DSD LAB2 107504507 資電四 吳葦誠
Lab_2.v

```
`timescale 1ns / 1ps

module Lab_2(
    input [1:8] SW_DIP,
    input clk_input,
    input rst,
    output reg [7:0] Enable,
    output reg [15:0] LED,
    output reg [7:0] SevenSeg
    );

    reg speed;                  //ON:0.5HZ      OFF:2HZ
    reg [3:0] mode;           //ON:+1               OFF:XX
    reg [3:0] pattern;
    reg [1:0] times;         //repeat times
    reg clk;
    reg [27:0] counter28;     //for counting frequency
    reg [1:0] group;
    reg up;                        //1 for going up
    reg [1:0] round;                //for counting times
    reg [3:0] pattern2;             //for adding pattern
    reg [1:0] times2;

    reg [7:0] SevenSegP;
    reg [7:0] SevenSegG;

    //assign Enable = 8'b0001_1000;

    always @(*) begin
        speed = SW_DIP[1];
        mode = {3'b000, SW_DIP[2]};
        pattern = SW_DIP[3:6];
        times = SW_DIP[7:8];
    end
```

```
always @(posedge clk_input)
begin
//case(counter28[1])
case(counter28[19])
	1'b0:begin Enable = 8'b0000_0010; SevenSeg = SevenSegP; end
	1'b1:begin Enable = 8'b0000_0001; SevenSeg = SevenSegG; end
endcase
end

always @(posedge clk_input)    //counter++ for timing
begin
	if(!rst) begin
		counter28 = 28'b0;
	end
	else begin
		counter28 = counter28 + 1'b1;
	end
end

always @(posedge clk_input) //for different spped
begin
	if(speed)begin //0.5hz
		clk = counter28[27];
		//clk = counter28[2];
	end
	else if(!speed)begin	//1hz
		clk = counter28[25];
		//clk = counter28[1];
	end
end

always @(posedge clk or negedge rst) begin
	if(!rst) begin
		group = 2'b00;
		up = 1;
		round = 0;
		pattern2 = pattern;
		times2 = times;
```

```
            end
            else begin

                if(round == times && group == 2'b00) begin
                    LED <= LEDSet(pattern2, group);
                    SevenSegP <= SevenSet1(pattern2);
                    SevenSegG <= SevenSet2(group);
                end
                else begin
                    LED <= LEDSet(pattern2, group);
                    SevenSegP <= SevenSet1(pattern2);
                    SevenSegG <= SevenSet2(group);

                    if(group == 2'b00 && up == 1'b0)
                        pattern2 = pattern2 + mode;

                    if(group == 2'b00)
                        up = 1'b1;
                    else if(group == 2'b11)
                    begin
                        up = 1'b0;
                        round = round + 1;
                    end

                    case(up)
                    1'b1 : group = group + 1;
                    1'b0 : group = group - 1;
                    endcase
                end

            end
        end


function [15:0] LEDSet;
input [3:0] pattern;
input [1:0] group;
//input pattern and group to set the LED output
```

```
begin
	LEDSet = 16'b0;
	case(group)
	2'b00: LEDSet[3:0] = pattern;
	2'b01: LEDSet[7:4] = pattern;
	2'b10: LEDSet[11:8] = pattern;
	2'b11: LEDSet[15:12] = pattern;
	endcase
end
endfunction

function [7:0] SevenSet1;
input [3:0] digits;

begin
	case(digits)
	4'h0: SevenSet1 = 8'b00111111;
	4'h1: SevenSet1 = 8'b00000110;
	4'h2: SevenSet1 = 8'b01011011;
	4'h3: SevenSet1 = 8'b01001111;
	4'h4: SevenSet1 = 8'b01100110;
	4'h5: SevenSet1 = 8'b01101101;
	4'h6: SevenSet1 = 8'b01111101;
	4'h7: SevenSet1 = 8'b00100111;
	4'h8: SevenSet1 = 8'b01111111;
	4'h9: SevenSet1 = 8'b01101111;
	4'ha: SevenSet1 = 8'b01110111;
	4'hb: SevenSet1 = 8'b01111100;
	4'hc: SevenSet1 = 8'b00111001;
	4'hd: SevenSet1 = 8'b01011110;
	4'he: SevenSet1 = 8'b01111001;
	4'hf: SevenSet1 = 8'b01110001;
	endcase
end
endfunction

function [7:0] SevenSet2;
input [3:0] digits;
```

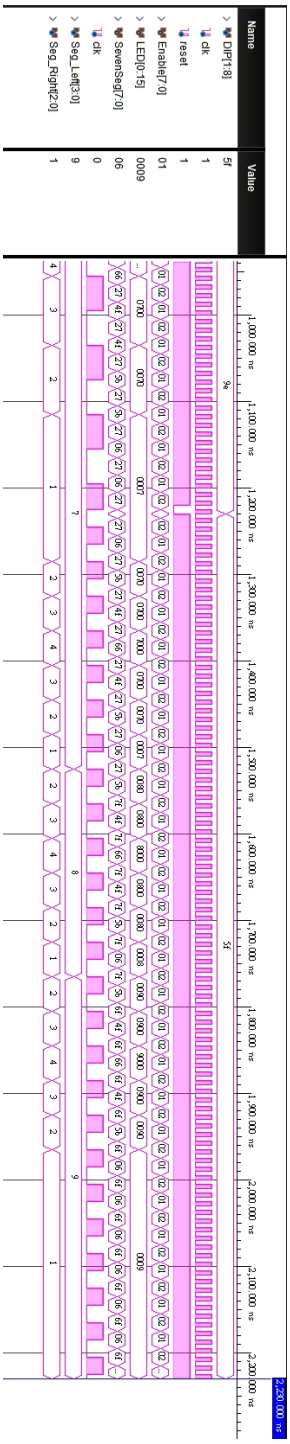
```
begin
	case(digits)
	4'h0: SevenSet2 = 8'b00000110;	//group1
	4'h1: SevenSet2 = 8'b01011011;	//g2
	4'h2: SevenSet2 = 8'b01001111;	//g3
	4'h3: SevenSet2 = 8'b01100110;	//g4
	endcase
end
endfunction

endmodule//main module end
```

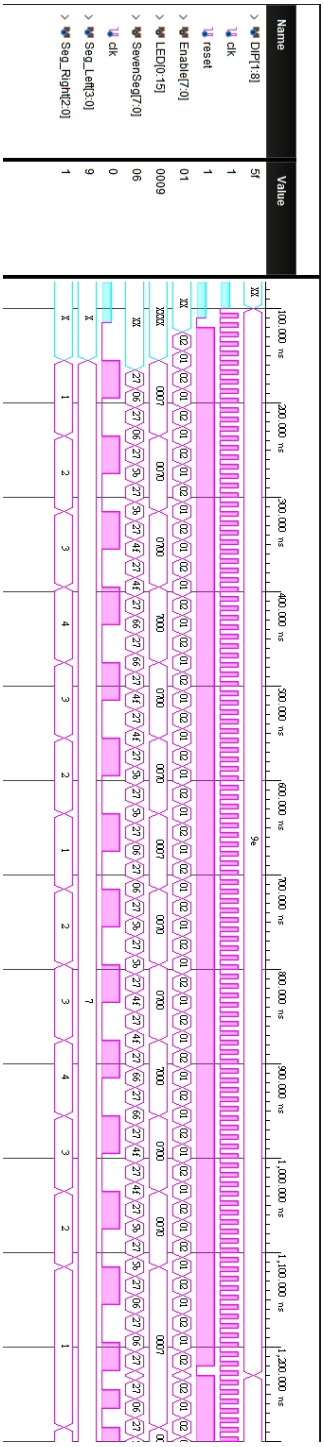
Lab_2_tb.v

```
`timescale 1ns / 1ps

module Lab_2_tb;
	//Input
	reg [1:8] DIP;
	reg clk;
	reg reset;
	//output
	wire [7:0] Enable;
	wire [0:15] LED;
	wire [7:0] SevenSeg;

	//uut
	Lab_2 uut(
	.SW_DIP(DIP),
	.clk_input(clk),
	.rst(reset),
	.Enable(Enable),
	.LED(LED),
	.SevenSeg(SevenSeg)
	);

	//reg speed;
	//reg [3:0] mode;
```

```
        //reg [3:0] pattern;
        //reg [1:0] times;

initial begin
        #100;
        clk = 1'b0;
        forever
        #5 clk = ~clk ;
end

initial
        begin
        #100;

        DIP = 8'b1001_1110;
        #10;
        reset=0;
        #10;
        reset=1;
        #600;
        reset=0;
        #10;
        DIP = 8'b0101_1111;
        reset=1;
        #600;
        $finish;
        end

endmodule
```

Behavior simulation(為方便橫向讀，左圖為 tesetbench b、右圖為 testbench a)





Synthesis timing report

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	7.495 ns	Worst Hold Slack (WHS):	0.131 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	13	Total Number of Endpoints:	13	Total Number of Endpoints:	14

All user specified timing constraints are met.