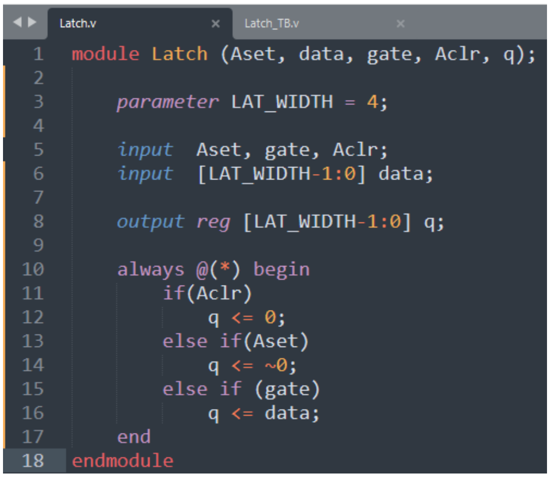
Assignment 3 Extra

By: Abdelrahman Maher Hassan

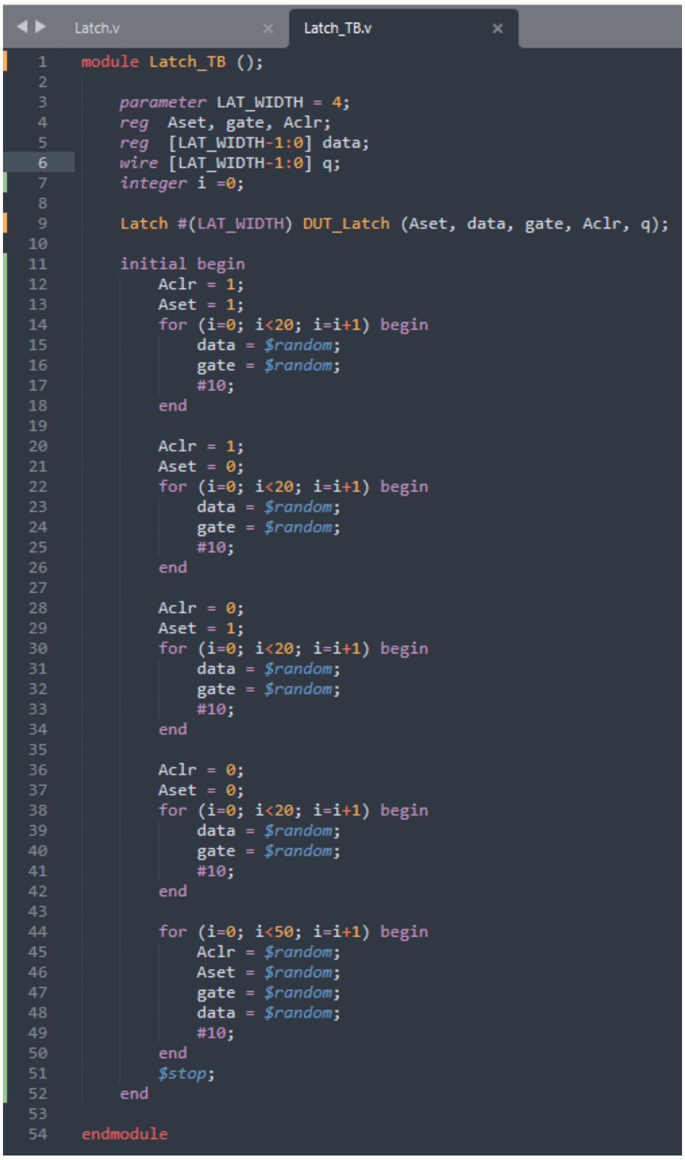
Question 1

Codes:



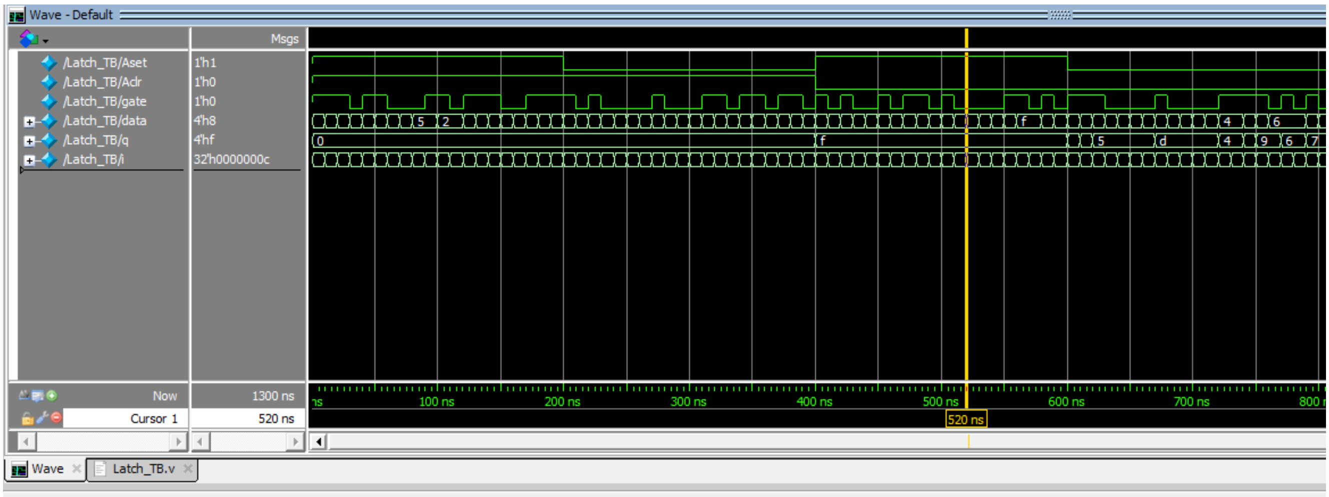
```verilog
module Latch (Aset, data, gate, Aclr, q);

    parameter LAT_WIDTH = 4;

    input  Aset, gate, Aclr;
    input  [LAT_WIDTH-1:0] data;

    output reg [LAT_WIDTH-1:0] q;

    always @(*) begin
        if(Aclr)
            q <= 0;
        else if(Aset)
            q <= ~0;
        else if (gate)
            q <= data;
    end
endmodule
```

Test Bench:



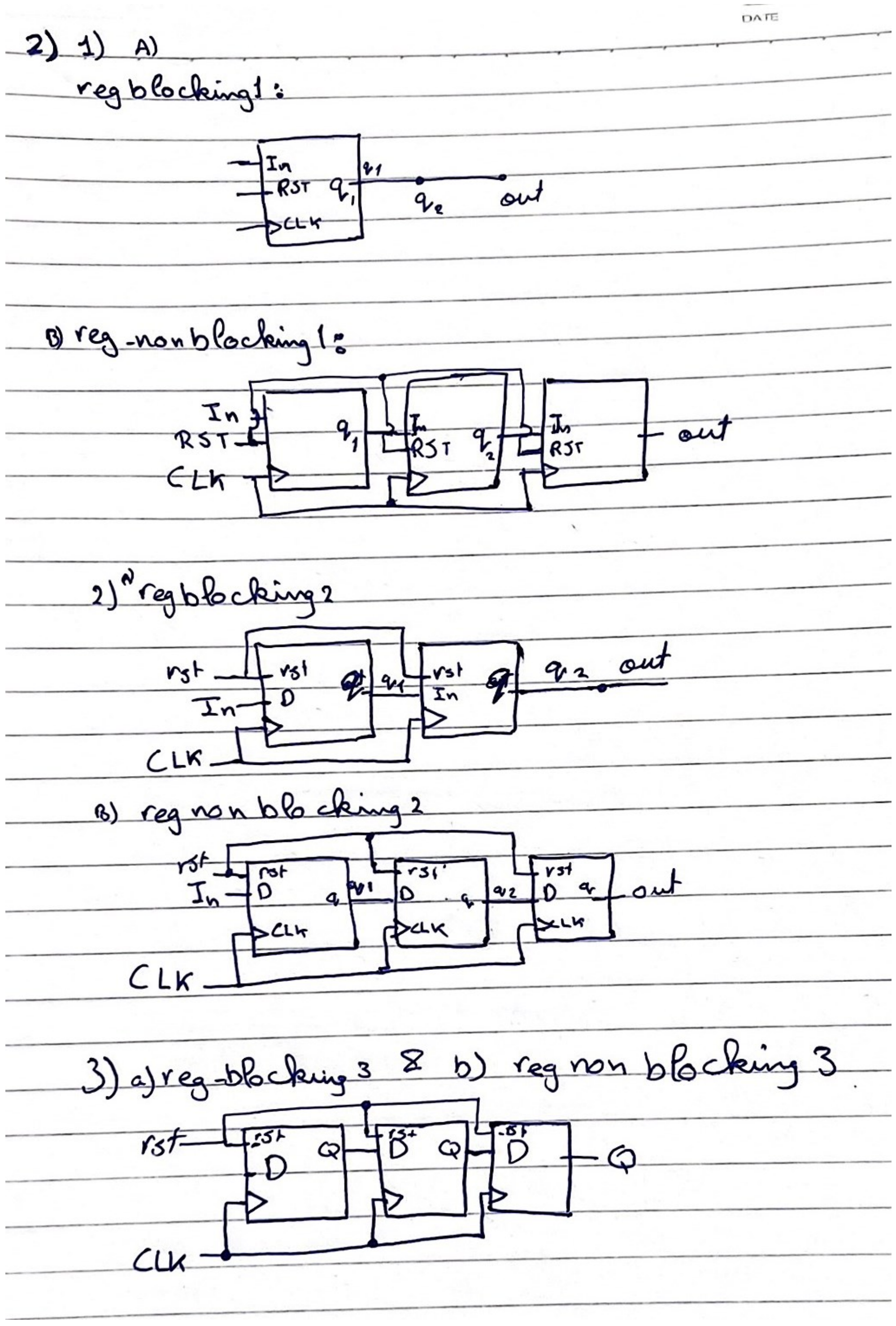
```verilog
module Latch_TB ();

    parameter LAT_WIDTH = 4;
    reg  Aset, gate, Aclr;
    reg  [LAT_WIDTH-1:0] data;
    wire [LAT_WIDTH-1:0] q;
    integer i =0;

    Latch #(LAT_WIDTH) DUT_Latch (Aset, data, gate, Aclr, q);

    initial begin
        Aclr = 1;
        Aset = 1;
        for (i=0; i<20; i=i+1) begin
            data = $random;
            gate = $random;
            #10;
        end

        Aclr = 1;
        Aset = 0;
        for (i=0; i<20; i=i+1) begin
            data = $random;
            gate = $random;
            #10;
        end

        Aclr = 0;
        Aset = 1;
        for (i=0; i<20; i=i+1) begin
            data = $random;
            gate = $random;
            #10;
        end

        Aclr = 0;
        Aset = 0;
        for (i=0; i<20; i=i+1) begin
            data = $random;
            gate = $random;
            #10;
        end

        for (i=0; i<50; i=i+1) begin
            Aclr = $random;
            Aset = $random;
            gate = $random;
            data = $random;
            #10;
        end
        $stop;
    end

endmodule
```

Simulation:



	Msgs
/Latch_TB/Aset	1'h1
/Latch_TB/Adr	1'h0
/Latch_TB/gate	1'h0
/Latch_TB/data	4'h8
/Latch_TB/q	4'hf
/Latch_TB/i	32'h0000000c
Now	1300 ns
Cursor 1	520 ns

Question 2 (A)



DATE

2) 1) A)

reg blocking1 :

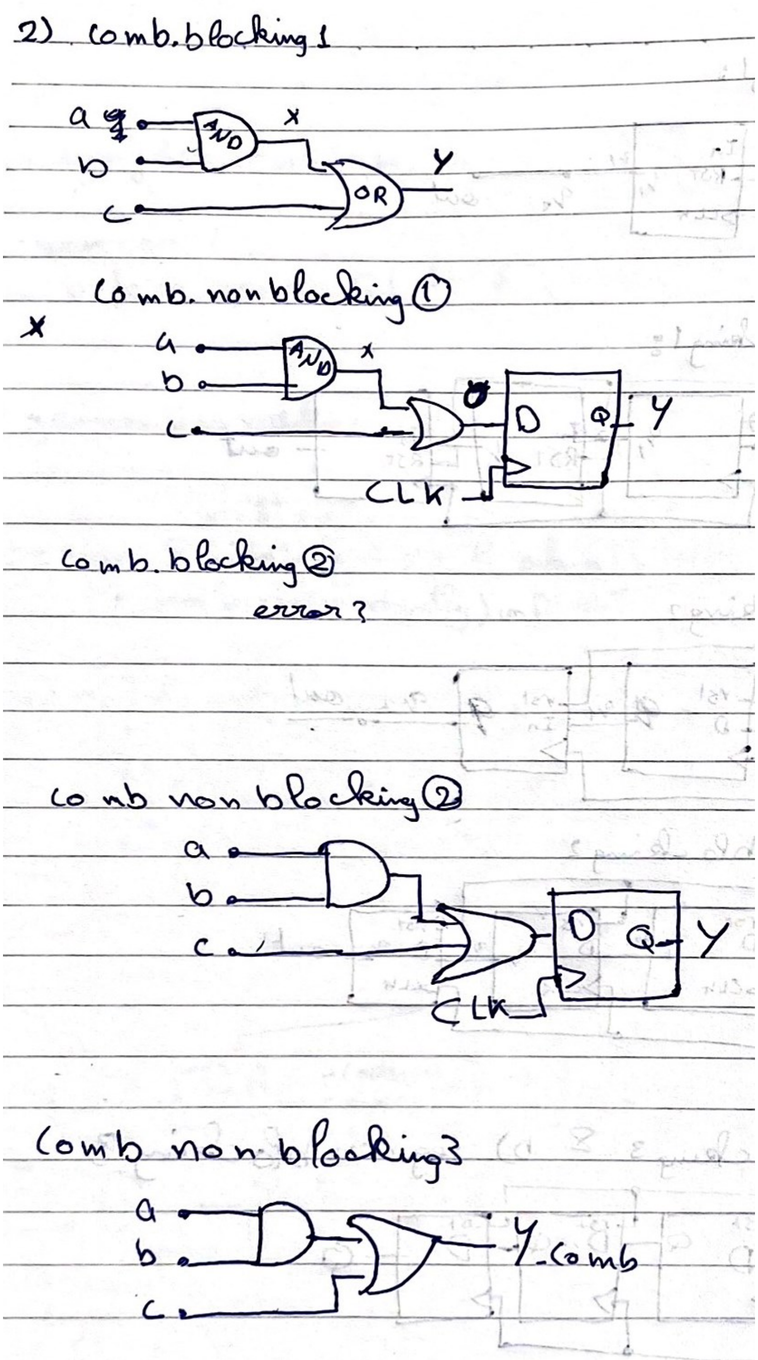
B) reg-nonblocking1 :

2) A) reg blocking 2

B) reg non blocking 2

3) a) reg-blocking 3 & b) reg non blocking 3

Question 2 (B)



2) comb.blocking 1

comb. nonblocking ①

comb. blocking ②

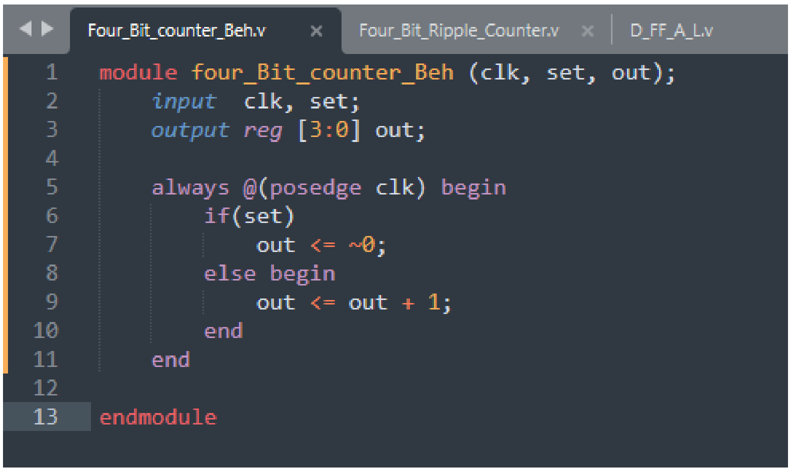
error?

comb non blocking ②

comb non-blocking3

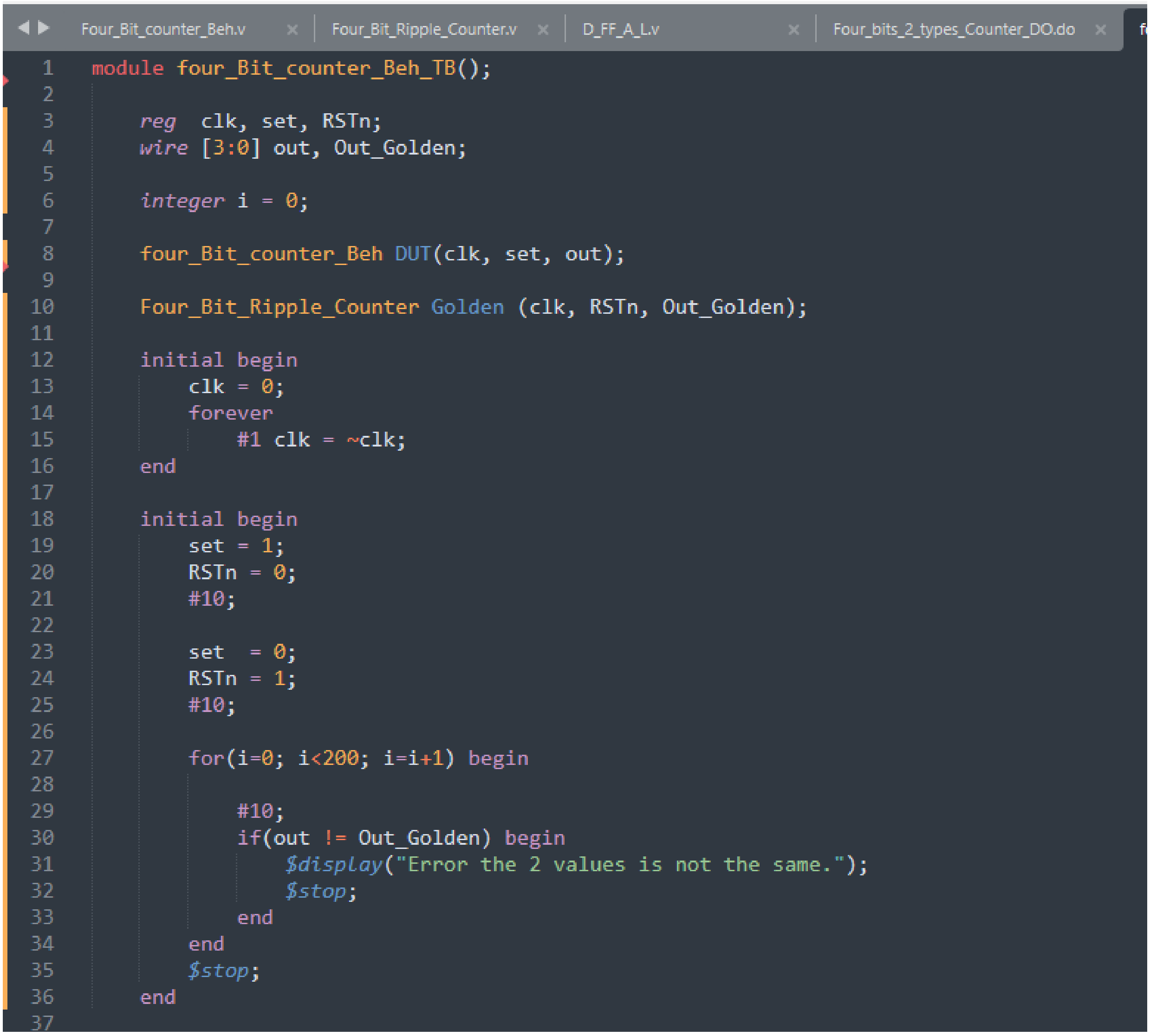
Question 3

Codes:



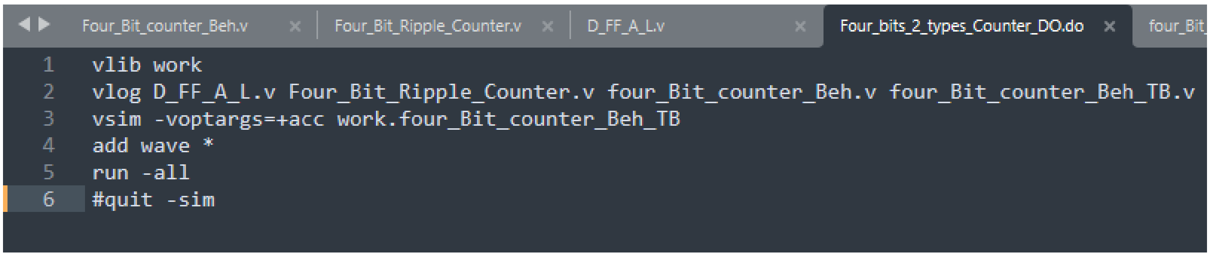
```verilog
module four_Bit_counter_Beh (clk, set, out);
    input  clk, set;
    output reg [3:0] out;

    always @(posedge clk) begin
        if(set)
            out <= ~0;
        else begin
            out <= out + 1;
        end
    end

endmodule
```

Test Bench:



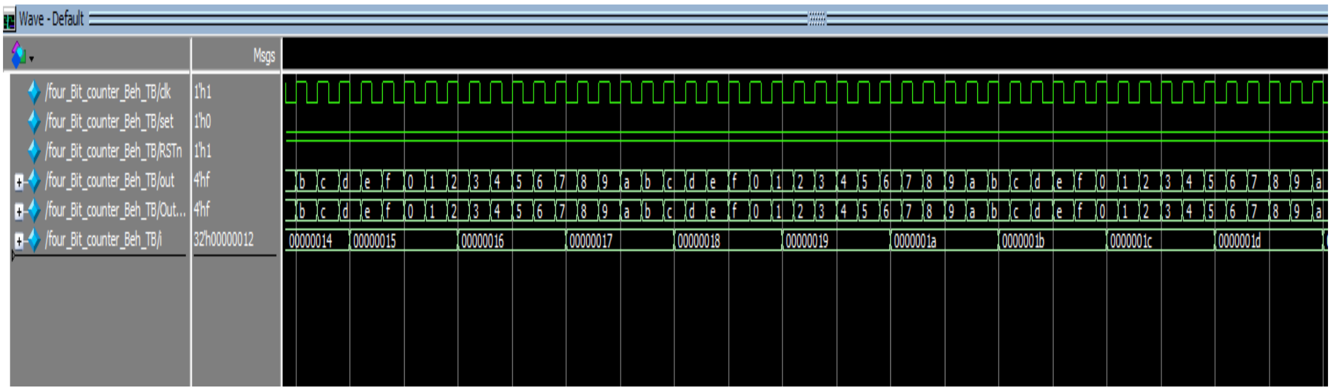
```verilog
module four_Bit_counter_Beh_TB();

    reg  clk, set, RSTn;
    wire [3:0] out, Out_Golden;

    integer i = 0;

    four_Bit_counter_Beh DUT(clk, set, out);

    Four_Bit_Ripple_Counter Golden (clk, RSTn, Out_Golden);

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

    initial begin
        set = 1;
        RSTn = 0;
        #10;

        set  = 0;
        RSTn = 1;
        #10;

        for(i=0; i<200; i=i+1) begin

            #10;
            if(out != Out_Golden) begin
                $display("Error the 2 values is not the same.");
                $stop;
            end
        end
        $stop;
    end
```

Do File:



```
vlib work
vlog D_FF_A_L.v Four_Bit_Ripple_Counter.v four_Bit_counter_Beh.v four_Bit_counter_Beh_TB.v
vsim -voptargs=+acc work.four_Bit_counter_Beh_TB
add wave *
run -all
#quit -sim
```

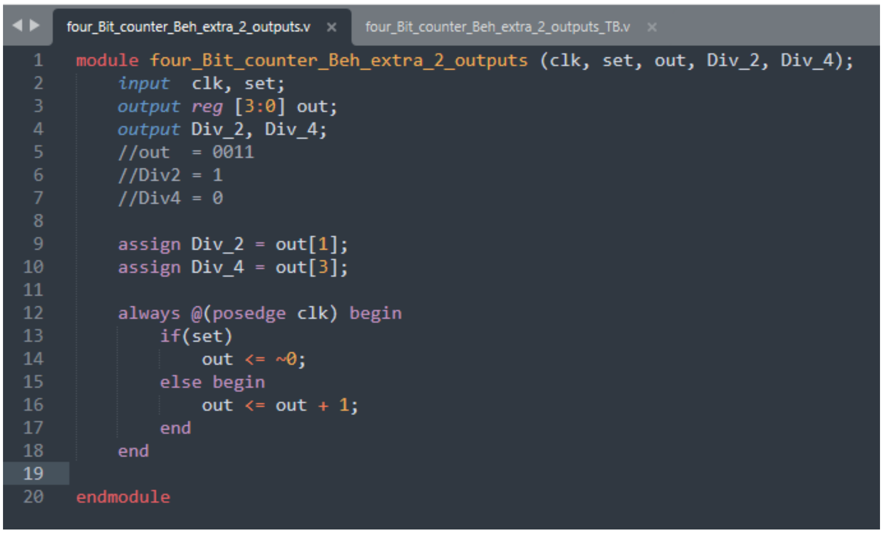
Simulation:


Wave - Default
Msgs
b c d e f 0 1 2 3 4 5 6 7 8 9 a b c d e f 0 1 2 3 4 5 6 7 8 9 a b c d e f 0 1 2 3 4 5 6 7 8 9 a
00000014 00000015 00000016 00000017 00000018 00000019 0000001a 0000001b 0000001c 0000001d

/four_Bit_counter_Beh_TB/clk	1'h1
/four_Bit_counter_Beh_TB/set	1'h0
/four_Bit_counter_Beh_TB/RSTn	1'h1
/four_Bit_counter_Beh_TB/out	4'hf
/four_Bit_counter_Beh_TB/Out...	4'hf
/four_Bit_counter_Beh_TB/i	32'h00000012

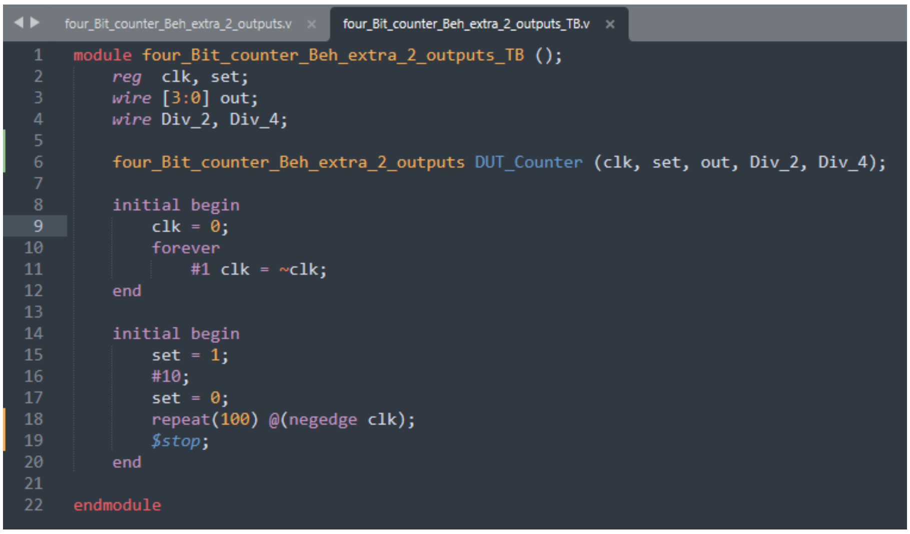
Question 4

Codes:


four_Bit_counter_Beh_extra_2_outputs.v
four_Bit_counter_Beh_extra_2_outputs_TB.v

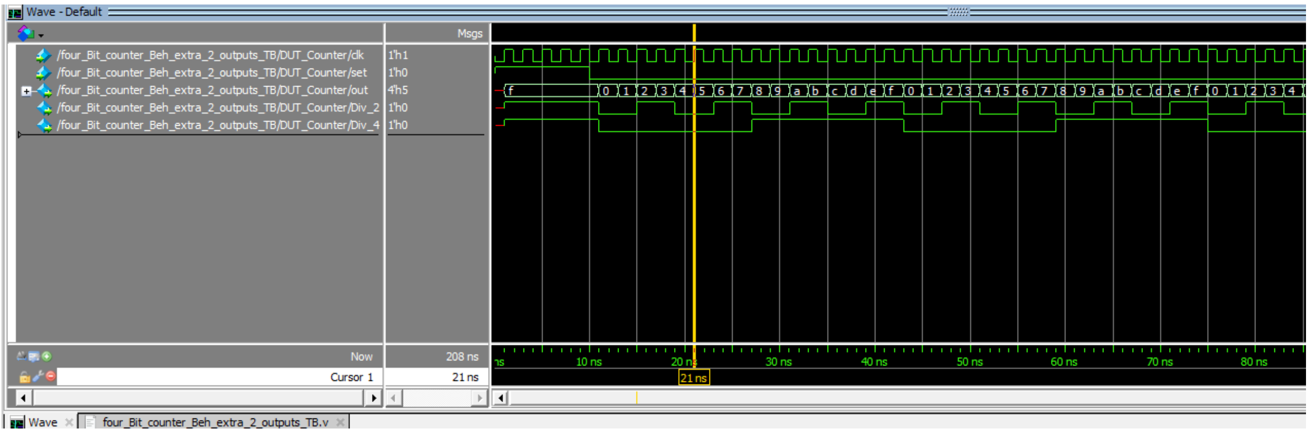
```verilog
module four_Bit_counter_Beh_extra_2_outputs (clk, set, out, Div_2, Div_4);
    input  clk, set;
    output reg [3:0] out;
    output Div_2, Div_4;
    //out  = 0011
    //Div2 = 1
    //Div4 = 0

    assign Div_2 = out[1];
    assign Div_4 = out[3];

    always @(posedge clk) begin
        if(set)
            out <= ~0;
        else begin
            out <= out + 1;
        end
    end

endmodule
```

Test Bench:


four_Bit_counter_Beh_extra_2_outputs.v
four_Bit_counter_Beh_extra_2_outputs_TB.v

```verilog
module four_Bit_counter_Beh_extra_2_outputs_TB ();
    reg  clk, set;
    wire [3:0] out;
    wire Div_2, Div_4;

    four_Bit_counter_Beh_extra_2_outputs DUT_Counter (clk, set, out, Div_2, Div_4);

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

    initial begin
        set = 1;
        #10;
        set = 0;
        repeat(100) @(negedge clk);
        $stop;
    end

endmodule
```

Simulation:



```
/four_Bit_counter_Beh_extra_2_outputs_TB/DUT_Counter/clk    1'h1
/four_Bit_counter_Beh_extra_2_outputs_TB/DUT_Counter/set    1'h0
/four_Bit_counter_Beh_extra_2_outputs_TB/DUT_Counter/out    4'h5
/four_Bit_counter_Beh_extra_2_outputs_TB/DUT_Counter/Div_2  1'h0
/four_Bit_counter_Beh_extra_2_outputs_TB/DUT_Counter/Div_4  1'h0
Now       208 ns
Cursor 1  21 ns
```