Assignment 3

By: Abdelrahman Maher Hassan

Question 1

Codes:

D_Latch_active_L.v

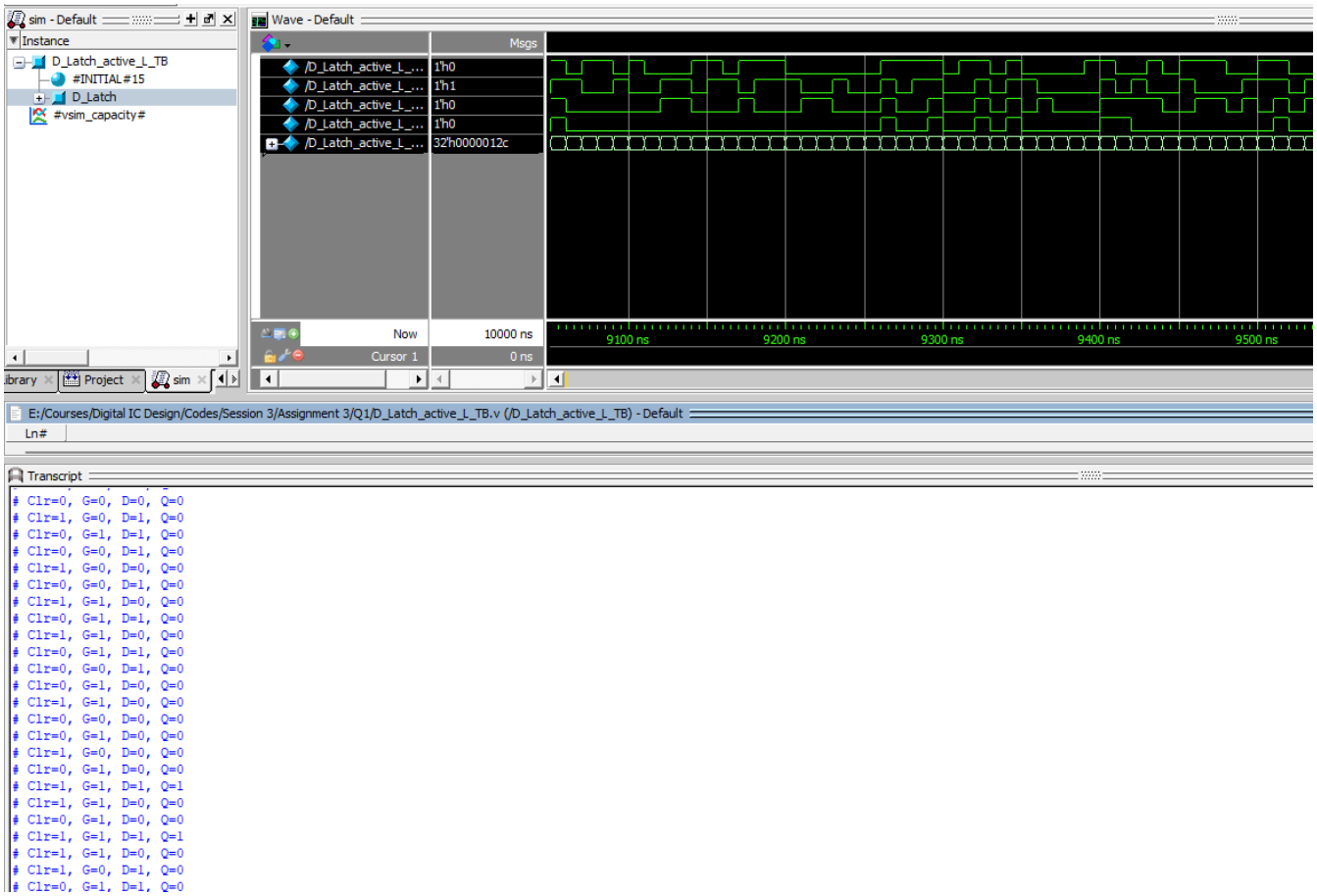
```
module D_Latch_active_L (D, G, CLRn, Q);

    input D, G, CLRn;
    output reg Q;

    always @(*)begin
        if (~CLRn)
            Q <= 0;
        else if(G)
            Q <= D;
    end

endmodule
```

Test Bench:

D_Latch_active_L_TB.v

```
module D_Latch_active_L_TB ();

    reg D, G, CLRn;
    wire Q;

    integer i = 0;

    D_Latch_active_L D_Latch (
        .D(D),
        .G(G),
        .CLRn(CLRn),
        .Q(Q)
        );

    initial begin
        CLRn = 0;
        G = 1;  //to force check this condition Q <= D

        $monitor("Clr=%b, G=%b, D=%d, Q=%b",CLRn,G,D,Q);

        for (i=0;i<100; i=i+1) begin
            D = $random;
            #10;
        end

        CLRn = 1;
        for (i=0;i<600; i=i+1) begin
            D = $random;
            #10;
        end

        for (i=0;i<300; i=i+1) begin
            CLRn = $random;
            G = $random;
            D = $random;
            #10;
        end
        $stop;
    end


endmodule
```

Simulation:


sim - Default
Instance
D_Latch_active_L_TB
#INITIAL#15
D_Latch
#vsim_capacity#
Wave - Default
Msgs
/D_Latch_active_L_...
1'h0
1'h1
32'h0000012c
Now
10000 ns
Cursor 1
0 ns
9100 ns
9200 ns
9300 ns
9400 ns
9500 ns
Project
sim
E:/Courses/Digital IC Design/Codes/Session 3/Assignment 3/Q1/D_Latch_active_L_TB.v (/D_Latch_active_L_TB) - Default
Ln#
Transcript
Clr=0, G=0, D=0, Q=0
Clr=1, G=0, D=1, Q=0
Clr=0, G=1, D=1, Q=0
Clr=0, G=0, D=1, Q=0
Clr=1, G=0, D=0, Q=0
Clr=0, G=0, D=1, Q=0
Clr=1, G=1, D=0, Q=0
Clr=0, G=1, D=1, Q=0
Clr=1, G=1, D=0, Q=0
Clr=0, G=1, D=1, Q=0
Clr=0, G=0, D=1, Q=0
Clr=0, G=1, D=0, Q=0
Clr=1, G=1, D=0, Q=0
Clr=0, G=0, D=0, Q=0
Clr=0, G=1, D=0, Q=0
Clr=1, G=0, D=0, Q=0
Clr=0, G=0, D=0, Q=0
Clr=1, G=1, D=1, Q=1
Clr=1, G=1, D=0, Q=0
Clr=0, G=1, D=0, Q=0
Clr=1, G=1, D=1, Q=1
Clr=1, G=1, D=0, Q=0
Clr=1, G=0, D=1, Q=0
Clr=0, G=1, D=1, Q=0

Question 2 (A)

Codes:

T_FF.v

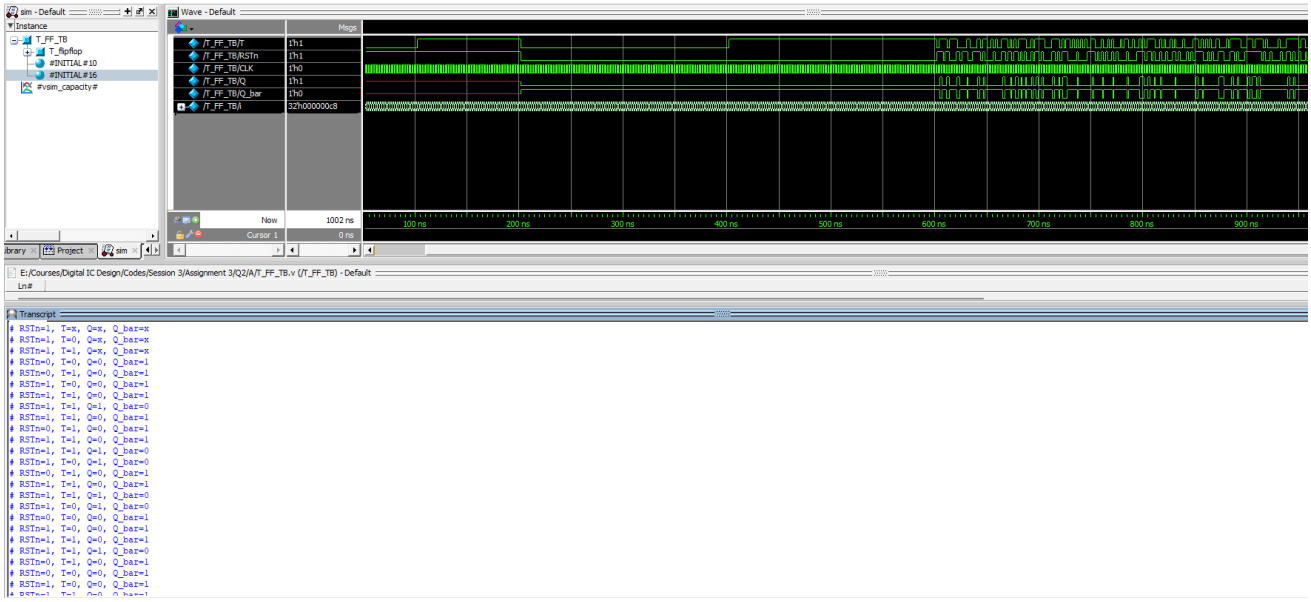
```
module T_FF (T, RSTn, CLK, Q, Q_bar);
    input  T, RSTn, CLK;
    output reg Q, Q_bar;

    always @(posedge CLK or negedge RSTn) begin

        if (~RSTn) begin
            Q <= 0;
            Q_bar <= 1;
        end
        else if (T)begin
            Q <= ~Q;
            Q_bar <= ~Q_bar;
        end
    end

endmodule
```

Test Bench:

T_FF_TB.v

```
module T_FF_TB ();

    reg  T, RSTn, CLK;
    wire Q, Q_bar;

    T_FF T_flipflop (T, RSTn, CLK, Q, Q_bar);

    integer i = 0;

    initial begin
        CLK = 0;
        forever
          #1 CLK = ~CLK;
    end

    initial begin
        $monitor("RSTn=%b, T=%b, Q=%b, Q_bar=%b",RSTn, T, Q, Q_bar);
        RSTn = 1;
        @(negedge CLK);

        for(i=0; i<50;i=i+1)begin
            T = 0;
            @(negedge CLK);
        end

        for(i=0; i<50;i=i+1)begin
            T = 1;
            @(negedge CLK);
        end

        RSTn = 0;
        for(i=0; i<100;i=i+1)begin
            T = 0;
            @(negedge CLK);
        end

        for(i=0; i<100;i=i+1)begin
            T = 1;
            @(negedge CLK);
        end

        for(i=0; i<200;i=i+1)begin
            T = $random;
            RSTn = $random;
            @(negedge CLK);
        end
        $stop;
    end
endmodule
```

Simulation:




```
# RSTn=1, T=x, Q=x, Q_bar=x
# RSTn=1, T=0, Q=x, Q_bar=x
# RSTn=1, T=1, Q=x, Q_bar=x
# RSTn=0, T=0, Q=0, Q_bar=1
# RSTn=0, T=1, Q=0, Q_bar=1
# RSTn=1, T=0, Q=0, Q_bar=1
# RSTn=1, T=1, Q=0, Q_bar=1
# RSTn=1, T=1, Q=1, Q_bar=0
# RSTn=1, T=0, Q=1, Q_bar=0
# RSTn=0, T=1, Q=0, Q_bar=1
# RSTn=1, T=0, Q=0, Q_bar=1
# RSTn=1, T=1, Q=1, Q_bar=0
# RSTn=1, T=0, Q=1, Q_bar=0
# RSTn=1, T=1, Q=0, Q_bar=1
# RSTn=0, T=0, Q=0, Q_bar=1
# RSTn=0, T=1, Q=0, Q_bar=1
# RSTn=1, T=0, Q=0, Q_bar=1
# RSTn=1, T=1, Q=1, Q_bar=0
# RSTn=0, T=1, Q=0, Q_bar=1
# RSTn=0, T=0, Q=0, Q_bar=1
# RSTn=1, T=0, Q=0, Q_bar=1
```

Question 2 (B)

Codes:

D_FF_A_L.v

```v
module D_FF_A_L (D, RSTn, CLK, Q, Qbar);
    input  D, RSTn, CLK;
    output reg Q;
    output Qbar;

    assign Qbar = ~Q;

    always@(posedge CLK or negedge RSTn) begin
        if(~RSTn) begin
            Q <= 0;
        end
        else
            Q <= D;
    end


endmodule

```

Test Bench:

D_FF_A_L_TB.v

```v
module D_FF_A_L_TB ();
    reg  D, RSTn, CLK;
    wire Q, Q_bar;

    D_FF_A_L D_flipflop (D, RSTn, CLK, Q, Q_bar);

    integer i = 0;

    initial begin
        CLK = 0;
        forever
          #1 CLK = ~CLK;
    end

    initial begin
        $monitor("RSTn=%b, D=%b, Q=%b, Q_bar=%b",RSTn, D, Q, Q_bar);
        RSTn = 0;
        @(negedge CLK);

        for(i=0; i<50;i=i+1)begin
            D = 0;
            @(negedge CLK);
        end

        for(i=0; i<50;i=i+1)begin
            D = 1;
            @(negedge CLK);
        end

        RSTn = 1;
        for(i=0; i<100;i=i+1)begin
            D = 0;
            @(negedge CLK);
        end

        for(i=0; i<100;i=i+1)begin
            D = 1;
            @(negedge CLK);
        end

        for(i=0; i<200;i=i+1)begin
            D = $random;
            RSTn = $random;
            @(negedge CLK);
        end
        $stop;
    end
endmodule
```

Simulation:


sim - Default
Instance
D_FF_A_L_TB
D_flipflop
#INITIAL#9
#INITIAL#15
#vsim_capacity#
Wave - Default
Msgs
/D_FF_A_L_TB/D
/D_FF_A_L_TB/RSTn
/D_FF_A_L_TB/CLK
/D_FF_A_L_TB/Q
/D_FF_A_L_TB/Q_bar
/D_FF_A_L_TB/i
1'h1
1'h1
1'h0
1'h1
1'h0
32'h000000c8
Now
1002 ns
Cursor 1
0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns
800 ns
Project
sim
E:/Courses/Digital IC Design/Codes/Session 3/Assignment 3/Q2/B/D_FF_A_L_TB.v (/D_FF_A_L_TB) - Default
Ln#
46
$stop;
Transcript
RSTn=1, D=1, Q=1, Q_bar=0
RSTn=1, D=0, Q=1, Q_bar=0
RSTn=1, D=0, Q=0, Q_bar=1
RSTn=0, D=0, Q=0, Q_bar=1
RSTn=1, D=0, Q=0, Q_bar=1
RSTn=1, D=1, Q=0, Q_bar=1
RSTn=1, D=1, Q=1, Q_bar=0
RSTn=0, D=1, Q=0, Q_bar=1
RSTn=0, D=0, Q=0, Q_bar=1
RSTn=1, D=0, Q=0, Q_bar=1
RSTn=1, D=1, Q=0, Q_bar=1
RSTn=1, D=1, Q=1, Q_bar=0
RSTn=0, D=0, Q=0, Q_bar=1
RSTn=1, D=1, Q=0, Q_bar=1
RSTn=1, D=1, Q=1, Q_bar=0
RSTn=0, D=1, Q=0, Q_bar=1
RSTn=0, D=0, Q=0, Q_bar=1
RSTn=0, D=1, Q=0, Q_bar=1
RSTn=1, D=0, Q=0, Q_bar=1
RSTn=0, D=1, Q=0, Q_bar=1
RSTn=0, D=0, Q=0, Q_bar=1
RSTn=1, D=1, Q=0, Q_bar=1
RSTn=1, D=1, Q=1, Q_bar=0

Question 2 (C)

Codes:

AFF_Paramterized_RSTn.v | D_FF_A_L.v | T_FF.v | AFF_Paramterized_RSTn_TB1.v

```
1  module AFF_Paramterized_RSTn (D, RSTn, CLK, Q, Q_Bar);
2
3      parameter Type = "DFF";
4
5      input D, RSTn, CLK;
6      output Q, Q_Bar;
7
8      generate
9          if (Type == "DFF" )
10             D_FF_A_L D_Flip_Flop (D, RSTn, CLK, Q, Q_Bar);
11         else begin
12             T_FF Toggle_Flip_Flop (D, RSTn, CLK, Q, Q_Bar);
13         end
14     endgenerate
15
16 endmodule
```

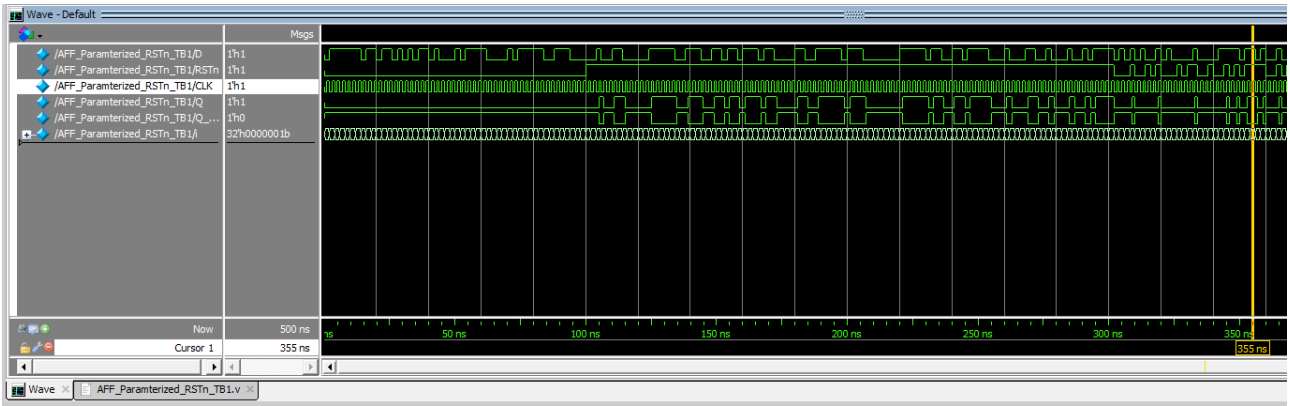
Question 2 (D)

Test Bench 1:

```
File  Edit  Selection  Find  View  Goto  Tools  Project  Preferences  Help
AFF_Paramterized_RSTn.v | D_FF_A_L.v | T_FF.v | AFF_Paramterized_RSTn
```

```
1  module AFF_Paramterized_RSTn_TB1 ();
2
3      parameter Type = "DFF";
4
5      reg  D, RSTn, CLK;
6      wire Q, Q_Bar;
7
8      AFF_Paramterized_RSTn #(Type) DUT1_DFF (D, RSTn, CLK, Q, Q_Bar);
9
10     integer i = 0;
11
12     initial begin
13         CLK =0;
14         forever
15             #1 CLK = ~CLK;
16     end
17
18     initial begin
19         RSTn = 0;
20         for (i=0;i<50;i=i+1) begin
21             D = $random;
22             @(negedge CLK);
23         end
24
25         RSTn = 1;
26         for (i=0;i<100;i=i+1) begin
27             D = $random;
28             @(negedge CLK);
29         end
30
31         for (i=0;i<100;i=i+1) begin
32             RSTn = $random;
33             D = $random;
34             @(negedge CLK);
35         end
36         $stop;
37     end
38
39 endmodule
```

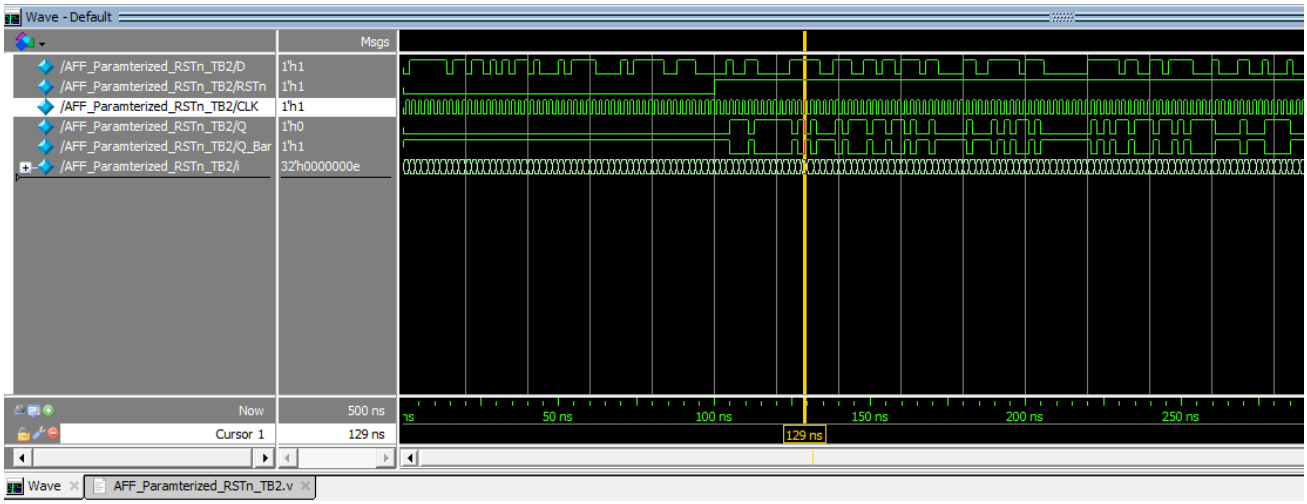
Simulation 1:



Test Bench 2:

```
AFF_Paramterized_RSTn.v | D_FF_A_L.v | T_FF.v | AFF_Paramterized_RSTn_TB
1  module AFF_Paramterized_RSTn_TB2 ();
2
3      parameter Type = "TFF";
4
5      reg  D, RSTn, CLK;
6      wire Q, Q_Bar;
7
8      AFF_Paramterized_RSTn #(Type) DUT2_TFF (D, RSTn, CLK, Q, Q_Bar);
9
10     integer i = 0;
11
12     initial begin
13         CLK =0;
14         forever
15             #1 CLK = ~CLK;
16     end
17
18     initial begin
19         RSTn = 0;
20         for (i=0;i<50;i=i+1) begin
21             D = $random;
22             @(negedge CLK);
23         end
24
25         RSTn = 1;
26         for (i=0;i<100;i=i+1) begin
27             D = $random;
28             @(negedge CLK);
29         end
30
31         for (i=0;i<100;i=i+1) begin
32             RSTn = $random;
33             D = $random;
34             @(negedge CLK);
35         end
36         $stop;
37     end
38
39 endmodule
```

Simulation 2:



Question 3

Codes:

D_FF_A_L.v | Four_Bit_Ripple_Counter.v | Four_Bit_Ripple_Counter_TB.v | Four_bit_

```
module Four_Bit_Ripple_Counter(CLK, RSTn, Out);
    input CLK, RSTn;
    output [3:0]Out;

    wire qn0, qn1, qn2, qn3;
    wire q0, q1, q2, q3;

     D_FF_A_L D1 (
        .D(qn0),
        .RSTn(RSTn),
        .CLK(CLK),
        .Q(q0),
        .Qbar(qn0)
     );

     D_FF_A_L D2 (
        .D(qn1),
        .RSTn(RSTn),
        .CLK(q0),
        .Q(q1),
        .Qbar(qn1)
     );

     D_FF_A_L D3 (
        .D(qn2),
        .RSTn(RSTn),
        .CLK(q1),
        .Q(q2),
        .Qbar(qn2)
     );
     D_FF_A_L D4 (
        .D(qn3),
        .RSTn(RSTn),
        .CLK(q2),
        .Q(q3),
        .Qbar(qn3)
     );

     assign Out = {qn3, qn2, qn1, qn0};

endmodule
```

Test Bench:

D_FF_A_L.v | Four_Bit_Ripple_Counter.v | Four_Bit_Ripple_Counter_TB.v

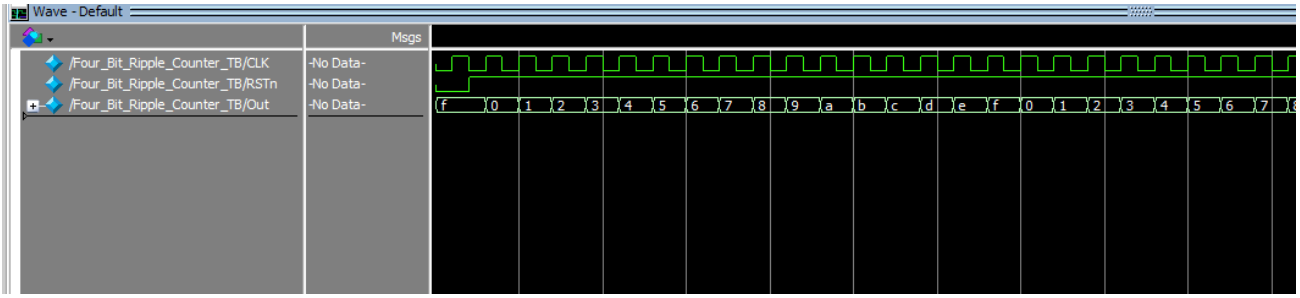
```
module Four_Bit_Ripple_Counter_TB ();
    reg CLK, RSTn;
    wire [3:0]Out;

    Four_Bit_Ripple_Counter R_Counter (CLK, RSTn, Out);

    initial begin
        CLK = 0;
        forever
            #1 CLK = ~CLK;
    end

    initial begin
        RSTn = 0;
        @(negedge CLK);

        RSTn = 1;
        repeat(50)  @(negedge CLK);
        $stop;
    end


endmodule
```

Do File:

D_FF_A_L.v | Four_Bit_Ripple_Counter.v | Four_Bit_Ripple_Counter_TB.v | Four_bit_Counter_DO.do

```
1  vlib work
2  vlog D_FF_A_L.v Four_Bit_Ripple_Counter.v Four_Bit_Ripple_Counter_TB.v
3  vsim -voptargs=+acc work.Four_Bit_Ripple_Counter_TB
4  add wave *
5  run -all
6  //#quit -sim
```

Simulation:



Question 4

Codes:

Paramterized_Shift.v | Sequential_Logic_Element.v

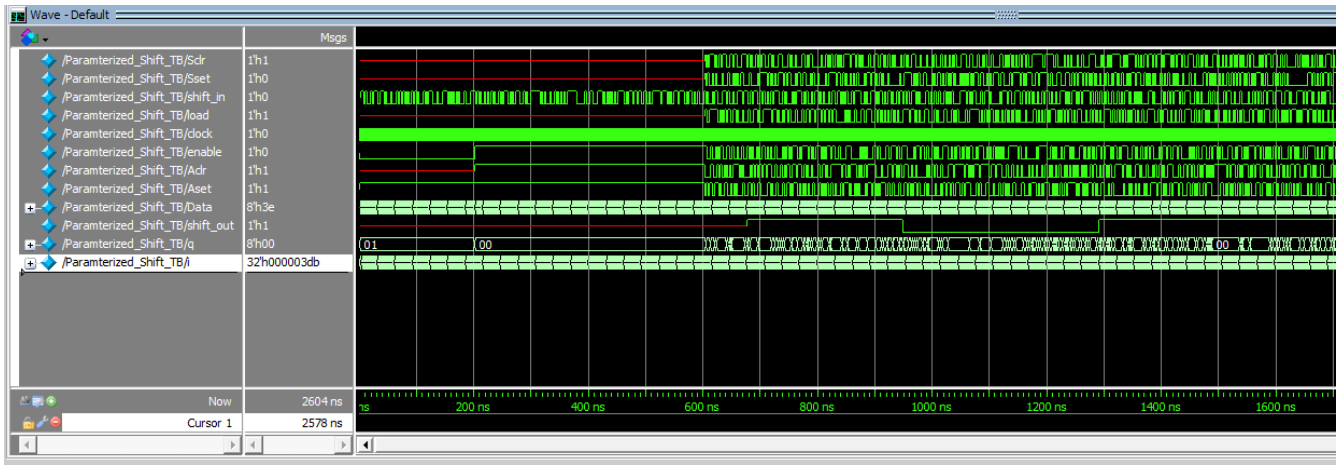
```
module Paramterized_Shift (Sclr, Sset, shift_in, load, Data, clock, enable, Aclr, Aset, shift_out, q);

    parameter LOAD_AVALUE = 1;
    parameter SHIFT_DIRECTION = "LEFT" ;
    parameter LOAD_SVALUE = 1;
    parameter SHIFT_WIDTH = 8;

    input  Sclr, Sset, shift_in, load, clock, enable, Aclr, Aset;
    input  [SHIFT_WIDTH-1:0]Data;

    output reg shift_out;
    output reg [SHIFT_WIDTH-1:0] q;

    always @(posedge clock or posedge Aclr or posedge Aset) begin

        if(Aclr)
            q <= 0;
        else if(Aset)
            q <=  LOAD_AVALUE;
        else if(enable)begin
            if(Sclr)
                q <= 0;
            else if (Sset)
                q <=  LOAD_SVALUE;
            else if (load)
                q <= Data;
            else begin
                if(SHIFT_DIRECTION == "LEFT") begin
                    shiftout <= Data[7];
                    q <= {Data[6:0],shift_in};
                end
                else begin
                    shiftout <= Data[0];
                    q <= {shift_in, Data[7:1]};
                end

            end
        end

    end

endmodule
```

Test Bench:

Paramterized_Shift.v | Paramterized_Shift_TB.v | Sequential_Logic_Element.v

```
module Paramterized_Shift_TB ();
    parameter LOAD_AVALUE = 1;
    parameter SHIFT_DIRECTION = "LEFT" ;
    parameter LOAD_SVALUE = 1;
    parameter SHIFT_WIDTH = 8;
    reg  Sclr, Sset, shift_in, load, clock, enable, Aclr, Aset;
    reg  [SHIFT_WIDTH-1:0] Data;
    wire shift_out;
    wire [SHIFT_WIDTH-1:0] q;
    integer i = 0;

    Paramterized_Shift #(LOAD_AVALUE, SHIFT_DIRECTION, LOAD_SVALUE, SHIFT_WIDTH) DUT_Param_Shift (Sclr, Sset, shift_in, load, Data, clock, enable, Aclr, Aset, shift_out, q);

    initial begin
        clock = 0;
        forever
        #1 clock = ~clock;
    end

    initial begin
    Aset = 1;
    enable = 0;
    @(negedge clock);

    for (i=0; i<100; i=i+1) begin
        Data = $random;
        shift_in = $random;
        @(negedge clock);
    end

    Aclr = 1;
    enable = 1;
    @(negedge clock);

    for (i=0; i<200; i=i+1) begin
        Data = $random;
        shift_in = $random;
        @(negedge clock);
    end

    for (i=0; i<1000; i=i+1) begin
        Sclr = $random;
        Sset= $random;
        shift_in = $random;
        load = $random;
        enable = $random;
        Aclr = $random;
        Aset = $random;
        Data = $random;
        @(negedge clock);
    end
        $stop;
    end
endmodule
```

Simulation:



Question 5

Codes:

Paramterized_Shift.v | Sequential_Logic_Element.v

```
1  module Sequential_Logic_Element(D, CLK, EN, ALn, ADn, SLn, SD, LAT, Q);
2
3      input  D, CLK, EN, ALn, ADn, SLn, SD, LAT;
4      output reg Q;
5
6      always @(posedge CLK or negedge ALn) begin
7          if(~ALn)
8              Q <= !ADn;
9          else if(LAT)
10             if(SLn)
11                 Q <= D;
12             else
13                 Q <= SD;
14         else begin
15             if(EN) begin
16                 if(SLn)
17                     Q <= D;
18                 else
19                     Q <= SD;
20             end
21         end
22
23     end
24
25 endmodule
```

Test Bench:

Sequential_Logic_Element.v | Sequential_Logic_Element_TB.v

```
1  module Sequential_Logic_Element_TB ();
2
3      reg  D, CLK, EN, ALn, ADn, SLn, SD, LAT;
4      wire Q;
5
6      Sequential_Logic_Element SLE1 (D, CLK, EN, ALn, ADn, SLn, SD, LAT, Q);
7
8      integer i = 0;
9
10     initial begin
11         CLK = 0;
12         forever
13             #1 CLK = ~CLK;
14     end
15
16     initial begin
17         ALn = 0;
18         LAT = 1;
19         SLn = 1;
20         @(negedge CLK);
21
22         for (i=0; i<50; i=i+1) begin
23             ADn = $random;
24             D = $random;
25             @(negedge CLK);
26         end
27
28         ALn = 1;
29         LAT = 1;
30         @(negedge CLK);
31
32         for (i=0; i<50; i=i+1) begin
33             ADn = $random;
34             D = $random;
35             SLn = $random;
36             @(negedge CLK);
37         end
38
39         LAT = 0;
40         @(negedge CLK);
41
42         for (i=0; i<50; i=i+1) begin
43             EN = $random;
44             D = $random;
45             SLn = $random;
46             @(negedge CLK);
47         end
48
49         for (i=0; i<200; i=i+1) begin
50             ALn = $random;
51             LAT = $random;
52             ADn = $random;
53             D   = $random;
54             SLn = $random;
55             EN  = $random;
56             @(negedge CLK);
57         end
58
59         $stop;
60     end
61 endmodule
```

Simulation: