邏輯設計實驗Lab5結報

105060012張育菘

**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.**

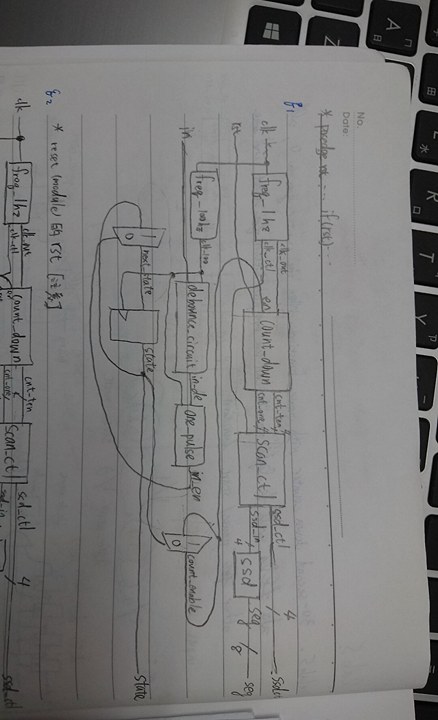
|  |  |  |  |
| --- | --- | --- | --- |
| clk | state | rst | in |
| W5 | U16 | V17 | W19 |

Design Specification

Input：clk,rst,in;

Output：ssd\_ctl[3:0],seg[7:0],state;

block diagram :



Design Implementation

Logic function :

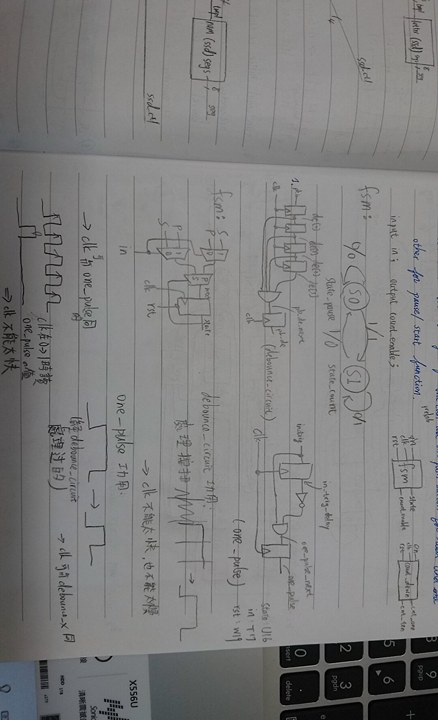
1. display：

此display module，專門拿來呼叫其他小moudule的。

還有做為finite state machine，以接出count\_enable至count\_down決定是否能開始執行count\_down的關鍵。

這裡的clk我是接100hz，因為clk在0🡪1時讀one\_pulse(其clk為100hz)處理過的值，因此display的clk不能太快。

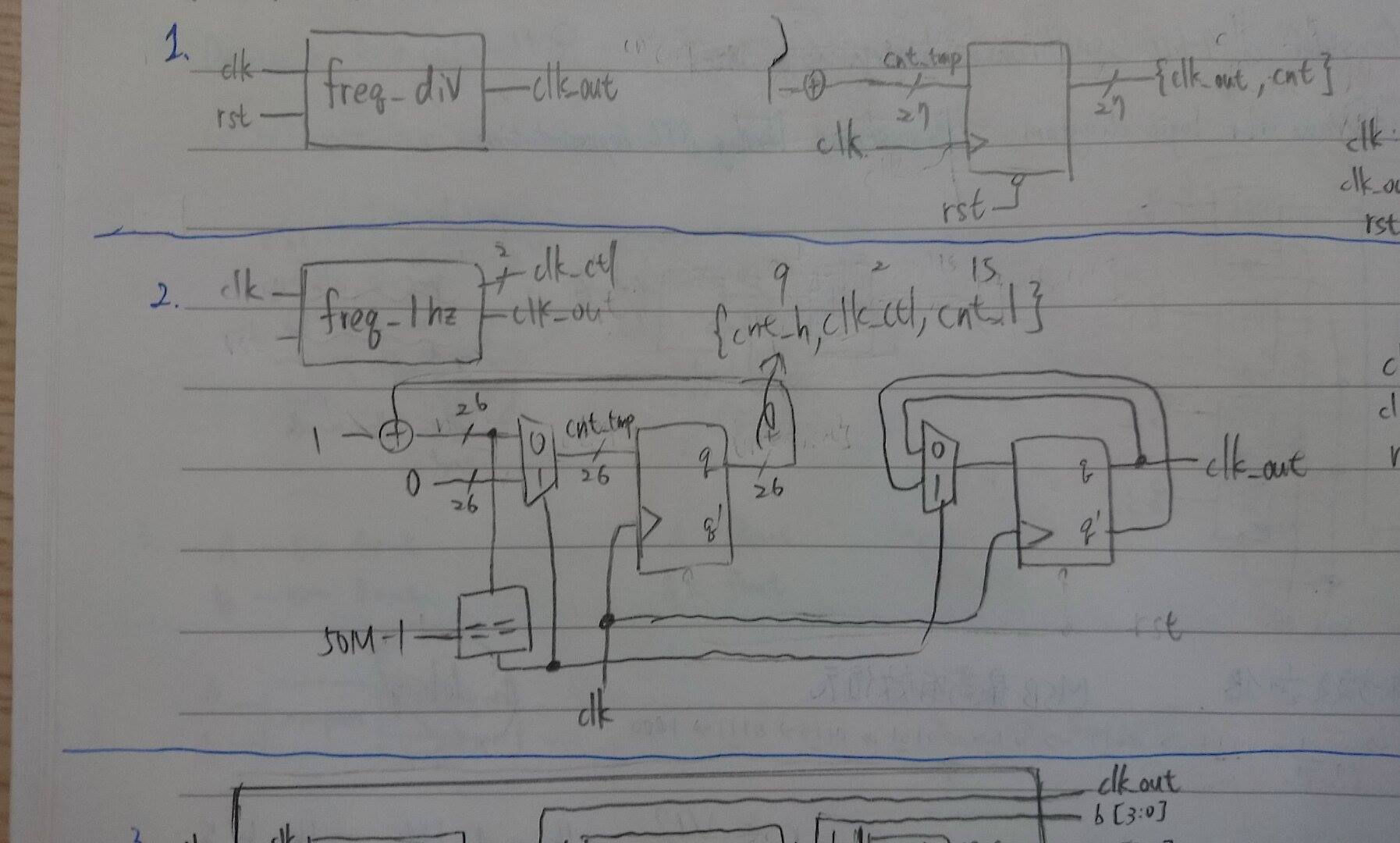
* fsm不能接display的input rst，因為那裡的rst是希望count\_down重新開始數。當rst=0時，freq\_100hz輸出的clk\_100持續為0🡪fsm不做事。



1. freq\_1hz：

此為除頻的module，分別輸出1hz的clk\_out控制count\_down以及控制scan\_ctl的clk\_ctl。

* 這裡不能接display的input rst，因為那裡的rst是希望count\_down重新開始數。當rst=0時，freq\_1hz輸出的clk持續為0🡪freq\_1hz不做事。



1. freq\_100hz：

此為除頻的module，輸出100hz的clk\_out當作debounce\_circuit、one\_pulse以及fsm的clk(在display內)。

* 這裡不能接display的input rst，因為那裡的rst是希望count\_down重新開始數。當rst=0時，freq\_100hz輸出的clk持續為0🡪freq\_100hz不做事。

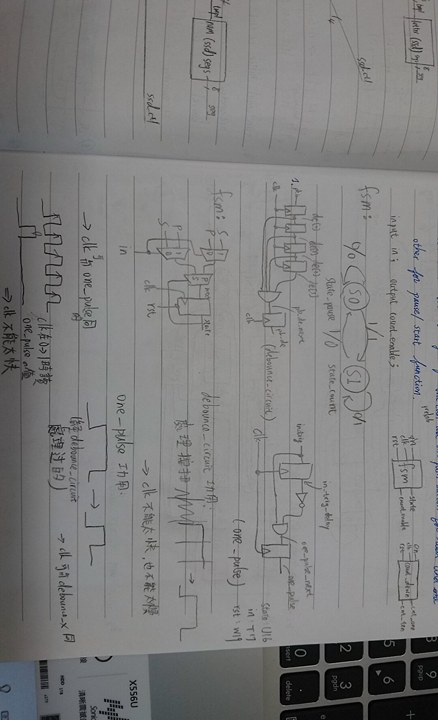


1. debounce\_circuit：

此為**除按鈕雜訊**的module，原理是，當debounce\_circuit收到連續4個1時才產生一個0🡪1的訊號，這樣可以避免收到前段起伏不定的雜訊，達到除雜訊的效果。

且要接100hz的clk，因為若接此的clk頻率太快就無法達到除雜訊的效果，太慢可能會超過按鈕維持穩定值的時間，因此我選擇100hz作為debounce\_circuit的clk。

* 這裡不能接display的input rst，因為那裡的rst是希望count\_down重新開始數。當rst=0時，freq\_100hz輸出的clk\_100持續為0🡪 debounce\_circuit不做事。

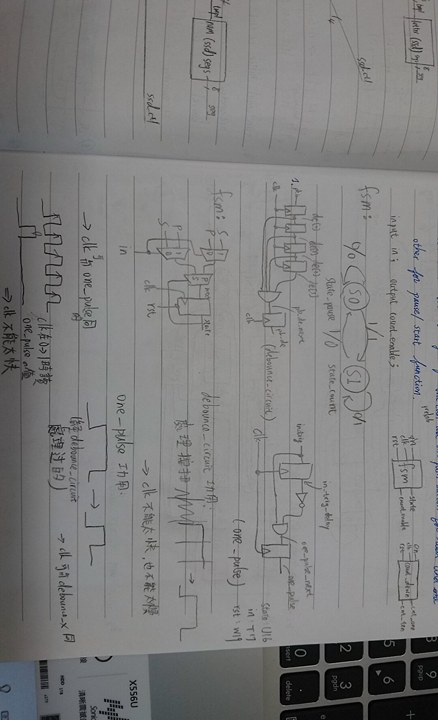


1. one\_pulse：

輸入訊號經過debounce\_circuit處理後，其週期可能會超過1個clk的時間，因此需要此module使訊號的週期為1個clk的時間。

這裡的clk我也是接與debounce\_circuit同的100hz。

* 這裡不能接display的input rst，因為那裡的rst是希望count\_down重新開始數。當rst=0時，freq\_100hz輸出的clk\_100持續為0🡪 one\_pulse不做事。



1. count\_down：

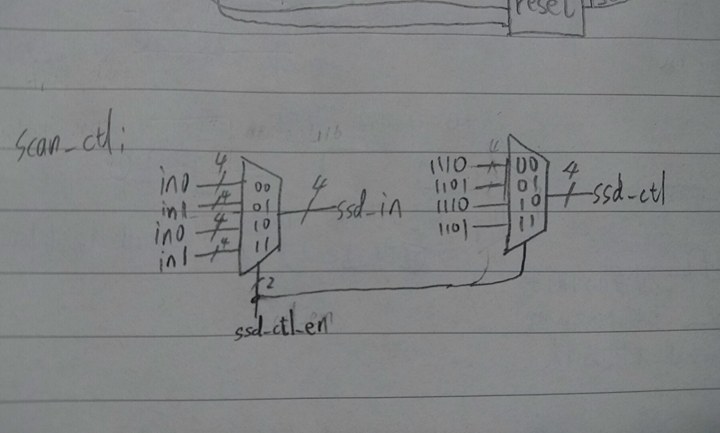
這是同時數兩個變數，個位數cnt\_one跟十位數cnt\_ten進行往下數。當cnt\_one=0時把cnt\_ten的數值減一以及cnt\_one的數值變成9；當cnt\_one跟cnt\_ten都為0時，讓兩個數都維持0。而初始化(rst=0)就是把數設為30(cnt\_ten=3 & cnt\_one=0)。

這裡還接一個en，作為是否能往下數的判斷值，若en=1，表示可以往下數；若en=0，則不能繼續往下數。

1. scan\_ctl：

此module是讓七段顯示器同時顯示不同位數的module。

ssd\_ctl\_en(from freq\_1hz的clk\_ctl[2:0])的用意為顯示兩個不同的數，因為七段顯示器一次只能顯示一種數次，因此需要一個頻率介於內建時間到1hz的ssd\_ctl\_en產生同時顯示兩個數的錯覺。



1. ssd：

當bcd 為 4'd0: segs = 8'b00000011; //七段顯示器顯示0

當bcd 為 4'd1: segs = 8'b10011111; //七段顯示器顯示1

當bcd 為 4'd2: segs = 8'b00100101; //七段顯示器顯示2

當bcd 為 4'd3: segs = 8'b00001101; //七段顯示器顯示3

當bcd 為 4'd4: segs = 8'b10011001; //七段顯示器顯示4

當bcd 為 4'd5: segs = 8'b01001001; //七段顯示器顯示5

當bcd 為 4'd6: segs = 8'b01000001; //七段顯示器顯示6

當bcd 為 4'd7: segs = 8'b00011111; //七段顯示器顯示7

當bcd 為 4'd8: segs = 8'b00000001; //七段顯示器顯示8

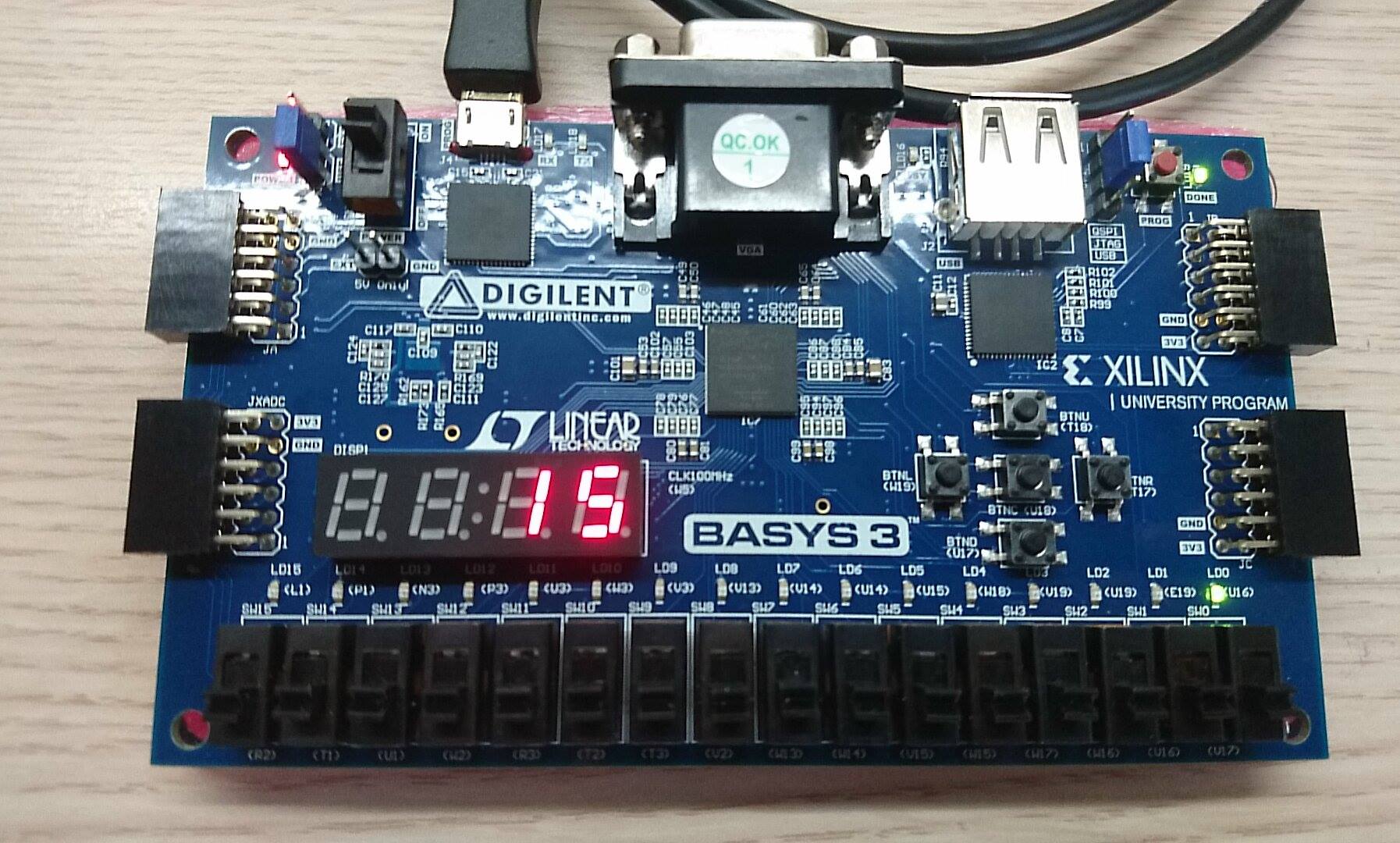
當bcd 為 4'd9: segs = 8'b00001001; //七段顯示器顯示9

default: segs = 8'b00000000; //七段顯示器顯示8.

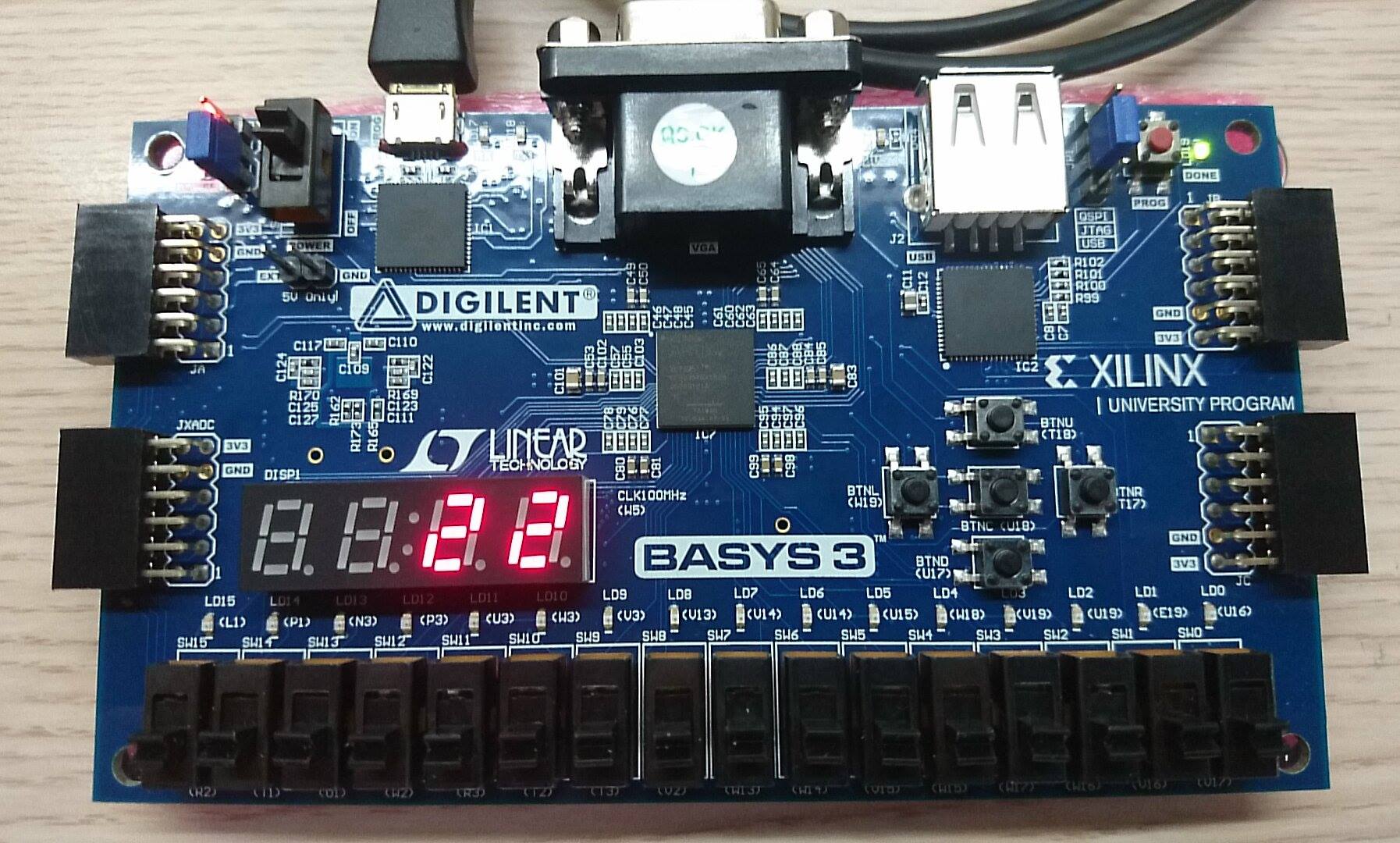
Result

1. 圖一：當U16亮，表示state=1🡪en=1🡪開始往下數。
2. 圖二：當U16暗，表示state=0🡪en=0🡪停止往下數。
3. 圖三：當按下V17，表示rst=1🡪數字重新回至30，且停止往下數。

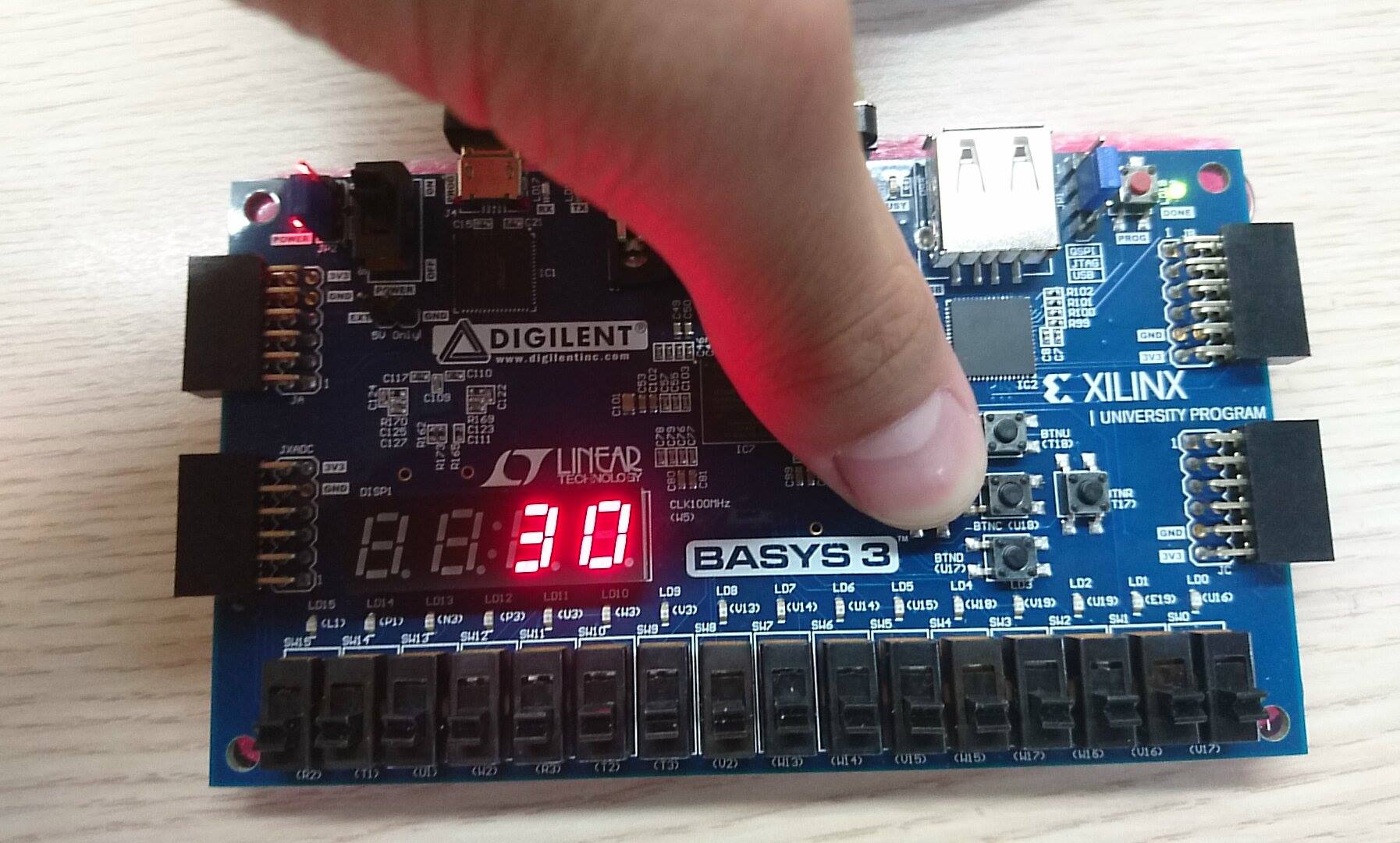
圖一



圖二



圖三



Discussion

1. 這個實驗最重要的要注意rst接的module，因為這裡的rst是希望count\_down重新開始數。
2. count\_down判斷rst要在rst從0🡪1時判斷，因為rst是一個按鈕，按下的過程是0🡪1，且要在rst==1時做出重置，這是跟以往實驗較不同之處。
3. 處理按鈕雜訊要使用debounce\_circuit，原理在上述已有做說明了。
4. 若要使多個週期的訊號簡化成週期為1個clk則要使用one\_pulse去做處理。

**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)**

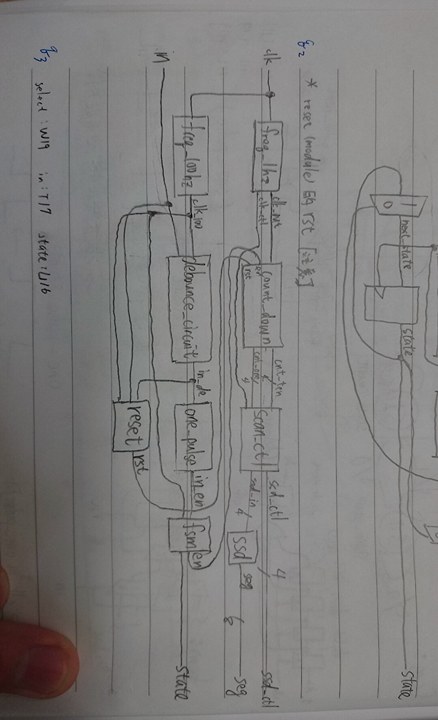
|  |  |  |
| --- | --- | --- |
| clk | state | in |
| W5 | U16 | W19 |

Design Specification

Input：clk,in;

Output：ssd\_ctl[3:0],seg[7:0],state;

block diagram :



Design Implementation

Logic function :

1. display：

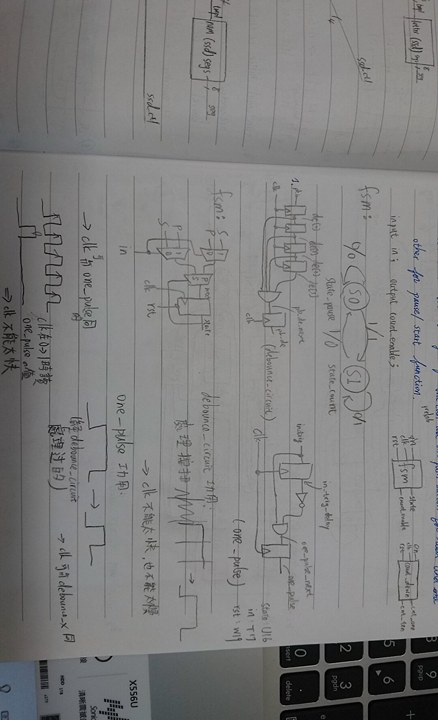
此display module，專門拿來呼叫其他小moudule的。

1. fsm：

此module做為finite state machine，以接出count\_enable至count\_down決定是否能開始執行count\_down的關鍵。

這裡的clk我是接100hz，因為clk在0🡪1時讀one\_pulse(其clk為100hz)處理過的值，因此display的clk不能太快。

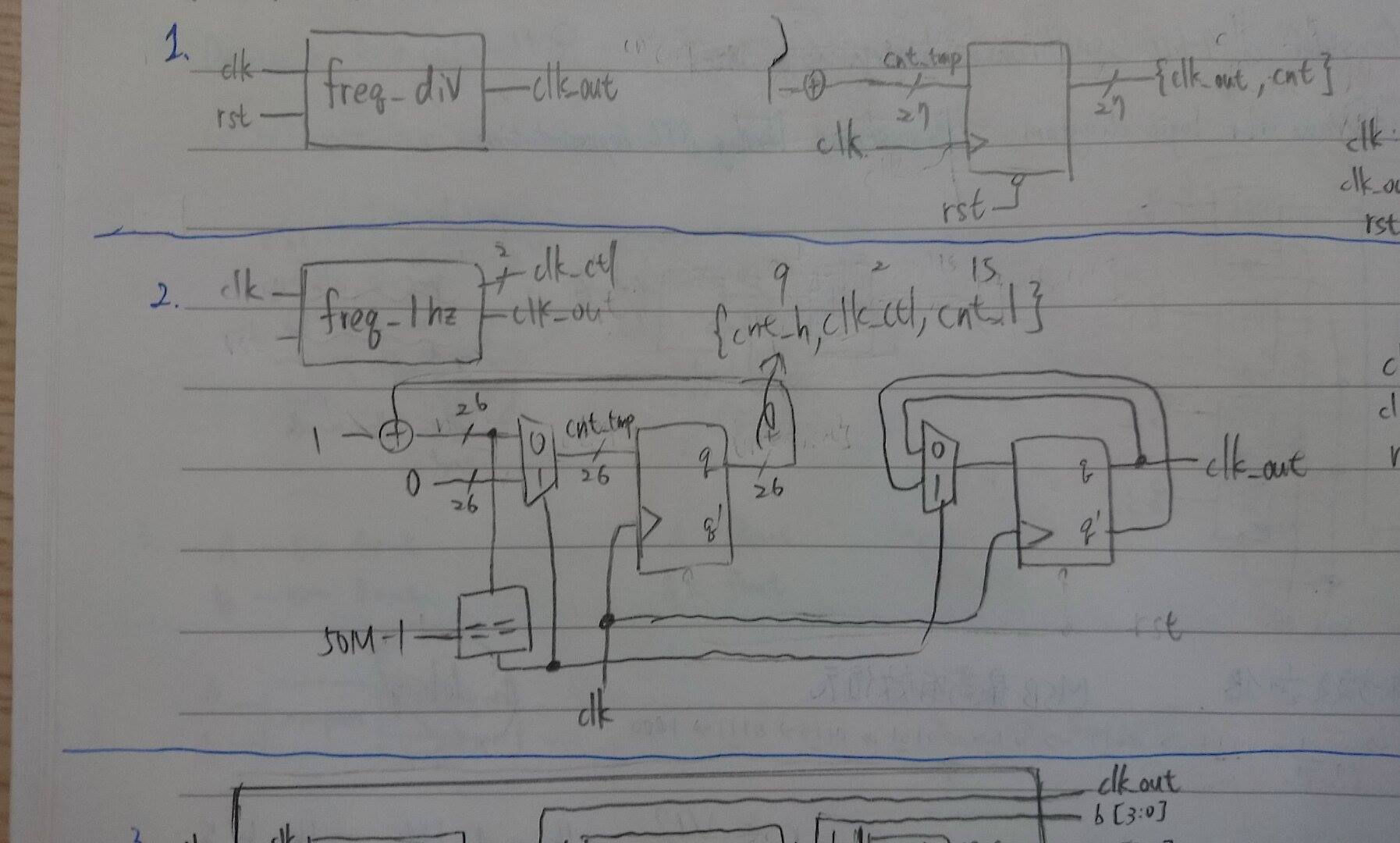
* fsm不能接display的input rst，因為那裡的rst是希望count\_down重新開始數。當rst=0時，freq\_100hz輸出的clk\_100持續為0🡪fsm不做事。



1. freq\_1hz：

此為除頻的module，分別輸出1hz的clk\_out控制count\_down以及控制scan\_ctl的clk\_ctl。

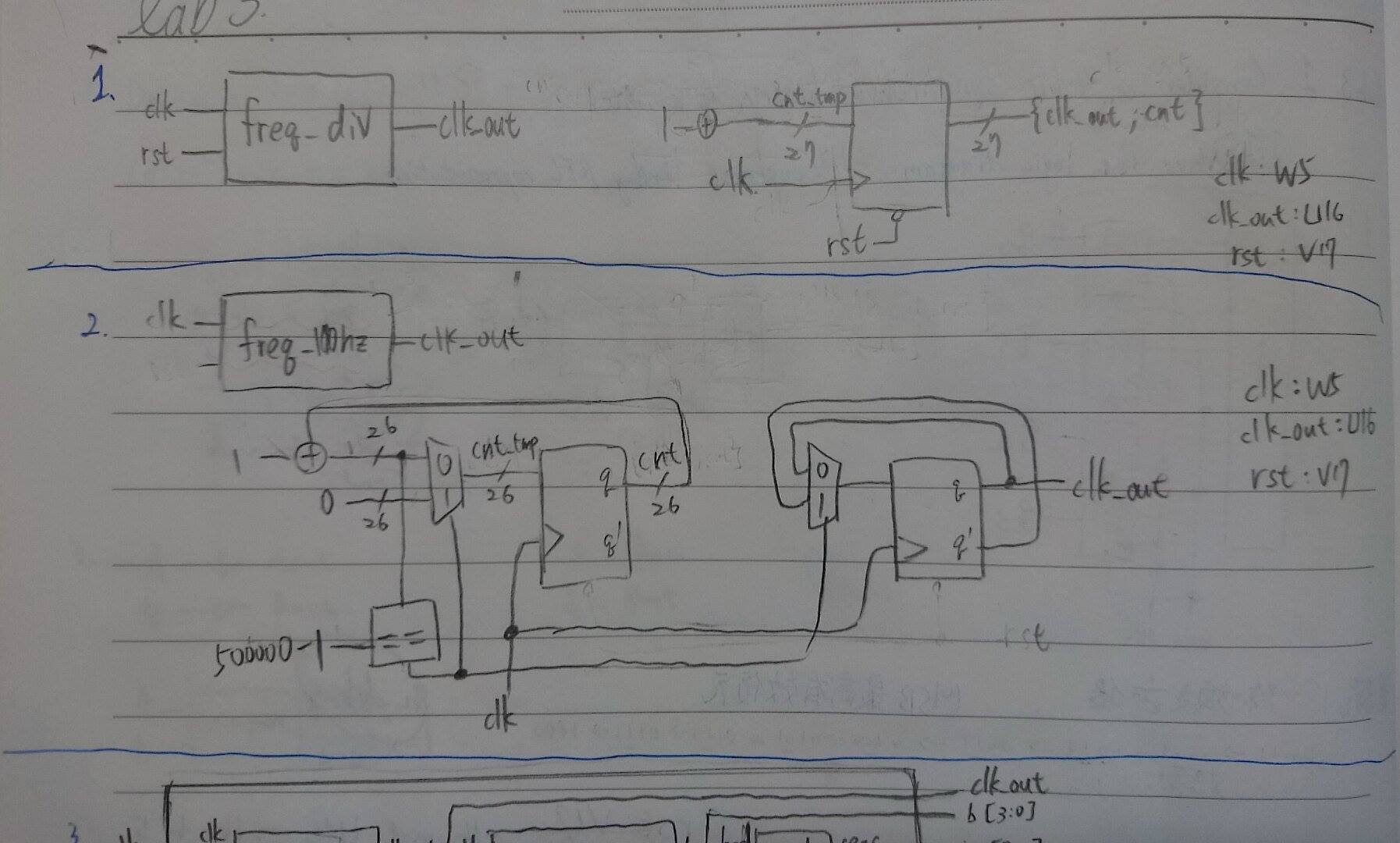
* 這裡不能接display的input rst，因為那裡的rst是希望count\_down重新開始數。當rst=0時，freq\_1hz輸出的clk持續為0🡪freq\_1hz不做事。



1. freq\_100hz：

此為除頻的module，輸出100hz的clk\_out當作debounce\_circuit、one\_pulse以及fsm的clk(在display內)。

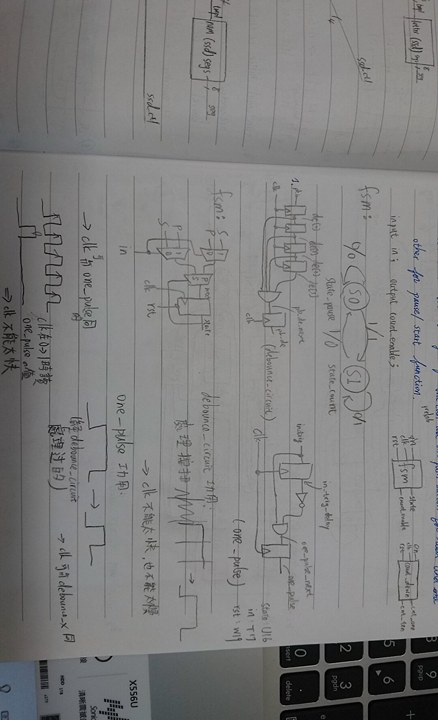
* 這裡不能接display的input rst，因為那裡的rst是希望count\_down重新開始數。當rst=0時，freq\_100hz輸出的clk持續為0🡪freq\_100hz不做事。



1. debounce\_circuit：

此為**除按鈕雜訊**的module，且要接100hz的clk，因為若接此的clk頻率太快就無法達到除雜訊的效果，太慢可能會超過按鈕維持穩定值的時間，因此我選擇100hz作為debounce\_circuit的clk。

* 這裡不能接display的input rst，因為那裡的rst是希望count\_down重新開始數。當rst=0時，freq\_100hz輸出的clk\_100持續為0🡪 debounce\_circuit不做事。

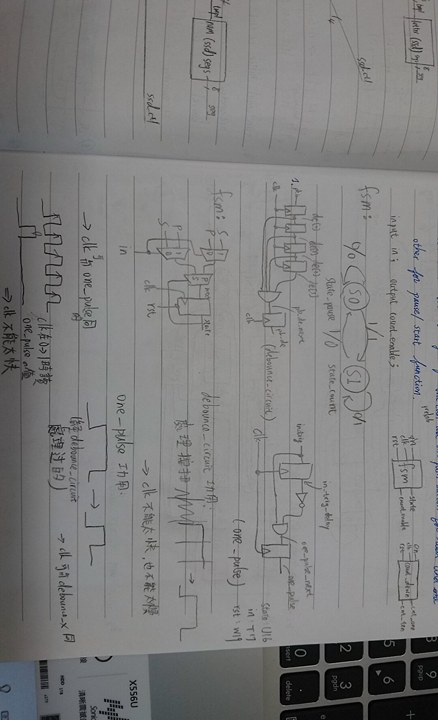


1. one\_pulse：

輸入訊號經過debounce\_circuit處理後，其週期可能會超過1個clk的時間，因此需要此module使訊號的週期為1個clk的時間。

這裡的clk我也是接與debounce\_circuit同的100hz。

* 這裡不能接display的input rst，因為那裡的rst是希望count\_down重新開始數。當rst=0時，freq\_100hz輸出的clk\_100持續為0🡪 one\_pulse不做事。



1. count\_down：

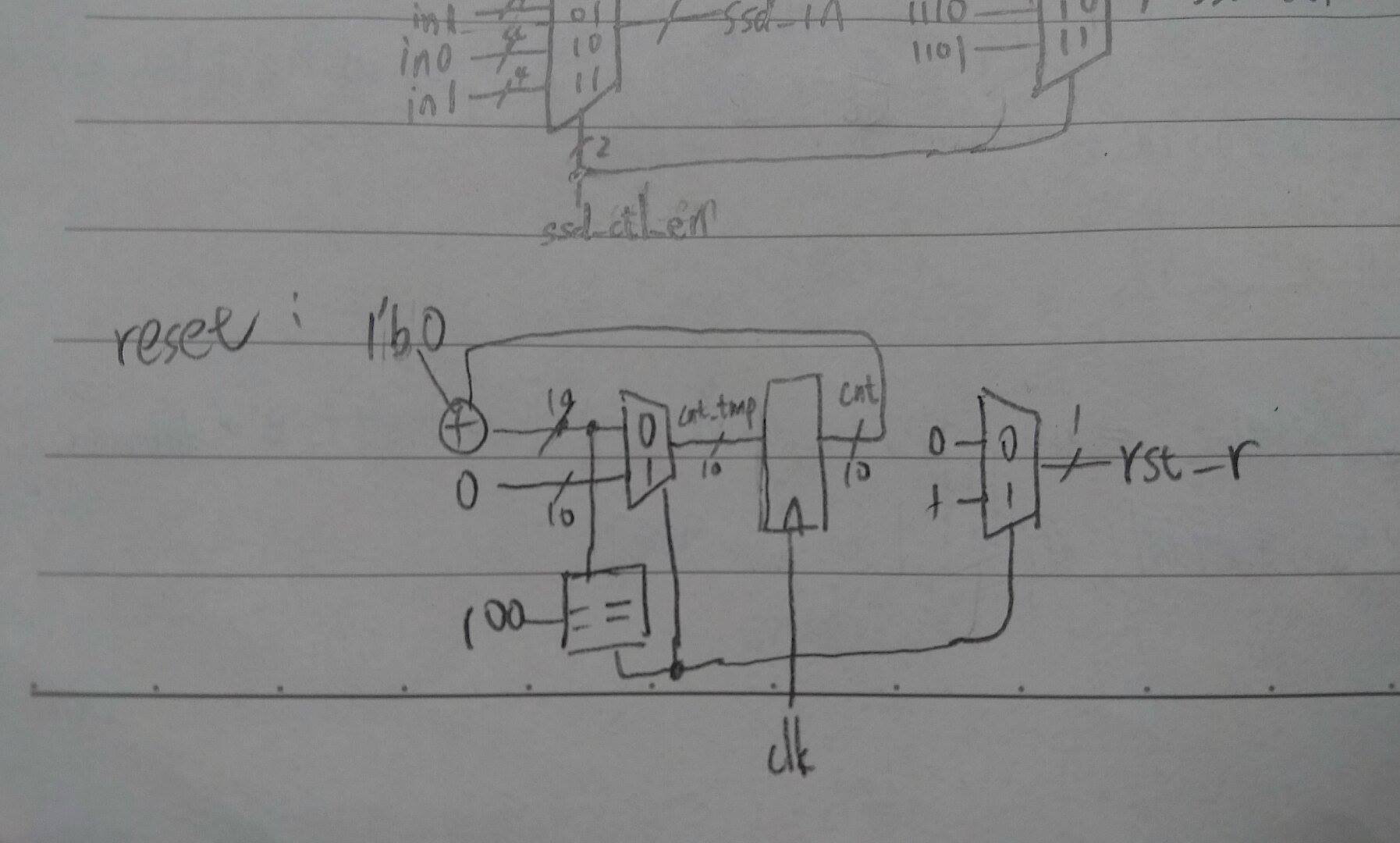
這是同時數兩個變數，個位數cnt\_one跟十位數cnt\_ten進行往下數。當cnt\_one=0時把cnt\_ten的數值減一以及cnt\_one的數值變成9；當cnt\_one跟cnt\_ten都為0時，讓兩個數都維持0。而初始化(rst=0)就是把數設為30(cnt\_ten=3 & cnt\_one=0)。

這裡還接一個en，作為是否能往下數的判斷值，若en=1，表示可以往下數；若en=0，則不能繼續往下數。

1. reset：

此module功能是為產生rst訊號。

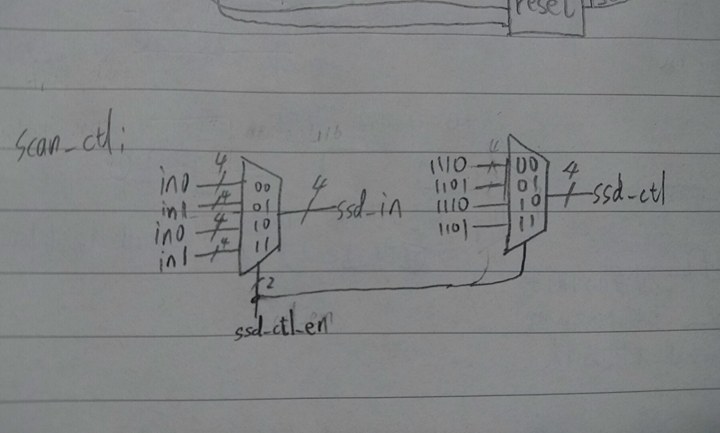
而此input訊號rst來自只經過debounce\_circuit處理過的輸入訊號in\_de，因為此週期可能為多個週期的clk\_100，因此可以拿來作為判斷長按的依據。

****

1. scan\_ctl：

此module是讓七段顯示器同時顯示不同位數的module。

ssd\_ctl\_en(from freq\_1hz的clk\_ctl[2:0])的用意為顯示兩個不同的數，因為七段顯示器一次只能顯示一種數次，因此需要一個頻率介於內建時間到1hz的ssd\_ctl\_en產生同時顯示兩個數的錯覺。



1. ssd：

當bcd 為 4'd0: segs = 8'b00000011; //七段顯示器顯示0

當bcd 為 4'd1: segs = 8'b10011111; //七段顯示器顯示1

當bcd 為 4'd2: segs = 8'b00100101; //七段顯示器顯示2

當bcd 為 4'd3: segs = 8'b00001101; //七段顯示器顯示3

當bcd 為 4'd4: segs = 8'b10011001; //七段顯示器顯示4

當bcd 為 4'd5: segs = 8'b01001001; //七段顯示器顯示5

當bcd 為 4'd6: segs = 8'b01000001; //七段顯示器顯示6

當bcd 為 4'd7: segs = 8'b00011111; //七段顯示器顯示7

當bcd 為 4'd8: segs = 8'b00000001; //七段顯示器顯示8

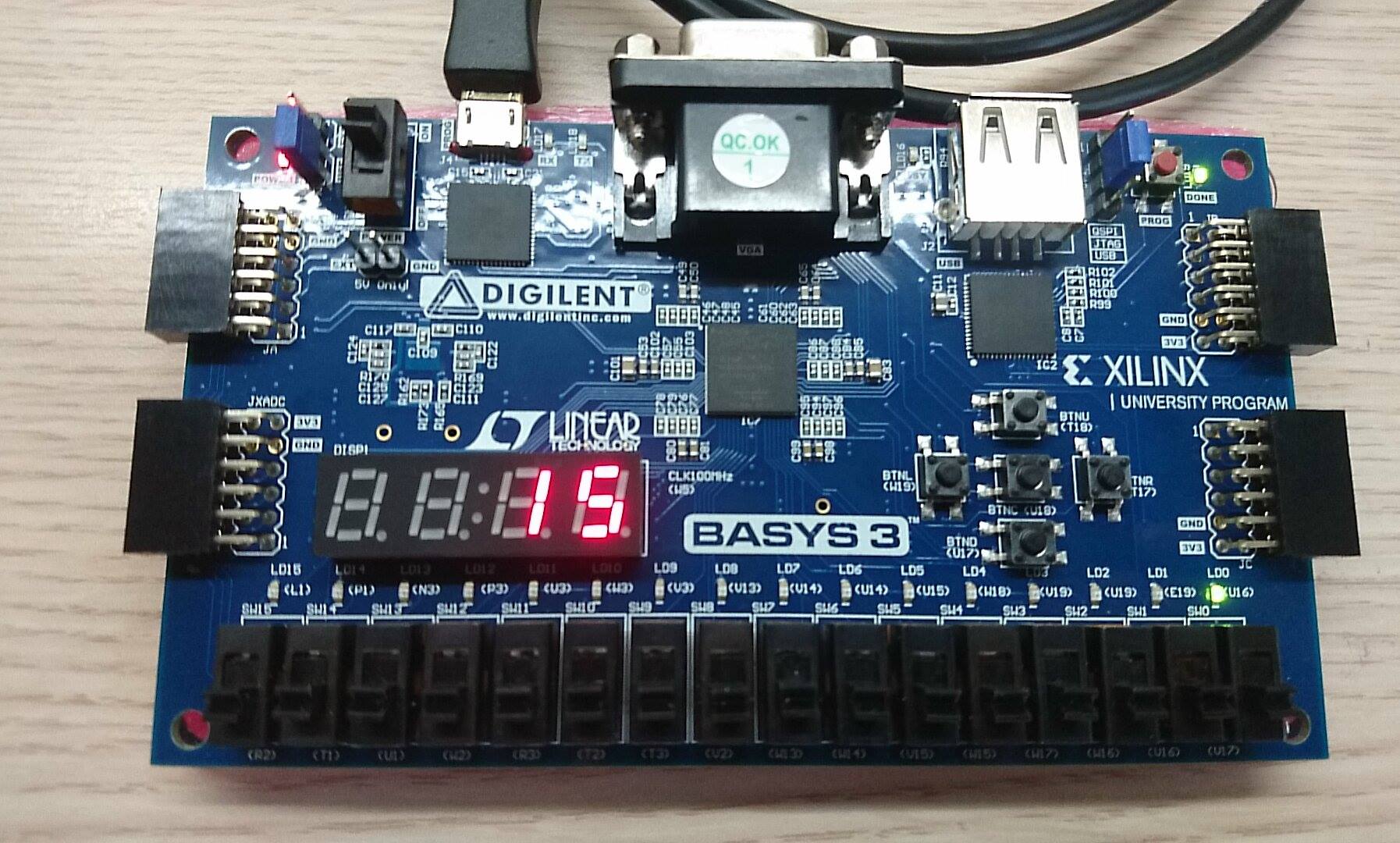
當bcd 為 4'd9: segs = 8'b00001001; //七段顯示器顯示9

default: segs = 8'b00000000; //七段顯示器顯示8.

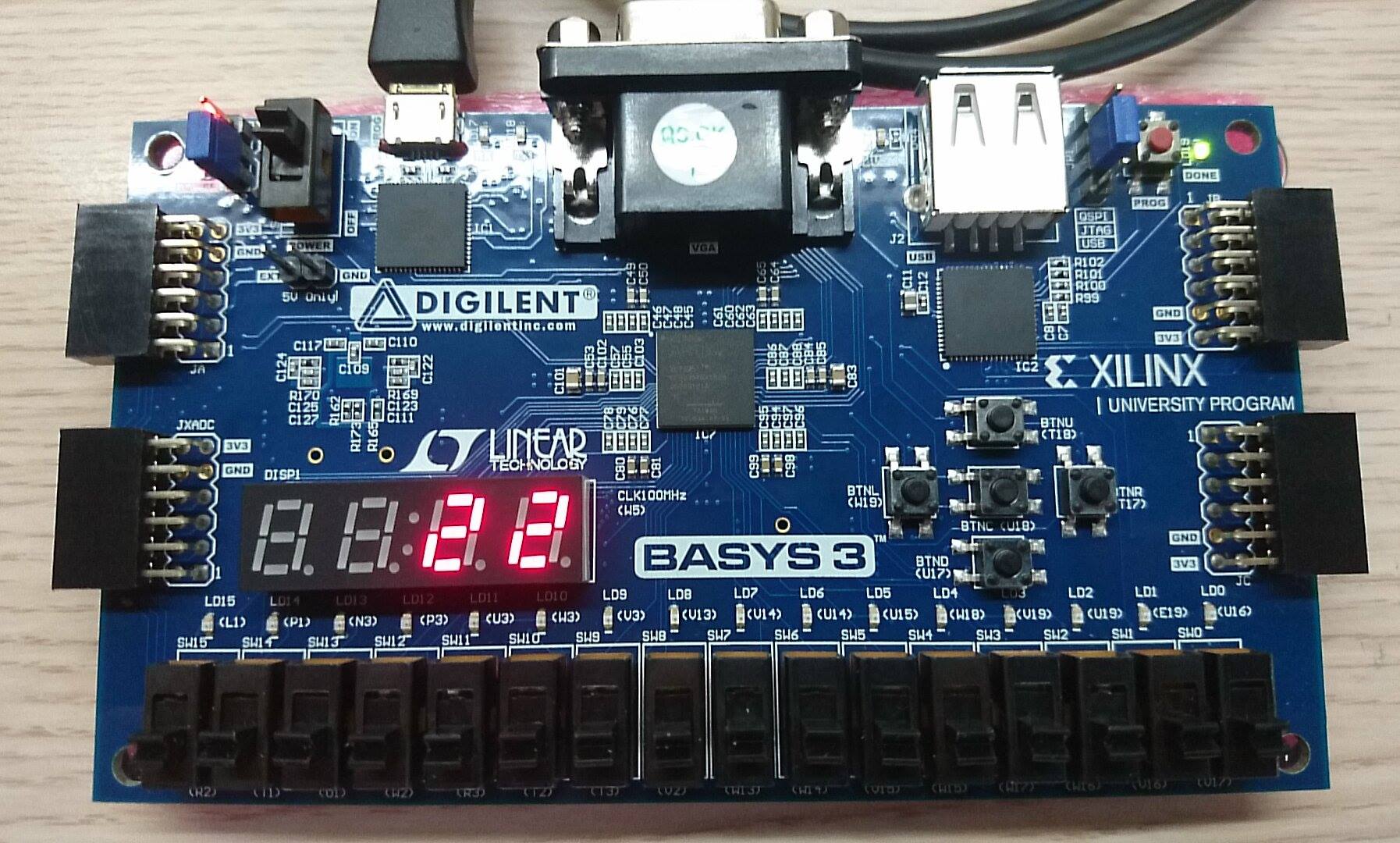
Result

1. 圖一：當U16亮，表示state=1🡪en=1🡪開始往下數。
2. 圖二：當U16暗，表示state=0🡪en=0🡪停止往下數。
3. 圖三：當長按(1秒)W19，表示rst=1🡪數字重新回至30，且停止往下數。

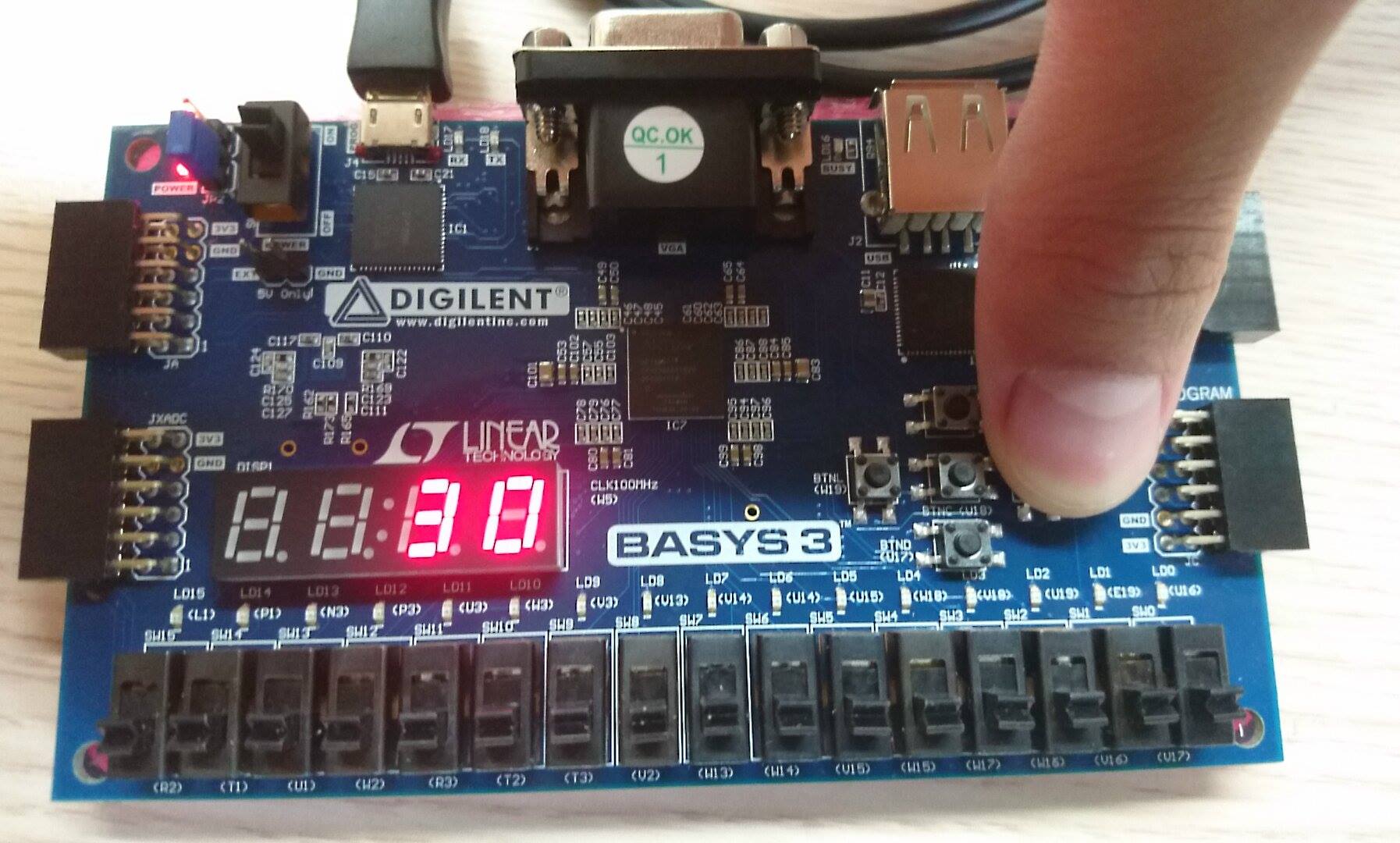
圖一



圖二



圖三



Discussion

1. 這個實驗最重要的要注意rst接的module，因為這裡的rst是希望count\_down重新開始數。
2. count\_down判斷rst要在rst從0🡪1時判斷，因為rst是一個按鈕，按下的過程是0🡪1，且要在rst==1時做出重置，這是跟以往實驗較不同之處。
3. 處理按鈕雜訊要使用debounce\_circuit，原理在上述已有做說明了。
4. 若要使多個週期的訊號簡化成週期為1個clk則要使用one\_pulse去做處理。
5. 此多用reset module產生rst訊號，是用加法器的概念去達成長按能產生rst效果。

**Conclusion**：

這次的lab更深入帶我們了解到如何使用少許的按鈕達成多種功能，以及清楚了解使用rst的時機，雖然過程中挫折不斷，但是打完真的是成就感滿滿，真的很開心。