邏輯設計實驗 Lab05 結報

104060012 邱怡庭

1 Construct a 30-second down counter with pause function. When the counter goes to 0, all the LEDs will be lighted up. You can use one push button for reset and one other for pause/start function.

1.1 Implement a periodic 30-second down counter and demo with the FPGA board.

1.2 Implement Prelab 1.3 and demo with the FPGA board.

1.3 Combine 1.2 and 1.3 to finish the experiment.

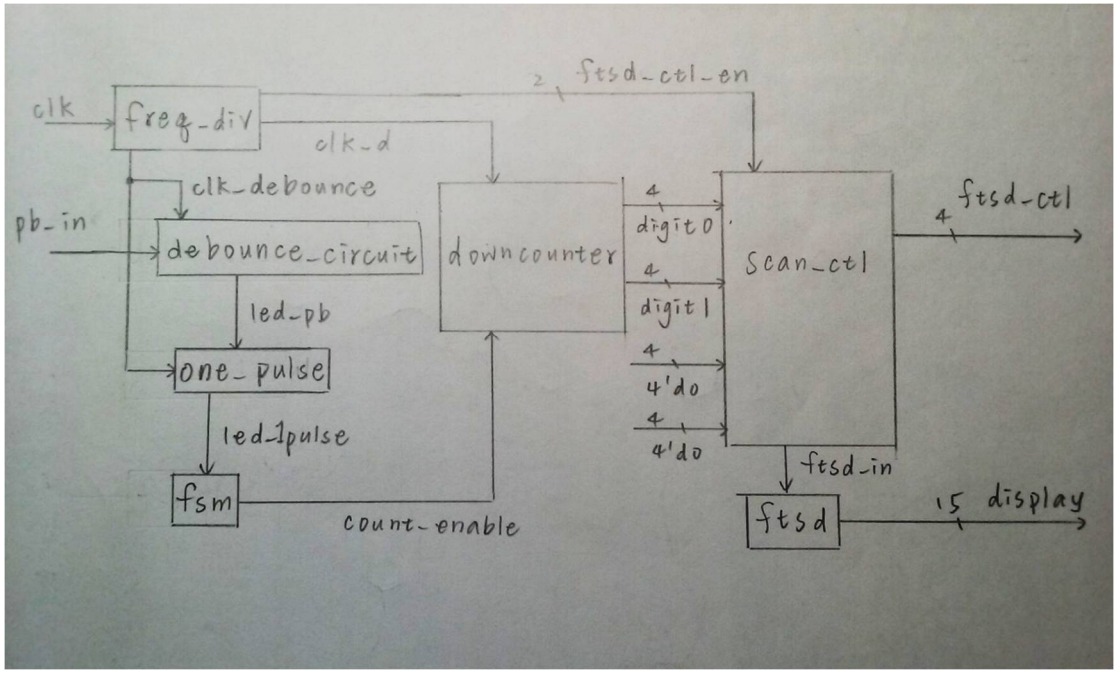
Design Specification

output : [3:0] ftsd_ctl, [14:0] display, [15:0] led

input : clk, pb_in, rst_n

wire : led_pb, led_1pulse, clk_d, [1:0]ftsd_ctl_en, [3:0]digit0, [3:0]digit1, count_enable, [3:0]ftsd_in, [3:0] in0, [3:0]in1, [3:0]in2, [3:0]in3, clk_debounce

block diagram:



Design Implementation

logic function / logic diagram:

downcounter

- if count_enable=1

  ->if digit1(個位數)=0 & digit0(十位數)!=0

  -> let digit1(個位數)=9 & let digit0(十位數)= digit0(十位數)-1

  Ex: 30->29 , 20->19, 10->9

  ->lest led=0(暗)

  ->if digit1(個位數)=0 & digit0(十位數)=0

  -> let digit1(個位數)=0 & let digit0(十位數)= 3

  periodic 30-second down counter

  -> lest led=1(亮)

  ->else

  -> let digit1(個位數)= digit1(個位數)-1 & let digit0(十位數)= digit0(十位數)

  ->lest led=0(暗)

- if count_enable=0

  ->if digit1(個位數)=0 & digit0(十位數)=0

  -> let digit1(個位數)=0 & let digit0(十位數)=0

  -> lest led=1(亮)

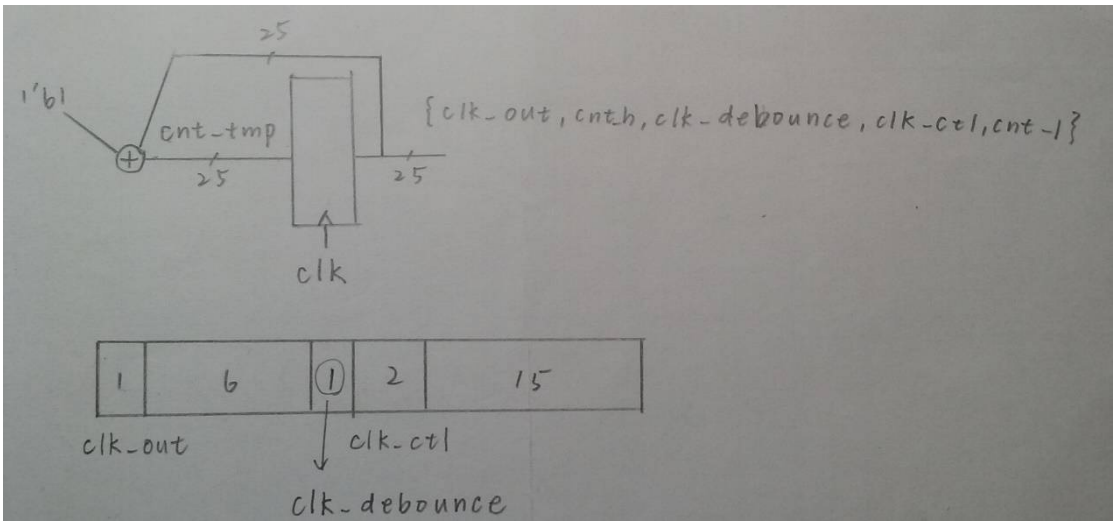
  ->else

  -> let digit1(個位數)= digit1(個位數) & let digit0(十位數)= digit0(十位數)

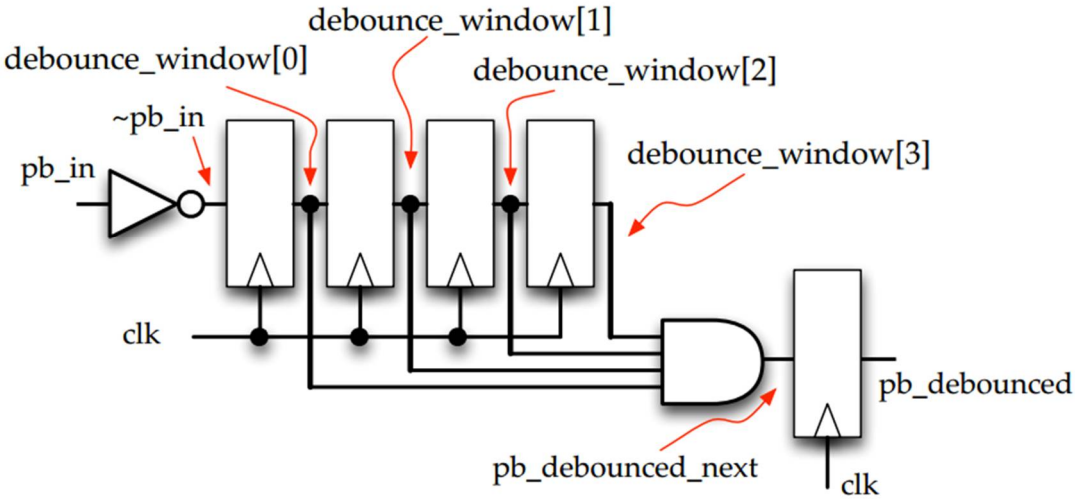
  維持原數

  ->lest led=0(暗)

freq_div:

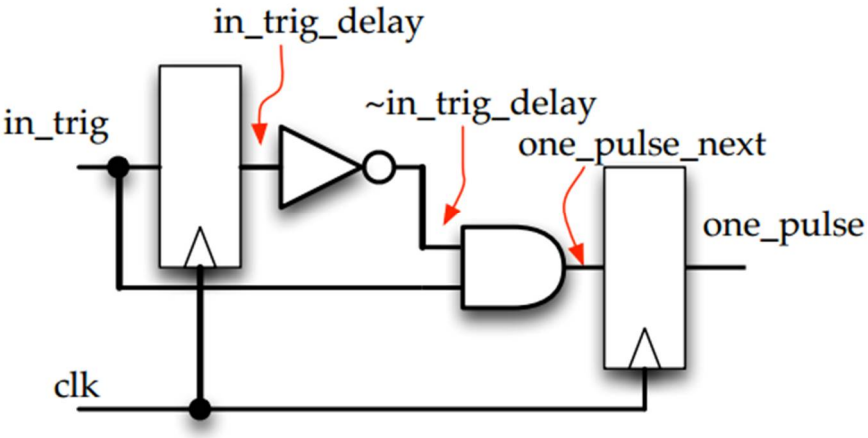


debounce_circuit



When all 4 bits of the registers are high the output of the debounce circuit changes to high

one_pulse



When one presses the push button for a short moment, the time that the switch is closed (ms range) is usually much longer than one clock period (μs or ns range). The one-pulse circuit generates only a one-clockperiod-long pulse every time the push button is hit, independent of the time one keeps the button pressed

scan_ctl

if ftsd_ctl_en=00

->控制第一個 14 段顯示器

if ftsd_ctl_en=01

->控制第二個 14 段顯示器

if ftsd_ctl_en=10

->控制第三個 14 段顯示器

if ftsd_ctl_en=11

->控制第四個 14 段顯示器

ftsd

當 bcd=4'd0: display = 15'b0000_0011_1111_111; //0

當 bcd=4'd1: display = 15'b1111_1111_1011_011; //1

當 bcd=4'd2: display = 15'b0010_0100_1111_111; //2

當 bcd=4'd3: display = 15'b0000_1100_1111_111; //3

當 bcd=4'd4: display = 15'b1001_1000_1111_111; //4

當 bcd=4'd5: display = 15'b0100_1000_1111_111; //5

當 bcd=4'd6: display = 15'b0100_0000_1111_111; //6

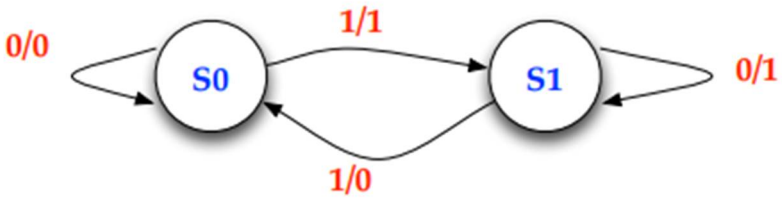
當 bcd=4'd7: display = 15'b0001_1111_1111_111; //7

當 bcd=4'd8: display = 15'b0000_0000_1111_111; //8

當 bcd=4'd9: display = 15'b0000_1000_1111_111; //9

default: display = 15'b1111_1111_1111_111; //DEF

fsm



I/O pin assignment:

NET "clk" LOC = R10;

NET "pb_in" LOC = U2;

NET "rst_n" LOC = U1;

NET "ftsd_ctl[0]" LOC = V8;

NET "ftsd_ctl[1]" LOC = U8;

NET "ftsd_ctl[2]" LOC = V6;

NET "ftsd_ctl[3]" LOC = T6;

NET "display[14]" LOC = P6;

NET "display[13]" LOC = N4;

```
NET "display[12]" LOC = V5;
NET "display[11]" LOC = T5;
NET "display[10]" LOC = U7;
NET "display[9]" LOC = R3;
NET "display[8]" LOC = N5;
NET "display[7]" LOC = R5;
NET "display[6]" LOC = T3;
NET "display[5]" LOC = T4;
NET "display[4]" LOC = V4;
NET "display[3]" LOC = V7;
NET "display[2]" LOC = R7;
NET "display[1]" LOC = T7;
NET "display[0]" LOC = U5;
NET "led[15]" LOC = H5;
NET "led[14]" LOC = H6;
NET "led[13]" LOC = F1;
NET "led[12]" LOC = F2;
NET "led[11]" LOC = J6;
NET "led[10]" LOC = J7;
NET "led[9]" LOC = G1;
NET "led[8]" LOC = G3;
NET "led[7]" LOC = K6;
NET "led[6]" LOC = L7;
NET "led[5]" LOC = H3;
NET "led[4]" LOC = H4;
NET "led[3]" LOC = K5;
NET "led[2]" LOC = L5;
NET "led[1]" LOC = K3;
NET "led[0]" LOC = K4;
```

Discussion:

一開始用同一個 40MHz 的 clock 控制所有功能，結果 pause 按鈕需要按很長的時間才會作用，因此，我取 clock 的第 18bit 做為控制 pause 的 clock，即能及時反應。

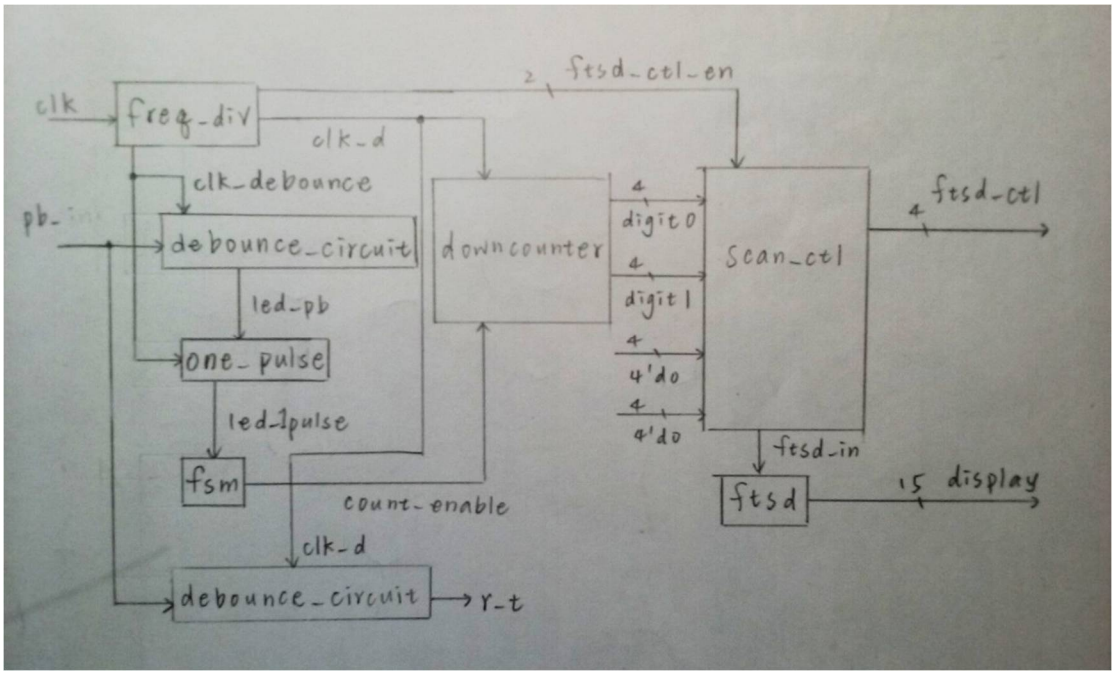
2 The same function as Exp. 1. Instead of using two push buttons for reset/pause/start, try to use just one push button to finish the design. (Hint: You can press the push button longer to represent the reset)

input clk, pb_in

wire rst_n, led_pb, led_1pulse, clk_d, [1:0]ftsd_ctl_en , [3:0]digit0, [3:0]digit1, count_enable, [3:0]ftsd_in, clk_debounce, r_t

output ftsd_ctl, display, led

block diagram:



rst_n=~r_t 當 debounce_window 四位都是 1 時，則表現 reset 的功能

Design Implementation

logic function / logic diagram:

downcounter

- if count_enable=1

  ->if digit1(個位數)=0 & digit0(十位數)!=0

  -> let digit1(個位數)=9 & let digit0(十位數)= digit0(十位數)-1

  Ex: 30->29 , 20->19, 10->9

  ->lest led=0(暗)

  ->if digit1(個位數)=0 & digit0(十位數)=0

  -> let digit1(個位數)=0 & let digit0(十位數)= 3

  periodic 30-second down counter

  -> lest led=1(亮)

->else

-> let digit1(個位數)= digit1(個位數)-1 & let digit0(十位數)= digit0(十位數)

->lest led=0(暗)

- if count_enable=0

->if digit1(個位數)=0 & digit0(十位數)=0

-> let digit1(個位數)=0 & let digit0(十位數)=0

-> lest led=1(亮)

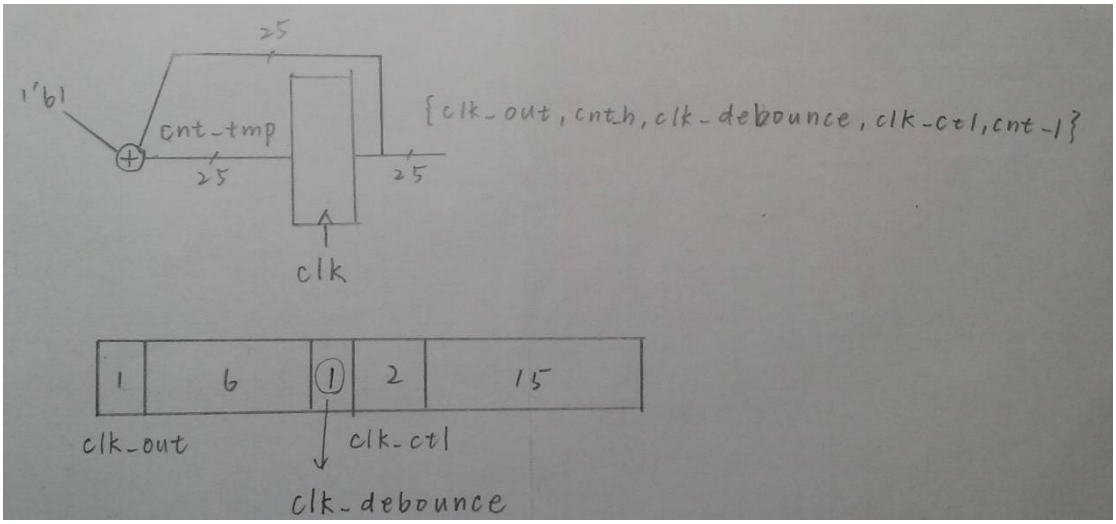
->else

-> let digit1(個位數)= digit1(個位數) & let digit0(十位數)= digit0(十位數)

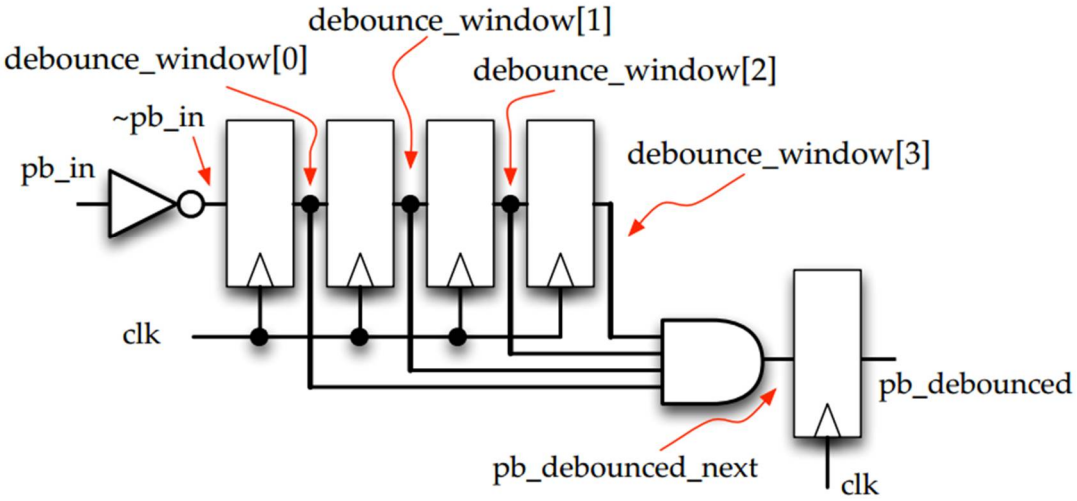
維持原數

->lest led=0(暗)

freq_div:

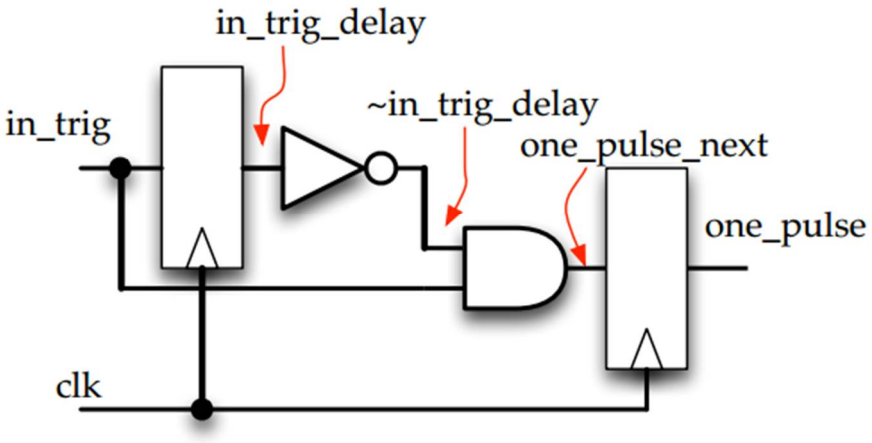


debounce_circuit



When all 4 bits of the registers are high the output of the debounce circuit changes to high

one_pulse



When one presses the push button for a short moment, the time that the switch is closed (ms range) is usually much longer than one clock period (μs or ns range). The one-pulse circuit generates only a one-clockperiod-long pulse every time the push button is hit, independent of the time one keeps the button pressed

scan_ctl

if ftsd_ctl_en=00

->控制第一個 14 段顯示器

if ftsd_ctl_en=01

->控制第二個 14 段顯示器

if ftsd_ctl_en=10

->控制第三個 14 段顯示器

if ftsd_ctl_en=11

->控制第四個 14 段顯示器

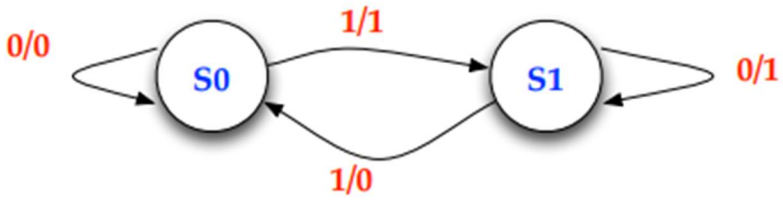
ftsd

當 bcd=4'd0: display = 15'b0000_0011_1111_111; //0

當 bcd=4'd1: display = 15'b1111_1111_1011_011; //1

當 bcd=4'd2: display = 15'b0010_0100_1111_111; //2
當 bcd=4'd3: display = 15'b0000_1100_1111_111; //3
當 bcd=4'd4: display = 15'b1001_1000_1111_111; //4
當 bcd=4'd5: display = 15'b0100_1000_1111_111; //5
當 bcd=4'd6: display = 15'b0100_0000_1111_111; //6
當 bcd=4'd7: display = 15'b0001_1111_1111_111; //7
當 bcd=4'd8: display = 15'b0000_0000_1111_111; //8
當 bcd=4'd9: display = 15'b0000_1000_1111_111; //9
default: display = 15'b1111_1111_1111_111; //DEF

fsm



I/O pin assignment:

```
NET "clk" LOC = R10;
NET "pb_in" LOC = U1;
NET "ftsd_ctl[0]" LOC = V8;
NET "ftsd_ctl[1]" LOC = U8;
NET "ftsd_ctl[2]" LOC = V6;
NET "ftsd_ctl[3]" LOC = T6;
NET "display[14]" LOC = P6;
NET "display[13]" LOC = N4;
NET "display[12]" LOC = V5;
NET "display[11]" LOC = T5;
NET "display[10]" LOC = U7;
NET "display[9]" LOC = R3;
NET "display[8]" LOC = N5;
NET "display[7]" LOC = R5;
NET "display[6]" LOC = T3;
NET "display[5]" LOC = T4;
NET "display[4]" LOC = V4;
NET "display[3]" LOC = V7;
NET "display[2]" LOC = R7;
```

```
NET "display[1]" LOC = T7;
NET "display[0]" LOC = U5;
NET "led[15]" LOC = H5;
NET "led[14]" LOC = H6;
NET "led[13]" LOC = F1;
NET "led[12]" LOC = F2;
NET "led[11]" LOC = J6;
NET "led[10]" LOC = J7;
NET "led[9]" LOC = G1;
NET "led[8]" LOC = G3;
NET "led[7]" LOC = K6;
NET "led[6]" LOC = L7;
NET "led[5]" LOC = H3;
NET "led[4]" LOC = H4;
NET "led[3]" LOC = K5;
NET "led[2]" LOC = L5;
NET "led[1]" LOC = K3;
NET "led[0]" LOC = K4;
```

Discussion:

此題相較於第一題，只需設置 1 個可以讀取長按功能的按鍵作為 reset 即可，方法即是去讀取一個 4bits 的 debounce_window，一個 clock 的時間裡只能從右邊讀進一位，當四位都是 1 時，則表現出長按時的功能，故可以僅以一個按鍵表現出更多的功能，其餘細節則都和第一題相同。

Conclusion:

在 lab05 中，我學會了如何去做一個自動倒數器，並控制她的開始、暫停、初始、以及不同狀態的初始值，另外也學會了如何控制長按時可以表現的功能！

total: 41/47
5-1 (15/16)
Design Specification (2/2)
block diagram of the design or Logic Diagram (4/4)
I/O pin assignment (2/2)
Discussion + Conclusion (3/4)
Function explanation (4/4)

5-2 (19/19)
Design Specification (2/2)
block diagram of the design or Logic Diagram (5/5)
I/O pin assignment (2/2)
Discussion + Conclusion (5/5)
Function explanation (5/5)

Bouns (0/2)
TA (7/10)