Lab 8-1 按鈕器進階操作

題目：設計一個 4 bits 計數器，當使用者按下 VeriLite PCB 版上的實體 Push-Button 一次才向上計數一次。按下 reset 則將計數器歸零。將結果顯示在 PCB 版的 LED 燈 D0~D3 上。

1. 請在" c:\logiclab\<你自己的學號>\ " 的路徑下新增一資料夾 lab8_1
2. 設定專案包含下列 Verilog 檔：
   - lab8_1.v
3. 完成設計後請 download 至 VeriLite FPGA 開發板來驗證結果。

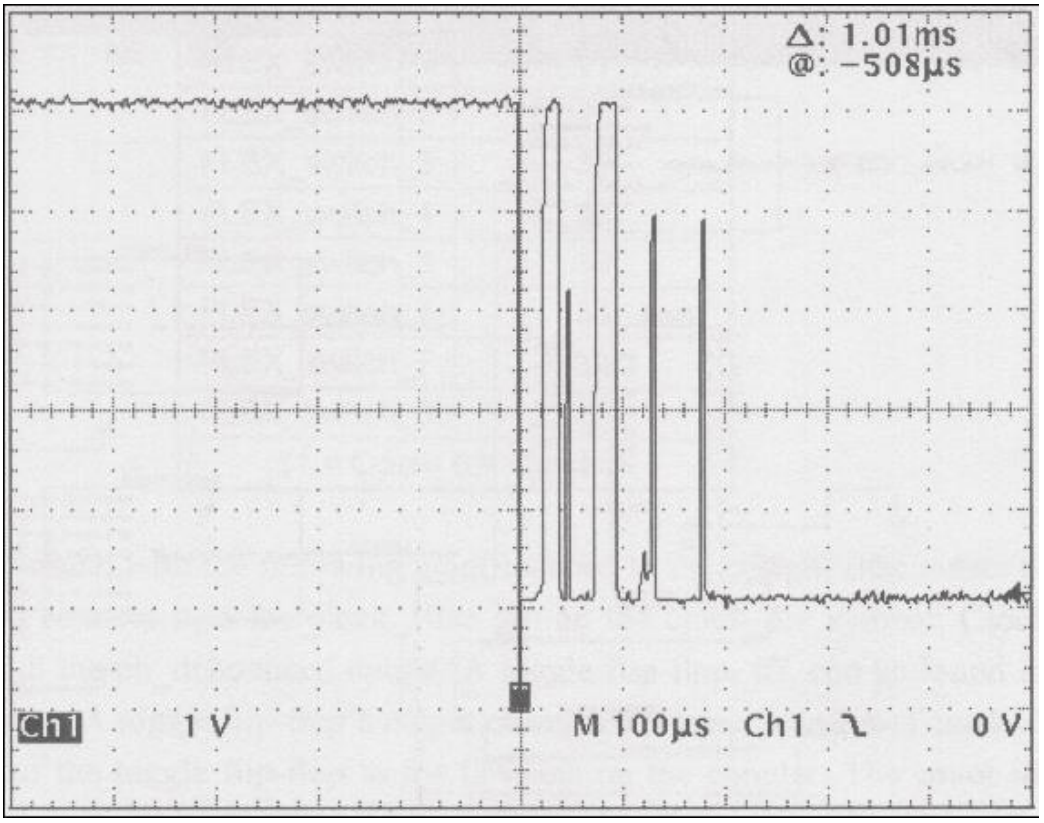
Lab8_1.v

```
module lab9_1(reset, pb, led);
    input     reset;
    input     pb;
    output    [3:0]       led;
    reg  [3:0]      counter;
    always@(posedge reset or posedge pb)
    begin
        if(reset)
            counter <= 4'd0;
        else
            counter <= counter + 4'd1;
    end

    assign led = counter;
endmodule
```

Lab 8-2 按鈕器 *De-bounce* 電路

題目：在 lab8-1 我們發現當按下 push button 時，計數器會不只計數一次，這是由於 push button 在切換高低電位時會有 bounce 現象，故 lab8-2 加入了 de-bounce 電路 oneshot.v 來消除 bounce 的影響。

1. 請在" **c:\logiclab\<你自己的學號>** " 的路徑下新增一資料夾 **lab8_2**
2. 設定專案包含下列 Verilog 檔：
   - **lab8_2.v**
   - **oneshot.v**
3. 完成設計後請 download 至 VeriLite FPGA 開發板來驗證結果。



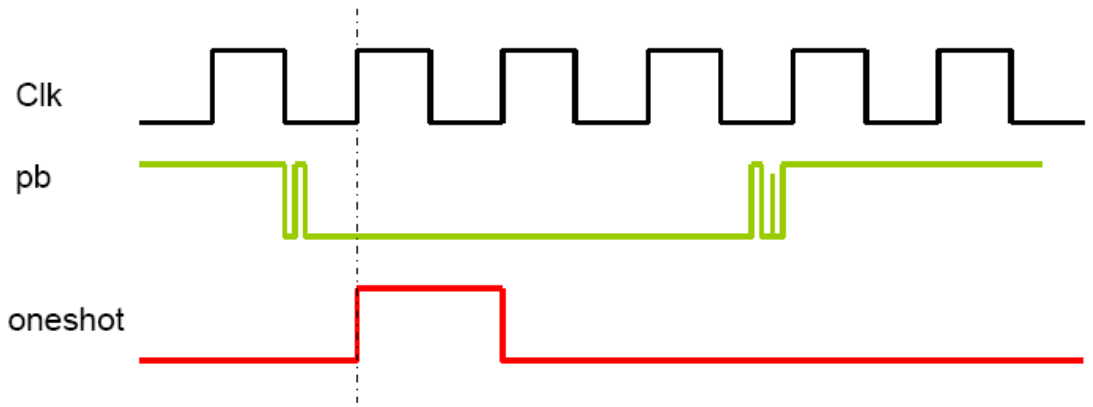
實體 **push button** 在高低電位切換時的 **bounce** 現象

註 1：上圖為實際的 push button 的輸出在高低電位切換時會有震盪的現象，因為 這個現象所以電路會不只讀到一個正(負)緣觸發，故使用實體 push button 時會有 de-bounce 電路，lab8-2 的 oneshot.v 即為一個 de-bounce 電路。VeriInstrument 虛擬儀器的 push button 不會有 bounce 現象。

註 2：VeriLite PCB 板上的 push button 為共陰極，即按下後輸出為 0，放開輸出為 1。 VeriInstrument 的虛擬儀器 push button 則為共陽極，按下輸出為 1，放開輸出為 0。

oneshot.v

```
module oneshot(clock, din, dout);

    input     clock;
    input     din;

    output    dout;

    reg [1:0] ss;

    always @(posedge clock)
    begin
    case(ss)
            2'b00:
                if(~din) //若 push button  為共陰極用~din，共陽用 din
                    ss<=2'b01;
            2'b01:
                ss<=2'b10;
            2'b10:
                if(din) //若 push button  為共陰極用 din，共陽用~din
                    ss<=2'b00;
            default:
                ss<=2'b00;
    endcase
    end

    assign dout=ss[0];

endmodule
```



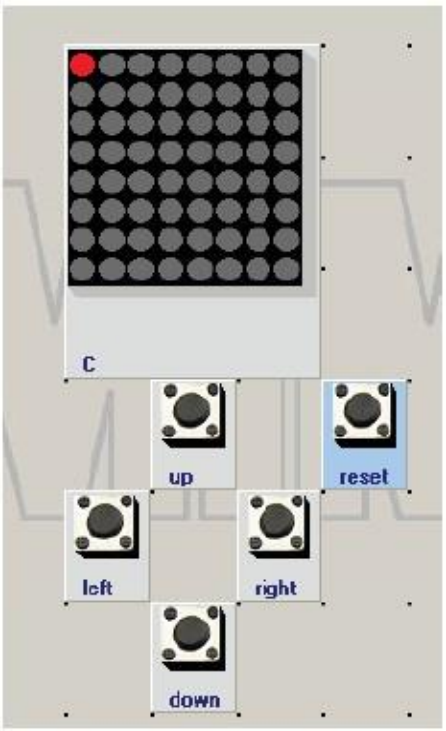
Lab8_2.v

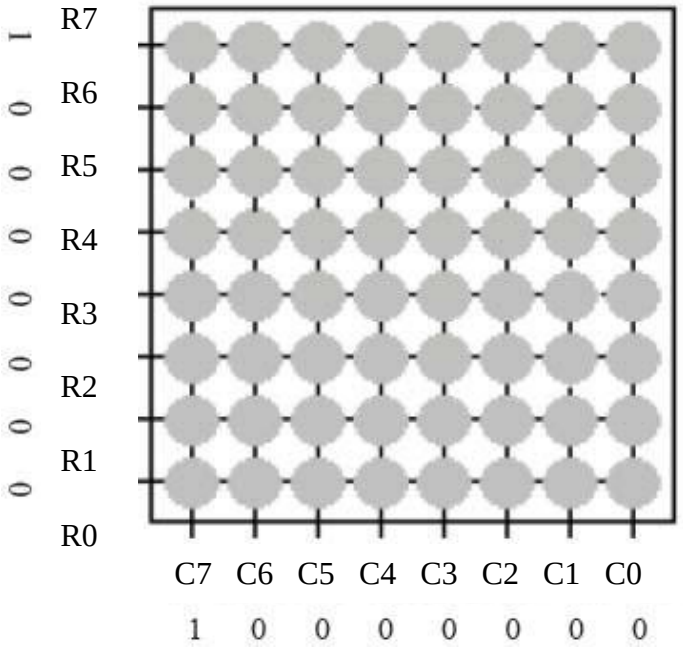
```
module lab8_2(clk, reset, pb, led);

    input           clk;
    input           reset;
    input           pb;
    output  [3:0]   led;

    reg     [3:0]   counter;
    wire            pb_oneshot;
    wire            reset_oneshot;

// De-bounce logic
    oneshot u_oneshot1(clk, pb, pb_oneshot);
    oneshot u_oneshot2(clk, reset, reset_oneshot);

// 4 bits counter
/* PS: PCB 板上的 push button 為共陰極 */

    always@(posedge reset_oneshot or posedge pb_oneshot)
    begin
        if(reset_oneshot)
            counter <= 4'd0;
        else
            counter <= counter + 4'd1;
    end

// led 為共陽極
assign led = counter;

endmodule
```

Lab 8-3　8x8 LED 矩陣之基本操作

題目：使 8x8 LED 只有一個 LED 燈亮，按下 reset 後，亮點一開始在最左上的 LED 燈，使用者可以按上下左右的 push button 來操控這個亮點的位置。(註：電路要按下 reset 後才開始運作。)

1. 請在" **c:\logiclab\<你自己的學號>** " 的路徑下新增一資料夾 **lab8_3**
2. 設定專案包含下列 Verilog 檔：
   - **lab8_3.v**
   - **oneshot.v**
3. 完成設計後請 download 至 VeriLite FPGA 開發板來驗證結果。





oneshot.v (適用於虛擬 push button)

```
module oneshot(clock, din, dout);
    input     clock;
    input     din;
    output    dout;

    reg [1:0] ss;

    always @(posedge clock)
    begin
    case(ss)
            2'b00:
                if(din) //若 push button 為共陰極用~din，共陽用 din
                    ss<=2'b01;
            2'b01:
                ss<=2'b10;
            2'b10:
                if(~din) //若 push button 為共陰極用 din，共陽用~din
                    ss<=2'b00;
            default:
                ss<=2'b00;
    endcase
    end

    assign dout=ss[0];

endmodule
```

Lab8_3.v

```
module lab8_3(clk, reset, up, down, left, right, R, C);

    input           clk;
    input           reset;
    input           up, down, left, right;
    output  [7:0]   R, C;

    reg     [7:0]   R, C;
    reg     [7:0]   R_t, C_t;
    wire    up_oneshot, down_oneshot, left_oneshot, right_oneshot;

// De-bounce logic
    oneshot     u_oneshot1(clk, up, up_oneshot);
    oneshot     u_oneshot2(clk, down, down_oneshot);
    oneshot     u_oneshot3(clk, left, left_oneshot);
    oneshot     u_oneshot4(clk, right, right_oneshot);

// 8x8 LED row control logic
    always@(posedge clk)
    begin
        if(reset)
            R <= 8'b00000001;
        else
            R <= R_t;
    end

    always@(*) /*請同學完成左右控制*/
    begin
        if(down_oneshot)
        begin
            R_t[7:1] <= R[6:0];
            R_t[0] <= R[7];
        end
        else if(up_oneshot)
        begin
            R_t[6:0] <= R[7:1];
            R_t[7] <= R[0];
        end
        else
            R_t <= R;
    end
```

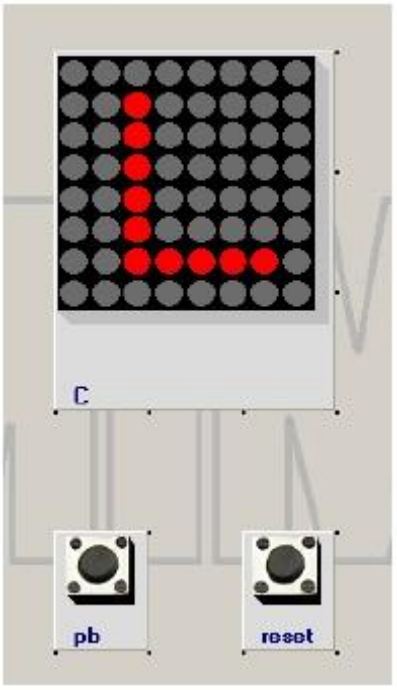
```
        always@(posedge clk)
            begin
              if(reset)
                    C <= 8'b00000001;
              else
              begin
                    C[6:0] <= C[7:1];
                    C[7] <= C[0];
              end
            end
endmodule
```

Lab 8-4　　8x8 LED 矩陣進階操作

題目：使用 8x8 LED 依序顯示文字 'L' 'O' 'G' 'I' 'C'。按下 reset 後一開始顯示'L'，每按一次 push button 就顯示下一個文字，到 'C' 再按下 push button 則回到 'L' 繼續循環。(註：電路需按下 reset 後才開始運作。)

1. 請在" **c:\logiclab\<你自己的學號>** " 的路徑下新增一資料夾 **lab8_4**
2. 設定專案包含下列 Verilog 檔：
   - **lab8_4.v**
   - **oneshot.v**
   - **clk_div.v**

3. 完成設計後請 download 至 VeriLite FPGA 開發板來驗證結果。



Lab8_4.v

```
module lab8_4(clk, reset, pb, R, C);

	input		clk;
	input		reset;
	input		pb;
	output [7:0]	R;
	output [7:0]	C;

	reg [7:0]	R1, R2, R3, R4, R5; //'L' 'O' 'G' 'I' 'C'
	reg [7:0]	R;
	reg [7:0]	C;
	reg [2:0]	counter;
	reg [2:0]	display;
	wire		pb_oneshot;
	wire		reset_oneshot;

// De-bounce logic
	oneshot		u_oneshot1(clk, pb, pb_oneshot);
	oneshot		u_oneshot2(clk, reset, reset_oneshot);

	clk_div		u_clk_div(clk, clk_1000HZ);

	always @(posedge clk_1000HZ or posedge reset_oneshot)
		if (reset_oneshot)
			counter <= 3'd0;
		else
			counter <= counter + 3'd1;

//8x8 LED row control logic
//'L'
	always @(counter)
	begin
		case(counter)
		3'd0:
			R1 = 8'b00000000;
		3'd1:
			R1 = 8'b01000000;
		3'd2:
			R1 = 8'b01000000;
		3'd3:
			R1 = 8'b01000000;
		3'd4:
			R1 = 8'b01000000;
		3'd5:
			R1 = 8'b01111110;
		3'd6:
			R1 = 8'b00000000;
		3'd7:
			R1 = 8'b00000000;
		endcase
end
```

```
//'O'
      always @(counter)
      begin
            case(counter)
            3'd0:
                  R2 = 8'b00000000;
            3'd1:
                  R2 = 8'b00111100;
            3'd2:
                  R2 = 8'b01000010;
            3'd3:
                  R2 = 8'b01000010;
            3'd4:
                  R2 = 8'b01000010;
            3'd5:
                  R2 = 8'b01000010;
            3'd6:
                  R2 = 8'b00111100;
            3'd7:
                  R2 = 8'b00000000;
            endcase
      end
//'G'
      always @(counter)
      begin
            .
            . /*請同學完成'G' , 'I' , 'C'的 row control logic */
            .
      end

//'I'

//'C'

      always@(posedge pb_oneshot or posedge reset_oneshot)
      begin if(reset_oneshot)
                  display <= 3'd0;
            else if(display == 3'd4)
                  display <= 3'd0;
            else
                  display <= display + 3'd1;
      end

      always@(*)
      begin
            case(display)
            3'd0: R = R1;
            3'd1: R = R2;
            3'd2: R = R3;
            3'd3: R = R4;
            3'd4: R = R5;
```

```
            default: R = R1;
            endcase
        end

    //8x8 LED coulumn control logic
        always@(posedge
        clk_1000HZ)
        begin
            if(rst)
                C <= 8'b10000000;
            else
            begin
                C[6:0] <= C[7:1];
                C[7] <= C[0];
            end
        end

    endmodule
```

clk_div.v

```
module clk_div (clk, clk_1000HZ);

    input clk;
    output clk_1000HZ;

    reg [31:0] count_1000HZ;
    reg clk_1000HZ;

    always@(posedge clk)
    begin
        if (count_1000HZ <= 32'd48000)
            count_1000HZ <= count_1000HZ + 32'd1;
        else
            count_1000HZ <= 32'd1;
    end

    always@(posedge clk)
    begin
        if (count_1000HZ <= 32'd24000)
            clk_1000HZ <= 1'd1;
        else
            clk_1000HZ <= 1'd0;
    end
endmodule
```