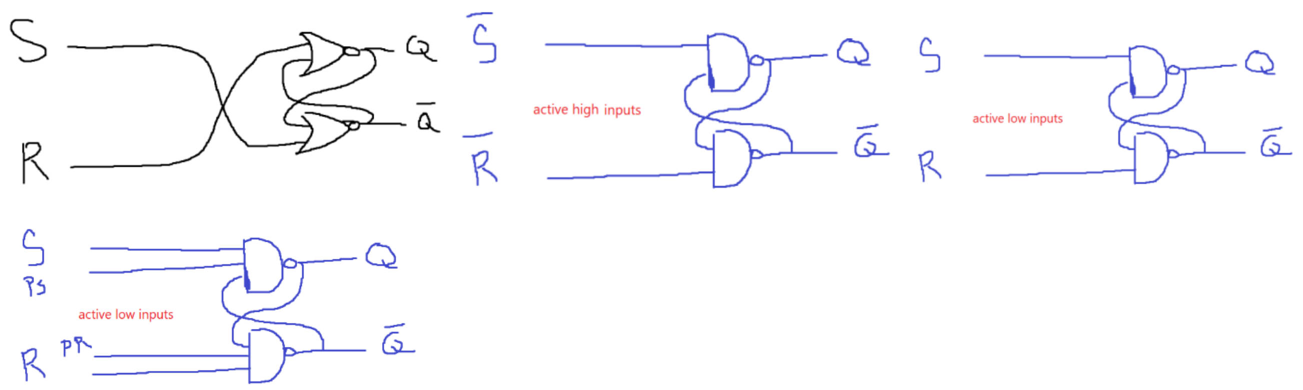
Q1:

a)



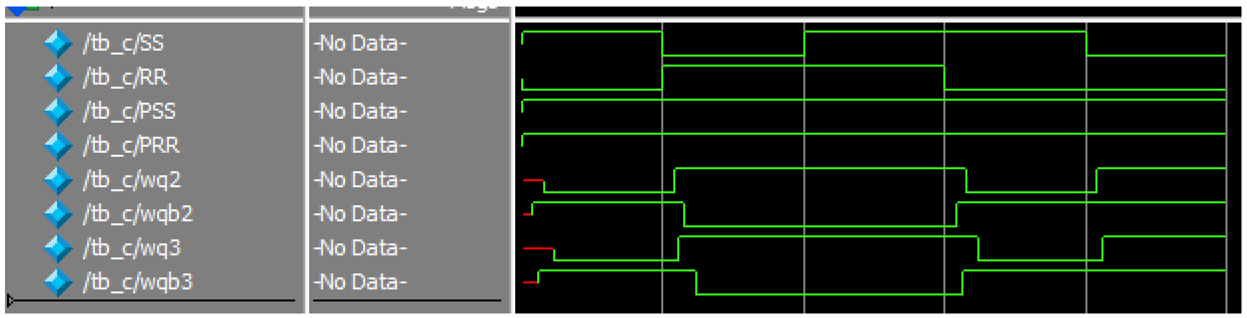
```
module SR_2_input (input S, R, output Q, QB);
    nand #8 (Q, S, QB);
    nand #8 (QB, R, Q);
endmodule

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.

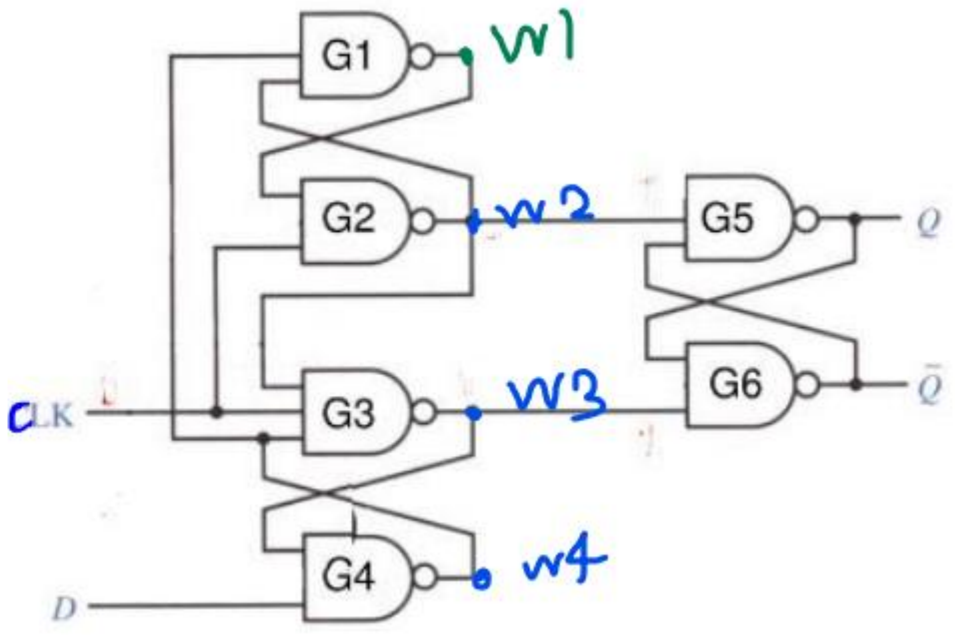
3_input_nand delay = 12ns.

c)



Q2:

a)

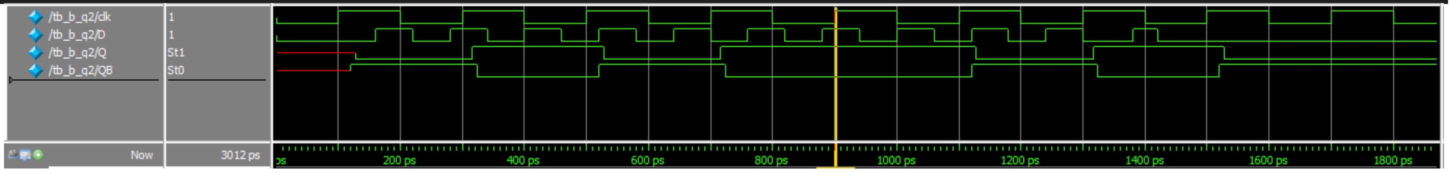


System Verilog code:

```
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:

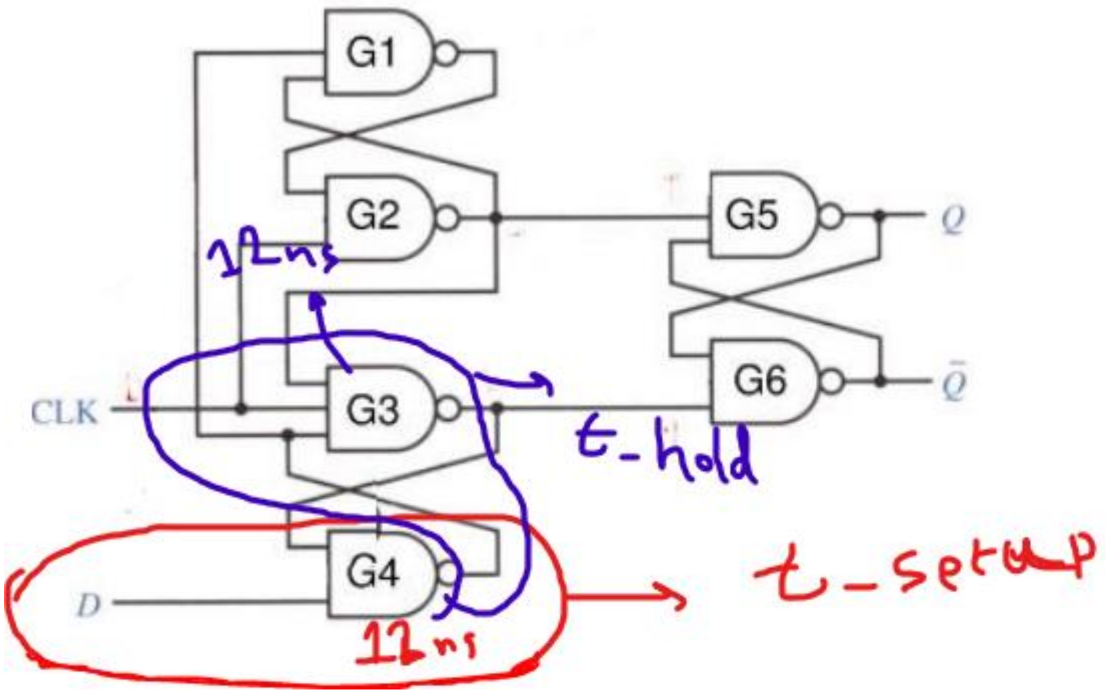
we have to stop transitions of G4 gates inputs (D & clk), And let the nand gate propagate its value on its output.

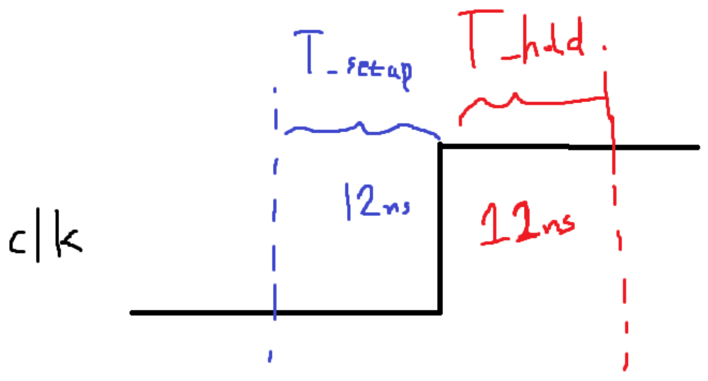
so the delay for this beautiful patience will be equal to G4s delay.
nand_3_input delay= 12ns.

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.

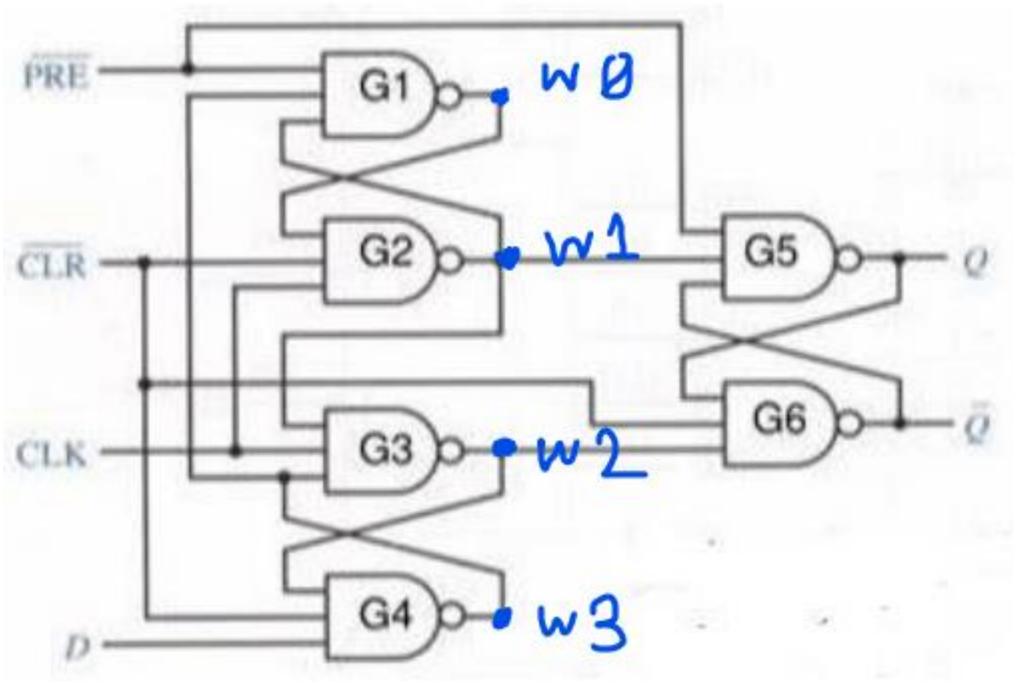
so the delay for this beautiful patience will be equal to G3s delay.
nand_3_input delay= 12ns.





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)

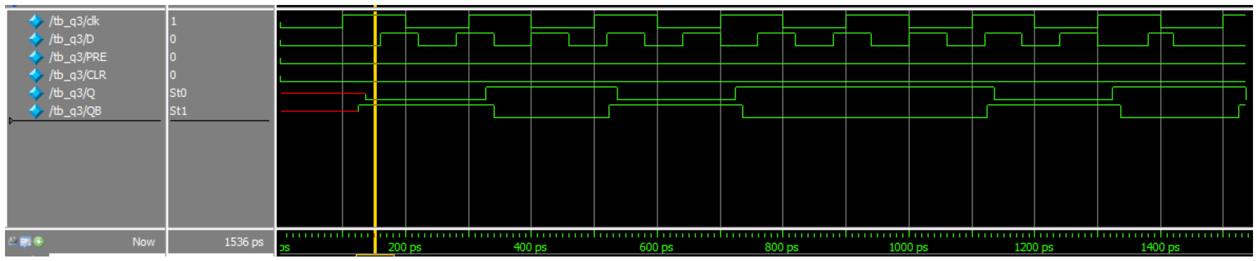
testbench without PRE & CLR transition:

```
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
```

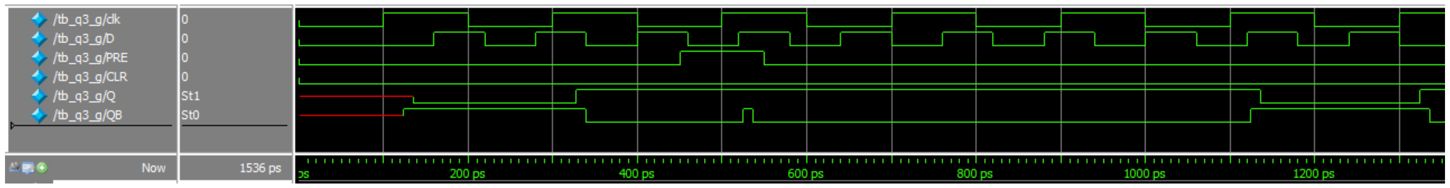
testbench with PRE transition:

```
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
```

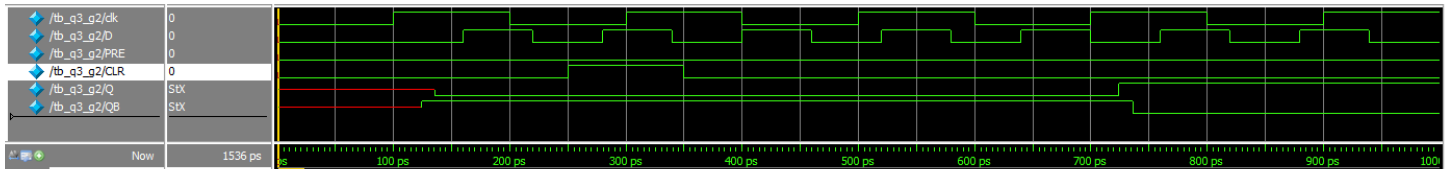
Waveform without PRE & CLR transition (which has wrong outputs in some points (f.e 500ps & 700ps) because of T_setup&hold violations like Q2).



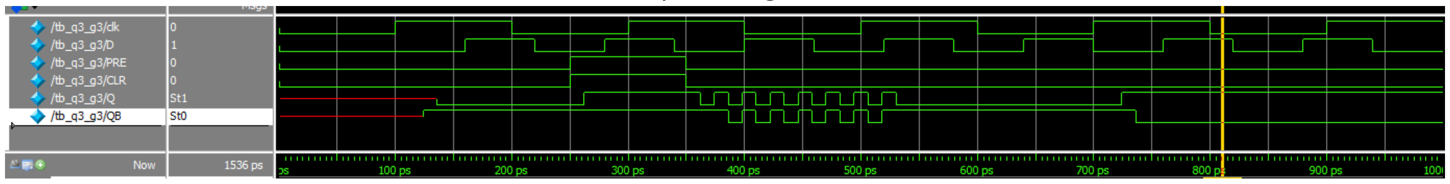
Waveform with PRE transition:
as we can see on (450ps till 550) PRE has priority over clk.



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.


/tb_q3_g3/clk 0
/tb_q3_g3/D 1
/tb_q3_g3/PRE 0
/tb_q3_g3/CLR 0
/tb_q3_g3/Q St1
/tb_q3_g3/QB St0
Now 1536 ps
ps
100 ps
200 ps
300 ps
400 ps
500 ps
600 ps
700 ps
800 ps
900 ps