Lab 6

Implementation of a 8 bit Ring Counter

Spring 2025

Submitted by: **Mohsin Sajjad**

Registration No: **22pwsce2149**

Class Section: **A**

"On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work."

Student Signature: _____

Submitted to:

Engr. Faheem Jan

Month Day, Year (23 03, 2025)

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

Objective:

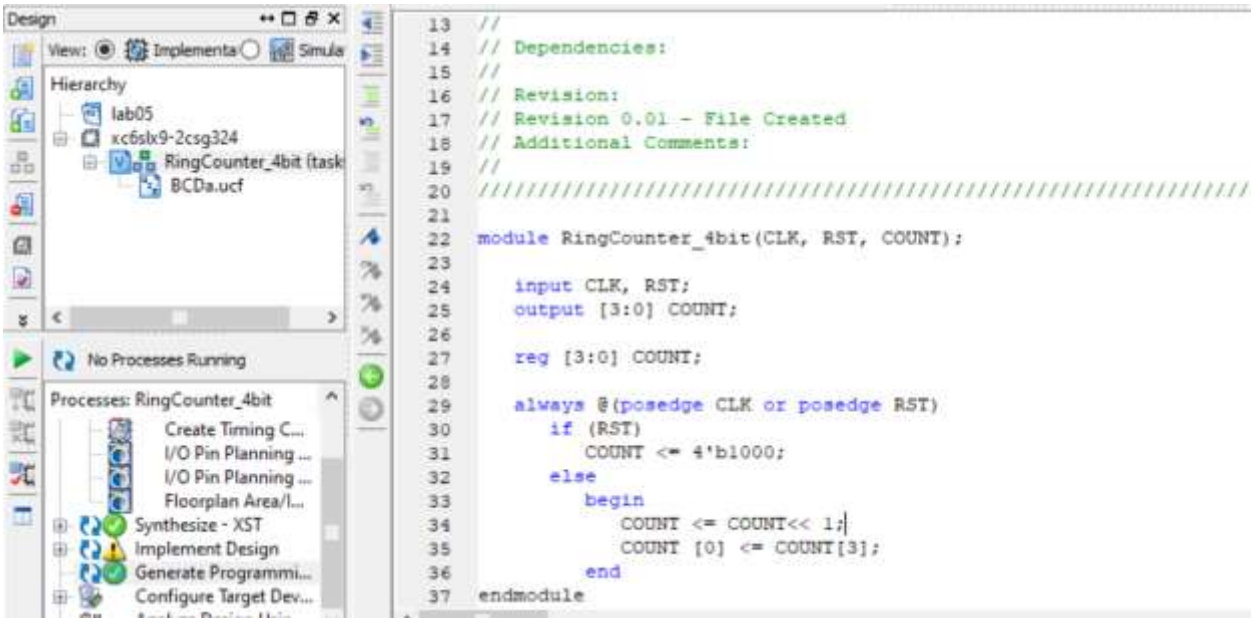
- To become familiarized with behavior level modeling
- To be able to implement sequential circuits using Verilog
- To Implement an 8 Bit Ring Counter on Spartan 6 FPGA starter kit.

.

Lab Task:

1-Implement 4 bit Ring Counter

CODE:




```
13  //
14  // Dependencies:
15  //
16  // Revision:
17  // Revision 0.01 - File Created
18  // Additional Comments:
19  //
20  ////////////////////////////////////////////////////////////////////////////////
21
22  module RingCounter_4bit(CLK, RST, COUNT);
23
24     input CLK, RST;
25     output [3:0] COUNT;
26
27     reg [3:0] COUNT;
28
29     always @(posedge CLK or posedge RST)
30        if (RST)
31           COUNT <= 4'b1000;
32        else
33           begin
34              COUNT <= COUNT<< 1;
35              COUNT [0] <= COUNT[3];
36           end
37  endmodule
```

UCF file:

```
96  ##############################################################################################
97     NET "RST"                    LOC = M18     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST | PULLUP; #SW1
98     # NET "SUM[3]"                LOC = L18     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST | PULLUP; #SW2
99     # NET "SUM[2]"                LOC = M16     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST | PULLUP; #SW3
100    #NET "SUM[1]"                LOC = L17     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST | PULLUP; #SW4
101    #NET "SUM[0]"                LOC = K17     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST | PULLUP; #SW5
102    NET "CLK"                    LOC = K18     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST | PULLUP; #SW6
103
104 ##############################################################################################
105 #                                                LEDs
106 ##############################################################################################
107    #NET "Cout"                   LOC = P15     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D1
108    #NET "Sum[1]"                 LOC = P16     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D2
109    #NET "Sum[2]"                 LOC = N15     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D3
110   # NET "Cout"                   LOC = N16     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D4
111    NET "COUNT[3]"               LOC = U17     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D5
112    NET "COUNT[2]"               LOC = U18     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D6
113    NET "COUNT[1]"               LOC = T17     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D7
114    NET "COUNT[0]"               LOC = T18     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D8
115
```

OUTPUT:

Conclusion:

This **4-bit Ring Counter** shifts a single '1' through its four-bit register on each clock pulse. When **reset (RST) is high**, it initializes to `1000`. On every **clock (CLK) pulse**, the bits shift left, and the leftmost bit wraps around to the rightmost position, creating a continuous rotating pattern. It is commonly used in **state machines** and **cyclic counting applications** in digital systems.

TASK 02:

Implement 8-bit Ring Counter

CODE:

```
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////module RingCounter (CLK, RST, C

module RingCounter_8bit(CLK, RST, COUNT);

   input CLK, RST;
   output [7:0] COUNT;

   reg [7:0] COUNT;

   always @(posedge CLK)
      if (RST)
         COUNT <= 8'b10000000;
      else
         begin
            COUNT <= COUNT<< 1;
            COUNT [0] <= COUNT [7];
         end

endmodule
```

Ucf file:

```
##############################################################################################
    NET "RST"                   LOC = M18     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST | PULLUP; #SW1
    # NET "SUM[3]"                LOC = L18     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST | PULLUP; #SW2
    # NET "SUM[2]"                LOC = M16     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST | PULLUP; #SW3
    #NET "SUM[1]"               LOC = L17     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST | PULLUP; #SW4
    #NET "SUM[0]"               LOC = K17     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST | PULLUP; #SW5
    NET "CLK"                   LOC = K18     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST | PULLUP; #SW6

##############################################################################################
#                                                LEDs
##############################################################################################
    NET "COUNT[7]"                LOC = P15     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D1
    NET "COUNT[6]"               LOC = P16     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D2
    NET "COUNT[5]"               LOC = N15     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D3
    NET "COUNT[4]"                LOC = N16     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D4
    NET "COUNT[3]"              LOC = U17     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D5
    NET "COUNT[2]"              LOC = U18     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D6
    NET "COUNT[1]"              LOC = T17     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D7
    NET "COUNT[0]"              LOC = T18     | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;  #D8

##############################################################################################
```

OUTPUT:

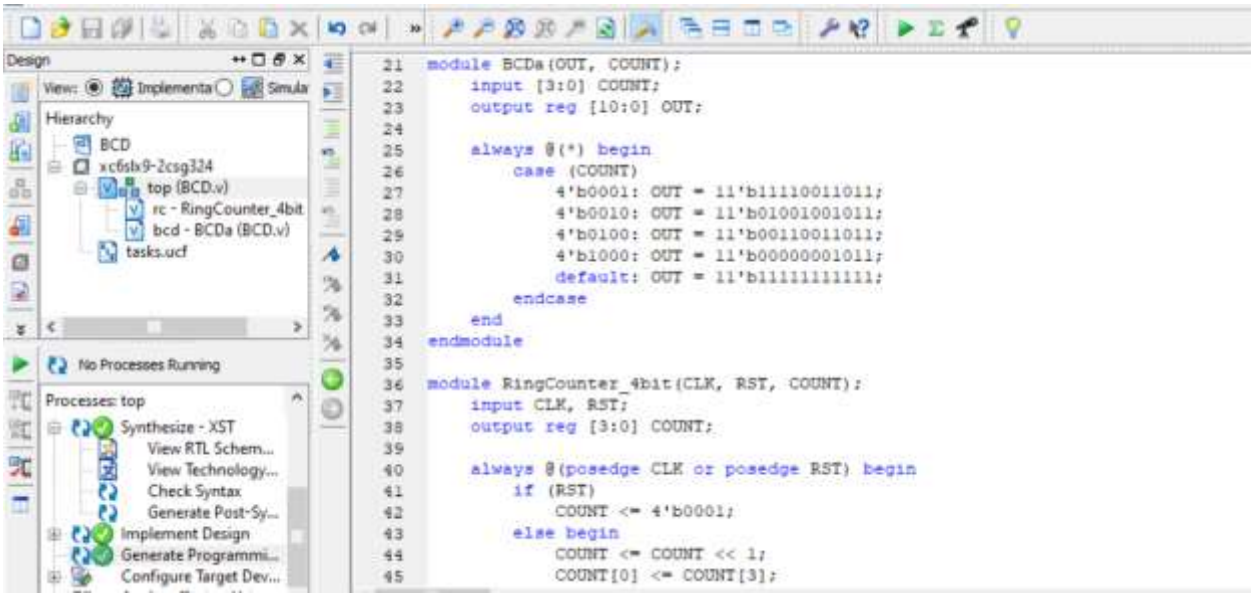
Conclusion:

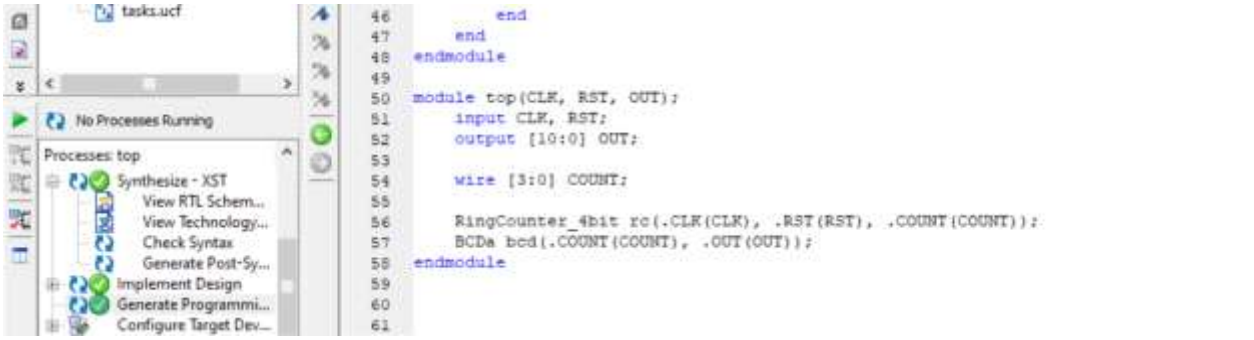
This 8-bit Ring Counter continuously shifts a single '1' through its 8-bit register on each clock cycle. When reset (RST) is high, it initializes to 10000000. On every clock (CLK) pulse, the bits shift left, and the leftmost bit wraps around to the rightmost position, creating a cyclic shifting pattern. This type of counter is useful in sequential circuits, state machines, and LED chasers.

TASK 03:

For 4 bit Ring counter display the count in the seven segment display.

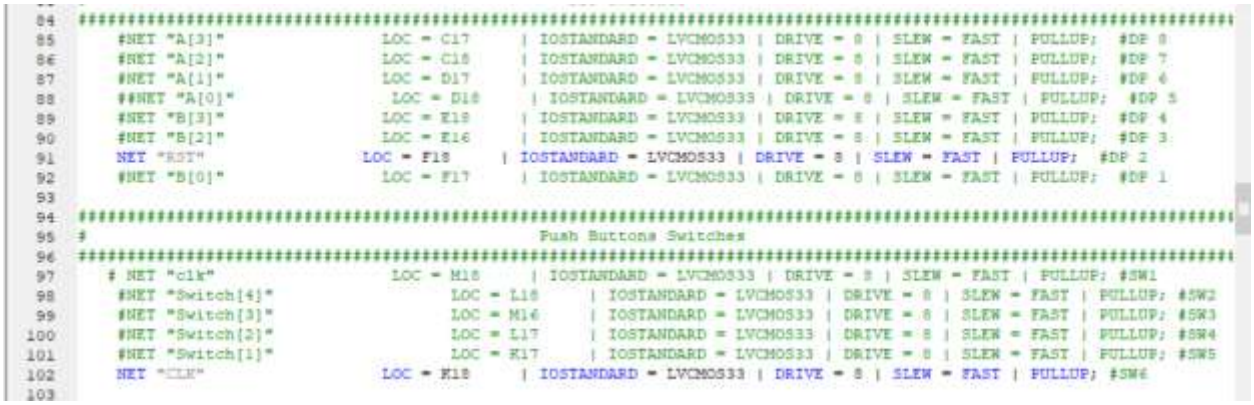
CODE:





UCF file:

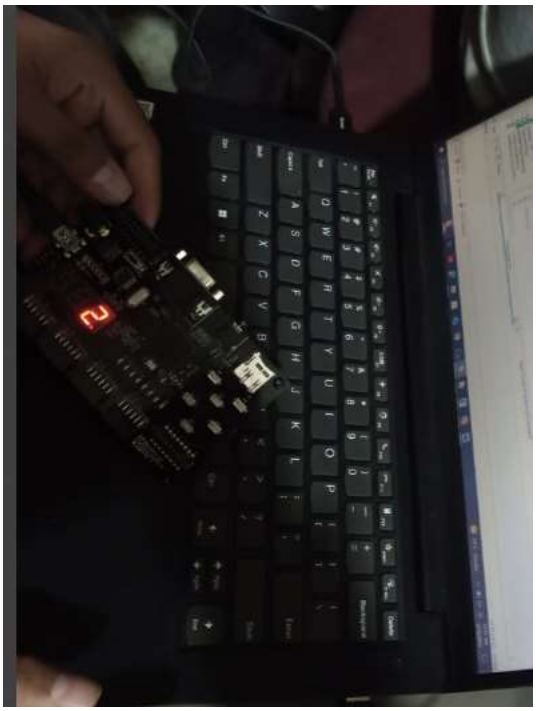
Output:

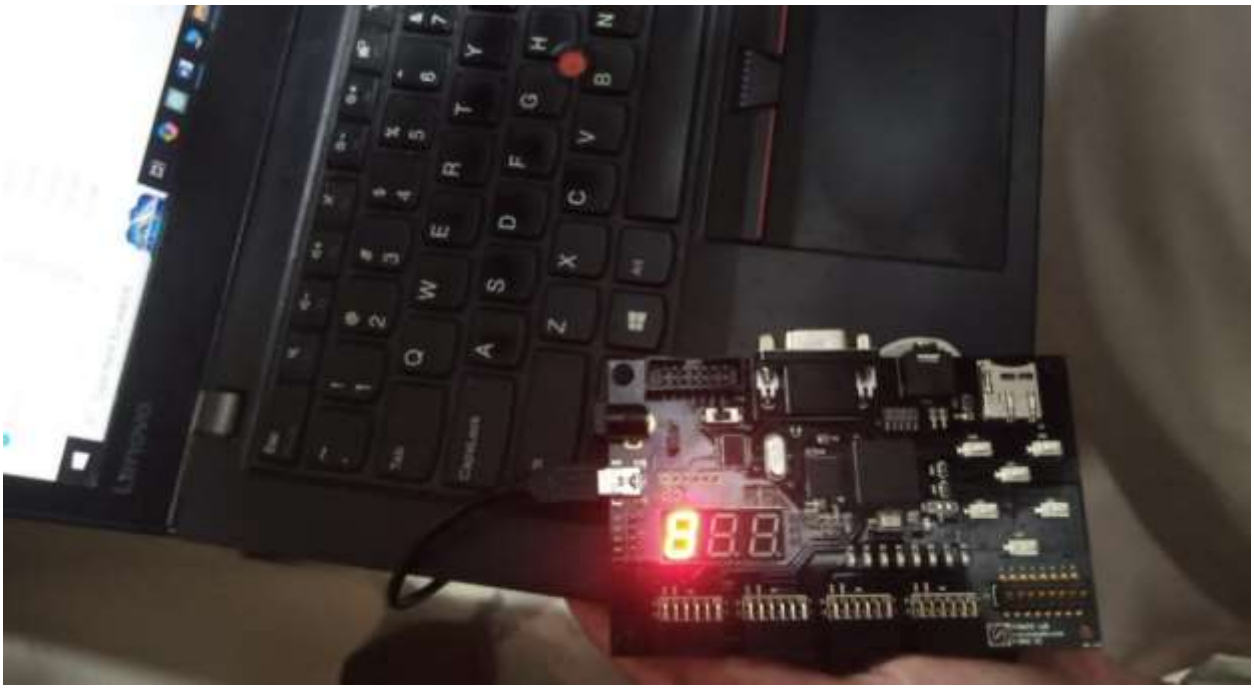


```
NET "OUT[4]"        LOC = A3    | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;   #a
NET "OUT[5]"        LOC = B4    | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;   #b
NET "OUT[6]"        LOC = A4    | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;   #c
NET "OUT[7]"        LOC = C4    | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;   #d
NET "OUT[8]"        LOC = C5    | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;   #e
NET "OUT[9]"        LOC = D6    | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;   #f
NET "OUT[10]"        LOC = C6    | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;   #g
NET "OUT[3]"        LOC = A5    | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;   #dot

NET "OUT[2]"    LOC = B3    | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;   #Enables for Seven Segment
NET "OUT[1]"    LOC = A2    | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;
NET "OUT[0]"    LOC = B2    | IOSTANDARD = LVCMOS33 | DRIVE = 8 | SLEW = FAST ;
```

Output:





Conclusion:

This Verilog code implements a 4-bit Ring Counter and a BCD to 7-segment display decoder, then connects them in a top module.

1. **Ring Counter Module**
    - It cycles a single '1' through four positions.
    - On reset (RST), it starts at 0001.
    - On each clock (CLK) pulse, the bits shift left, and the last bit moves to the first position.
2. **BCD Decoder (BCDa) Module**
    - It converts a 4-bit binary count into an 11-bit output for display.
    - Specific BCD values (0001, 0010, 0100, 1000) map to predefined outputs.
3. **Top Module**
    - It connects the Ring Counter to the BCD Decoder.
    - The counter generates a sequence, and the decoder outputs the corresponding display value.

This setup is useful for sequential circuits, display systems, and LED animations.