Progress this week: *Logic design lab5*

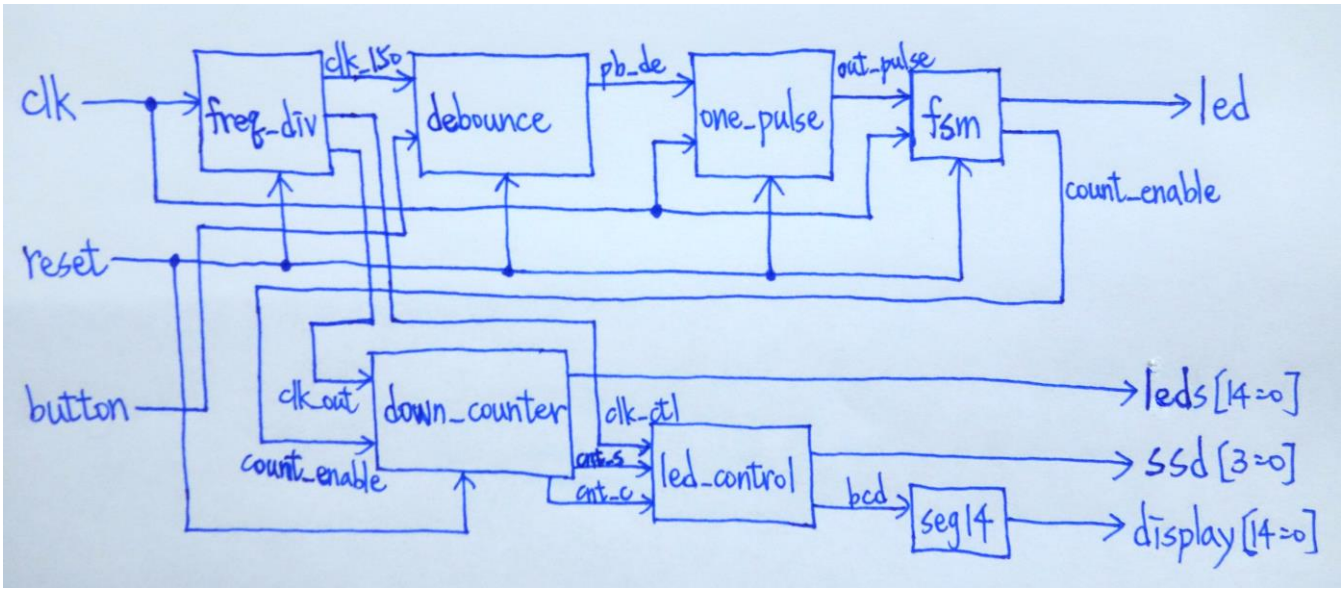
Lab5-1: Construct a 30-second down counter with pause function.

1. Design Specification
(1) Specification of my design:

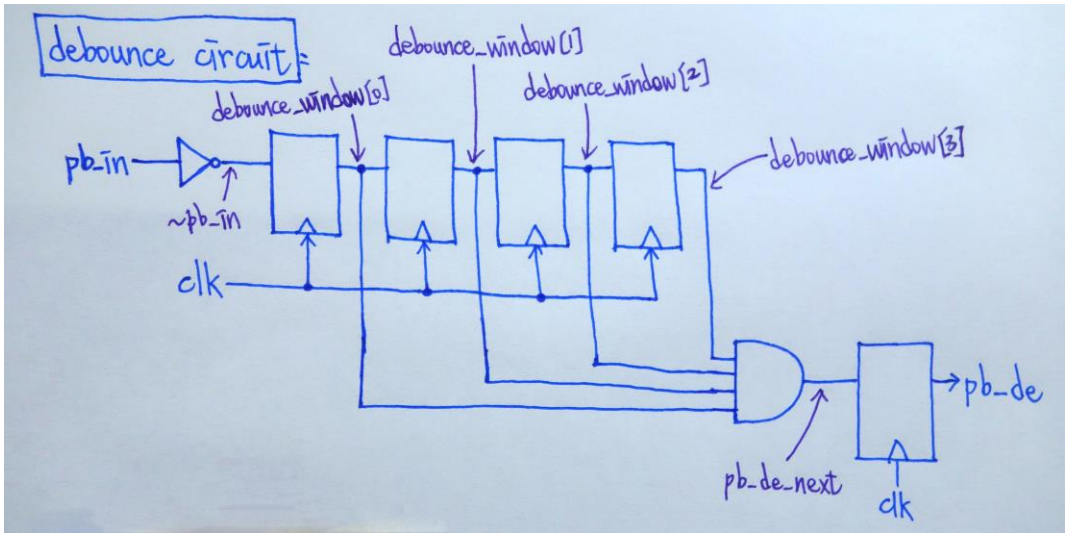
本實驗要設計一個能以按鈕控制的 30 秒倒數計時器，其中一個按鈕要控制 reset，另一個按鈕要能同時控制開始(start)與暫停(pause)，並且在時間倒數至 0 時所有 LED 燈亮起來。以下為本實驗的輸入與輸出訊號：

Input: clk, reset, button
Output: [14:0]display, [3:0]ssd, led, [14:0]leds

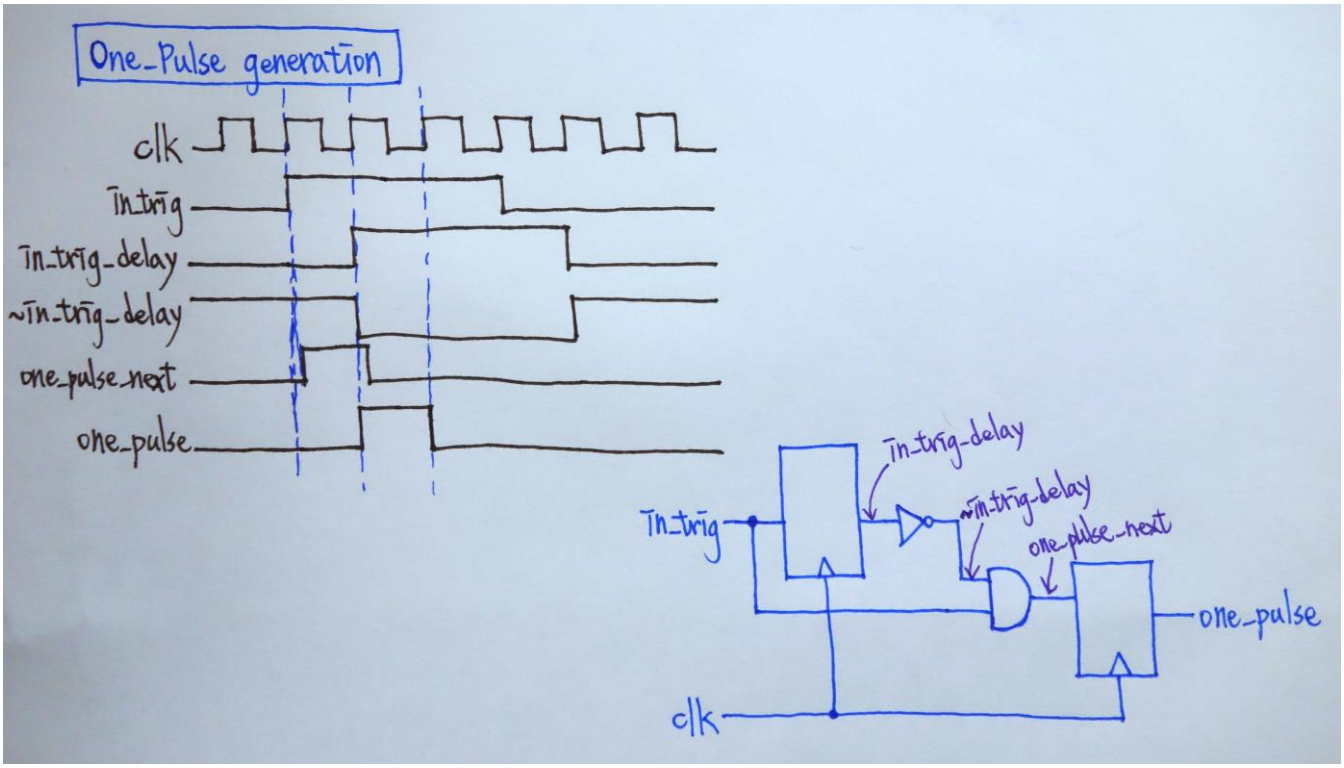
2. Design Implementation
(1) The related logic functions & Block diagram:



比起之前單純的倒數計時器，這裡多使用了 debounce 電路、one_pulse 電路、fsm 有限狀態機，由於 FPGA 板子對按鈕觸發是很敏感的，所以按下 button 一開始會有訊號不穩定的狀況，雖然不穩定的時間非常短暫，但一接收到這段不穩定的訊號就會使 input 不穩定，導致狀態判斷錯誤，所以用了 debounce 電路來去除雜訊、取得我們真正需要的訊號；不過我們還需要將 debounce 取得的訊號輸出成一個 clk 的 pulse 來觸發按鈕的功能，所以使用了 one_pulse 電路，接著就能用輸出的 one_pulse 作為 fsm 狀態轉換的條件；在 fsm 中，我將 s0 設定為 pause 狀態，s1 為 start 狀態；在 s0 狀態時，當 one_pulse=1，代表按鈕觸發動作，則 next_state=s1(開始倒數)，若 one_pulse=0，也就是沒有觸發動作，那狀態會一直維持在 s0(持續暫停)；在 s1 狀態時，當 one_pulse=1，則 next_state=s0(暫停倒數)，若 one_pulse=0，沒有觸發動作，那狀態會一直維持在 s1(持續倒數)，接著只需要與先前的倒數計時器結合修改，就能順利設計出完整的電路了。



在 debounce 電路裡運用了 4 個 shift register 來達到能取得正確值的效果，其中輸入的 clk 我使用的是約 150Hz 的頻率。



在 one_pulse 電路設計中讓我們能得到真正需要的一個 clk 能觸發的 pulse，這樣一來不管 push 的時間多長，只要有 push 到都能產生一個適當的 pulse 來觸發 fsm 避免發生錯誤。

(2) I/O pin assignment:

```
//input
NET "clk" LOC = R10;
NET "reset" LOC = T2;        //reset
NET "button" LOC = U1;       //start, pause

//output
```

```
NET "display[0]" LOC = U5;
NET "display[1]" LOC = T7;
NET "display[2]" LOC = R7;
NET "display[3]" LOC = V7;
NET "display[4]" LOC = V4;
NET "display[5]" LOC = T4;
NET "display[6]" LOC = T3;
NET "display[7]" LOC = R5;
NET "display[8]" LOC = N5;
NET "display[9]" LOC = R3;
NET "display[10]" LOC = U7;
NET "display[11]" LOC = T5;
NET "display[12]" LOC = V5;
NET "display[13]" LOC = N4;
NET "display[14]" LOC = P6;
NET "ssd[0]" LOC = V8;
NET "ssd[1]" LOC = U8;
NET "ssd[2]" LOC = V6;
NET "ssd[3]" LOC = T6;
NET "led" LOC = K4;              //current_state
NET "leds[0]" LOC = K3;
NET "leds[1]" LOC = L5;
NET "leds[2]" LOC = K5;
NET "leds[3]" LOC = H4;
NET "leds[4]" LOC = H3;
NET "leds[5]" LOC = L7;
NET "leds[6]" LOC = K6;
NET "leds[7]" LOC = G3;
NET "leds[8]" LOC = G1;
NET "leds[9]" LOC = J7;
NET "leds[10]" LOC = L6;
NET "leds[11]" LOC = F2;
NET "leds[12]" LOC = F1;
NET "leds[13]" LOC = H6;
NET "leds[14]" LOC = H5;
```

(3) Verification:

Demo 影片 : http://youtu.be/n-Vq4sT1f6g?list=UUDO7r33dqFCA5nvWzUu3mOQ

3. Discussion

一開始很直接的就想到了之前做過的倒數計時器，於是就把類似的 block 拿過來使用，而 fsm 也只有兩個 state 所以並不難設計，但 demo 時發現 push button 反應好像不太靈敏，有時候要按很多下狀態才會改變，原本以為是正常的不用理會，後來詳細看了老師上課的投影片才了解原來是因為有雜訊，最後加上了 debounce 和 one_pulse 電路就沒有這樣的問題了！

Lab5-2: Use just one push button to construct a 30-second down counter.

1. Design Specification

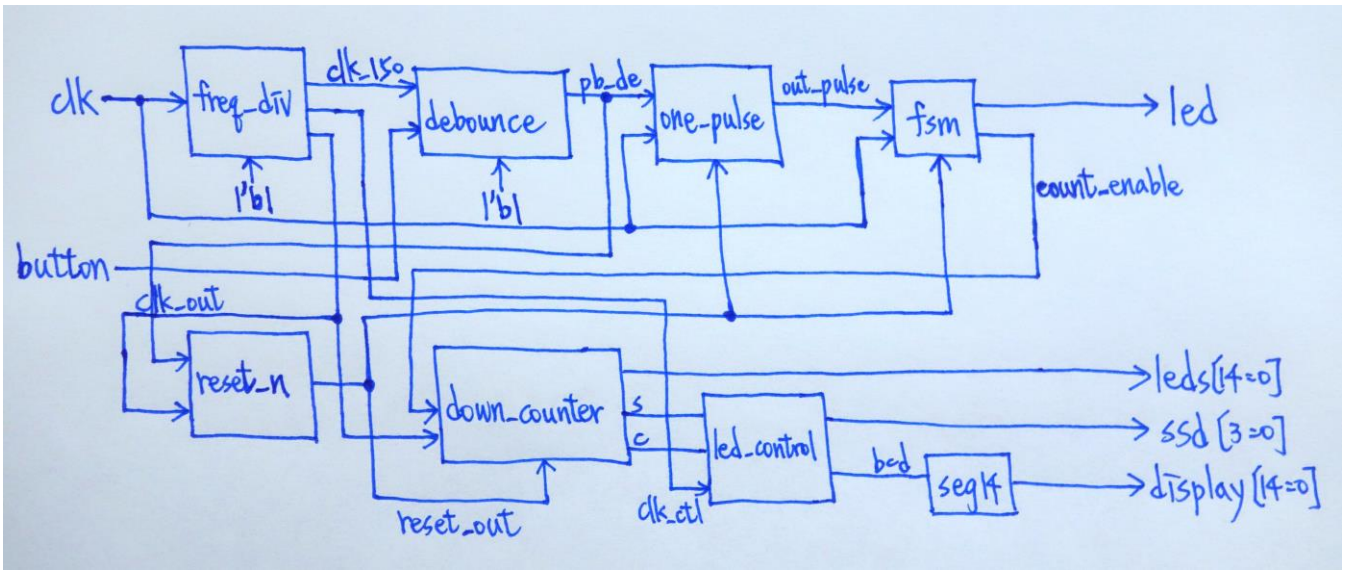
(1) Specification of my design:

本實驗要以一個按鈕來控制 30 秒倒數計時器能實現開始(start)、暫停(pause)以及重設(reset)三項功能。以下為本實驗的輸入與輸出訊號：

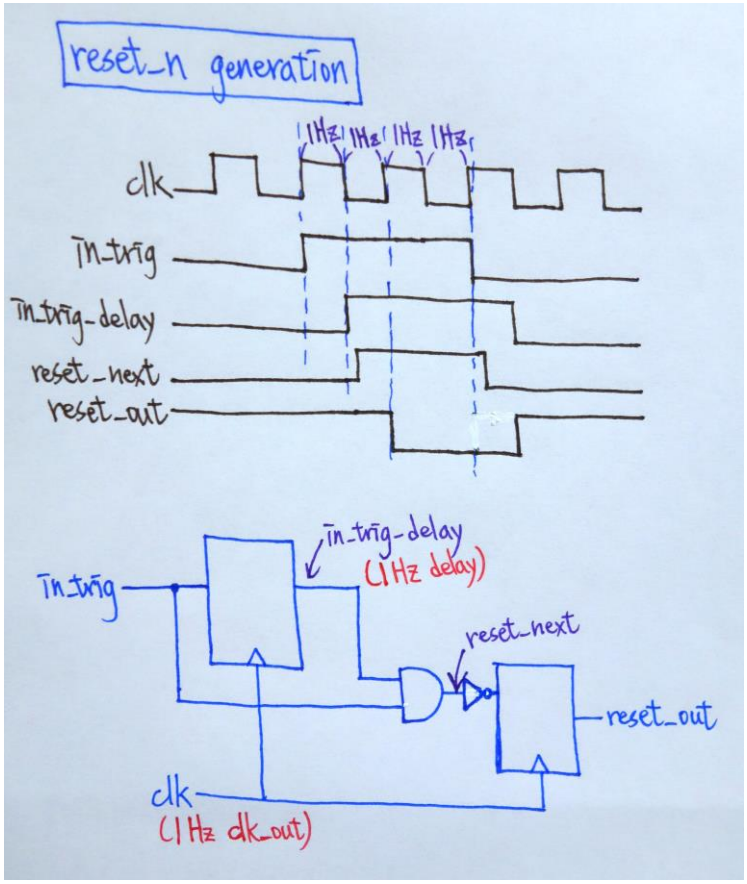
Input: clk, button
Output: [14:0]display, [3:0]ssd, led, [14:0]leds

2. Design Implementation

(1) The related logic functions & Block diagram:



這部分與上一題類似，但必須把 reset 跟 start/pause 做在同一個 button 上，所以另外設計了一個 reset_n 的電路，讓 button 持續按一秒以上就能產生負緣觸發的 reset_out；由於 reset_n 必須等 frequency diver 和 debounce 電路跑完輸出值給 reset_n 才能運作，所以直接讓 freq_div 和 debounce 的 reset=1’ b1，因為在這兩個電路中 reset 的觸發比較沒那麼必要，只要讓他們能一直運作就可以了。



這部分我自己設計了一個 reset_n，藉由輸入 1Hz 的 clk，讓 in_trig_delay 延遲一秒，那麼只要 push 超過一秒，in_trig 就會與 in_trig_delay 有重疊在 1 的狀態，用 and 來得到持續一秒以上 push 的結果，再用反向器得到我們所需要的負緣觸發的 reset_out。

(2) I/O pin assignment:

```
//input
NET "clk" LOC = R10;
NET "button" LOC = T2;        //start, pause, reset

//output
NET "display[0]" LOC = U5;
NET "display[1]" LOC = T7;
NET "display[2]" LOC = R7;
NET "display[3]" LOC = V7;
NET "display[4]" LOC = V4;
NET "display[5]" LOC = T4;
NET "display[6]" LOC = T3;
NET "display[7]" LOC = R5;
NET "display[8]" LOC = N5;
NET "display[9]" LOC = R3;
NET "display[10]" LOC = U7;
NET "display[11]" LOC = T5;
NET "display[12]" LOC = V5;
NET "display[13]" LOC = N4;
NET "display[14]" LOC = P6;
```

```
NET "ssd[0]" LOC = V8;
NET "ssd[1]" LOC = U8;
NET "ssd[2]" LOC = V6;
NET "ssd[3]" LOC = T6;
NET "led" LOC = K4;            //current_state
NET "leds[0]" LOC = K3;
NET "leds[1]" LOC = L5;
NET "leds[2]" LOC = K5;
NET "leds[3]" LOC = H4;
NET "leds[4]" LOC = H3;
NET "leds[5]" LOC = L7;
NET "leds[6]" LOC = K6;
NET "leds[7]" LOC = G3;
NET "leds[8]" LOC = G1;
NET "leds[9]" LOC = J7;
NET "leds[10]" LOC = L6;
NET "leds[11]" LOC = F2;
NET "leds[12]" LOC = F1;
NET "leds[13]" LOC = H6;
NET "leds[14]" LOC = H5;
```

(3) Verification:

Demo 影片 : http://youtu.be/_-kVZkprYpk?list=UUDO7r33dqFCA5nvWzUu3mOQ

3. Discussion

在這部分遇到的困難是每個 block 的 reset 要怎麼處理，不能等 reset_n 輸出的 reset_out 來回授給自己的 block 以及前面的 block 使用，必須有個初始的 input，block 才能開始運作，所以嘗試了很多種方法，一開始一直想辦法要用暫存器來存取 reset_out 的值來使用，但不管再 main block 裡還是 reset_n 電路裡，似乎都還是錯誤的，後來也嘗試改變電路設計方法，把 reset 去掉，但這樣的設計會變得沒那麼嚴謹，最後還是回到原本的設計，但只需要給除頻器跟 debounce 的 reset 為 1 就可以順利運作了！

Lab5-3: Use two push buttons to control a two-mode down counter with pause function.

1. Design Specification

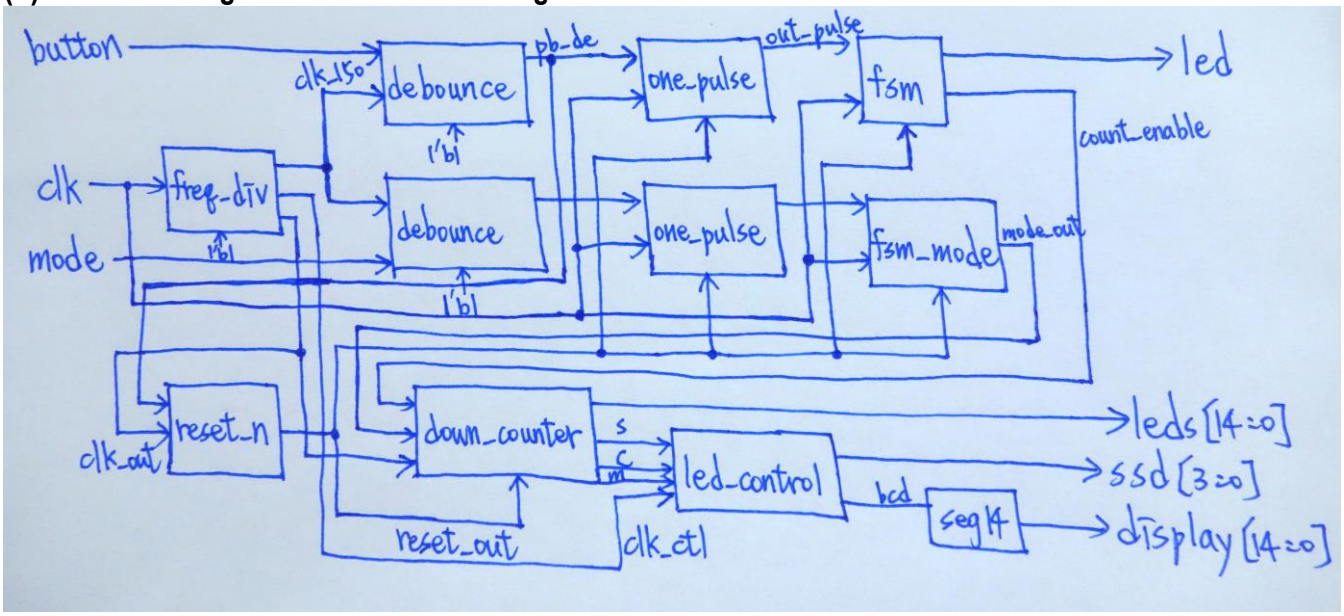
(1) Specification of my design:

本實驗要使用兩個按鈕來控制兩種模式的倒數計時器，其中一個按鈕要控制模式的切換(30 秒/1 分鐘)，而另一個按鈕則是要控制開始(start)、暫停(pause)以及重設(reset)。以下為本實驗的輸入與輸出訊號：

Input: clk, button, mode
Output: [14:0]display, [3:0]ssd, led, [14:0]leds

2. Design Implementation

(1) The related logic functions & Block diagram:



這部分的設計原理與上一題相似，只是多了一個 mode 的 button 作為電路判斷的條件，所以在 debounce、one_pulse 這些相同的按鈕觸發動作，我直接運用相同的 block 複製一份來兜 mode 電路，不過在 fsm 部分，由於 button 的輸出是要控制 counter 的 enable，而 mode 的輸出是要控制 counter 的初始值，所以在 fsm 上的設計會有些微的不同，就沒有用同一個 block 了。

(2) I/O pin assignment:

```
//input
NET "clk" LOC = R10;
NET "button" LOC = U1;        //start, pause, reset
NET "mode" LOC = T2;          //30s or 1min

//output
NET "display[0]" LOC = U5;
NET "display[1]" LOC = T7;
NET "display[2]" LOC = R7;
NET "display[3]" LOC = V7;
NET "display[4]" LOC = V4;
NET "display[5]" LOC = T4;
NET "display[6]" LOC = T3;
NET "display[7]" LOC = R5;
NET "display[8]" LOC = N5;
NET "display[9]" LOC = R3;
NET "display[10]" LOC = U7;
NET "display[11]" LOC = T5;
NET "display[12]" LOC = V5;
NET "display[13]" LOC = N4;
NET "display[14]" LOC = P6;
NET "ssd[0]" LOC = V8;
NET "ssd[1]" LOC = U8;
NET "ssd[2]" LOC = V6;
```

```
NET "ssd[3]" LOC = T6;
NET "led" LOC = K4;            //current_state
NET "leds[0]" LOC = K3;
NET "leds[1]" LOC = L5;
NET "leds[2]" LOC = K5;
NET "leds[3]" LOC = H4;
NET "leds[4]" LOC = H3;
NET "leds[5]" LOC = L7;
NET "leds[6]" LOC = K6;
NET "leds[7]" LOC = G3;
NET "leds[8]" LOC = G1;
NET "leds[9]" LOC = J7;
NET "leds[10]" LOC = L6;
NET "leds[11]" LOC = F2;
NET "leds[12]" LOC = F1;
NET "leds[13]" LOC = H6;
NET "leds[14]" LOC = H5;
```

(3) Verification:

Demo 影片 : http://youtu.be/tN2IEiXE7xk?list=UUDO7r33dqFCA5nvWzUu3mOQ

3. Discussion

這部分比較困難的是 down_counter 電路中與 mode、button 之間的關聯，判斷式比較複雜，必須謹慎把每一種情況都設計到，否則在計數上與模式的轉換就會有 bug，並且這部分使用了非常多 block，所以在連結的地方也要很仔細檢查才不會有誤。

4. Conclusion

這次的實驗困難許多，讓我花費很多時間與心力來完成，但也從中學習到很多，設計電路也越來越嚴謹，尤其是懂了 debounce 和 one_pulse 原理，了解了之後就能一步步解開困難了，還能將理解的東西融會貫通，讓自己在電路設計上多了一些不同的思維，真的成長很多呢！

5. References

Samir Palnitkar, “Verilog HDL:a guide to digital design and synthesis , 2nd ed”
M. Morris Mano/Michael D. Ciletti, “Digital Design , fourth edition”
His-Pin Ma, “Using Push Buttons”, Logic design lab05