**Lab 07**

105060016 謝承儒

1. **Lab07-1 : 有凍結功能的分秒計時器**

(凍結:表面上不動，實際上內部仍然還在計時。)

1. **Design Specification :**
2. Input : clk //輸入的頻率(100MHz)

rst //當=1時，使數字回到00並暫停

pb\_s //暫停/開始按鈕

pb\_f //凍結/重製按鈕

1. Output : [7:0] display //七段顯示器的顯示碼

[3:0] display\_c //控制4個顯示器中哪個會改變

1. Wire : clk\_1 //除頻後的頻率(1Hz)

clk\_100 //除頻後的頻率(100Hz)

[1:0] ssd\_ctl //從除頻器輸出，解碼後為display\_c

pb\_s/f\_de //經過Debounce後的pb\_s/f

one\_pulse\_s/f //經過One\_pulse處理後，產生為期1週期的

freeze\_en、rst\_f、count\_en

//從FSM輸出，分別代表是否凍結/重置/計時

co\_s1/ co\_min / co\_m1

//代表進位，分別是秒十位數、分個位數、分十位數

[3:0] sec\_s0、sec\_s1、min\_m0、min\_m1

//秒和分的個位數、十位數

[3:0] d0、d1、d2、d3

//經過是否凍結的區塊後，準備顯示在顯示器的值

[7:0] display0、display1、display2、display3

//d經過解碼後得到的顯示碼

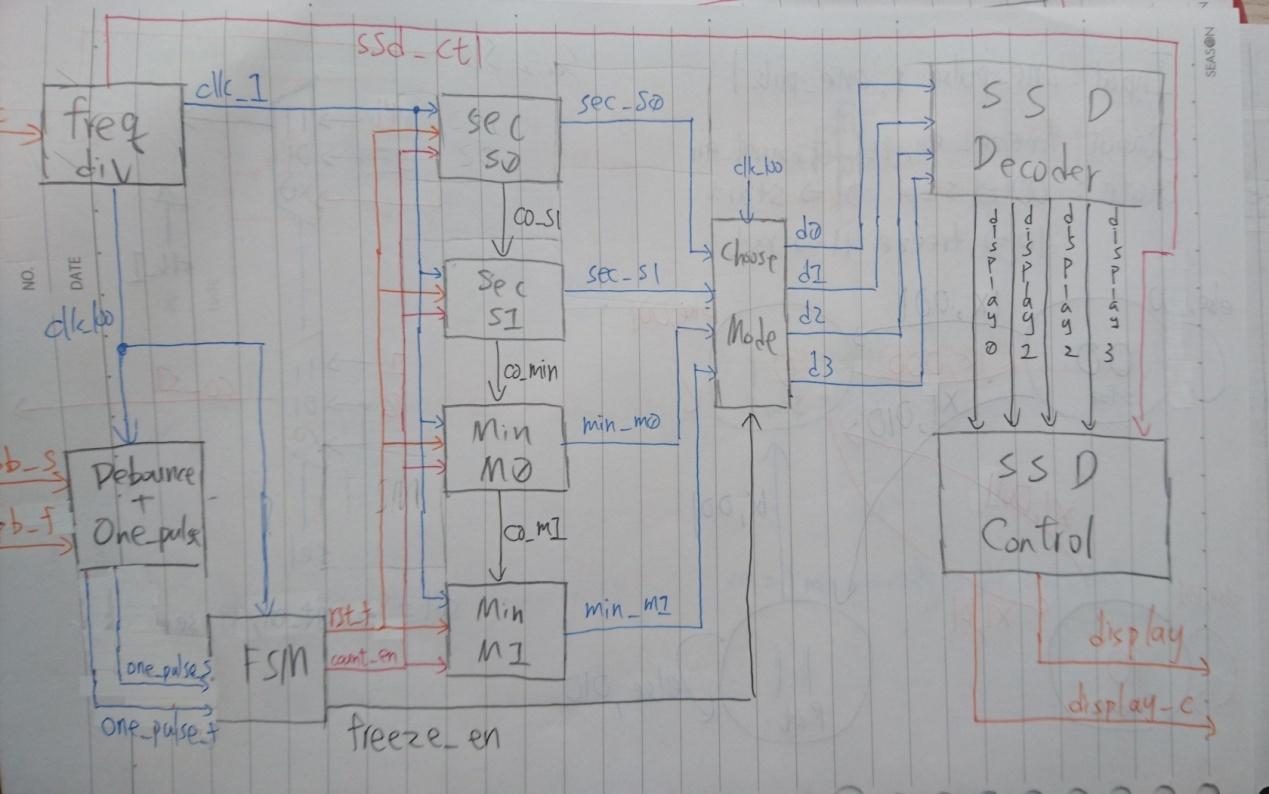


圖1 Lab07-1的區塊圖

1. **Design Implementation :**
2. Logic diagram :

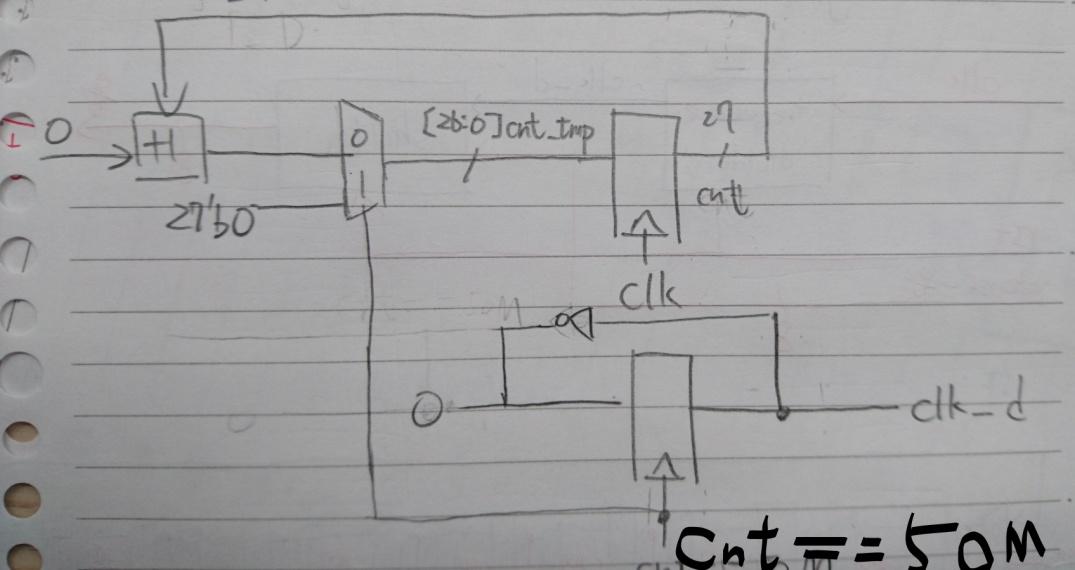


圖2 Lab07-1的除頻器(100MHz->1Hz)

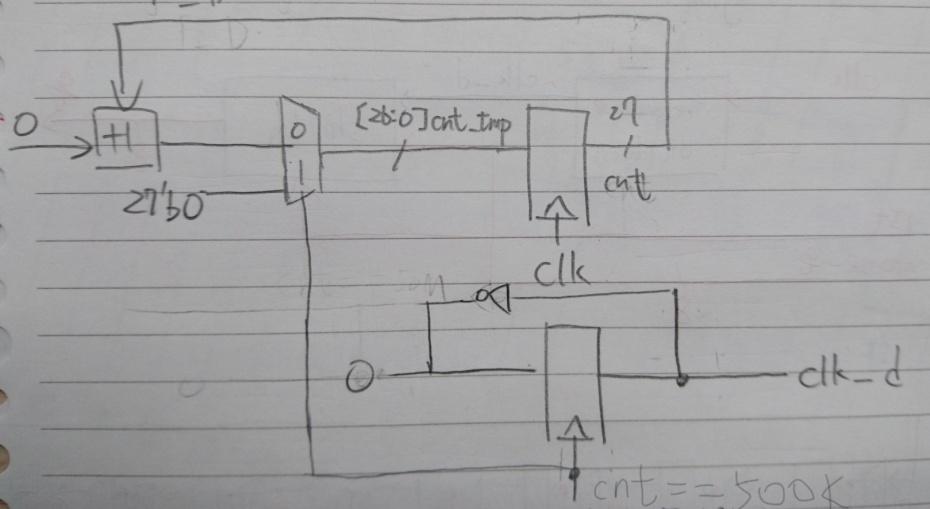


圖3 Lab07-1的除頻器(100MHz->100Hz)

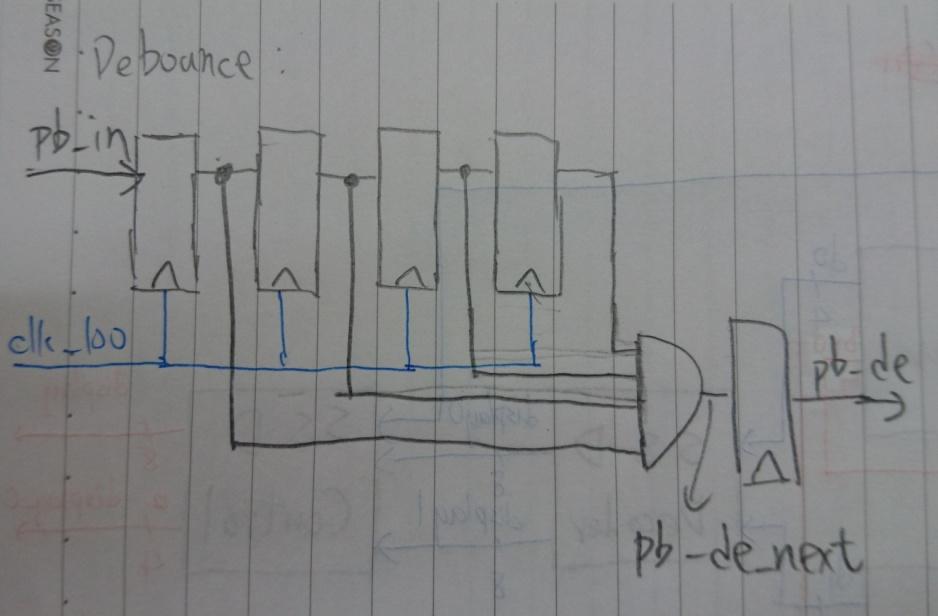


圖4 Lab07-1的Debounce

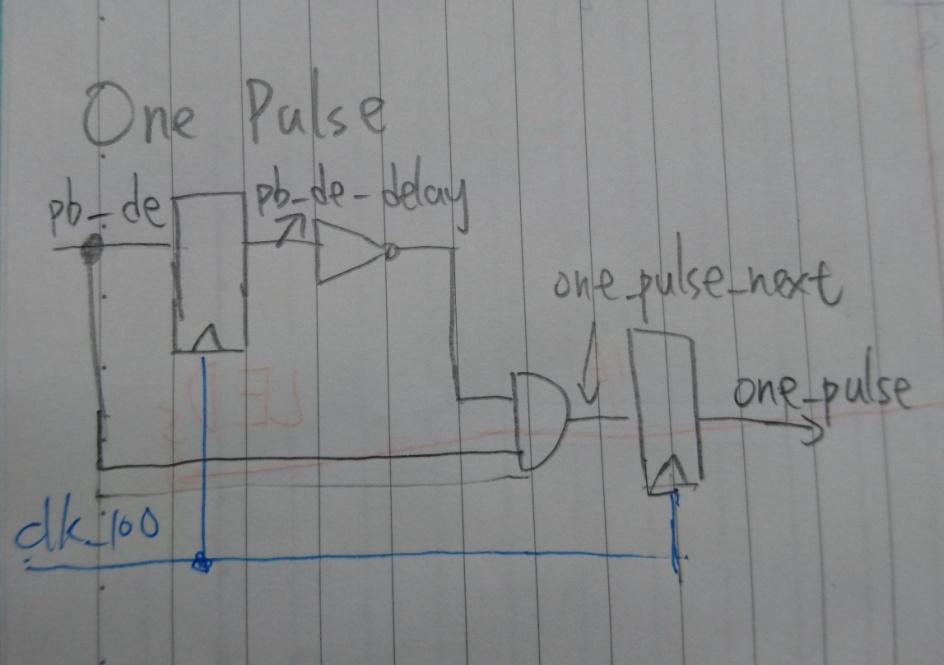


圖5 Lab07-1的One pulse

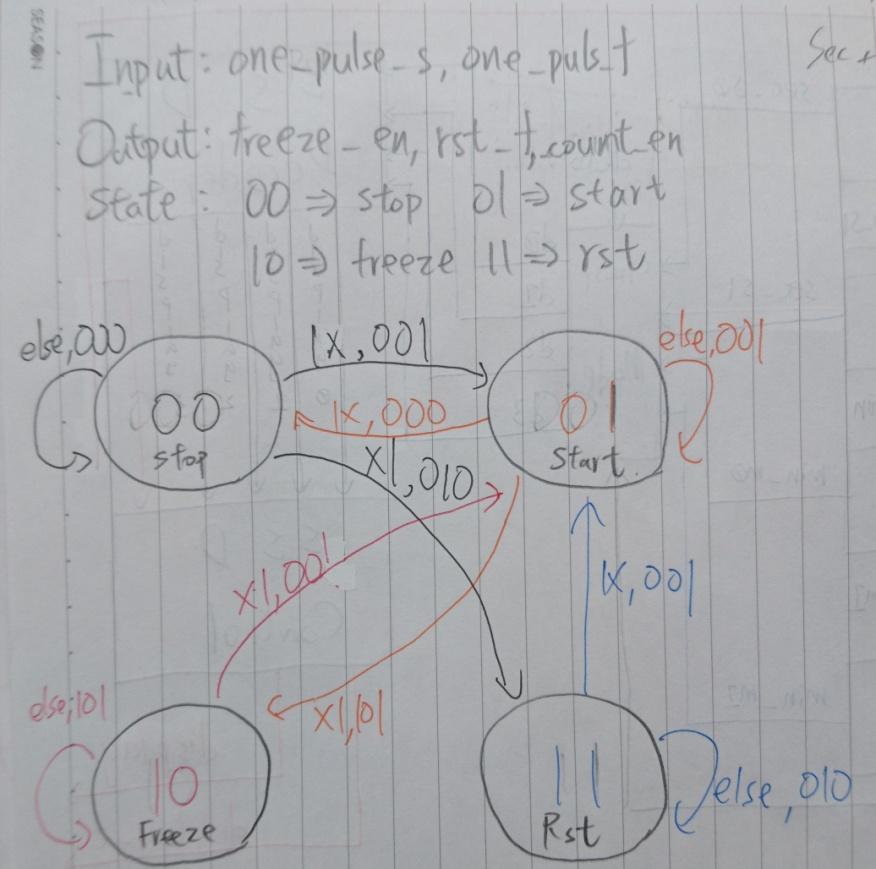


圖6 Lab07-1的FSM

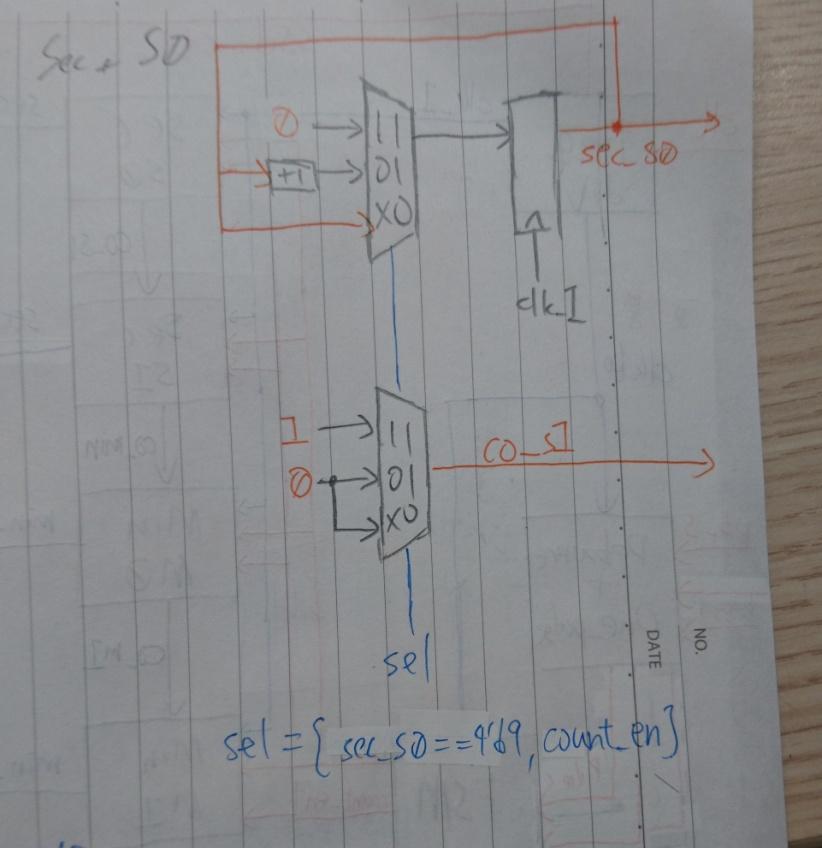


圖7 Lab07-1的sec\_s0的up counter(個位數)

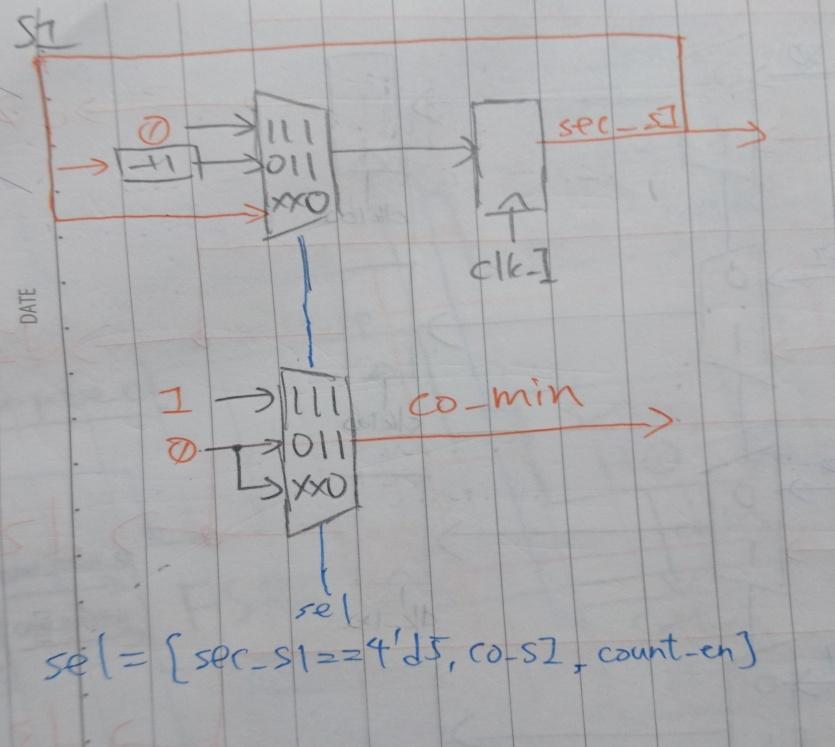


圖8 Lab07-1的sec\_s1的 up counter(十位數)

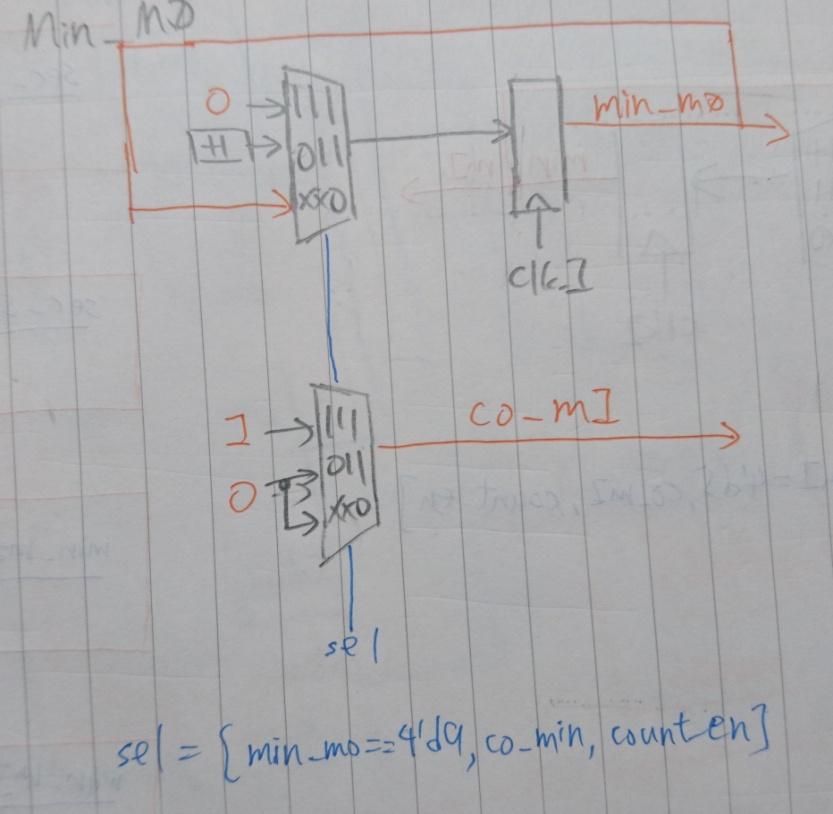


圖9 Lab07-1的min\_m0的 up counter(個位數)

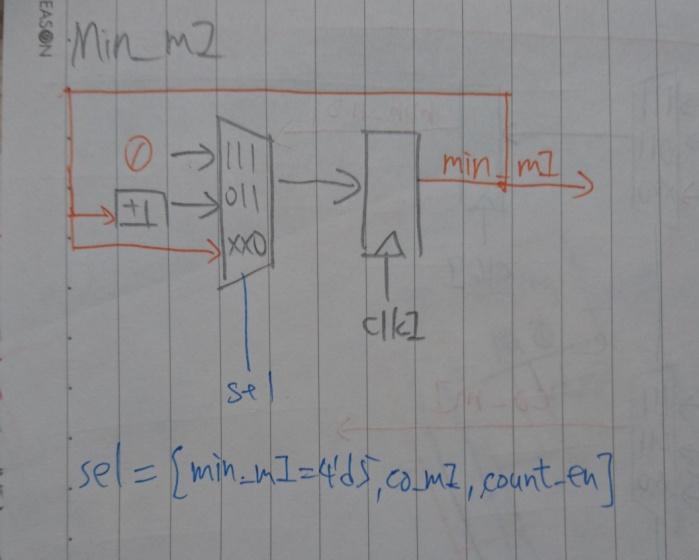


圖10 Lab07-1的min\_m0的 up counter(十位數)

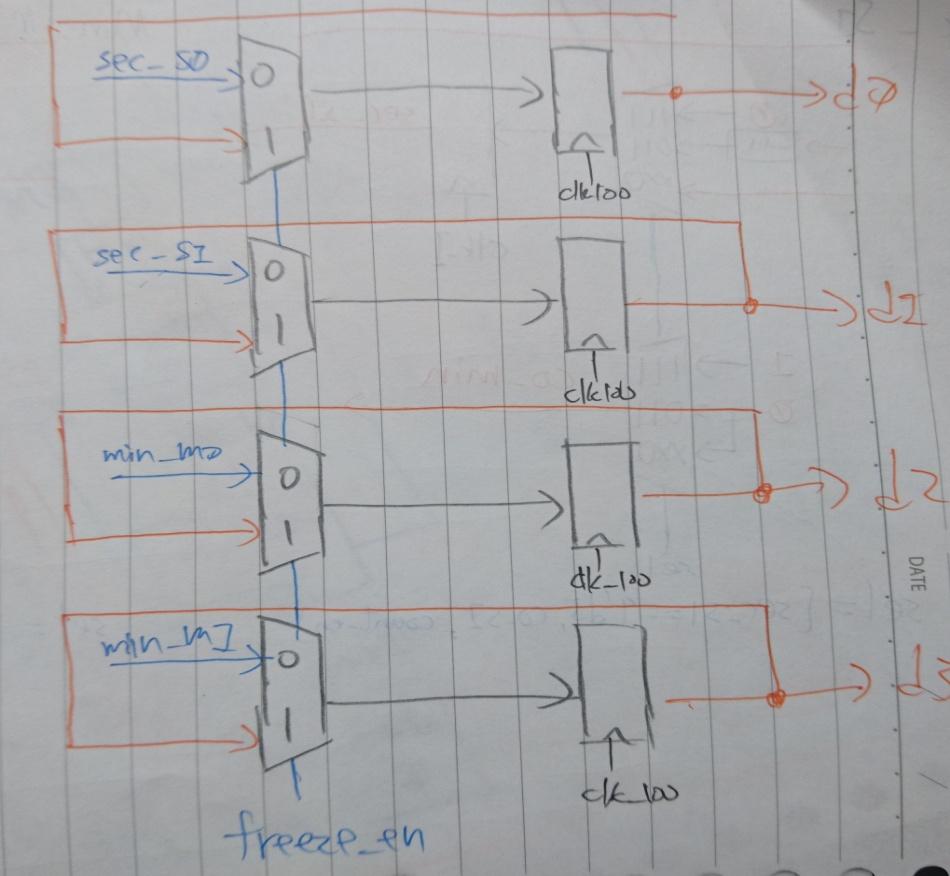


圖11 Lab07-1的choose\_mode

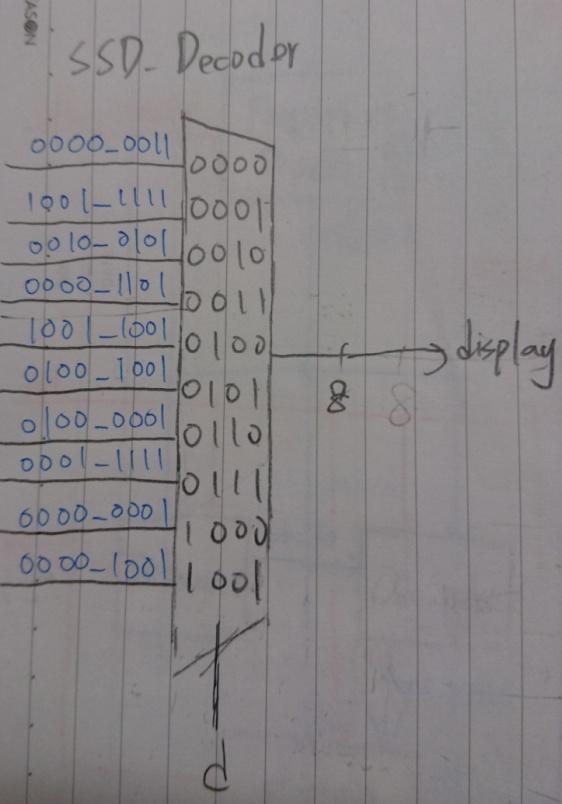


圖12 Lab07-1的SSD Decoder

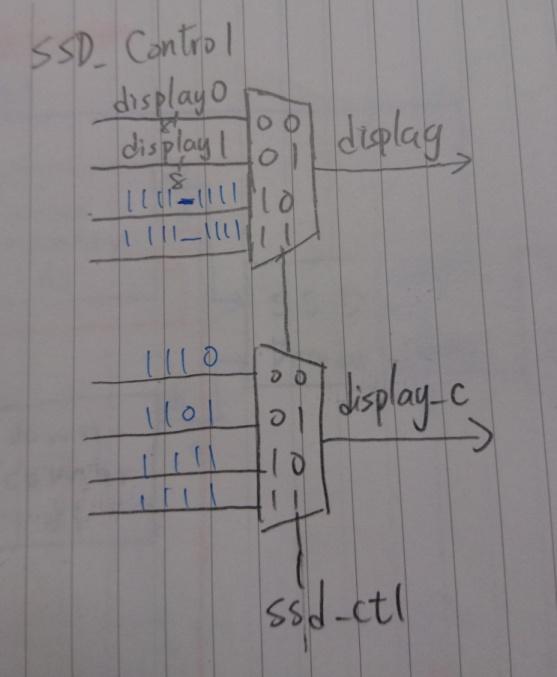


圖13 Lab07-1的SSD Control

1. Pin assignment :
2. Input :
3. clk = W5
4. rst = R2
5. pb\_f = W19
6. pb\_s = T17
7. Output :
8. display[0] = V7
9. display[1] = U7
10. display[2] = V5
11. display[3] = U5
12. display[4] = V8
13. display[5] = U8
14. display[6] = W6
15. display[7] = W7
16. display\_c[0] = U2
17. display\_c[1] = U4
18. display\_c[2] = V4
19. display\_c[3] = W4
20. **Discussion :**
21. 整體運作過程 :
    * + - 1. 將clk輸入進除頻器(freq\_div)，得到1Hz和100Hz的新clk。1Hz的用來驅動counter；100Hz的用來驅動Debonce、One\_pulse、FSM。

此外，輸出ssd\_ctl來做為待會SSD\_Control的其中一個Input。

* + - * 1. 將經過處理後的one\_pulse\_s、one\_pulse\_f輸入到FSM，得到freeze\_en、rst\_f、count\_en，用來在後面的counter中作為條件來達到不同狀態。
        2. 將4個counter的值(sec\_s0/1、min\_m0/1)輸入到choose\_mode，並利用freeze\_en來判斷是否處於凍結狀態，然後輸出要顯示出來的值(d0~3)。
        3. 最後，把d0~3解碼成顯示碼(display0~3)，再加上ssd\_ctl來控制顯示板(display\_c)和將每個值逐一輸出(display)。

1. 各block的構想 :
2. **除頻器**:用來製造1Hz、100Hz

跟之前的Lab相同，用輸入的clk(clk\_c)來驅動27-bit的Counter(其值為[26:0]cnt)。在cnt=50M時，讓clk\_d = ~clk\_d且cnt歸零，達到1Hz的效果。

而將條件改成cnt=500K，便可以製造出100Hz。

除此之外，將其中的兩位輸出(ssd\_ctl)，此次選擇cnt[15:14]。

1. **Debounce** : 消除不穩定的震盪

利用一個4-bit的Shfiter，每次clk都將現在按鈕的狀態輸入(pb\_s/f)，若4個都為1，則輸出pb\_s/f\_de=1。因為是由電腦模擬出來的，不會有不穩定的震盪。

1. **One Pulse** : 製造只會為期1週期的1=>只會觸發一次效果

將剛得到的pb\_s/f\_de輸入進Flip Flop(FF)，只有當clk posedge時，FF才會打開讓值通過，因此FF後的值(pb\_de\_delay)會和進來前的(pb\_de)有1clk的誤差。

將~(pb\_de\_delay)和pb\_de做and，便可以製造出只會為期1週期的1。

1. **FSM**:用來決定暫停、計時、凍結、重置4種狀態

因為有4種狀態，所以state是2-bit，00為暫停、01為計時、10為凍結、11為重置。

狀態的變換是經由兩個按鈕來控制；其中一個(pb\_s)控制暫停還是計時；另一個則是凍結/重置(pb\_f)，按下時，若為暫停狀態則會重置，若為計時狀態則會凍結。

1. **sec/min up counter**

基本上只是一個簡單的BCD up counter，只是需要注意秒和分都是六十進位制，當為59時，下一個則是00。

1. **choose\_mode:** 是否處於凍結狀態

把剛剛在FSM得到的freeze\_en作為選擇依據。

當freeze\_en=0時，就代表為一般的計時，直接將輸入的各值(sec\_s0/1、min\_m0/1)輸出就好。

當freeze\_en=1時，就代表是處於凍結狀態，把上個瞬間輸出的值(d1~3)再次輸出。

1. **SSD Decoder :** 將數字轉換成顯示碼

將Choose\_mode得到的值(D0、D1)輸入，作為多工器的依據(在code裡可用case得到相同效果)，解碼成8-bit2的顯示碼(display0、display1)，將它們輸入到下個SSD Control。

1. **SSD Control :** 決定哪塊板子的值改變

把在除頻器得到的2-bit的ssd\_ctl輸入，作為選擇SSD四個顯示器的依據。

當ssd\_ctl\_en=00時，便將display0輸出(display)，並將display\_c輸出為1110。如此顯示器上便只有最後一位可以改變，保留其他三個的字母，其他ssd\_ctl的情況也是一樣。

1. 過程中的Bug :

一開始，在freeze的狀況下，再按一次凍結鍵應回到計時狀態，但反而會先重置後，再開始計時。把兩個按鍵的Debounce和One\_pulse分開做就正常了。

1. **Lab07-2 : 可設定時間的倒數器**
   1. **Design Specification :**
      1. Input : clk //輸入的頻率(100MHz) 。

rst //當=1時，重置時間

setting //是否在設定狀態

pb\_p\_hr

//設定時為加小時按鈕，倒數時為暫停/倒數

pb\_s\_min

//設定時為加小時按鈕，倒數時為開始倒數

* + 1. Output : [7:0] display //七段顯示器的顯示碼。

[3:0] display\_c //決定哪個顯示器改變。

[15:0]LEDs //16個LED燈

* + 1. Wire : clk\_d //除頻後頻率(40Hz)

clk\_f //除頻後頻率(100Hz)

[1:0] ssd\_ctl //從除頻器輸出，解碼後為display\_c。

pb\_s/p\_de、one\_pulse\_s/p

//經過處理後的按鍵，給FSM相關的block，100Hz

pb\_cs/cp、one\_pulse\_cs/cp

//經過處理後的按鍵，給setting和counter，40Hz

rst\_f、de\_en、set\_en、light

//從FSM輸出、代表是否歸零、倒數、設定、發亮

[3:0] set\_m0,set\_m1,set\_h0,set\_h1

//代表設定的值

co\_m1、co\_hr、co\_h1

//代表是否進位到下一個時間單位

[3:0] min\_m0/1、hr\_h0/1

//分、時的個、十位數值

br\_m1、br\_h0、br\_h1

//代表借位

[7:0] display0、display1、display1、display1

//各個位數的顯示碼。

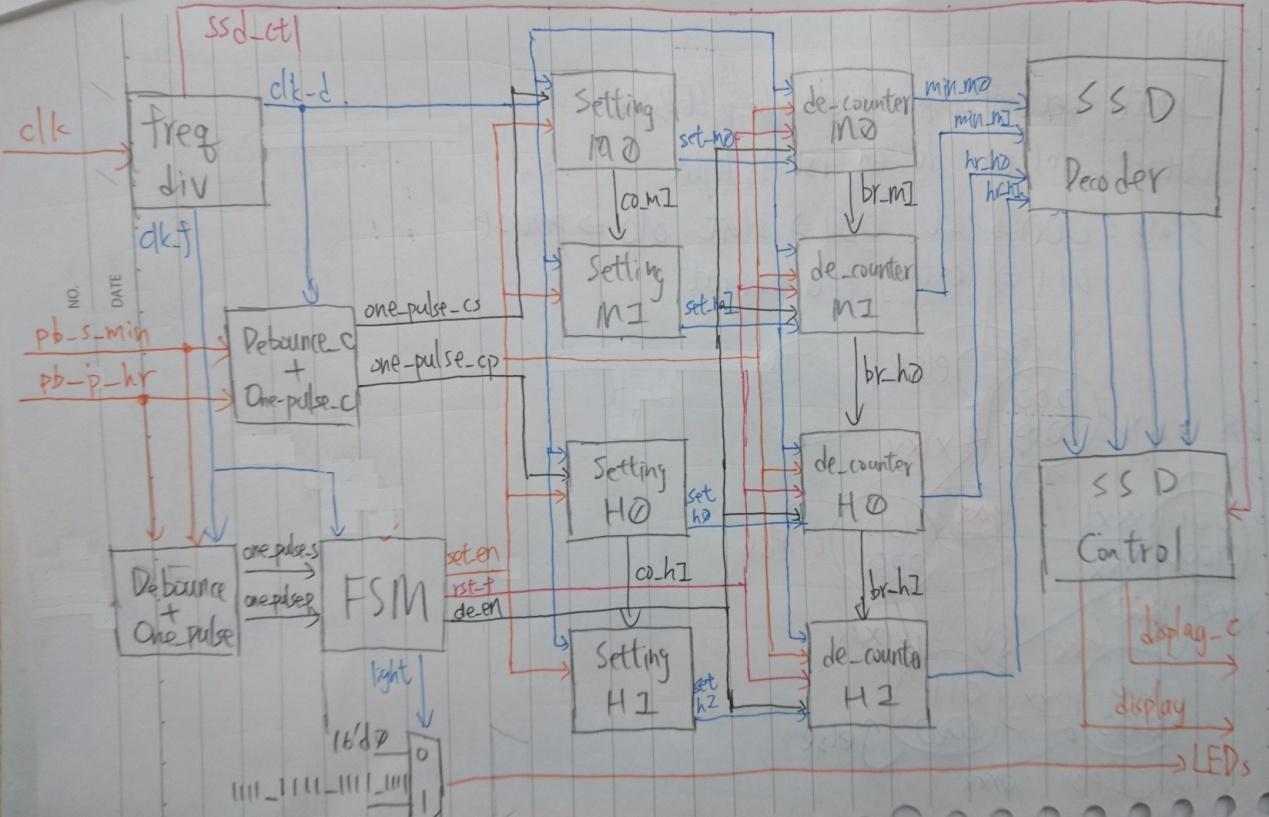


圖14 Lab07-2的區塊圖

* 1. **Design Implementation :**
     1. Logic function :

LEDs\_en = min\_m0==4'd0 && min\_m1==4'd0 && hr\_h0==4'd0 && hr\_h1==4'd0

* + 1. Logic diagram :

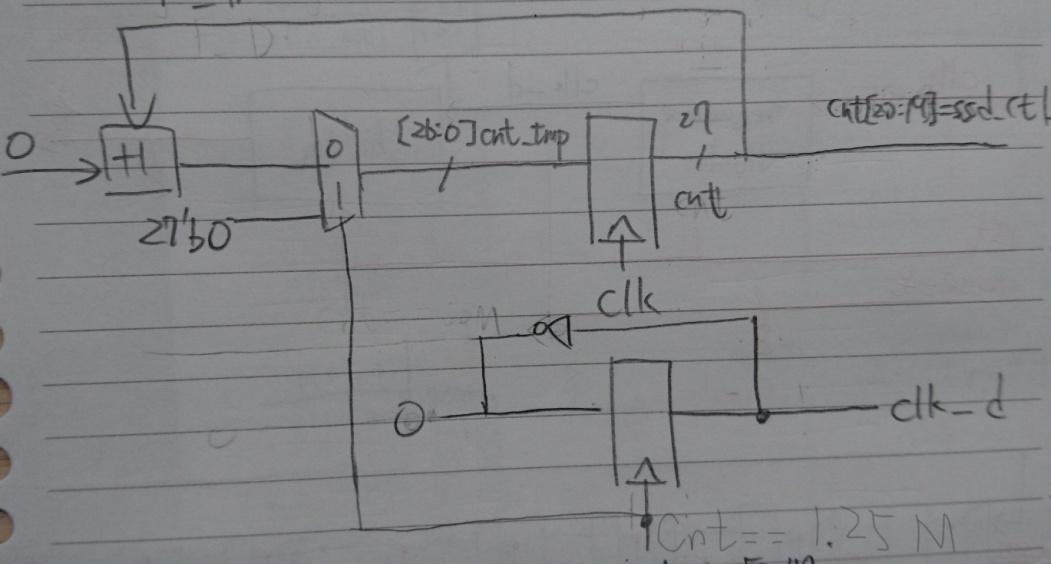


圖15 Lab07-2的除頻器(100MHz->40Hz)

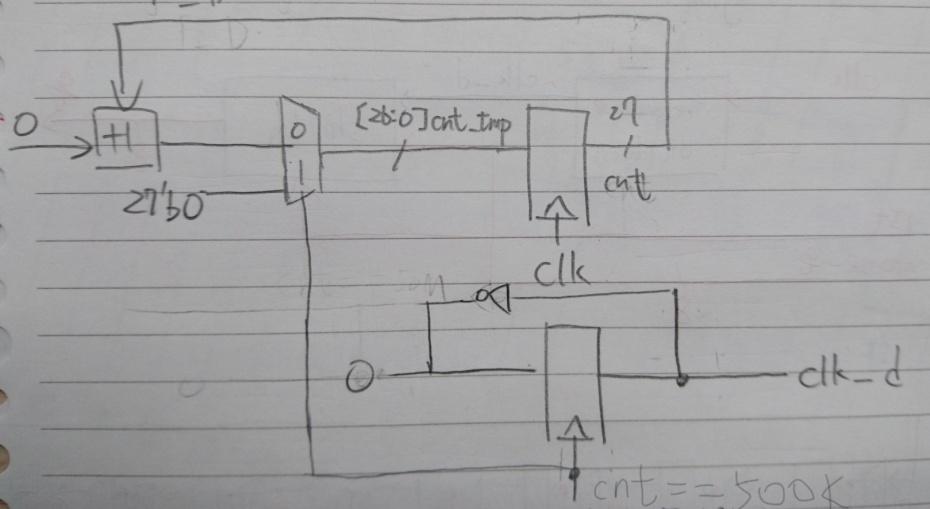


圖16 Lab07-2的除頻器(100MHz->100Hz)

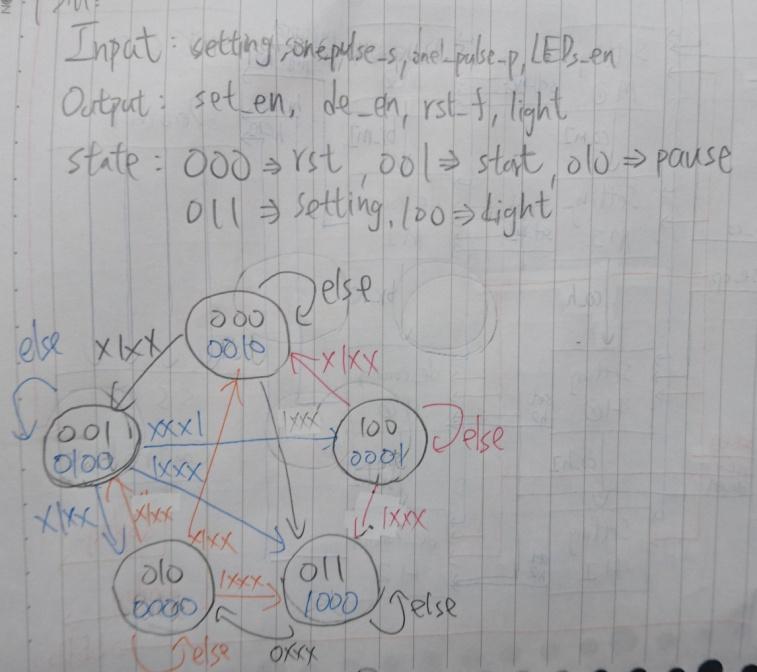


圖17 Lab07-2的FSM

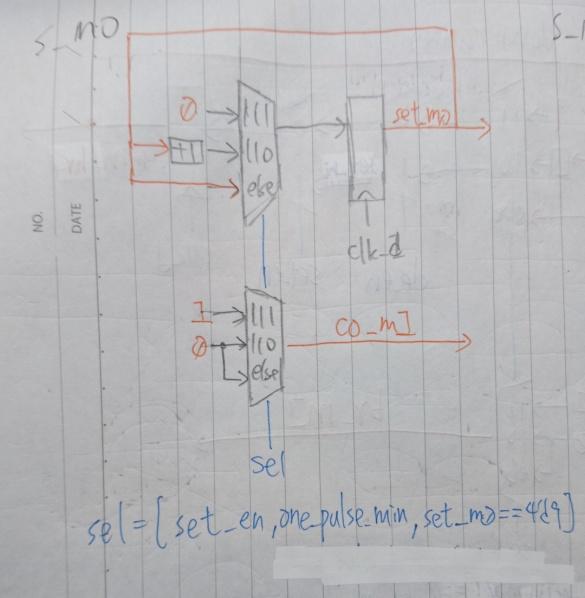


圖18 Lab07-2的 set\_m0

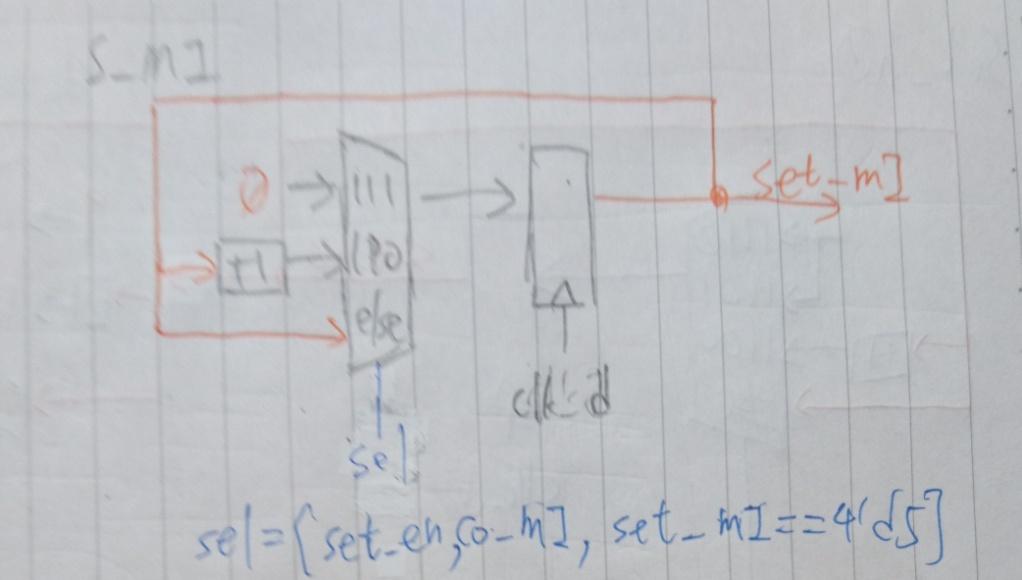


圖19 Lab07-2的 set\_m1

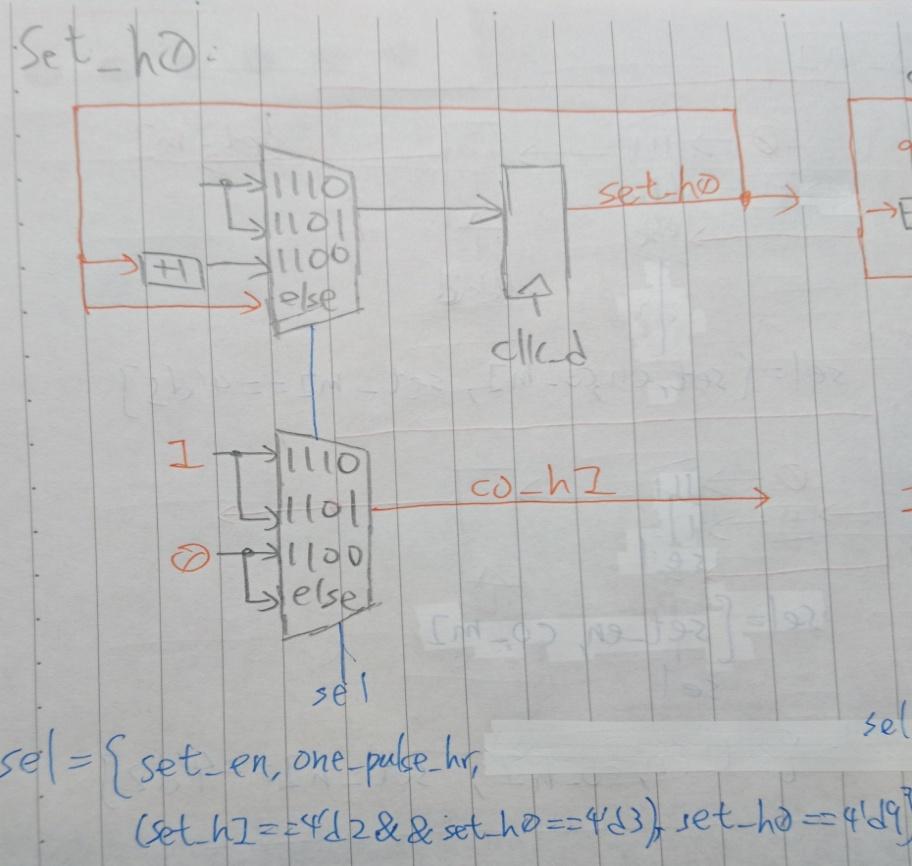


圖20 Lab07-2的 set\_h0

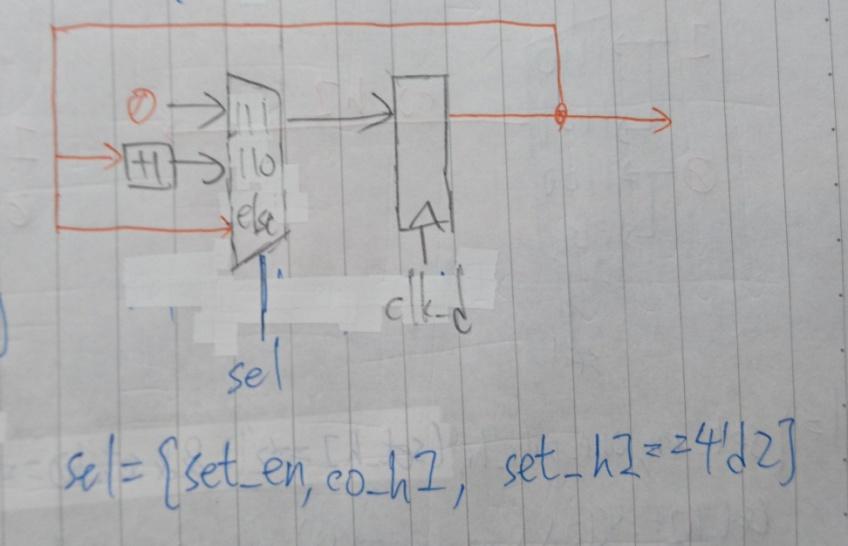


圖21 Lab07-2的 set\_h1

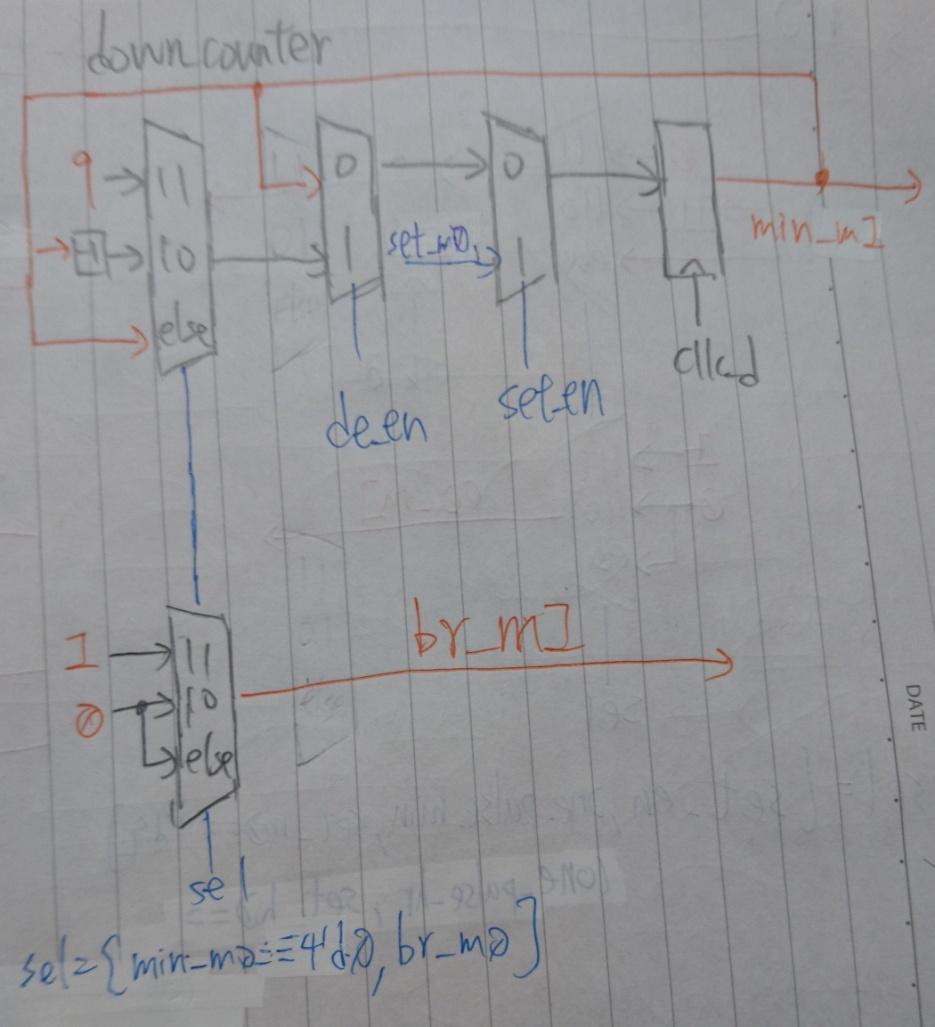


圖22 Lab07-2的 down counter

(各位數的結構差異不大)

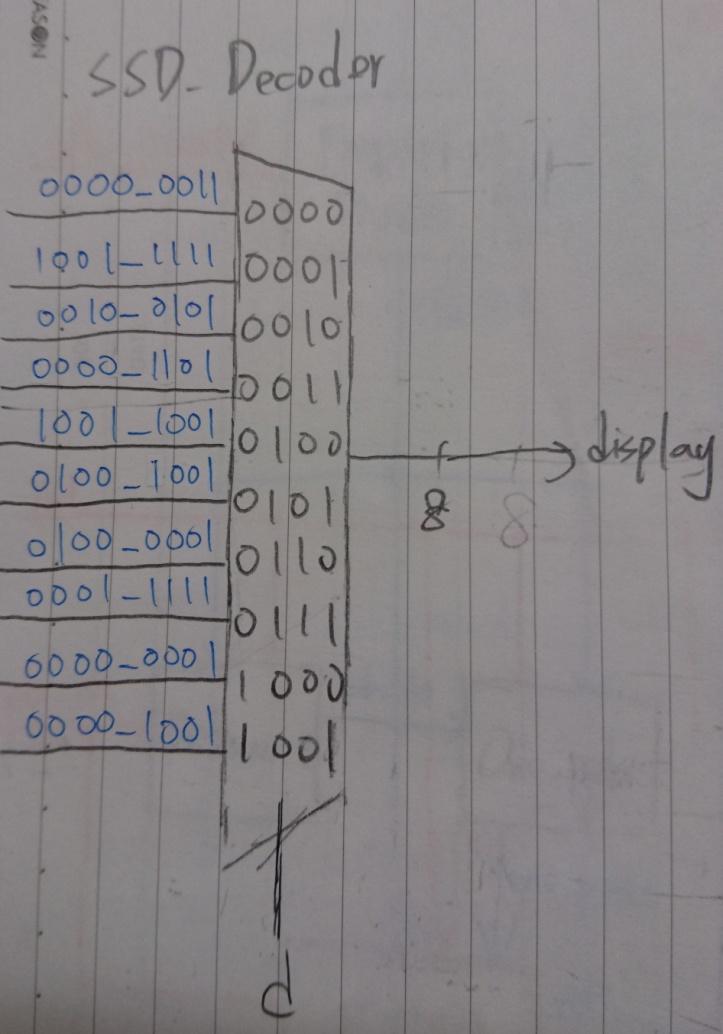


圖23 Lab07-2的SSD Decoder

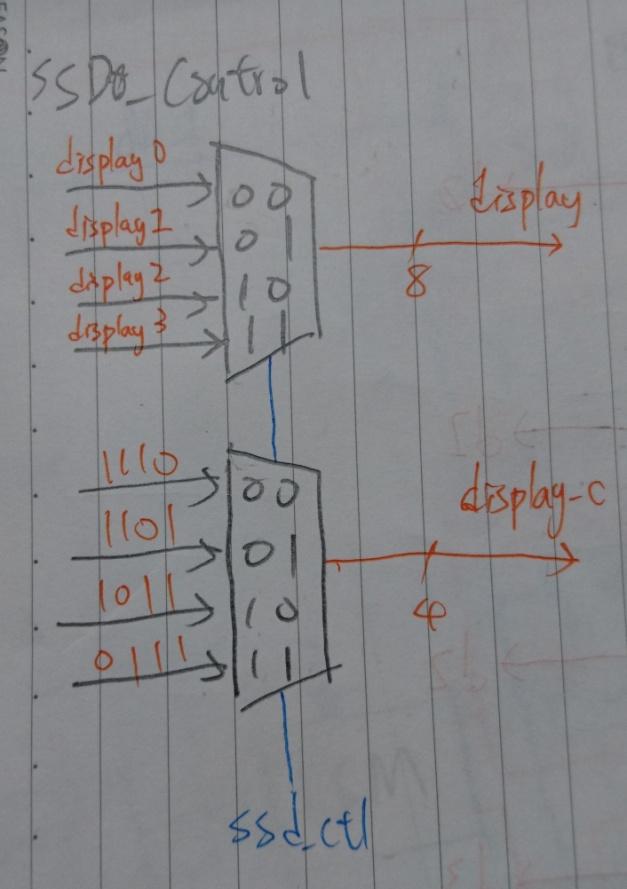


圖24 Lab07-2的SSD Control

* + 1. Pin assignment :
       - 1. Input :

clk : W5

rst : V17

setting : R2

pb\_p\_hr: W19

pb\_s\_min : T17

* + - * 1. Output :

display[0] = V7

display[1] = U7

display[2] = V5

display[3] = U5

display[4] = V8

display[5] = U8

display[6] = W6

display[7] = W7

display\_c[0] = U2

display\_c[1] = U4

display\_c[2] = V4

display\_c[3] = W4

[15:0] LED = L1~U16(全部的LED燈的Pin腳)

1. **Discussion :**
   * 1. 整體運作過程 :
        + 1. 將clk輸入進除頻器(freq\_div)，得到40Hz(clk\_d)和100Hz(clk\_f)的新clk。40Hz的用來驅動down counter和setting相關的block(debounce\_c、ome\_pulse\_c)；100Hz的用來驅動和FSM相關的block(debounce、ome\_pulse)。

此外，輸出ssd\_ctl來做為待會SSD\_Control的其中一個Input。

* + - * 1. 把clk\_f、和經過處理的按鍵(one\_pulse\_s/p)輸入到FSM來決定各種狀態，並輸出set\_en、de\_en、rst\_f、light來作為後面block的條件。
        2. 把clk\_d、和經過處理的按鍵(one\_pulse\_cs/cp)輸入到各個setting counter block，得到各個set的值(set\_m0、m1、h0、h1)。
        3. 把set\_m0、m1、h0、h1分別輸入到其所屬的down counter，得到各位數的值(min\_m0/1、hr\_h0/1)。
        4. 將min\_m0/1、hr\_h0/1輸入到SSD Decoder解碼得到顯示碼(display0~3)。
        5. 最後，將display0~3輸入到SSD Control，再加上ssd\_ctl來控制顯示板(display\_c)和把每個值逐一輸出(display)。
    1. 各block的構想 :

部分block可以沿用上個實驗Lab07-1的結構，而下面是說的是新block。

* + - * 1. **Debounce\_c+One\_pulse\_c :** 用於設定有關的block(Setting)

和原本的不同，這裡使用的clk是clk\_d，而不是clk\_f。因為輸出出來的one\_pulse\_cs/cp是要用在Setting blocks上，加上Setting是和 Down counter連動的，所以才採用相同的clk來避免bug。

* + - * 1. **Setting :** 設定時/分的值

這些block和一般的BCD up counter沒有太大的差別，主要是+1的機制不是由clk來控制，而是改為按鈕來控制。此外，當從59分到00分時，並沒有進位，其他的就沒什改變。

* + - * 1. **修改FSM :**

這次我將state分成5種狀態，000是歸零、001是開始到數、010是暫停、011是設定、100是數完然後發光。

利用setting,one\_pulse\_s/p,LEDs\_en來讓state互相轉換。

* + - * 1. **Down counter :** 倒數計時器

其實這和up counter相同，只是把+1改成-1和進位轉成退位就可以達成了。

* + 1. 過程中的Bug :

起初在設定值時，只要一按按鍵便會直接加到超過9，讓顯示器上顯示F，儘管之後把按鍵One\_pulse化也一樣。

最後，按照同學說得把setting和down counter分開來寫bug就消失了。

1. **Conclusion :**

上次才說有上手的感覺，結果這次就被打臉了。這次花了許多時間在消除bug，甚至為此改變整體結構才成功完成實驗。

1. **Reference :**
   1. 老師給的實驗講義

讓我知道大概的架構是時麼。