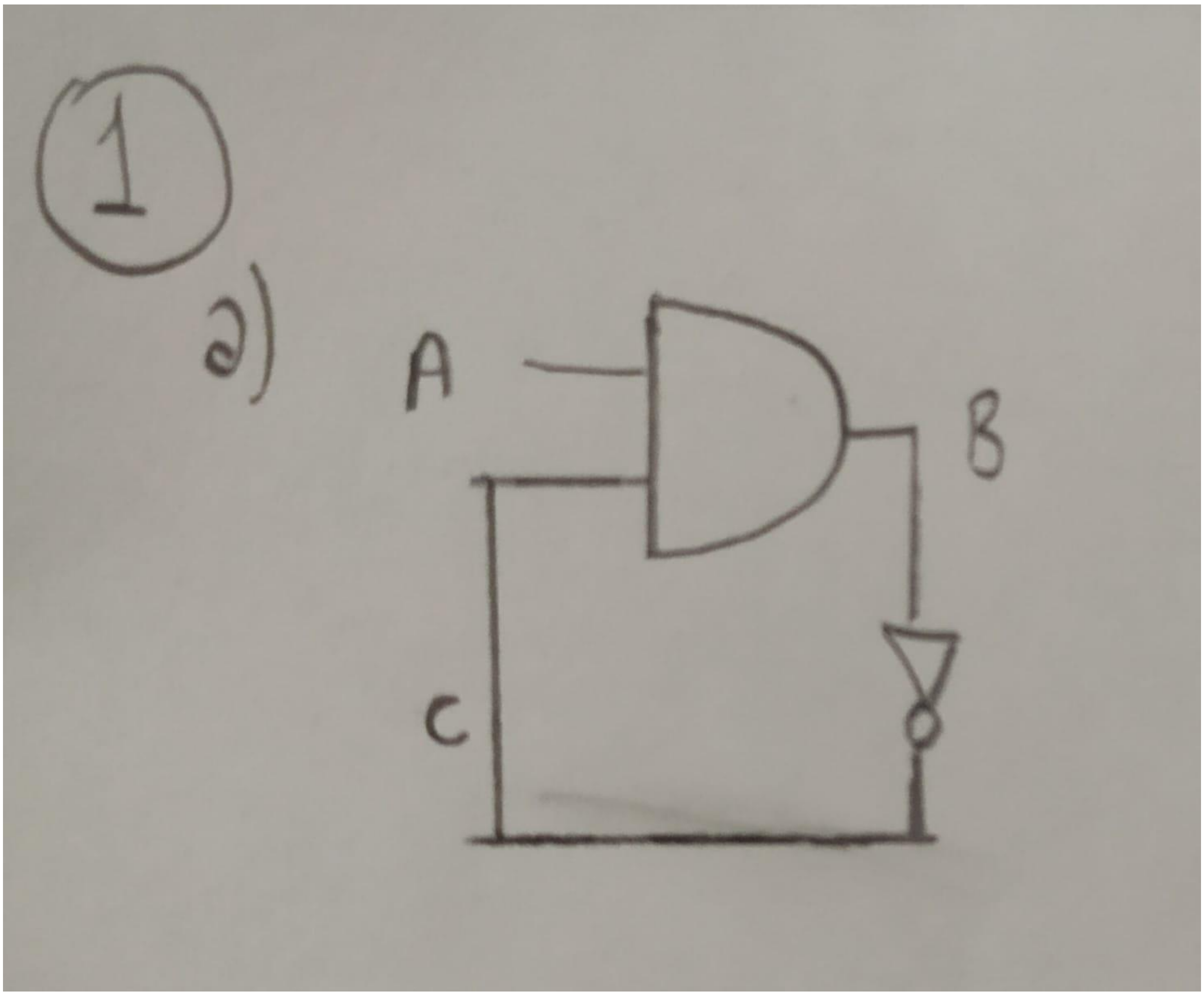
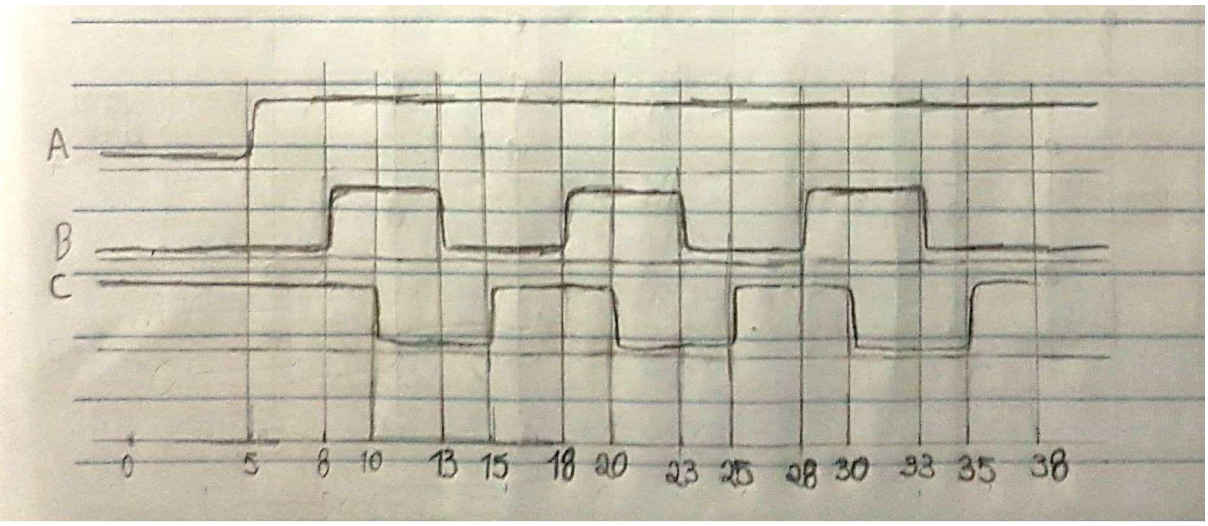
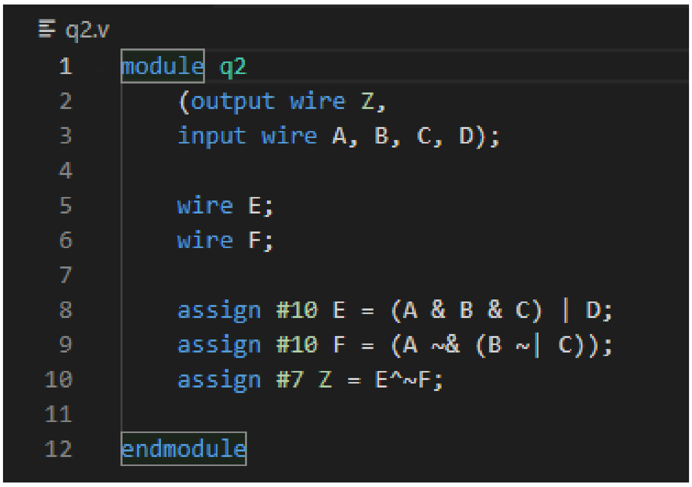
Questão 1
A)



B)



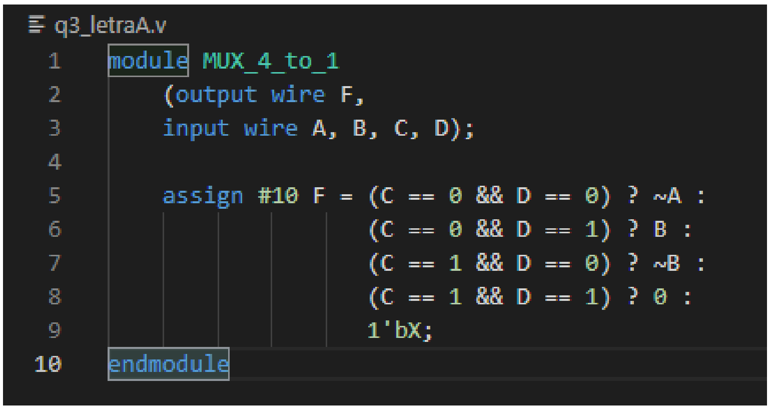
Questão 2



```
module q2
    (output wire Z,
    input wire A, B, C, D);

    wire E;
    wire F;

    assign #10 E = (A & B & C) | D;
    assign #10 F = (A ~& (B ~| C));
    assign #7 Z = E^~F;

endmodule
```

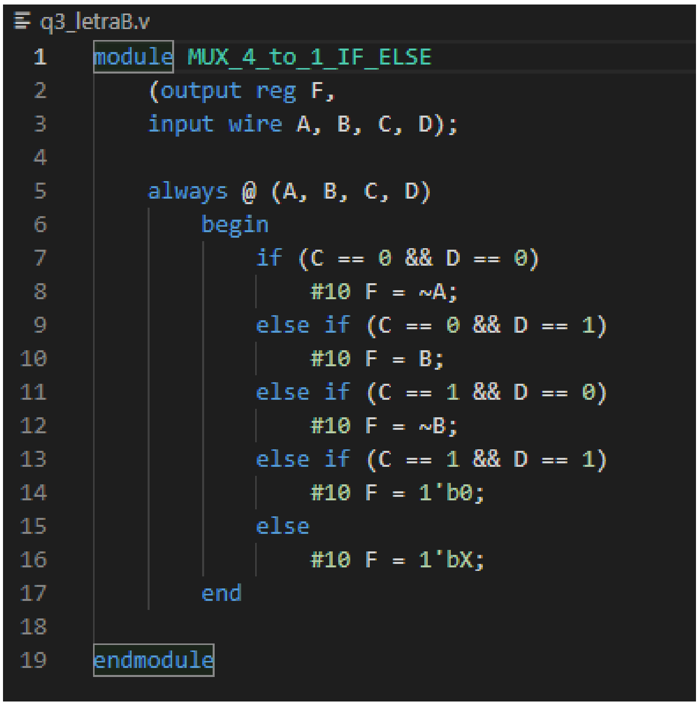
Questão 3

A)



```
module MUX_4_to_1
    (output wire F,
    input wire A, B, C, D);

    assign #10 F = (C == 0 && D == 0) ? ~A :
                   (C == 0 && D == 1) ? B :
                   (C == 1 && D == 0) ? ~B :
                   (C == 1 && D == 1) ? 0 :
                   1'bX;
endmodule
```

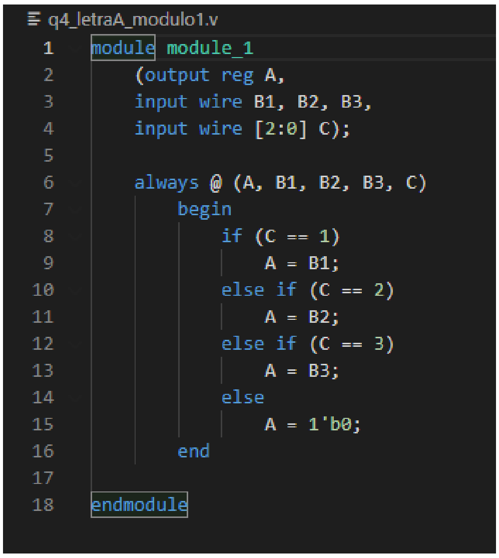
B)



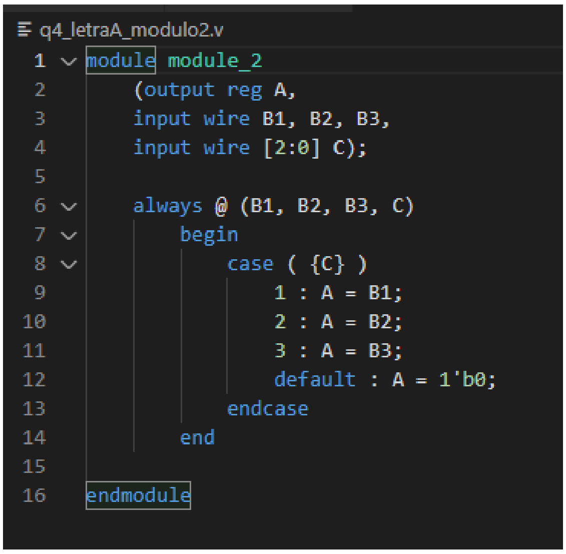
```verilog
module MUX_4_to_1_IF_ELSE
    (output reg F,
    input wire A, B, C, D);

    always @ (A, B, C, D)
        begin
            if (C == 0 && D == 0)
                #10 F = ~A;
            else if (C == 0 && D == 1)
                #10 F = B;
            else if (C == 1 && D == 0)
                #10 F = ~B;
            else if (C == 1 && D == 1)
                #10 F = 1'b0;
            else
                #10 F = 1'bX;
        end

endmodule
```

Questão 4

A)

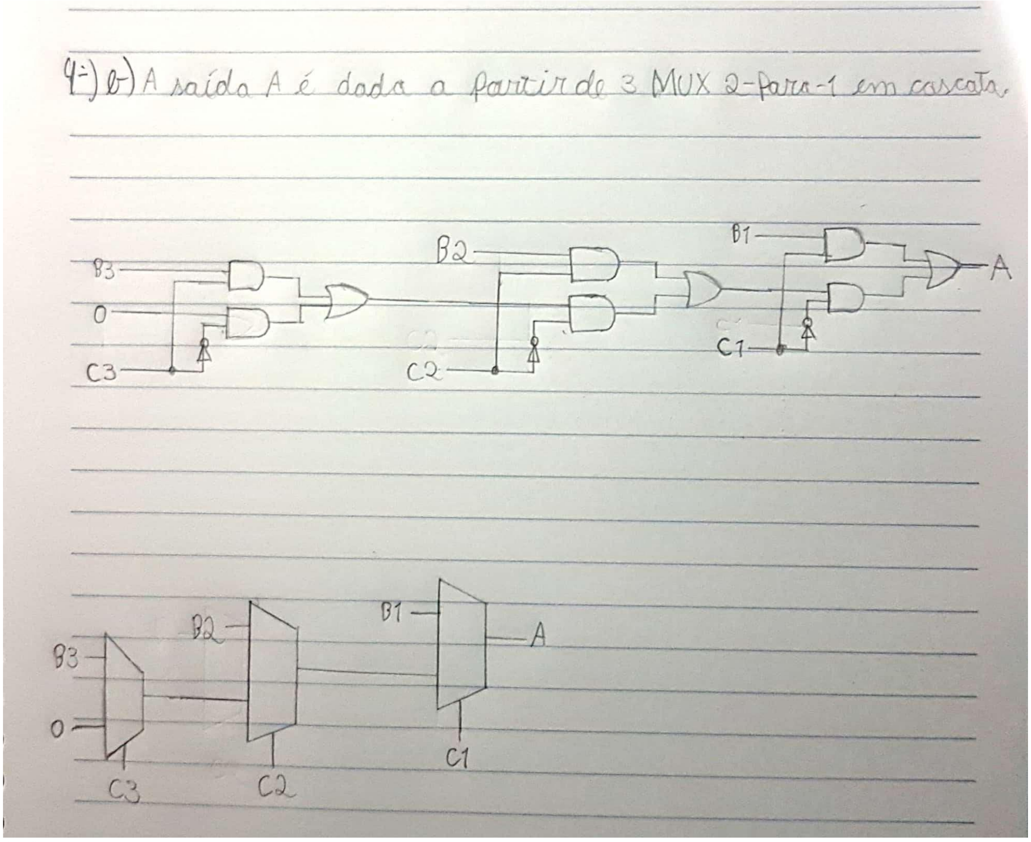


```verilog
module module_1
    (output reg A,
    input wire B1, B2, B3,
    input wire [2:0] C);

    always @ (A, B1, B2, B3, C)
        begin
            if (C == 1)
                A = B1;
            else if (C == 2)
                A = B2;
            else if (C == 3)
                A = B3;
            else
                A = 1'b0;
        end

endmodule
```



```verilog
module module_2
    (output reg A,
    input wire B1, B2, B3,
    input wire [2:0] C);

    always @ (B1, B2, B3, C)
        begin
            case ( {C} )
                1 : A = B1;
                2 : A = B2;
                3 : A = B3;
                default : A = 1'b0;
            endcase
        end

endmodule
```

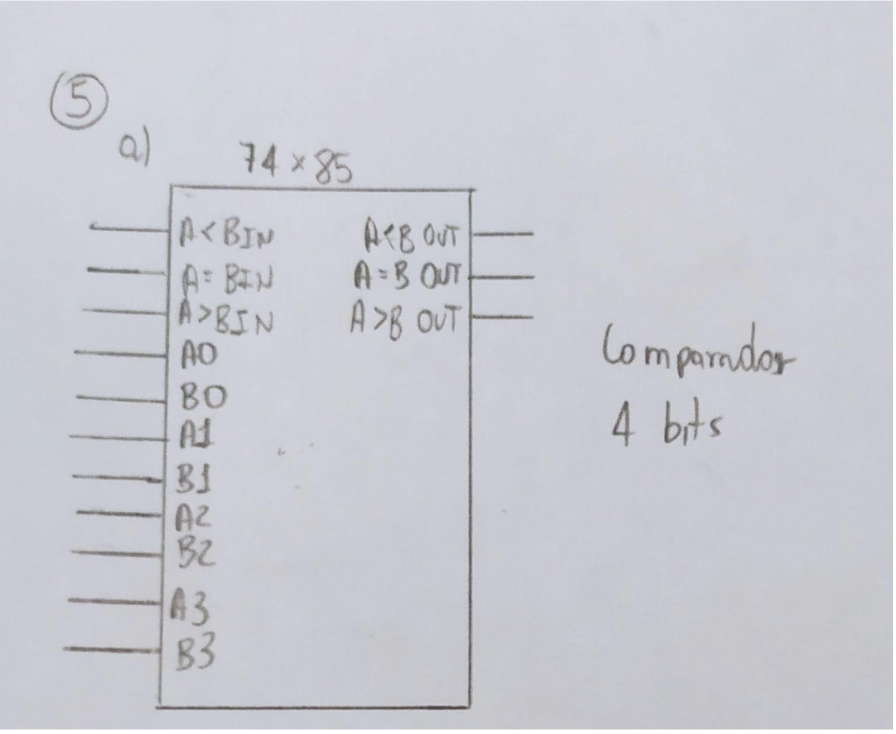
B)



4-) b) A saída A é dada a partir de 3 MUX 2-para-1 em cascata.

Questão 5

A)

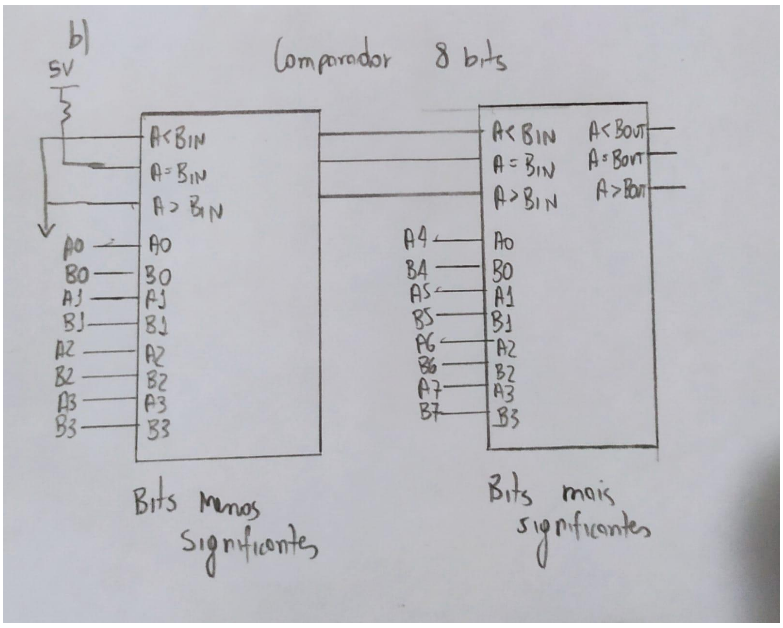


5

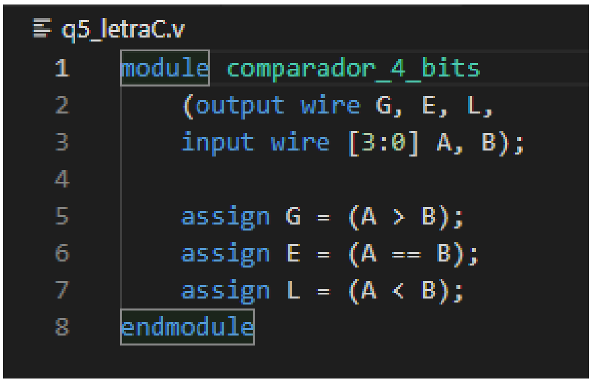
a)

Comparador 4 bits

B)

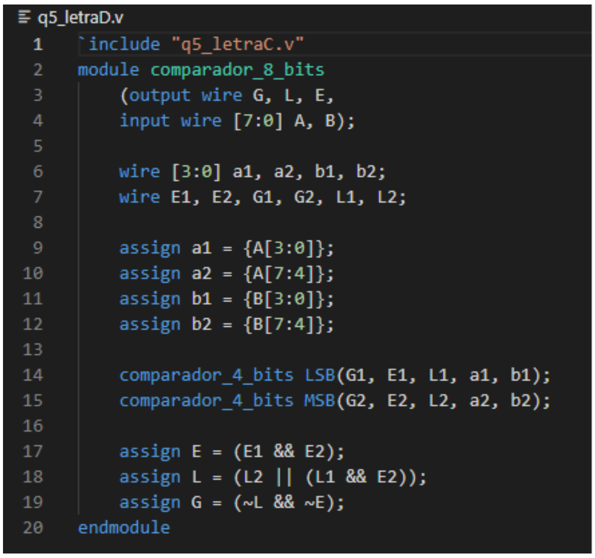


C)



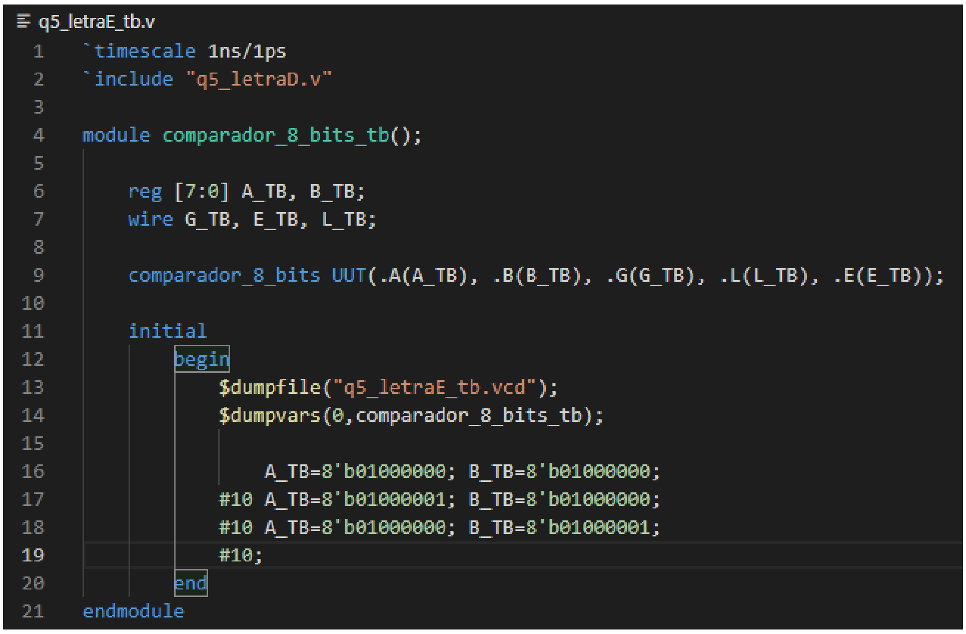
```verilog
module comparador_4_bits
    (output wire G, E, L,
    input wire [3:0] A, B);

    assign G = (A > B);
    assign E = (A == B);
    assign L = (A < B);
endmodule
```

D)



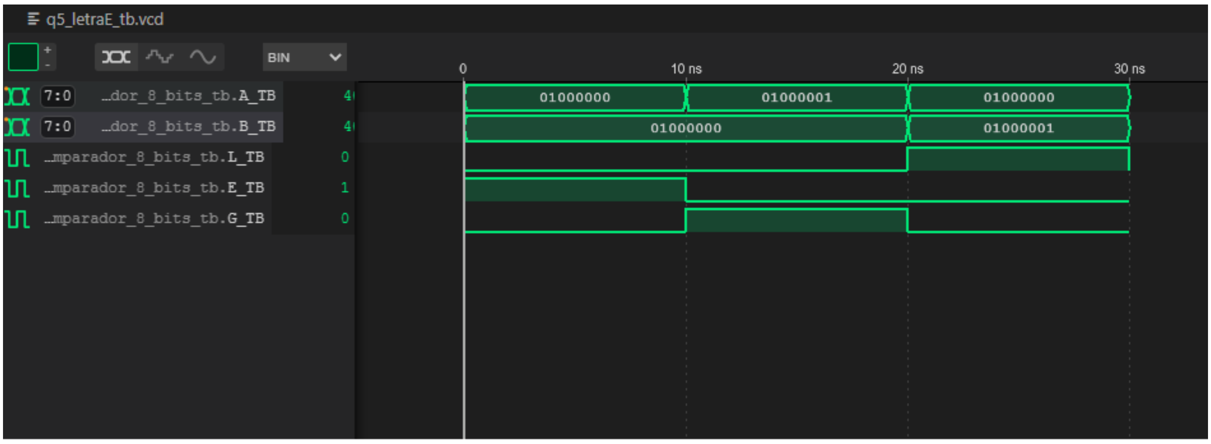
```verilog
`include "q5_letraC.v"
module comparador_8_bits
    (output wire G, L, E,
    input wire [7:0] A, B);

    wire [3:0] a1, a2, b1, b2;
    wire E1, E2, G1, G2, L1, L2;

    assign a1 = {A[3:0]};
    assign a2 = {A[7:4]};
    assign b1 = {B[3:0]};
    assign b2 = {B[7:4]};

    comparador_4_bits LSB(G1, E1, L1, a1, b1);
    comparador_4_bits MSB(G2, E2, L2, a2, b2);

    assign E = (E1 && E2);
    assign L = (L2 || (L1 && E2));
    assign G = (~L && ~E);
endmodule
```

E)



q5_letraE_tb.v

```
`timescale 1ns/1ps
`include "q5_letraD.v"

module comparador_8_bits_tb();

    reg [7:0] A_TB, B_TB;
    wire G_TB, E_TB, L_TB;

    comparador_8_bits UUT(.A(A_TB), .B(B_TB), .G(G_TB), .L(L_TB), .E(E_TB));

    initial
        begin
            $dumpfile("q5_letraE_tb.vcd");
            $dumpvars(0,comparador_8_bits_tb);

                A_TB=8'b01000000; B_TB=8'b01000000;
            #10 A_TB=8'b01000001; B_TB=8'b01000000;
            #10 A_TB=8'b01000000; B_TB=8'b01000001;
            #10;
        end
endmodule
```



q5_letraE_tb.vcd