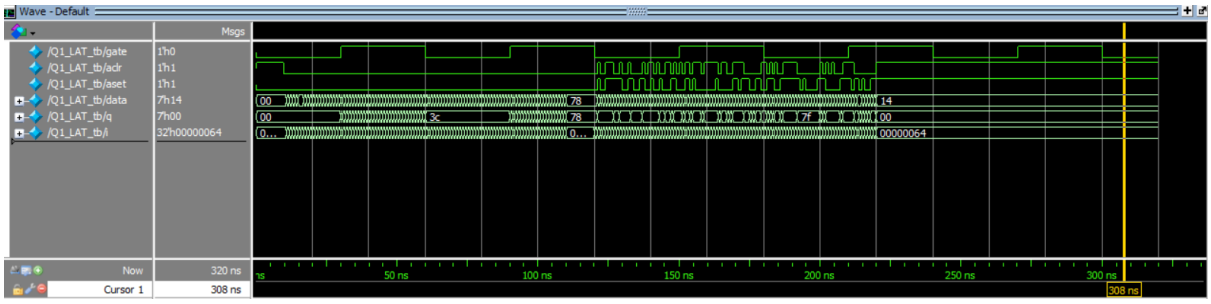
```
module Q1_LAT(gate,aclr,aset,data,q);
parameter LAT_WIDTH=1;
input gate/*work as a clk*/,aclr,aset;
input [LAT_WIDTH-1:0]data;
output reg [LAT_WIDTH-1:0]q;
always @(*) begin
	if (aclr) begin
		q<=0;
	end
	else if (aset) begin
		q<={LAT_WIDTH{1'b1}};
	end
	else if(gate) begin
		q<=data;
	end
end
endmodule
module Q1_LAT_tb();
parameter LAT_WIDTH=7;
reg gate,aclr,aset;
reg [LAT_WIDTH-1:0]data;
wire [LAT_WIDTH-1:0]q;
integer i=0;
Q1_LAT #(LAT_WIDTH) dut(gate,aclr,aset,data,q);
initial begin
gate<=0;
forever
#30 gate<=~gate;
end
initial begin
aclr=1;
```

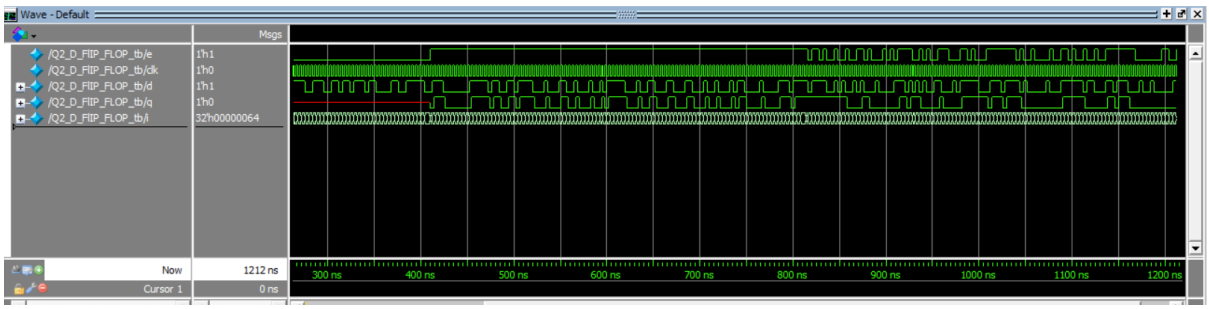
```
data=0;
aset=0;
#10 aclr=0;
for (i=0;i<100;i=i+1)begin
#1
data=$random;
end
#10
for (i=0;i<100;i=i+1)begin
#1
data=$random;
aclr=$random;
aset=$random;
end
#100;
$stop;
end
endmodule
```




```
/**************************************************************************Q2*********
**************************************************************************/

module Q2_D_FlIP_FLOP(d,e,clk,q);
parameter N=1;
input e,clk;
input [N-1:0]d;
```

```
output reg [N-1:0]q;
always @(posedge clk ) begin
        if (e)
                q<=d;
end
endmodule
module Q2_D_FlIP_FLOP_tb();
parameter N=1;
reg e,clk;
reg [N-1:0]d;
wire [N-1:0]q;
integer i=0;
 Q2_D_FlIP_FLOP #(N) dut(d,e,clk,q);
 initial begin
 clk=0;
 forever
 #2 clk=~clk;
 end
 initial begin
 d=0;
 e=0;
 #6;
 for (i=0;i<100;i=i+1)begin
@(negedge clk);
d=$random;
end
#6
e=1;
 for (i=0;i<100;i=i+1)begin
@(negedge clk);
d=$random;
```

```
end
#6;
 for (i=0;i<100;i=i+1)begin
@(negedge clk);
d=$random;
e=$random;
end
$stop;
 end

 endmodule
```



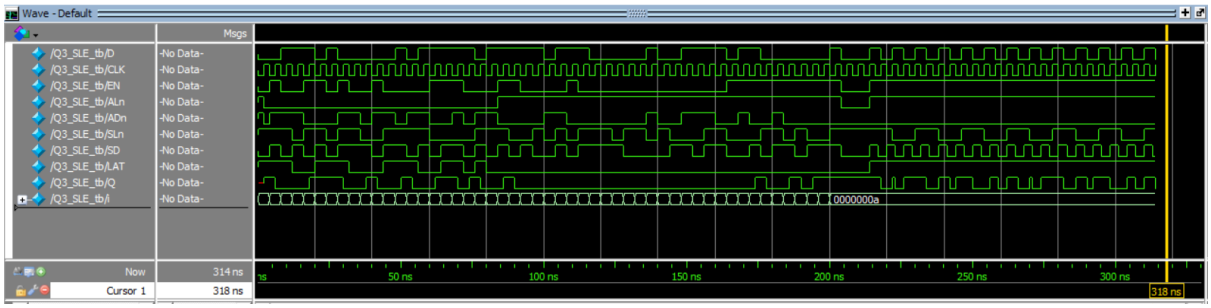

```
/**********************************************************************************Q3*
**********************************************************************/



module Q3_SLE(D,CLK,EN,ALn,ADn,SLn,SD,LAT,Q);


input D,CLK,EN,ALn,ADn,SLn,SD,LAT;
output reg Q;
reg in;
always @(posedge CLK ) begin
        if (~ALn) begin
                in=!ADn;
        end
        else if(EN) begin
```

```
                if (~SLn) begin
                        in=SD;
                end
                else begin
                        in=D;
                end
        end
end
always @(*) begin
 if (~LAT|| ~ALn) begin
                Q=in;
        end
        else if(CLK) begin
                        if (EN) begin
                                if (~SLn) begin
                                        Q=SD;
                                end
                                else
                                Q=D;
                end

        end

end
endmodule

module Q3_SLE_tb();
reg D,CLK,EN,ALn,ADn,SLn,SD,LAT;
wire Q;
integer i=0;
Q3_SLE dut(D,CLK,EN,ALn,ADn,SLn,SD,LAT,Q);
```

```
initial begin
CLK=0;
forever
#2 CLK=~CLK;
end
initial begin
D=0;
EN=0;
ALn=1;
ADn=1;
SLn =1;
SD=0;
LAT=1;
#2
ALn=0;
ADn=0;
for (i=0;i<20;i=i+1)begin
@(negedge CLK);
D=$random;
EN=$random;
ADn=$random;
SLn =$random;
SD=$random;
LAT=$random;
end
#4
ALn=1;
LAT=0;
for (i=0;i<10;i=i+1)begin
@(negedge CLK);
D=$random;
```

```
EN=$random;
ADn=$random;
SLn =$random;
SD=$random;
end
#4
ALn=1;
LAT=0;
EN=0;
for (i=0;i<10;i=i+1)begin
@(negedge CLK);
D=$random;
ADn=$random;
SLn =$random;
SD=$random;
end
#4
ALn=1;
LAT=0;
EN=1;
for (i=0;i<10;i=i+1)begin
@(negedge CLK);
D=$random;
ADn=$random;
SLn =$random;
SD=$random;
end
#4
ALn=1;
LAT=1;
for (i=0;i<10;i=i+1)begin
```

```
D=$random;
EN=$random;
ADn=$random;
SLn =$random;
SD=$random;
end
D=0;
EN=0;
ALn=0;
ADn=0;
SLn =1;
SD=0;
LAT=0;
#10;
$stop;
end
endmodule
```



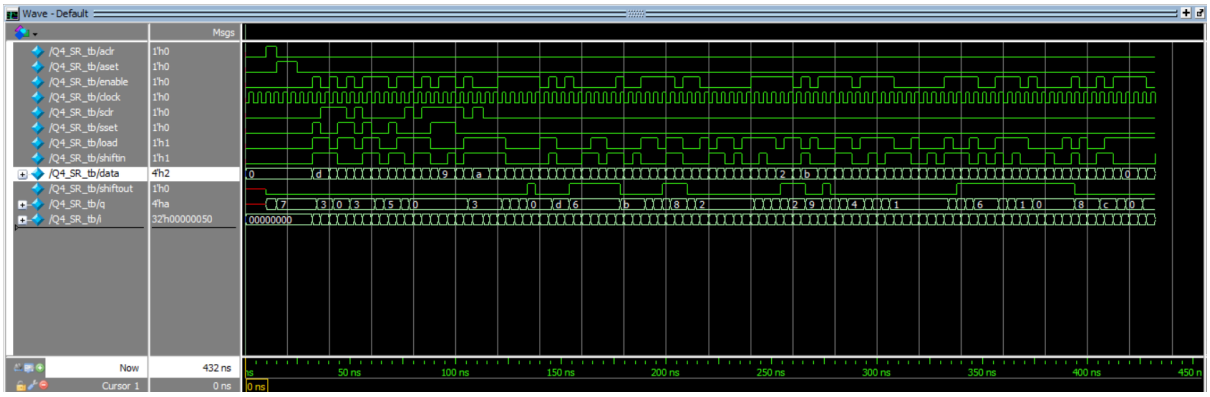

```
/**************************************************************************************
****Q4*************************************************************/


module Q4_SR(sclr,sset,shiftin,load,data,clock,enable,aclr,aset,shiftout,q);
parameter LOAD_AVALUE =7;
parameter SHIFT_DIRICTION ="LEFT";
parameter LOAD_SVALUE =3;
parameter SHIFT_WIDTH =4;
```

```
input sclr,sset,shiftin,load,clock,enable,aclr,aset;
input [SHIFT_WIDTH-1:0]data;
output reg shiftout;
output reg [SHIFT_WIDTH-1:0]q;
always @(posedge clock or posedge aclr or posedge aset) begin
        if (aclr) begin
                shiftout=0;
                q=0;
        end
        else if (aset) begin
                q=LOAD_AVALUE;
        end
        else if (enable) begin
                if(sclr) begin
                q=0;
        end
        else if (sset) begin
                q=LOAD_SVALUE;
        end
        else if (load) begin
                q=data;
        end
        else if (SHIFT_DIRICTION=="RIGHT") begin
                shiftout<=q[0];
                q<={shiftin,q[SHIFT_WIDTH-1:1]};
        end
        else begin
                shiftout<=q[SHIFT_WIDTH-1];
                q<={q[SHIFT_WIDTH-2:0],shiftin};
        end
        end
```

```
end



endmodule


module Q4_SR_tb();
parameter LOAD_AVALUE =7;
parameter SHIFT_DIRICTION ="RIGHT";
parameter LOAD_SVALUE =3;
parameter SHIFT_WIDTH =4;
reg sclr,sset,shiftin,load,clock,enable,aclr,aset;
reg [SHIFT_WIDTH-1:0]data;
wire shiftout;
wire [SHIFT_WIDTH-1:0]q;
Q4_SR
#(.SHIFT_DIRICTION(SHIFT_DIRICTION),.LOAD_AVALUE(LOAD_AVALUE),.LOAD_SVALUE(LOAD_SVAL
UE),.SHIFT_WIDTH(SHIFT_WIDTH)) dut (sclr,sset,shiftin,load,data,clock,enable,aclr,aset,shiftout,q);
integer i=0;
initial begin
clock=0;
forever
#2 clock=~clock;
end
initial begin
        sclr=0;
        sset=0;
        shiftin=0;
        load=0;
        enable=0;
        aclr=0;
        aset=0;
```

```
        data=0;
        #10
        aclr=1;
        #5
        aclr=0;
        aset=1;
        #10
        aset=0;
        #5;
        for (i=0;i<20;i=i+1)begin
                @(negedge clock);
        sclr=$random;
        sset=$random;
        shiftin=$random;
        load=$random;
        enable=$random;
        data=$random;
        end
        #5sclr=0;
        sset=0;
        for (i=0;i<80;i=i+1)begin
                @(negedge clock);

        shiftin=$random;
        load=$random;
        enable=$random;
        data=$random;
        end
        $stop;
end
```

endmodule



/**********Q5**********/

/**********1**********/

```
module Q5_1_COUNTER(set,clk,out);
input clk,set;
output reg [3:0]out;
always @(posedge clk or negedge set) begin
        if (~set) begin
                out=4'b1111;

        end
        else begin
                out=out+4'd1;
        end
end
endmodule
```

/**********2**********/

```
module Q5_2_D_FLIP_FLOP(d,rstn,clk,q,qbar);
input d,rstn,clk;
output reg q,qbar;
always @(posedge clk or negedge rstn) begin
        if (~rstn) begin
```

```
                q=0;

        end
        else  begin
                q=d;
        end
        qbar=!q;
end
endmodule

/*****************3**************/
module Q5_3_COUNTER(rstn,clk,out);
input clk,rstn;
output  [3:0]out;
wire q1,q2,q0,q3;
Q5_2_D_FLIP_FLOP dff1(out[0],rstn,clk,q0,out[0]);
Q5_2_D_FLIP_FLOP dff2(out[1],rstn,q0,q1,out[1]);
Q5_2_D_FLIP_FLOP dff3(out[2],rstn,q1,q2,out[2]);
Q5_2_D_FLIP_FLOP dff4(out[3],rstn,q2,q3,out[3]);
endmodule
/***************4****************/
module Q5_4_tb();
reg clk,rstn;
wire  [3:0]out1,out2;
integer i=0;
Q5_3_COUNTER d1(rstn,clk,out1);
Q5_1_COUNTER d2(rstn,clk,out2);
initial begin
        clk=0;
        forever
        #2 clk = ~clk;
```

```
end
initial begin
rstn=0;
#5
rstn=1;
for (i=0;i<1000;i=i+1)begin
@ (edge clk);
if(out2 != out1)
begin
        $display("Error");
        $stop;
end
end
$stop;
end
initial begin
$monitor("out1 = %d    out2=%d",out1,out2);
end
endmodule
```