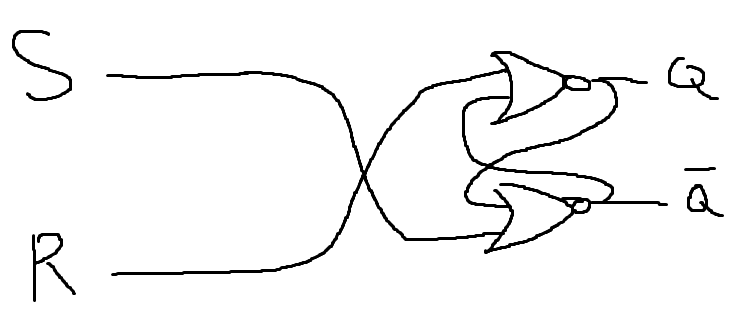
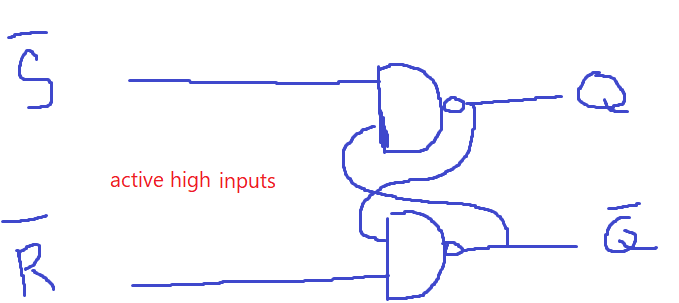
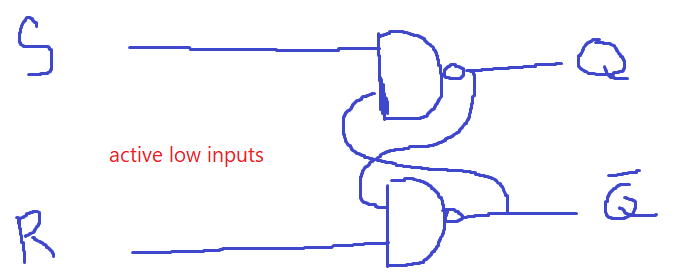
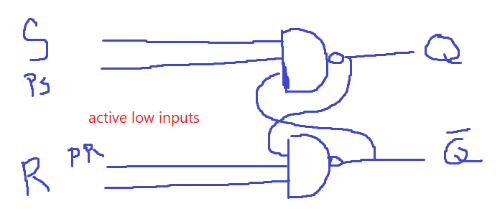
Q1:

a)



module SR\_2\_input (input S, R, output Q, QB);

    nand #8 (Q, S, QB);

    nand #8 (QB, R, Q);

endmodule

module SR\_3\_input (input S, PS, R, PR,  output Q, QB);

    nand #12 (Q, PS, S, QB);

    nand #12 (QB, PR, R, Q);

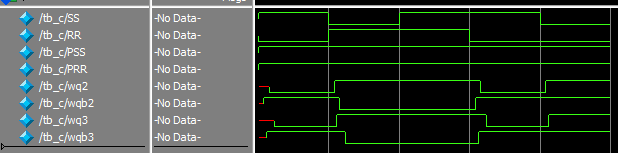
endmodule

b)

2\_input\_nand delay = 8ns.

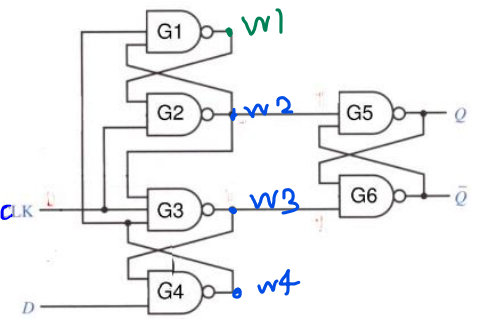
3\_input\_nand delay = 12ns.

c)



Q2:

a)



System Verilog code:

module SR\_2\_input (input S, R, output Q, QB);

    nand #8 (Q, S, QB);

    nand #8 (QB, R, Q);

endmodule

module SR\_3\_input (input S, PS, R, PR,  output Q, QB);

    nand #12 (Q, PS, S, QB);

    nand #12 (QB, PR, R, Q);

endmodule

module D\_flip\_flop (input D, clk, output Q, QB);

wire w1, w2, w3, w4;

SR\_2\_input L1 (w4, clk, w1, w2);

SR\_3\_input L2 (w2, clk, D, 1'b1, w3, w4);

SR\_2\_input L3 (w2, w3, Q, QB);

endmodule

testbench:

module tb\_b\_q2 ();

    `timescale 1ns/1ns

    logic D, clk;

    wire Q, QB;

    D\_flip\_flop my\_ff(D, clk, Q, QB);

    initial {D, clk} = 2'b00;

    initial repeat (30) #100 clk = ~clk;

    // initial repeat (20) #98 D = ~D;

    initial begin

        #100

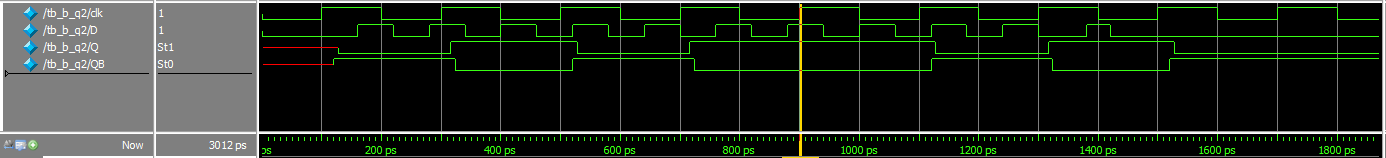
        repeat (20) #60 D = ~D;

        #80 D = ~D;

        #40 D = ~D;

    end

endmodule



It works correctly unless we have a T\_setup or hold violation, which the output in these situations should be unknown or X, BUT since we have simulation limit for our modelsim, it wont work correctly if we don’t respect T\_Setup & T\_hold time.  
And that’s why we can see f.e 1 on the output at 700ps.

C & d)

T\_setup: Before the riseedge of the clk:

we have to stop transitions of G4 gates inputs (D & clk), And let the nand gate propagate its value on its output.

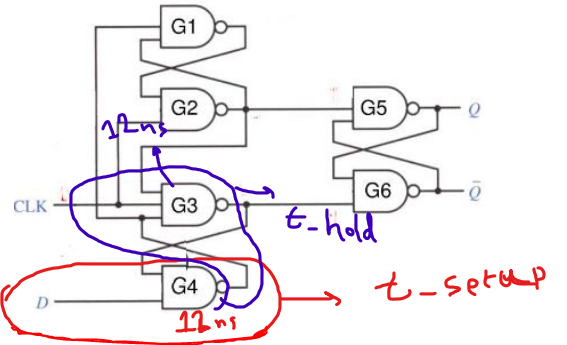
so the delay for this beautiful patience will be equal to G4s delay.  
nand\_3\_input delay= 12ns.

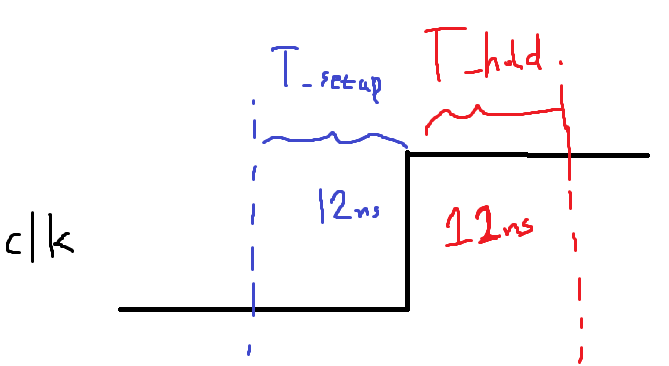
------------------------------------------------------------------------------------------------------------------------------------------------------------

T\_Hold: After the riseedge of the clk:

clk is an input for SR\_latch (G3, G4), so just like D input we will have…  
To stop transitions of G3 gates inputs (clk) , And let the nand gate propagate its value on its output.  
And since we know we wont have clk transition in our circuit, we have to make D stop its transitions and wait until G3 propagate its value. because if we let D changes its value, the output of G4 wont be available for G3 input.

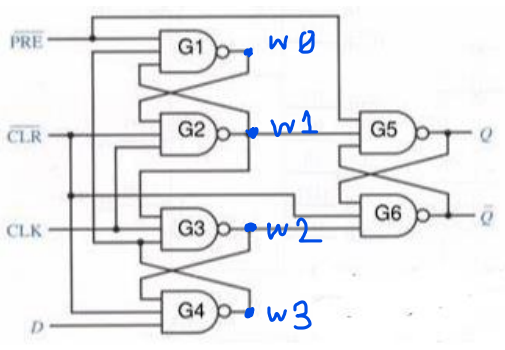
so the delay for this beautiful patience will be equal to G3s delay.  
nand\_3\_input delay= 12ns.





Q3:

e)



Here is the system Verilog code:

module SR\_2\_input (input S, R, output Q, QB);

    nand #8 (Q, S, QB);

    nand #8 (QB, R, Q);

endmodule

module SR\_3\_input (input S, PS, R, PR,  output Q, QB);

    nand #12 (Q, PS, S, QB);

    nand #12 (QB, PR, R, Q);

endmodule

module D\_flip\_flop\_PR (input D, clk, PRE, CLR,  output Q, QB);

    wire wi[3:0];

    SR\_3\_input L1 (wi[3], ~ PRE, clk, ~ CLR, wi[0], wi[1]);

    SR\_3\_input L2 (wi[1], clk, D, ~CLR, wi[2], wi[3]);

    SR\_3\_input L3 (wi[1], ~ PRE, wi[2], ~ CLR, Q, QB);

endmodule

The T\_Setup & T\_Hold delays for this circuit are equal to G3 & G4 NAND gates:  
T\_Setup = 12ns.  
T\_Hold = 12ns.

f)

testbench without PRE & CLR transition:

module tb\_q3\_b ();

    `timescale 1ns/1ns

    logic D, clk, PRE, CLR;

    wire Q, QB;

    D\_flip\_flop\_PR my\_ff(D, clk, PRE, CLR, Q, QB);

    initial {D, clk, PRE, CLR} = 4'b0;

    initial repeat (15) #100 clk = ~clk;

    // initial repeat (20) #98 D = ~D;

    initial begin

        #100

        repeat (20) #60 D = ~D;

        #80 D = ~D;

        #40 D = ~D;

    end

endmodule

testbench with PRE transition:

module tb\_q3\_g ();

    `timescale 1ns/1ns

    logic D, clk, PRE, CLR;

    wire Q, QB;

    D\_flip\_flop\_PR my\_ff(D, clk, PRE, CLR, Q, QB);

    initial {D, clk, PRE, CLR} = 4'b0;

    initial repeat (15) #100 clk = ~clk;

    initial begin

        #100

        repeat (20) #60 D = ~D;

        #80 D = ~D;

        #40 D = ~D;

    end

    initial begin

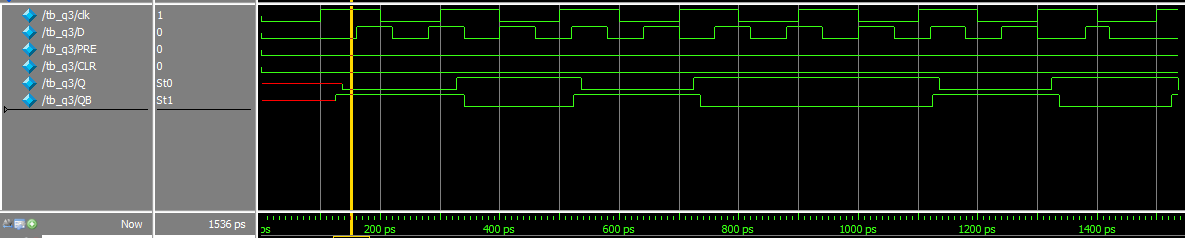
        #450 PRE = ~ PRE;

        #100 PRE = ~ PRE;

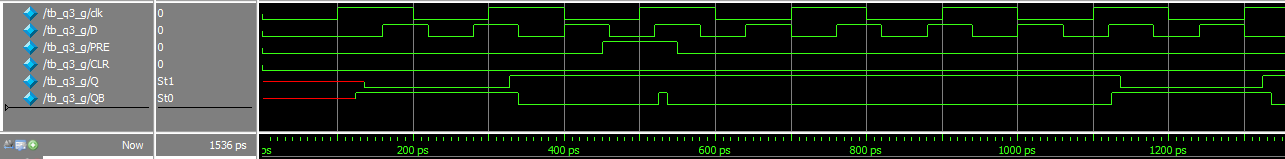
    end

endmodule

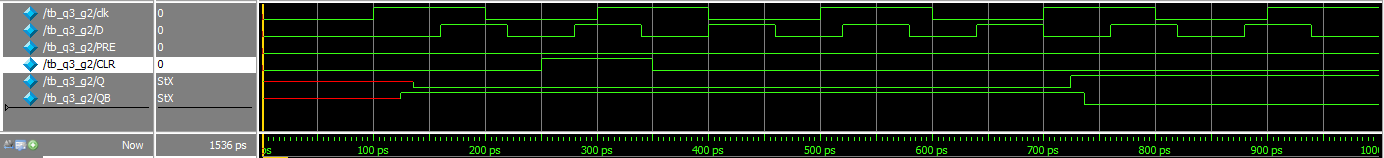
Waveform without PRE & CLR transition (which has wrong outputs in some points (f.e 500ps & 700ps) because of T\_setup&hold violations like Q2).



Waveform with PRE transition:  
as we can see on (450ps till 550) PRE has priority over clk.



Waveform with CLR transition:  
as we can see on (250ps till 350) CLR has priority over clk too.



Waveform with both CLR & PRE transition:  
if we active both of them at the same time, the memory will be gone.  
