ASSIGNMENT - 3

DIGITAL SYSTEMS ES204

Guntas Singh Saran
guntassingh.saran@iitgn.ac.in
22110089

Nihar Shah
nihar.shah@iitgn.ac.in
22110237

PART - 1

4-bit combined UP/DOWN BIN/BCD Counter

VERILOG CODE

```
module tff(
    input T,
    input rstn, preset, clk,
    output reg Q
    );

    always @(posedge clk or negedge rstn or negedge preset) begin
        if (!rstn) begin
            Q <= 0;
        end
        else if (!preset) begin
            Q <= 1;
        end
        else begin
            Q <= (T ? ~Q : Q);
        end
```

```
   end

endmodule


module counter(
   input En,
   input rstn, clk,
   input [3:0] preset,
   input SEL1, SEL2,
   output [3:0] Q
   );

// rstn = 0 -> the Flip Flop resets or clears
// [3:0] preset -> 0000 sets all T flip flops to Q1, Q2, Q3, Q4
= 1, 1, 1, 1


// SEL1 -> Decides for UP/DOWN
// f1 = UP.(SEL1) + DOWN.(~SEL1)
// f1 = UP when SEL1 = 1
// f1 = DOWN when SEL1 = 0

// SEL2 -> Decides for BCD/BIN where (BCD -> MOD 10 counter) and
(BIN -> MOD 16 counter)
// f2 = BCD.(SEL2) + BIN.(~SEL2)
// f2 = BCD when SEL2 = 1
// f2 = BIN when SEL2 = 0


/*
LOGIC -> for SEL1 (UP/DOWN)
   t1 = En
   t2 = q1.SEL1 + (~q1).(~SEL1)
   t3 = q2.q1.SEL1 + (~q2).(~q1).(~SEL1)
   t4 = q3.q2.q1.SEL1 + (~q3).(~q2).(~q1).(~SEL1)
```

```
*/

nand (new_rstn, Q[1], Q[3], SEL2, SEL1);

// it will check for resetting the flip flops to 0000 when they
are at
// the state 1010 (Asynchronous RESET) and the UP-BCD is chosen
(SEL1 = 1 and SEL2 = 1)
// 1000 -> 1001 -> 1010 (Normally)
// 1000 -> 1001 -> 0000 (Wanted)


nand (new_preset, Q[0], Q[1], Q[2], Q[3], ~SEL1, SEL2);
// we wish to PRESET Q1 and Q4 and RESET Q2 and Q3 giving
Q4Q3Q2Q1 = 1001
// when we reach Q4Q3Q2Q1 = 1111 (Asynchronous PRESET) and the
DOWN-BCD is chosen (SEL1 = 0 and SEL2 = 1)
// 0001 -> 0000 -> 1111 (Normally)
// 0001 -> 0000 -> 1001 (Wanted)

wire q1, q2, q3, q4;
wire t2, t3, t4;
wire temp1_0, temp1_1;
wire temp2_0, temp2_1;

tff T1(.T(En), .rstn(new_rstn & rstn), .preset(preset[0] &
new_preset), .clk(clk), .Q(q1));
xnor (t2, q1, SEL1);

tff T2(.T(t2), .rstn(new_rstn & rstn & new_preset),
.preset(preset[1]), .clk(clk), .Q(q2));
and (temp1_0, q2, q1, SEL1);
and (temp1_1, ~q2, ~q1, ~SEL1);
or (t3, temp1_0, temp1_1);

tff T3(.T(t3), .rstn(new_rstn & rstn & new_preset),
```

```
.preset(preset[2]), .clk(clk), .Q(q3));
and (temp2_0, q3, q2, q1, SEL1);
and (temp2_1, ~q3, ~q2, ~q1, ~SEL1);
or (t4, temp2_0, temp2_1);

tff T4(.T(t4), .rstn(new_rstn & rstn), .preset(preset[3] &
new_preset), .clk(clk), .Q(q4));

assign Q = {q4, q3, q2, q1};

endmodule
```

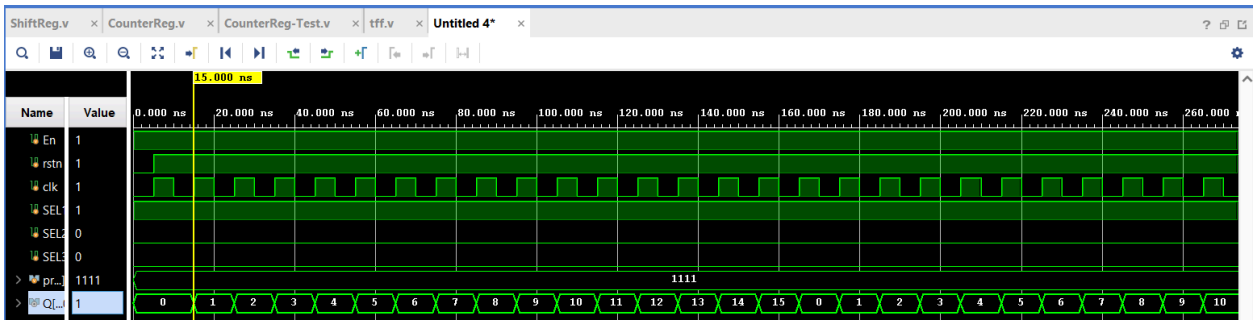
TESTBENCH

```
module testbench;

reg En, rstn, clk, SEL1, SEL2;
reg [3:0] preset;
wire [3:0] Q;

counter count(.En(En), .rstn(rstn), .clk(clk), .preset(preset),
.SEL1(SEL1), .SEL2(SEL2), .Q(Q));


always #5 clk = ~clk;

initial begin
   /*
   MODE ->
       00 -> DOWN-BIN
       01 -> DOWN-BCD
```

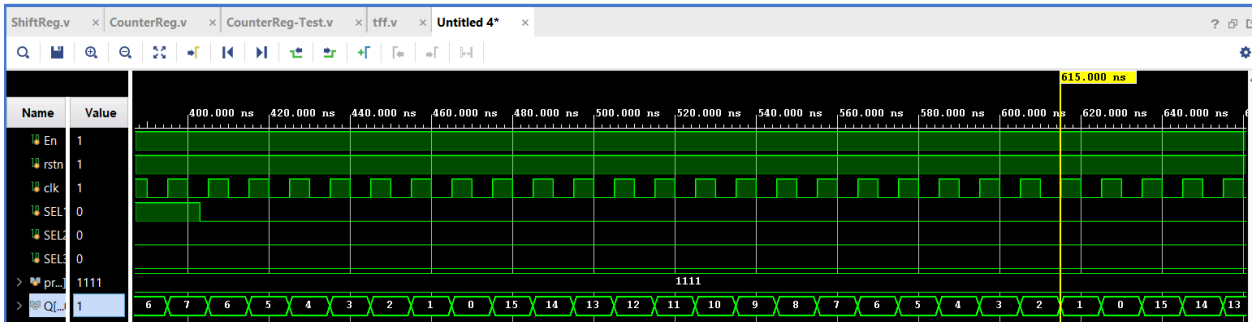
```
        10 -> UP-BIN
        11 -> UP-BCD
    */

    rstn <= 0; SEL1 = 1; SEL2 = 0;  // MODE = 10
    preset <= 4'b1111;
    clk <= 0;
    En = 1;
    $monitor($time, " MODE = %b%b, Q4Q3Q2Q1 = %b%b%b%b, Q = %d", SEL1,
SEL2, Q[3], Q[2], Q[1], Q[0], Q);
    #5 rstn <= 1;
    #398 SEL1 = 0;              // MODE = 00
    #398 SEL1 = 1; SEL2 = 1;    // MODE = 11
    #398 SEL1 = 0;              // MODE = 01
    #398 $finish;
end


endmodule
```

RESULTS OF SIMULATION OF UP-DOWN BIN-BCD COUNTER

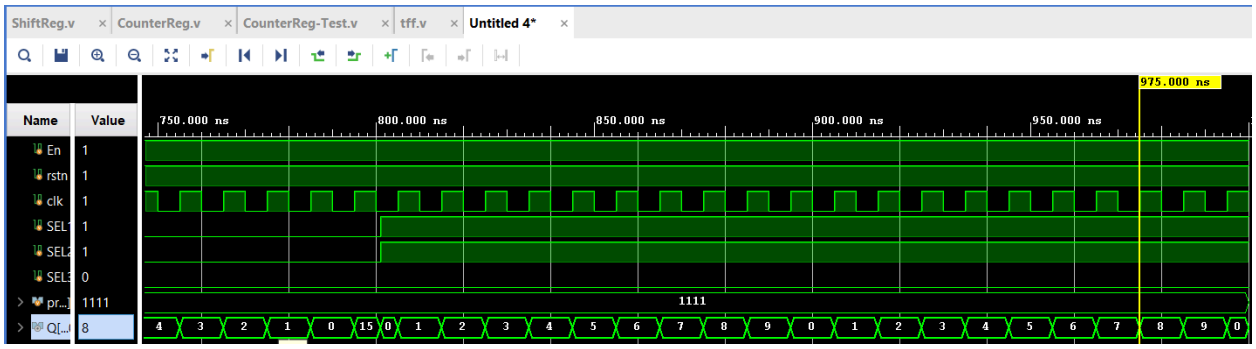
1. BIN UP Counter



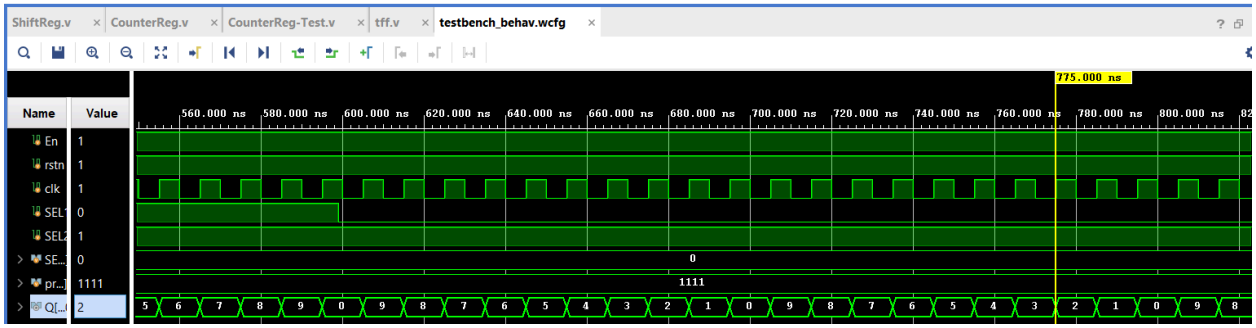
2. BIN DOWN Counter



3. BCD UP Counter



4. BCD DOWN Counter



PART – 2

4-bit Shift Reg with modes to Parallely shift (No Shift), Right Shift and Left Shift the 4 bits.

VERILOG CODE

```
module shiftreg(
    input En,
    input clk,
    input [3:0] preset,
    input [1:0] SEL3,
    output reg [3:0] Q
    );

// SEL3 = 0 -> PARALLEL SHIFT
// SEL3 = 1 -> RIGHT SHIFT
// SEL3 = 2 -> LEFT SHIFT
// En = 0 -> Shift Register brings down its values from the
received 4 bits of Counter

and (pr1, En, preset[0]);
and (pr2, En, preset[1]);
and (pr3, En, preset[2]);
and (pr4, En, preset[3]);
```

```
always @(posedge clk) begin
   if (SEL3 == 2'b00) begin
       Q <= {pr4, pr3, pr2, pr1};
   end

   else if (SEL3 == 2'b01) begin
       Q <= Q >> 1;
   end

   else if (SEL3 == 2'b10) begin
       Q <= Q << 1;
   end

end


endmodule
```

COMBINED TESTBENCH

```
module testbench;


reg En, rstn, clk, SEL1, SEL2, [1:0] SEL3;
reg [3:0] preset;
wire [3:0] Q;
wire [3:0] Q1;

counter count(.En(En), .rstn(rstn), .clk(clk), .preset(preset),
.SEL1(SEL1), .SEL2(SEL2), .Q(Q));
shiftreg sftreg(.En(~En), .clk(clk), .preset(Q), .SEL3(SEL3),
.Q(Q1));
```

```
always #5 clk = ~clk;

initial begin
   /*
   MODE ->
       En = 1 -> Counter ON
       En = 0 -> ShiftReg ON

       000 -> DOWN-BIN-ParallelShift
       010 -> DOWN-BCD-ParallelShift
       100 -> UP-BIN-ParallelShift
       110 -> UP-BCD-ParallelShift

       001 -> DOWN-BIN-RightShift
       011 -> DOWN-BCD-RightShift
       101 -> UP-BIN-RightShift
       111 -> UP-BCD-RightShift

       002 -> DOWN-BIN-LeftShift
       012 -> DOWN-BCD-LeftShift
       102 -> UP-BIN-LeftShift
       112 -> UP-BCD-LeftShift
   */

   rstn <= 0; SEL1 = 1; SEL2 = 0; SEL3 = 2'b00;  // MODE = 100
   preset <= 4'b1111;
   clk <= 0;
   En = 1;
   $monitor($time, " En = %b, MODE = %b%b%d, Q4Q3Q2Q1 =
%b%b%b%b, Q = %d, ShiftRegOut = %b%b%b%b", En, SEL1, SEL2, SEL3,
Q[3], Q[2], Q[1], Q[0], Q, Q1[3], Q1[2], Q1[1], Q1[0]);
   #5 rstn <= 1;
   #398 SEL1 = 0;              // MODE = 000
   #398 SEL1 = 1; SEL2 = 1;    // MODE = 110
   #398 SEL1 = 0;              // MODE = 010
   #409 En = 0;                // Shift Reg statrs
```

```
    #10 SEL3 = 2'b01;                  // Right Shift
    #40 SEL3 = 2'b00; SEL3 = 2'b10;            // Left Shift
    #40 $finish;
end

endmodule
```

RESULTS OF FINAL SIMULATION