module lab4\_2(input wire en, input wire reset,input wire clk,input wire dir,input wire record,

output wire [3:0]DIGIT,output wire[6:0]DISPLAY,output wire max,output wire min);

        wire clk23;

        wire clk25;

        wire clk13;

        wire en\_de;

        wire en\_pul;

        wire dir\_de;

        wire dir\_pul;

        wire record\_de;

        wire record\_pul;

        clk\_divider #(23)clk1 (.clk(clk),.clk\_div(clk23));

        clk\_divider #(25)clk2 (.clk(clk),.clk\_div(clk25));

        clk\_divider #(13)clk3 (.clk(clk),.clk\_div(clk13));

        clk\_divider #(16)clk4 (.clk(clk),.clk\_div(clk16));

        debounce de1(.pb\_debounced(en\_de),.pb(en),.clk(clk));

        onepulse on1(.clk(clk),.pb\_debounced(en\_de),.pb\_1pulse(en\_pul));

        debounce de2(.pb\_debounced(dir\_de),.pb(dir),.clk(clk));

        onepulse on2(.clk(clk),.pb\_debounced(dir\_de),.pb\_1pulse(dir\_pul));

        debounce de3(.pb\_debounced(record\_de),.pb(record),.clk(clk16));

        onepulse on3(.clk(clk23),.pb\_debounced(record\_de),.pb\_1pulse(record\_pul));

        wire [3:0]lastnum;

        wire [3:0]num;

        reg [3:0]nextBCD0 ;

        reg [3:0]nextBCD1 ;

        reg [3:0]nextBCD2 ;

        reg [3:0]nextBCD3 ;

        reg [6:0] display;

        reg [3:0] digit;

        reg enable;

        reg nextenable;

        reg [3:0] val;

        reg [3:0] BCD0;

        reg [3:0] BCD1;

        reg [3:0] BCD2;

        reg [3:0] BCD3;

        reg mini;

        reg maxi;

        reg dire;

        reg nextdir;

        reg rcd;

        always@(\*)

            begin

                if(reset==1'b1) begin nextBCD0=4'b0000;nextBCD1=4'b0000;nextBCD2=4'b0000;nextBCD3=4'b0000;

                nextenable=1'b0; nextdir=1'b1;maxi=1'b0; mini=1'b0; end

                else

                    begin

                        if(en\_pul==1'b1) begin nextenable=~enable; end

                        if(enable==1'b0)

                            begin

                                maxi=1'b0;

                                mini=1'b0;

                                nextBCD1=BCD1;

                                nextBCD0=BCD0;

                                nextBCD2=BCD2;

                                nextBCD3=BCD3;

                            end

                        else

                            begin

                            if(record\_pul==1'b1) begin nextBCD2=BCD0; nextBCD3=BCD1; end

                             else begin nextBCD2=BCD2; nextBCD3=BCD3; end

                            if(dir\_pul==1'b1) begin nextdir=~dire; end

                                if(dire==1'b1)

                                    begin

                                        if(BCD0==4'b1001&&BCD1==4'b1001)

                                            begin

                                                maxi=1'b1;

                                                mini=1'b0;

                                                nextBCD1=BCD1;

                                                nextBCD0=BCD0;

                                            end

                                        else if(BCD0==4'b1001&&BCD1!=4'b1001)

                                            begin

                                                maxi=1'b0;

                                                mini=1'b0;

                                                nextBCD1=BCD1+1;

                                                nextBCD0=4'b0000;

                                            end

                                        else

                                            begin

                                                maxi=1'b0;

                                                mini=1'b0;

                                                nextBCD0=BCD0+1;

                                                nextBCD1=BCD1;

                                            end

                                    end

                                else

                                    begin

                                        if(BCD0==4'b0000&&BCD1==4'b0000)

                                            begin

                                                maxi=1'b0;

                                                mini=1'b1;

                                                nextBCD1=BCD1;

                                                nextBCD0=BCD0;

                                            end

                                        else if(BCD0==4'b0000&&BCD1!=4'b0000)

                                            begin

                                                maxi=1'b0;

                                                mini=1'b0;

                                                nextBCD1=BCD1-1;

                                                nextBCD0=4'b1001;

                                            end

                                        else

                                            begin

                                                maxi=1'b0;

                                                mini=1'b0;

                                                nextBCD0=BCD0-1;

                                                nextBCD1=BCD1;

                                            end

                                    end

                            end

                    end

                case(val)

                    4'b0000: begin display=7'b0000001; end

                    4'b0001: begin display=7'b1001111; end

                    4'b0010: begin display=7'b0010010; end

                    4'b0011: begin display=7'b0000110; end

                    4'b0100: begin display=7'b1001100; end

                    4'b0101: begin display=7'b0100100; end

                    4'b0110: begin display=7'b0100000; end

                    4'b0111: begin display=7'b0001111; end

                    4'b1000: begin display=7'b0000000; end

                    4'b1001: begin display=7'b0000100; end

                    default:  begin display=7'b0000001; end

                endcase

            end

        always@(posedge clk23) //update FSM

            begin

                enable<=nextenable;

                dire<=nextdir;

            end

        always@(posedge clk25, posedge reset) //update counting number

            begin

                if(reset==1'b1)

                    begin

                        BCD0<=4'b0000;

                        BCD1<=4'b0000;

                        BCD2<=4'b0000;

                        BCD3<=4'b0000;

                    end

                else

                    begin

                        BCD0<=nextBCD0;

                        BCD1<=nextBCD1;

                        BCD2<=nextBCD2;

                        BCD3<=nextBCD3;

                    end

                /\*if(reset==1'b1) begin enable=1'b0; end

                else \*/

            end

        always@(posedge clk13)

            begin

                case(digit)

                4'b1110: begin val=BCD1; digit=4'b1101; end

                4'b1101: begin val=BCD2; digit=4'b1011; end

                4'b1011: begin val=BCD3; digit=4'b0111; end

                4'b0111: begin val=BCD0; digit=4'b1110; end

                default: begin val=BCD0; digit=4'b1110; end

                endcase

            end

        assign DISPLAY=display;

        assign DIGIT=digit;

        assign max=maxi;

        assign min=mini;

endmodule

module clk\_divider(clk,clk\_div);

    parameter n = 4;

    input clk;

    output clk\_div;

    reg [n-1:0] num;

    wire [n-1:0] nextnum;

    always@(posedge clk)

        begin

            num=nextnum;

        end

    assign nextnum=num+1;

    assign clk\_div=num[n-1];

endmodule

module debounce(pb\_debounced,pb,clk);

    output pb\_debounced;

    input pb;

    input clk;

    reg[3:0] shift\_reg;

    always@(posedge clk)

        begin

            shift\_reg[3:1] <= shift\_reg[2:0];

            shift\_reg[0] <= pb;

        end

    assign pb\_debounced=(shift\_reg==4'b1111)?1'b1:1'b0;

endmodule

/\*module onepulse(input wire rst,input wire clk,input wire pb\_debounced,output reg pb\_1pulse);

    reg pb\_1pulse\_next;

    reg pb\_debounced\_delay;

    always@(\*)

        begin

            pb\_1pulse\_next=pb\_debounced & (~pb\_debounced\_delay);

        end

    always@(posedge clk,posedge rst)

        begin

            if(rst==1'b1)

                begin

                    pb\_1pulse=1'b0;

                    pb\_debounced\_delay=1'b0;

                end

            else

                begin

                    pb\_1pulse=pb\_1pulse\_next;

                    pb\_debounced\_delay=pb\_debounced;

                end

        end

endmodule\*/

module onepulse (pb\_debounced, clk, pb\_1pulse);

    input pb\_debounced;

    input clk;

    output pb\_1pulse;

    reg pb\_1pulse;

    reg pb\_debounced\_delay;

    always @(posedge clk)

    begin

        if (pb\_debounced == 1'b1 & pb\_debounced\_delay == 1'b0)

            begin

                pb\_1pulse <= 1'b1;

            end

        else

            begin

                pb\_1pulse <= 1'b0;

                pb\_debounced\_delay <= pb\_debounced;

            end

    end

endmodule