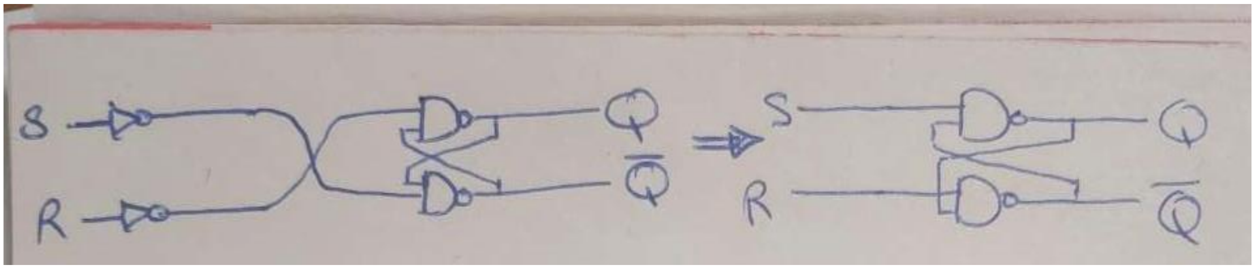
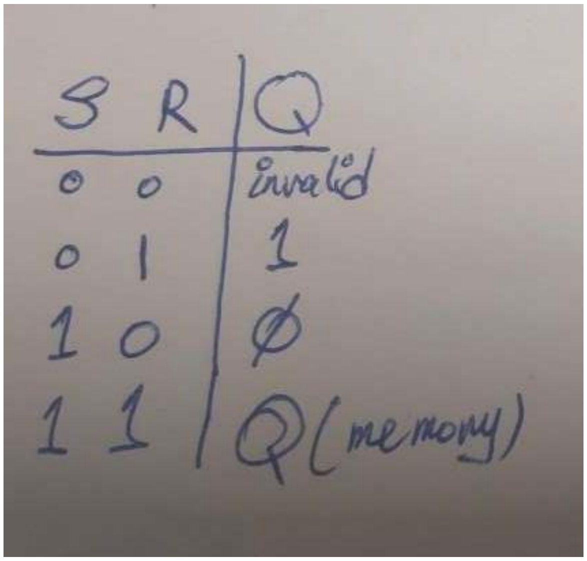
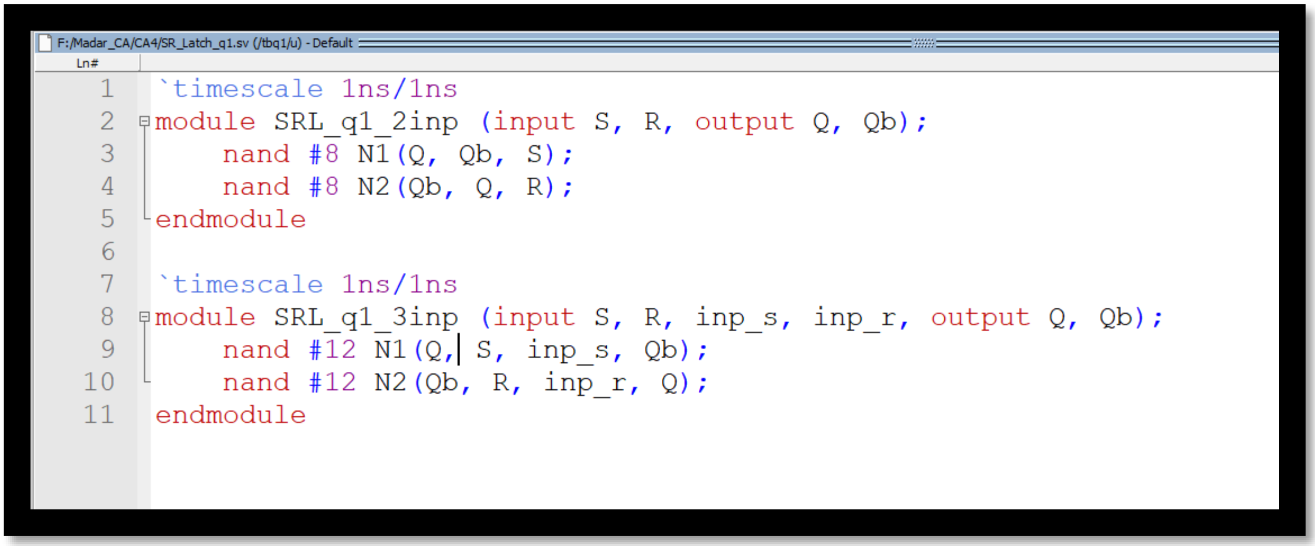
Q1:

Part a:



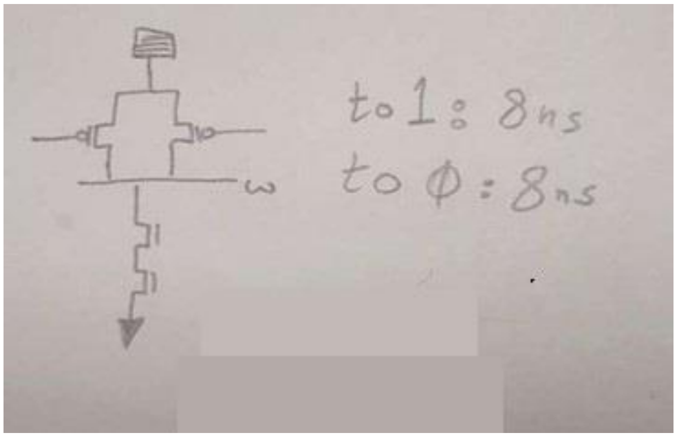


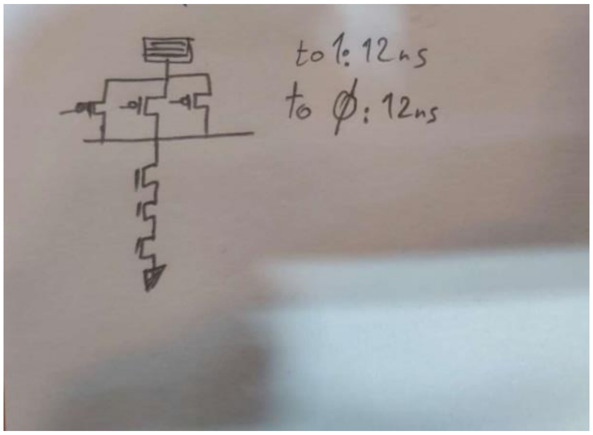
S	R	Q
0	0	invalid
0	1	1
1	0	Ø
1	1	Q(memory)



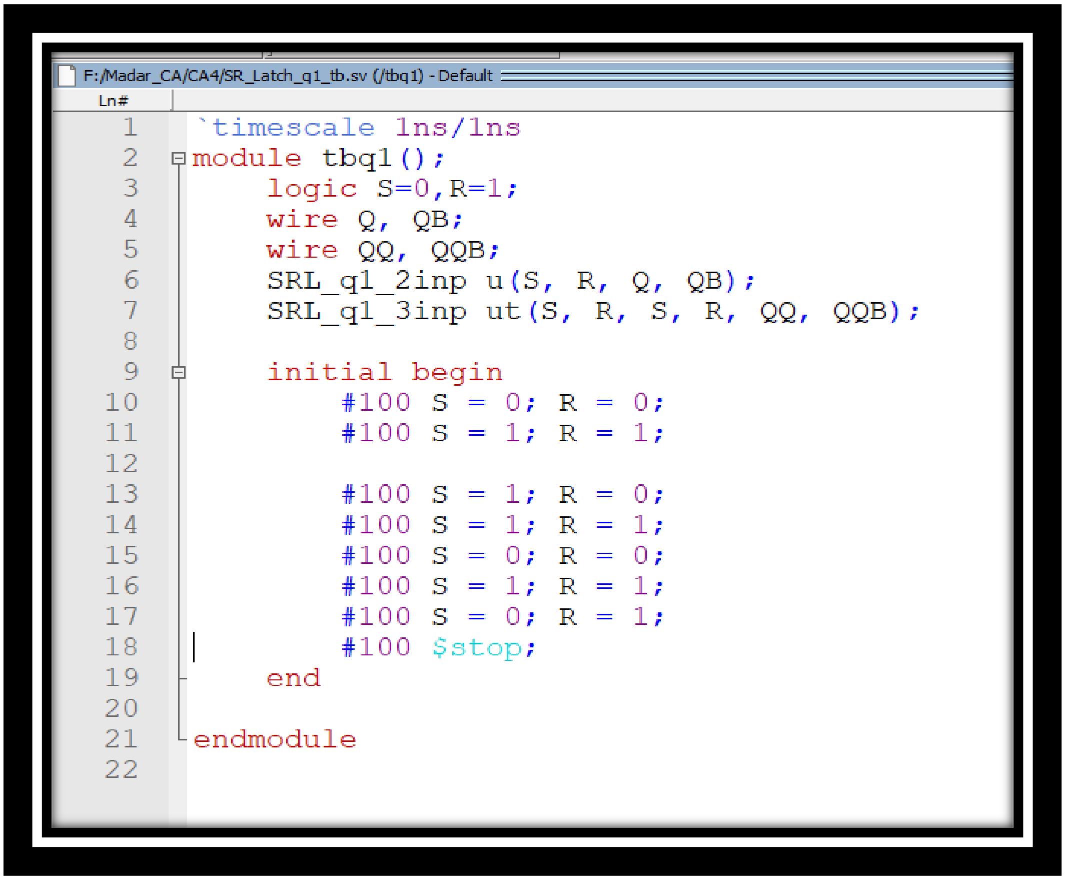
```
`timescale 1ns/1ns
module SRL_q1_2inp (input S, R, output Q, Qb);
    nand #8 N1(Q, Qb, S);
    nand #8 N2(Qb, Q, R);
endmodule

`timescale 1ns/1ns
module SRL_q1_3inp (input S, R, inp_s, inp_r, output Q, Qb);
    nand #12 N1(Q, S, inp_s, Qb);
    nand #12 N2(Qb, R, inp_r, Q);
endmodule
```

Part b:

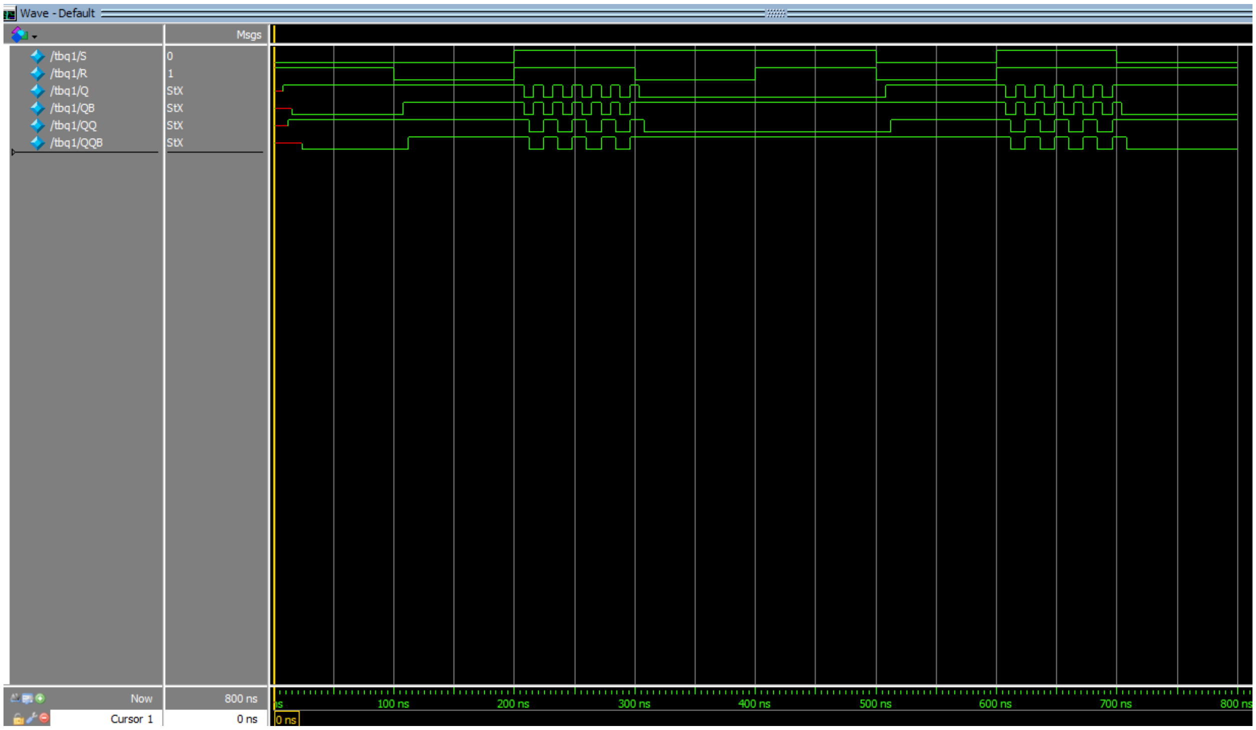




Part c:



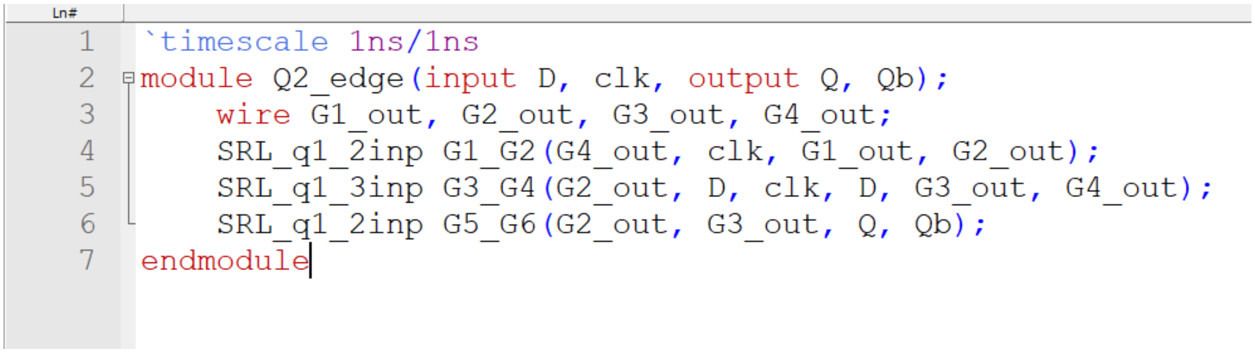
```
`timescale 1ns/1ns
module tbq1();
    logic S=0,R=1;
    wire Q, QB;
    wire QQ, QQB;
    SRL_q1_2inp u(S, R, Q, QB);
    SRL_q1_3inp ut(S, R, S, R, QQ, QQB);

    initial begin
        #100 S = 0; R = 0;
        #100 S = 1; R = 1;

        #100 S = 1; R = 0;
        #100 S = 1; R = 1;
        #100 S = 0; R = 0;
        #100 S = 1; R = 1;
        #100 S = 0; R = 1;
        #100 $stop;
    end

endmodule
```



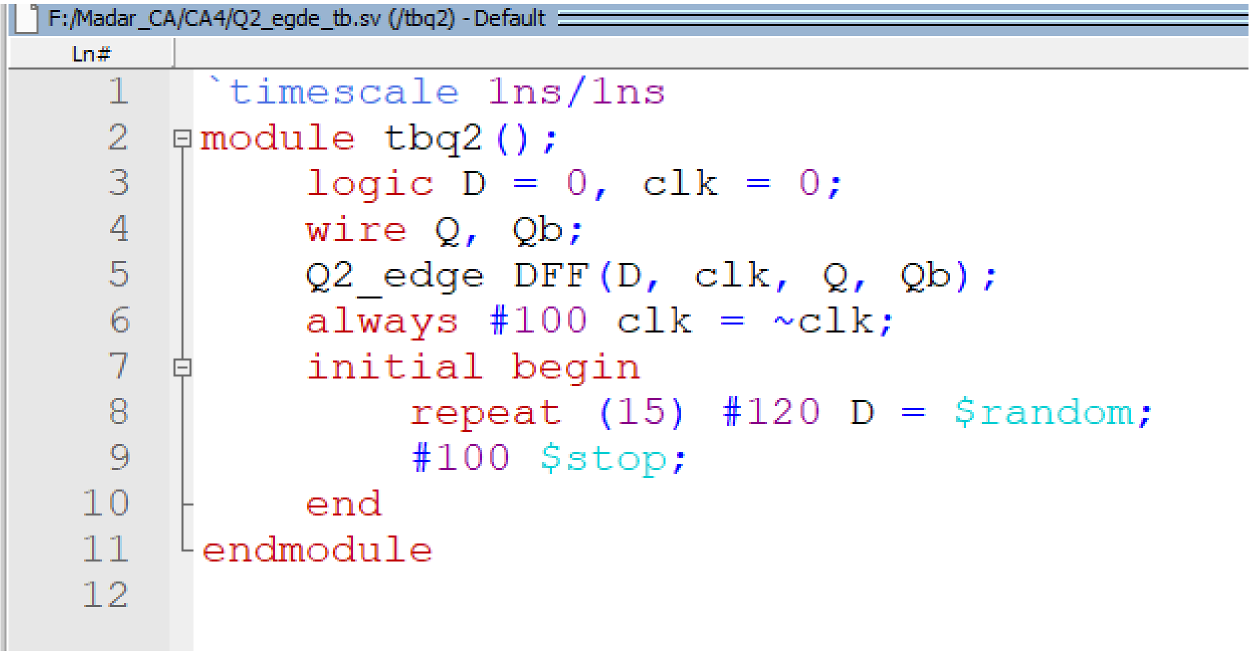
When S & R are simultaneously active, the loss of memory occurs. And because they are active low, when S & R are 0, that memory loss happens. When we change 00 to 11 we can see that because there are 2 ways to calculate the Q & Q complement and because of these things we see waveform changes quickly (somehow glitch occurs).

Q2:

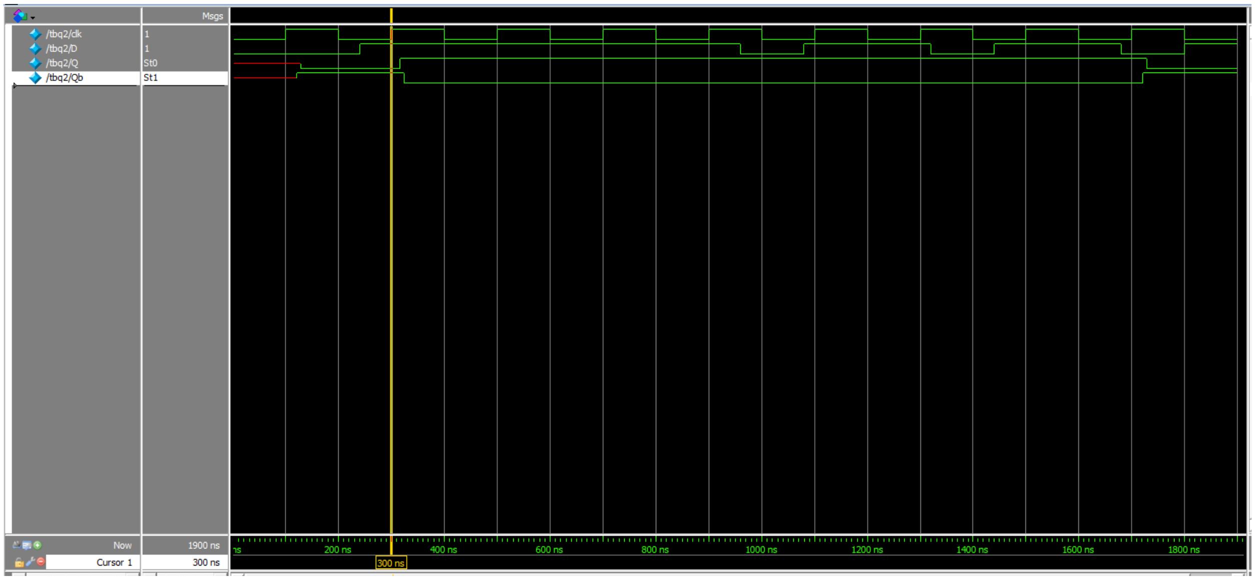
Part a:

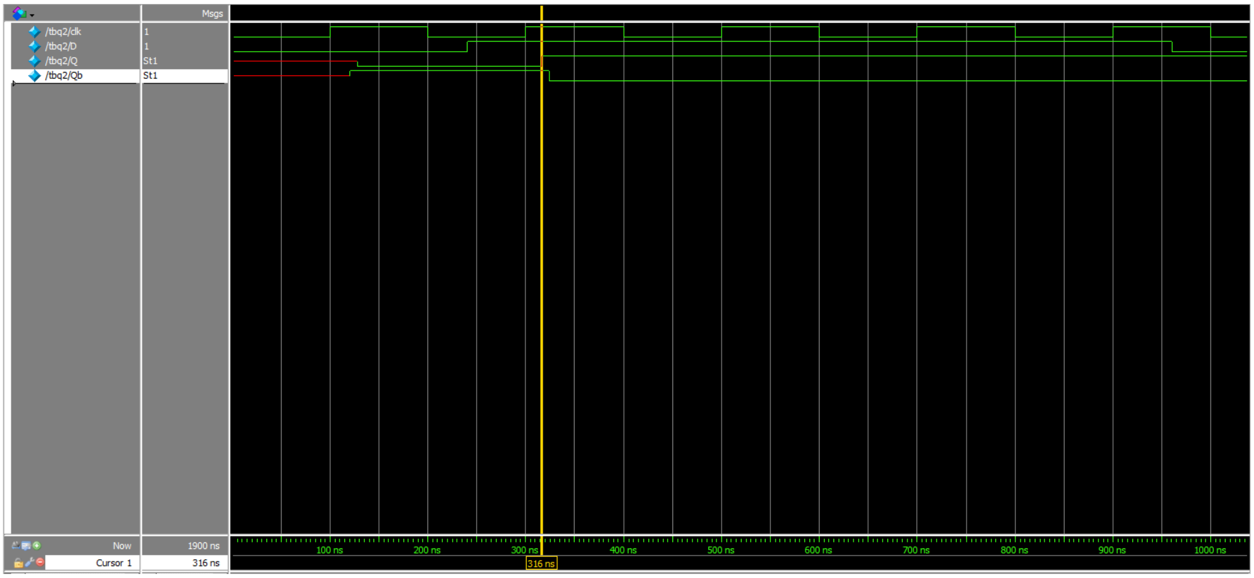


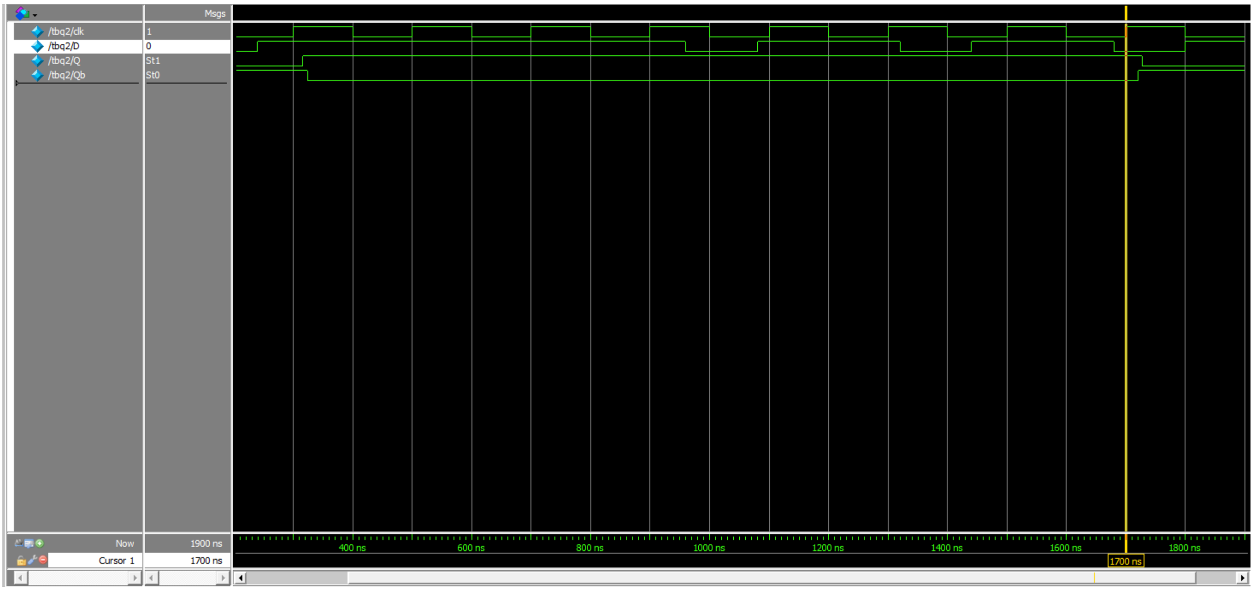
```
`timescale 1ns/1ns
module Q2_edge(input D, clk, output Q, Qb);
    wire G1_out, G2_out, G3_out, G4_out;
    SRL_q1_2inp G1_G2(G4_out, clk, G1_out, G2_out);
    SRL_q1_3inp G3_G4(G2_out, D, clk, D, G3_out, G4_out);
    SRL_q1_2inp G5_G6(G2_out, G3_out, Q, Qb);
endmodule
```

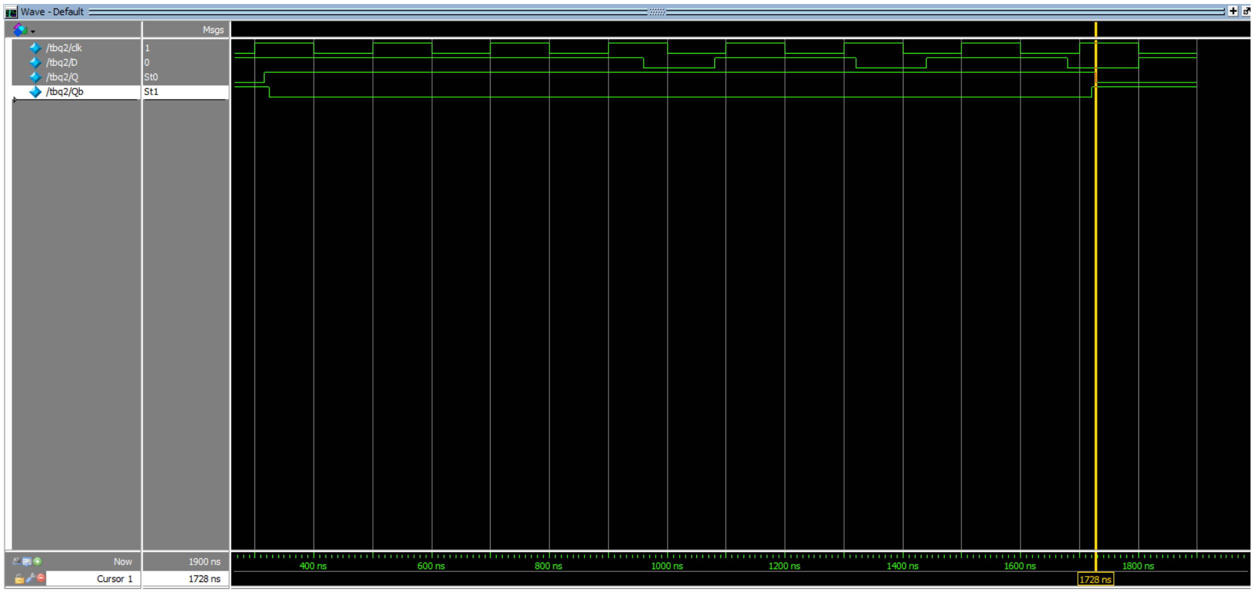


```
`timescale 1ns/1ns
module tbq2();
    logic D = 0, clk = 0;
    wire Q, Qb;
    Q2_edge DFF(D, clk, Q, Qb);
    always #100 clk = ~clk;
    initial begin
        repeat (15) #120 D = $random;
        #100 $stop;
    end
endmodule
```


Msgs
/tbq2/clk 1
/tbq2/D 1
/tbq2/Q St0
/tbq2/Qb St1
Now 1900 ns
Cursor 1 300 ns
ns
200 ns
400 ns
600 ns
800 ns
1000 ns
1200 ns
1400 ns
1600 ns
1800 ns
300 ns


Msgs
/tbq2/clk 1
/tbq2/D 1
/tbq2/Q St1
/tbq2/Qb St1
Now 1900 ns
Cursor 1 316 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns
800 ns
900 ns
1000 ns
316 ns





At the positive edge of the clock, the output will get the value of D input.

(1 to 0) :

Q = 28

Qbar = 20

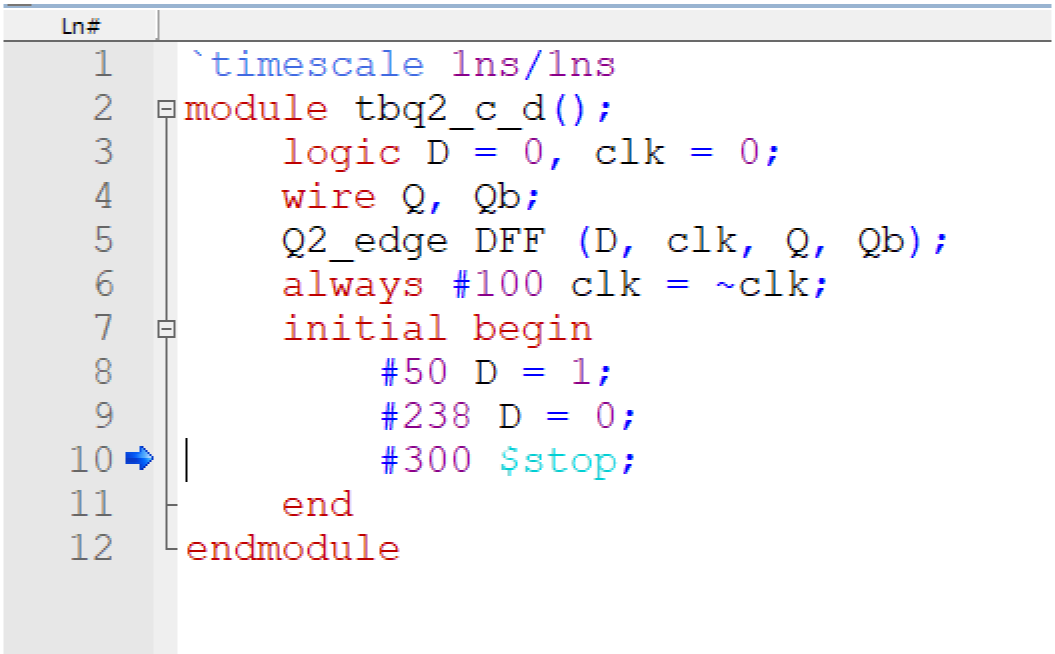
(0 to 1):

Q = 16

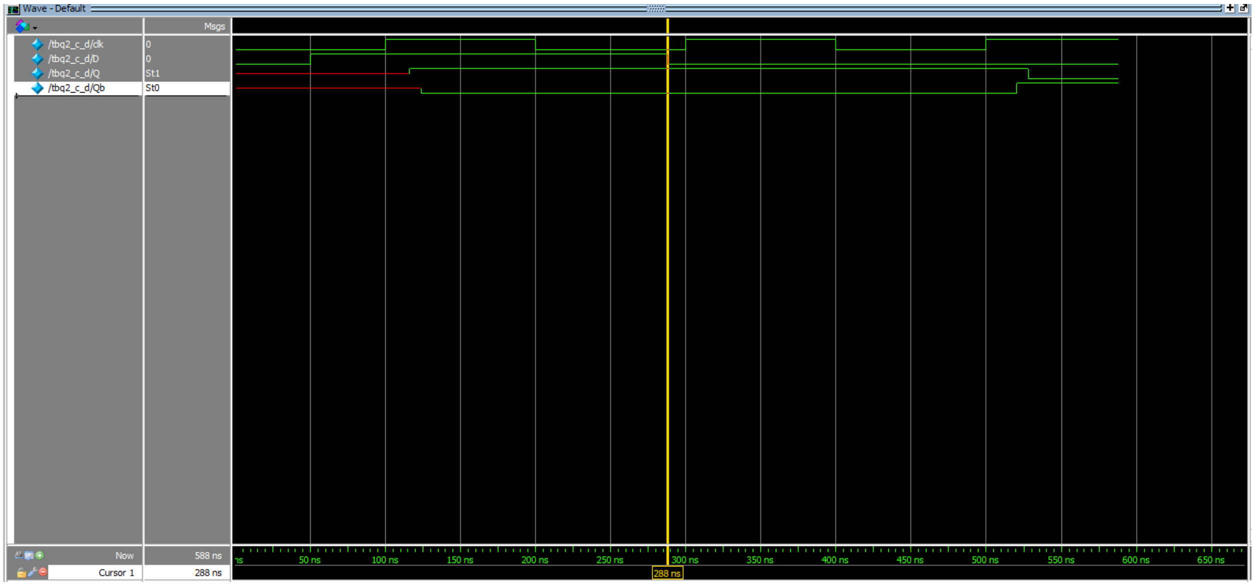
Qbar = 24

Part c:

Setup time is 13ns 🙄

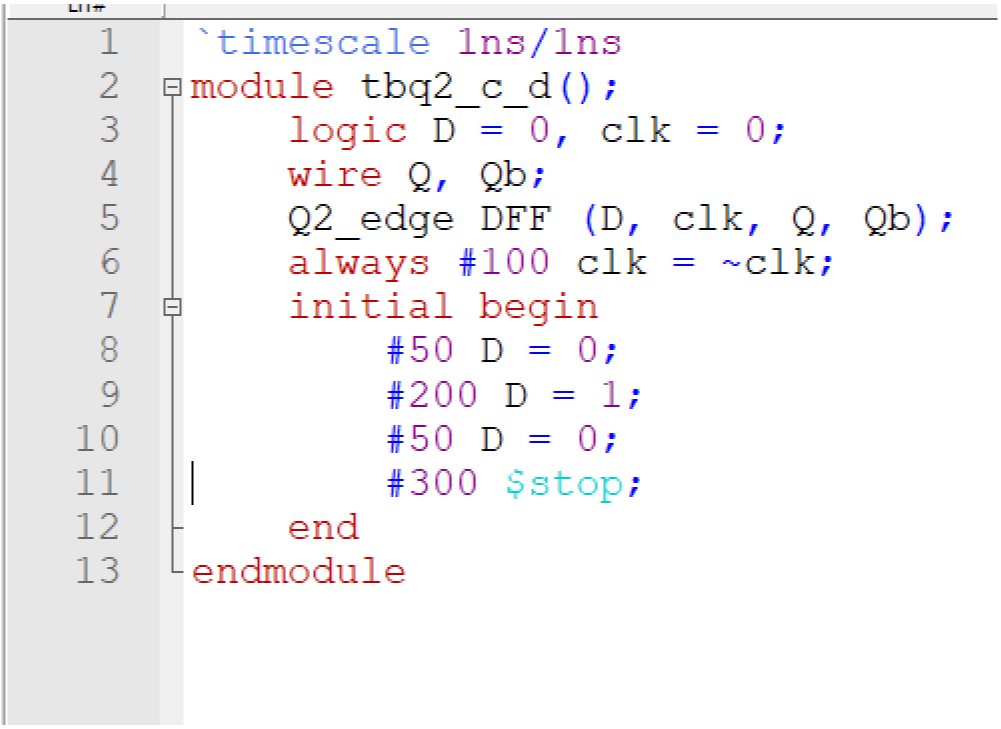


```
`timescale 1ns/1ns
module tbq2_c_d();
    logic D = 0, clk = 0;
    wire Q, Qb;
    Q2_edge DFF (D, clk, Q, Qb);
    always #100 clk = ~clk;
    initial begin
        #50 D = 1;
        #238 D = 0;
        #300 $stop;
    end
endmodule
```

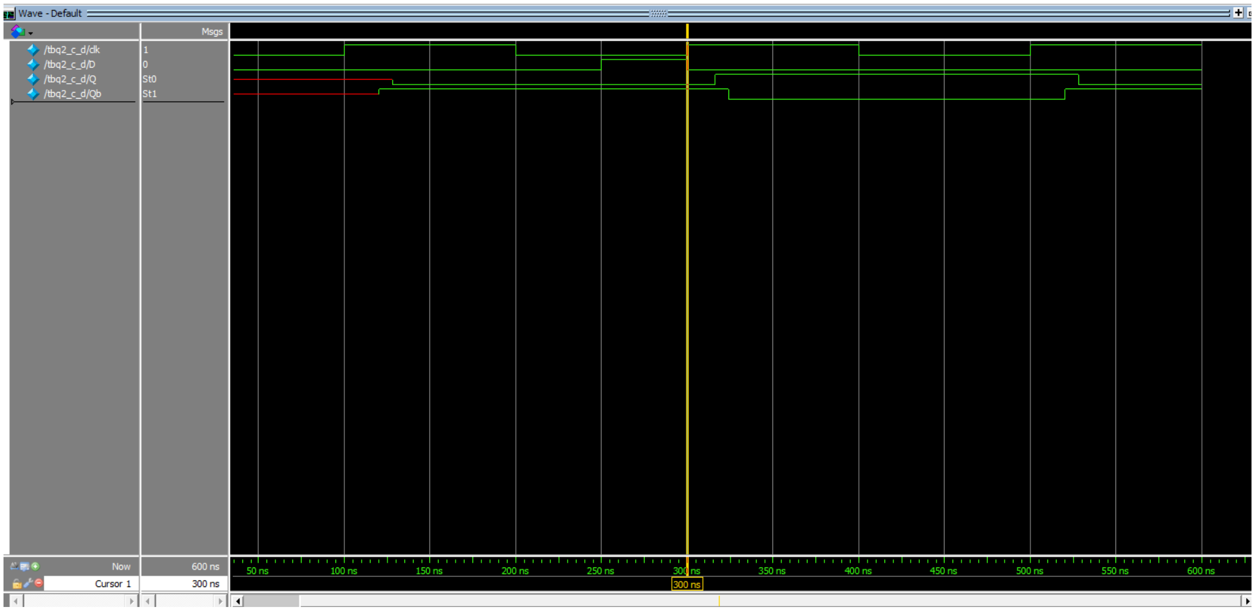


Part d:

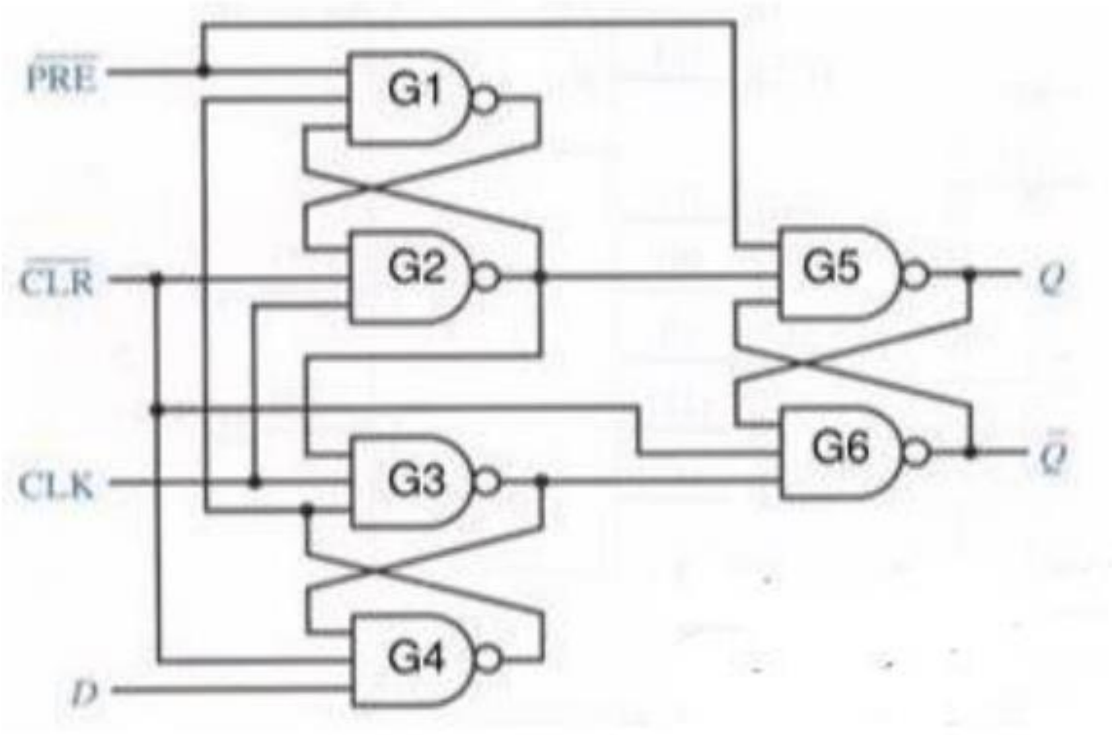
Hold time is 1ns 😳



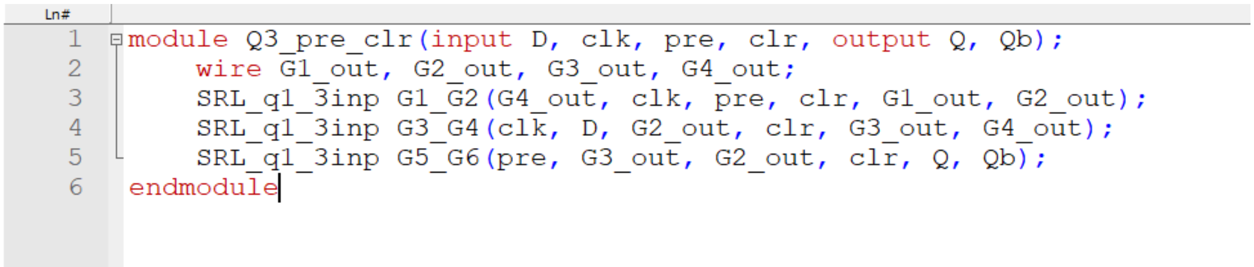
```
`timescale 1ns/1ns
module tbq2_c_d();
    logic D = 0, clk = 0;
    wire Q, Qb;
    Q2_edge DFF (D, clk, Q, Qb);
    always #100 clk = ~clk;
    initial begin
        #50 D = 0;
        #200 D = 1;
        #50 D = 0;
        #300 $stop;
    end
endmodule
```



Question 3:

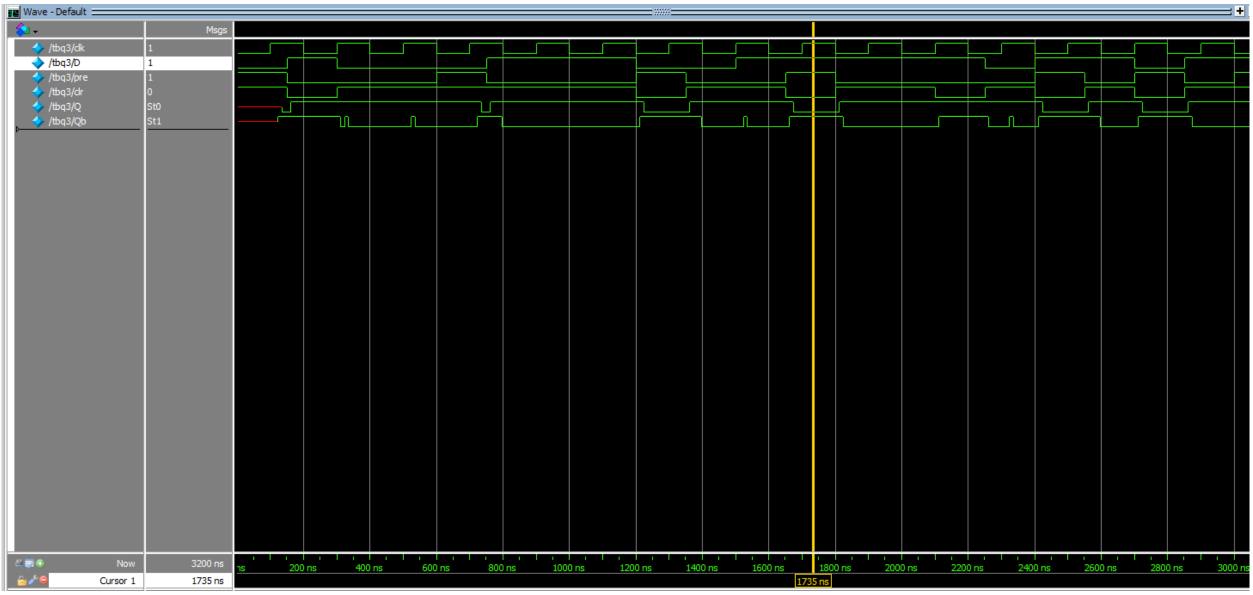


Part e:



```
module Q3_pre_clr(input D, clk, pre, clr, output Q, Qb);
    wire G1_out, G2_out, G3_out, G4_out;
    SRL_q1_3inp G1_G2(G4_out, clk, pre, clr, G1_out, G2_out);
    SRL_q1_3inp G3_G4(clk, D, G2_out, clr, G3_out, G4_out);
    SRL_q1_3inp G5_G6(pre, G3_out, G2_out, clr, Q, Qb);
endmodule
```

Part f:



When D changes from 0 to 1: delay(Q) = 24ns and delay(Qb) = 36ns

When D changes from 1 to 0: delay(Q) = 36ns and delay(Qb) = 24ns

When Pre is active: delay(Q) = 12ns and delay(Qb) = 48ns

When CLR is active: delay(Q) = 24ns and delay(Qb) = 12ns

Part g:

the output retains what it was (nothing happens), because PRE and CLR are asynchronous and they are independent from clock. When we clock, if one of PRE or CLR is active, it doesn't pay attention to clock and the output changes to PRE(Q = 1) or CLR(Q = 0).

Part h:

PRE → Q = 1

CLR → Q = 0

PRE and CLR → Q = Qb = 1 → memory loss occurs.