_OneCount <= internal_OneCount + 1;

always @ (posedge clk)
        if (reset)    count <= 1;
   else if (hold)     count <= count;
   else if (shift_R)  count <= {count[0], count [word_size-1 : 1]};
   else if (shift_L)  count <= {count [word_size-2:0], count[word_size-1]};

endmodule
```

```
module Counter_Controler  (clk, reset, shift_R, shift_L, hold, OneCount_rst,
                           OneCount_reached, MSBisONE, LSBisONE);

input                                   clk, reset;
output     reg                          shift_R, shift_L, hold, OneCount_rst;
input                                   OneCount_reached, MSBisONE, LSBisONE;

reg   [1:0]   state, next_state;

parameter  hold_one = 2'b00;
parameter  R_shift  = 2'b01;
parameter  L_shift  = 2'b10;

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

always @ *
     begin
          case (state)
               hold_one :  begin
                           hold = 1;
                           OneCount_rst = 0;   shift_R = 0;   shift_L = 0;
                           if  (OneCount_reached)  next_state = L_shift;
                            else                   next_state = hold_one;     end
               L_shift  :  begin
                           shift_L = 1;
                           hold = 0;           shift_R = 0;   OneCount_rst = 0;
                           if  (MSBisONE)          next_state = R_shift;
                            else                   next_state = L_shift;      end
               R_shift  :  begin
                           shift_R = 1;
                           shift_L = 0;    OneCount_rst = 0;   hold = 0;
                           if(LSBisONE)      begin next_state = hold_one;
                                                   OneCount_rst = 1;   end
                               else                next_state = R_shift;      end
                default :                          next_state = hold_one;
           endcase
      end

endmodule
```



At cursor 1 the module encounters the first positive edge after the reset was deactivated. This is when the counting starts. After 4 counts of 1's, over 4 clock cycles, the shifting left starts as given at the cursor 2. The numbers in the simulation diagram is given in hexadecimal. 01 indicates that the one bit is at the LSB right side and 80 shows that the one bit is at the MSB left side. At the cursor 3 the right shifting starts. At the cursor 4 the 1 reaches the LSB, and the hold starts for 4 clock cycles and the left shifting starts at cursor 5. During the right shifting the reset is activated forcing the count to gi back to 01. Since there is one clock cycle delay between the controller output signals and the DataPath output flag signals, the DataPath raises the flag of MSBisOne when the count is at 02, and the flag for the LSBisONE is raised when thecount is at 40, and the count is forced back to 01.