Lab7 Report

7-1.Stop Watch with lap

●Design Specification

Input:

clk,(接上板子的原本的震盪頻率 W5)

rst_n, (控制整個功能的開關)

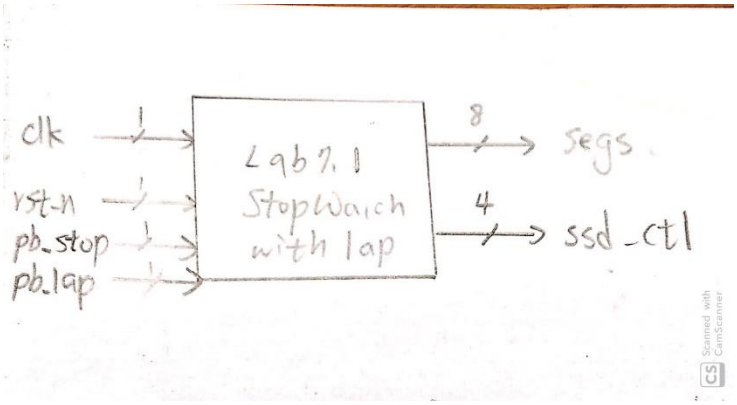
pb_stop,(這個功能是為了暫停功能，可以開始和暫停計時)

pb_lap,(這個短按是 lap 的功能，長按則是 reset 的功能)

Output:

[7:0]segs,(傳給七段顯示器的 output)

[3:0] ssd_ctl,(控制每一個七段顯示器的亮與暗)



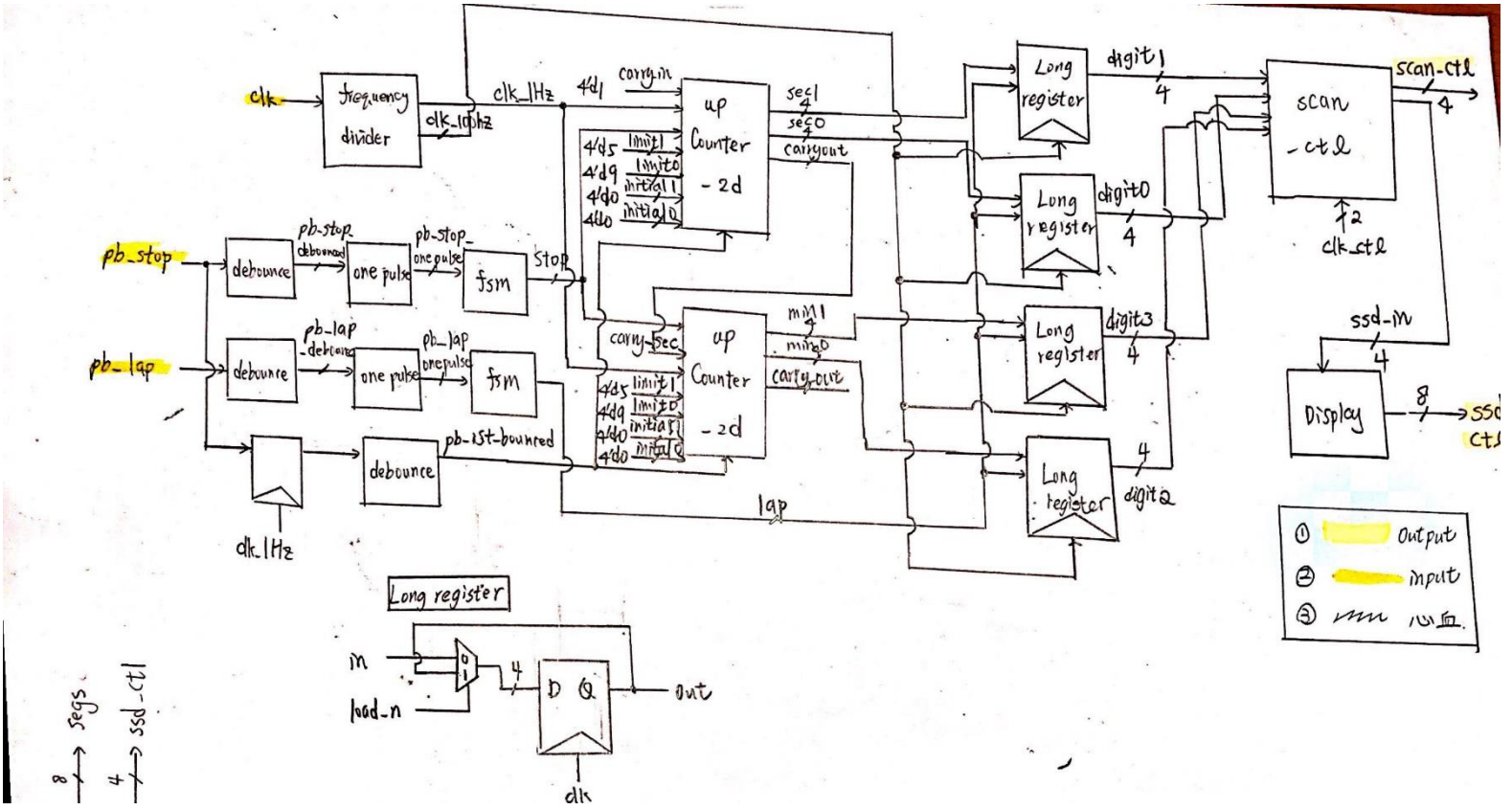
●Design Implementation(1.1)

1.Outline

這個實驗主要是做一個可以顯示暫停秒數的上數 counter，但是 counter 還要繼續數，也就是碼表的功能，如果繼續延伸功能應該可以做出可以暫停好幾個時間的功能，蠻像是那種在紀錄多人短跑成績體育課老師都有的那種碼表。

大部分使用到的 module 都是之前有的，只有多寫了 load register 而已，功能是要讓四個 D-flip flop 分別記住 lap 的數字；另外還有關於長按短按的功能，我是利用把 debounce、one pulse 的 clk_in 都設成 100Hz，讓按鈕只要輕輕按一下功能就會產生了，至於長按的部分，我是利用在接上一個 clk 是 1Hz 的 D-flipflop，這樣可以規定如果要長按的效果就一定要按超過一秒以上，這樣比較好分別出長按短按的差別，詳細的功能介紹會補充在 logic diagram 下方。

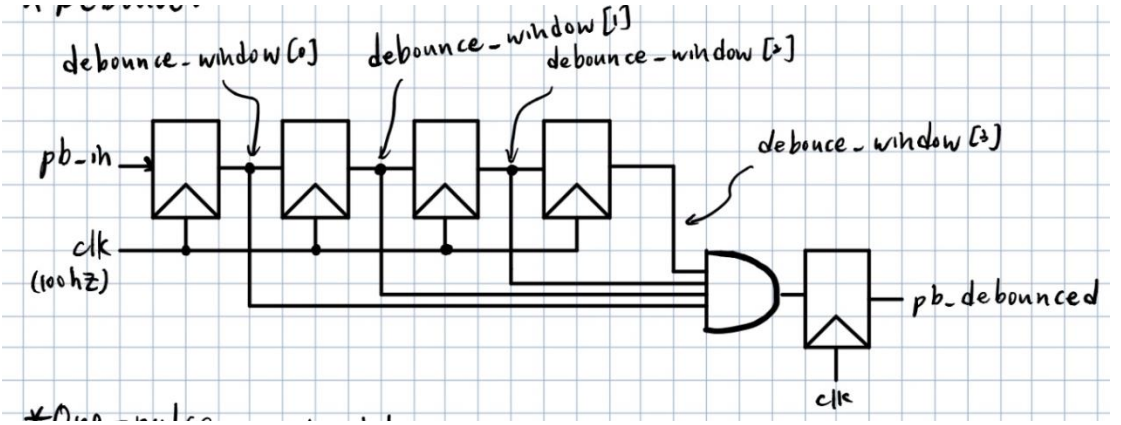
2.Logic Diagram



↑ 上圖是整個系統的邏輯圖，主要小功能會在下面解釋。

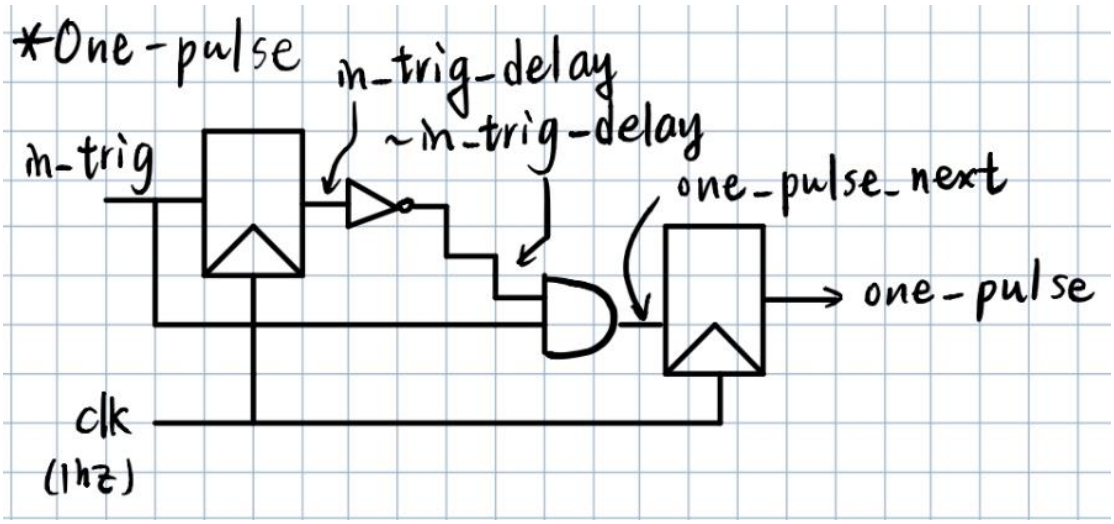
※以上的 module 都沒有接上 rst_N，是為了畫面比較簡潔，其實每個都有接。

▲Debounce

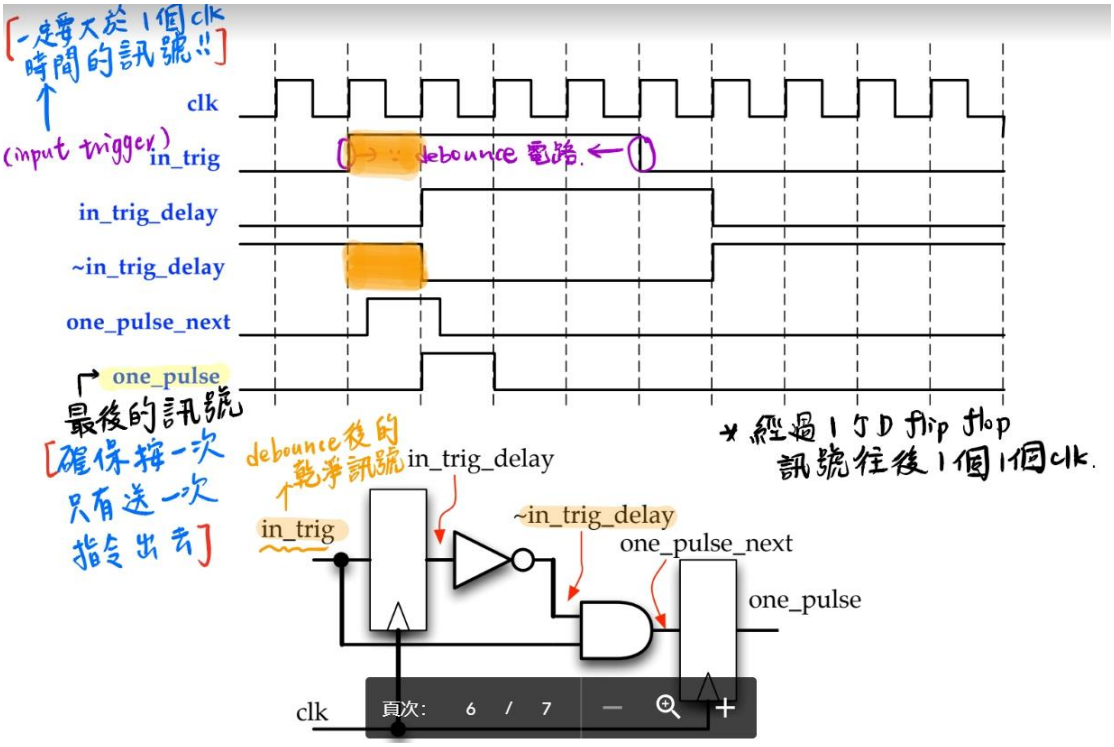


↑ 上圖是利用四個 D-flip flop 來改善按下按鈕會震盪的問題，如果把經過四個 D-flip flop 的結果全部 and 起來，就要當所有值(debounce_window[3:0])全部都等於 1 才會有一個 step 出來，後來再接上一個 D-flip flop 是為了要讓訊號更穩定，因為可以讓 pb_debounced 的結果與 clk 排隊。另外值得一提的是這些 D-flip flop 接的 clk 是經過 clock generator 出來的 100hz，因為與按鈕震盪的頻率比較接近，可以比較準確的判斷按鈕的 trigger 與否。

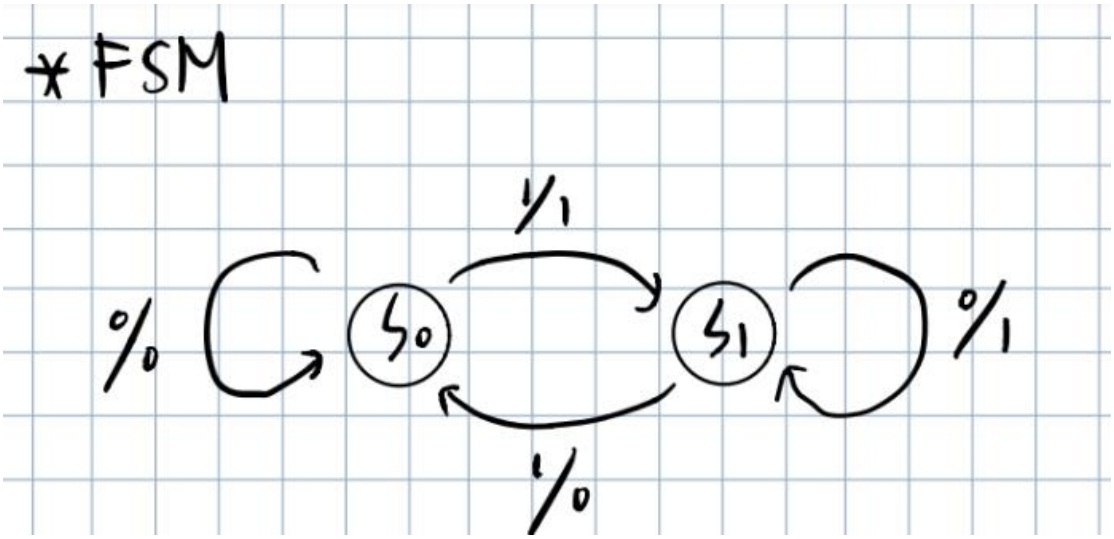
▲One-pulse



↑ 上圖是利用本身的訊號再經過 clk 經過一個 1hz 的 delay 後，利用 invert 來反轉信號，再將兩個訊號 and 起來，最後再利用一個 D flip flop 來把訊號調整到 clk posedge trigger 的位置上，即可以成為控制 down-counter 的 en，也就是 count-enable。(如下圖所示)

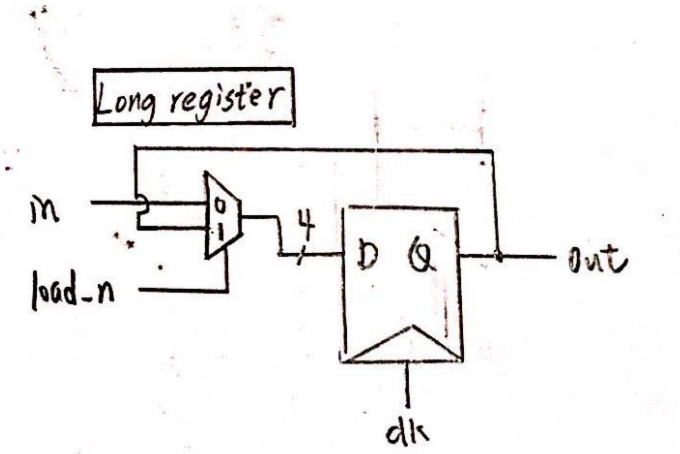


▲FSM



↑利用 FSM 來控制 Upcounter，S0 是暫停的狀態、S1 則是開始數的的狀態，箭頭上的分子代表的是 input、而分母則代表的是 output，利用這個圖表及可以用 case 的語法來寫 verilog 了。(用 case 的語法，state 為括號裏面的變數，利用 input 來變換 state)

▲Long register



↑這個東西是之前沒有寫過的 module，主要的功能是要存起一組數字，in 接的是 sec 或是 min 的資料，當有 load_in 進去的時候，也就是 pb_lap 給他按下去的時候會讓 D flipflop 的數值停留在同個數值，不會繼續和 counter 繼續更新，達到計時的功能目的。

↓以下是 I/O 接腳

I/O	segs [7]	segs [6]	segs [5]	segs [4]	segs [3]	segs [2]	segs [1]	segs [0]	Clk	Rst_n
Pin	V14	U14	U15	W18	V19	U19	E19	U16	W5	R2

I/O	ssd_ctl[3]	ssd_ctl[2]	ssd_ctl[1]	ssd_ctl[0]
Pin	W4	V4	U4	U2

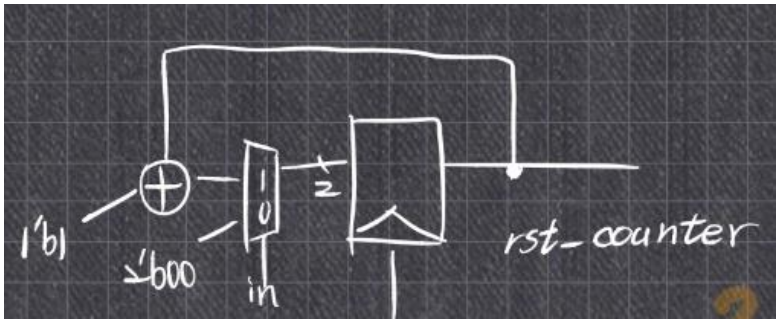
I/O	Pb_lap	pb_stop
Pin	U17	U18

●Discussion

在打這個實驗的時候遇到以下幾個問題:

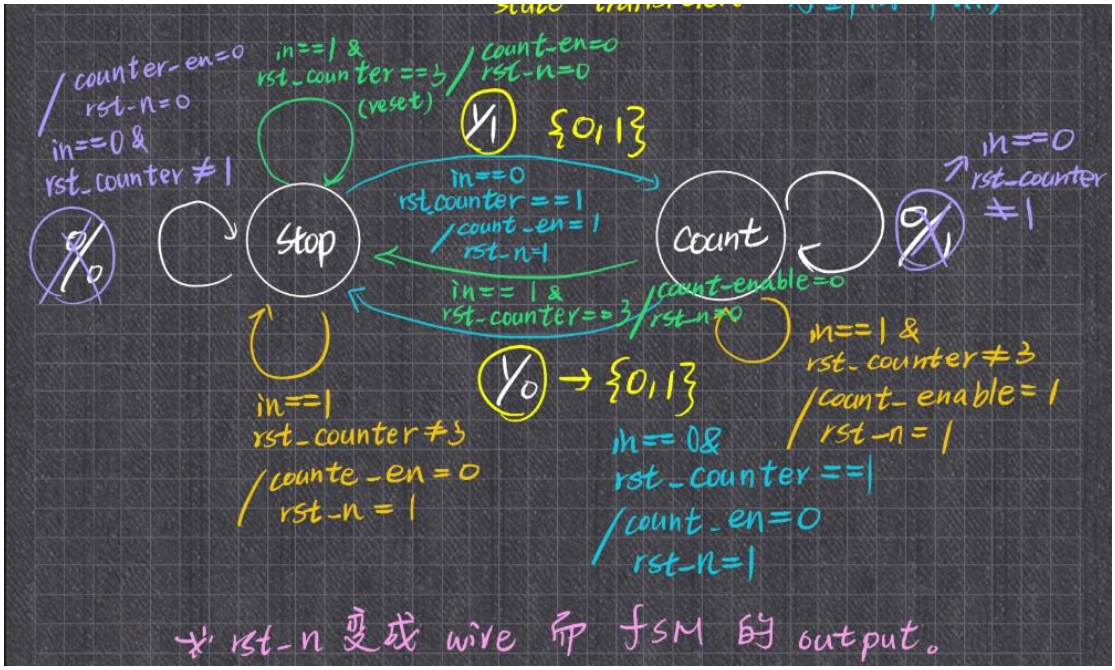
在 pb_lap 長按短按的功能地方，雖然可以成功執行短按就 lap、長按就 reset 的功能，但是問題是當時我還沒有想到要怎麼樣可以分開長按短按的功能方式，也就是說我現在的 code 功能是長按會 reset 但是短按的效果也會執行，雖然因為功能本身的性質，在 reset 的時候前先 lap 一下並不會造成太大的麻煩，頂多就在短按一下就好，但是如果之後的功能是短按還有長按的效果差太多的話就會遇到困難，e.g. 如果短按的功能是 reset 長按的是 lap 就完蛋了。

馬教授上課講的那個方式似乎是一個不錯的解決辦法，就是利用 counter 在接上一個 D flipflop 來延遲短按執行的方式，但是仔細想想其實雖然這個方式解決的第一波的長按可以不會執行短按效果，但是之後長按後，除非你剛剛好按到 4 個 clk，不然你的短按效果還是會執行下去，所以只是延遲了這個問題，到目前還有想到有什麼可以把兩種功能”完完全全”分開執行的方法。

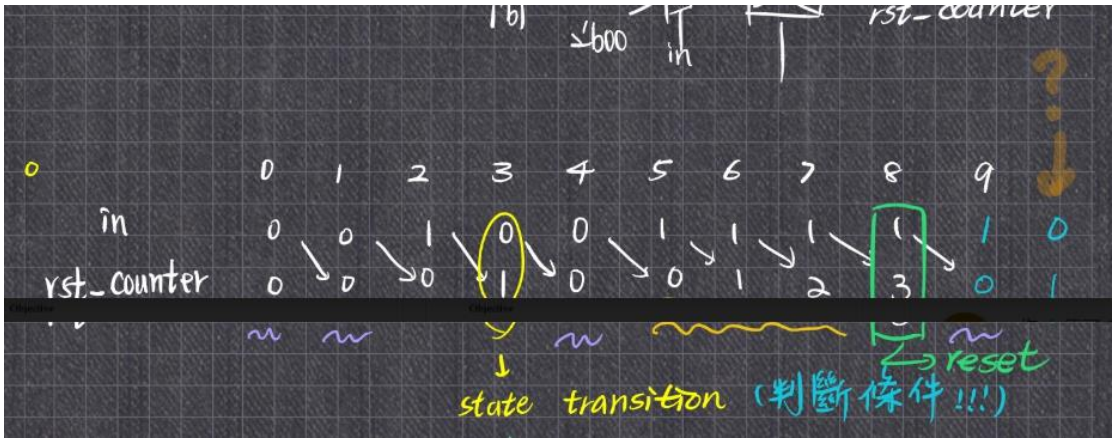


↑

↑控制長按短按效果的 logic diagram



↑這個東西是老師上課講的 fsm，簡單來說就是(in,rst_counter)=(0,1)執行短按效果；(in,rst_counter)=(1,3)的時候執行長按效果。



↑這個問題就是給他出現在最左邊的那個問號，在這個情況中，如果我沒有剛剛好按四個 clk，像是上面就是按了 5 個 clk 時間的按鈕，結果就會多出一個(in,rst_counter)=(0,1)的情況，也就是執行短按效果的條件，所以這樣的話還是沒辦法把兩種效果分開執行的非常好。

7-2. Timer with setting function

●Design Specification

Input:

clk,(接上板子的原本的震盪頻率 W5)

rst_n, (控制整個功能的開關)

switch_mode(這個 switch 可以切換倒數/設定時間的狀態)

pb_pause_resume_hour(這個按鈕有兩個功能，在倒數狀態的時候可以回去 initial 數值;在設定時間的狀態是負責小時的設定功能)

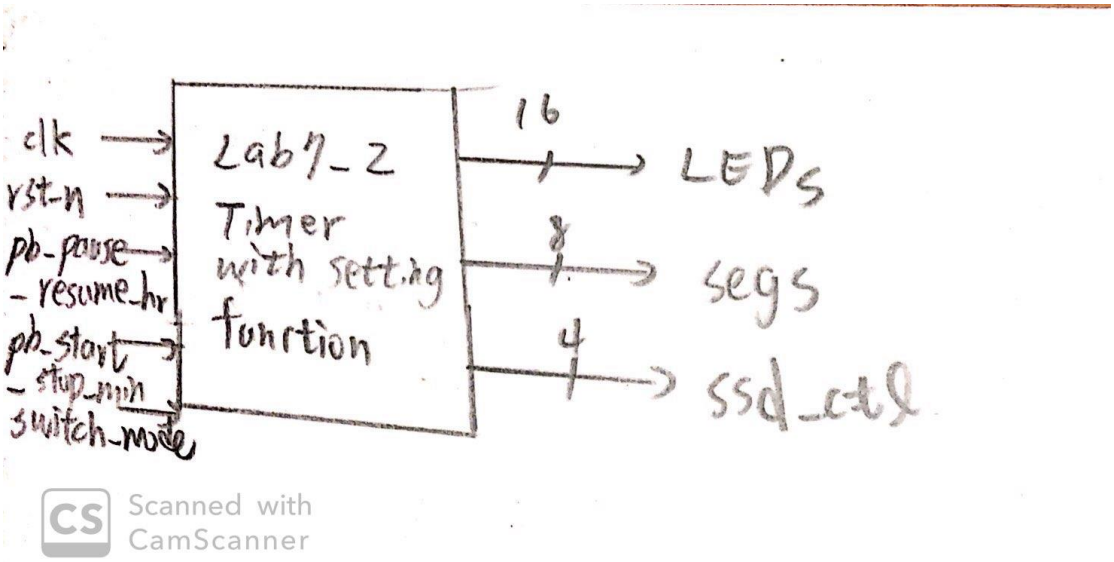
pb_start_stop_min(這個功能也有兩個功能，在倒數狀態的時候可以暫停/開始;在設定時間的狀態時是負責分鐘的設定功能)

Output:

[7:0]segs,(傳給七段顯示器的 output)

[3:0] ssd_ctl,(控制每一個七段顯示器的亮與暗)

[15:0]LEDs(當顯示時間是 0000 時，會全亮；其他時間都不會)



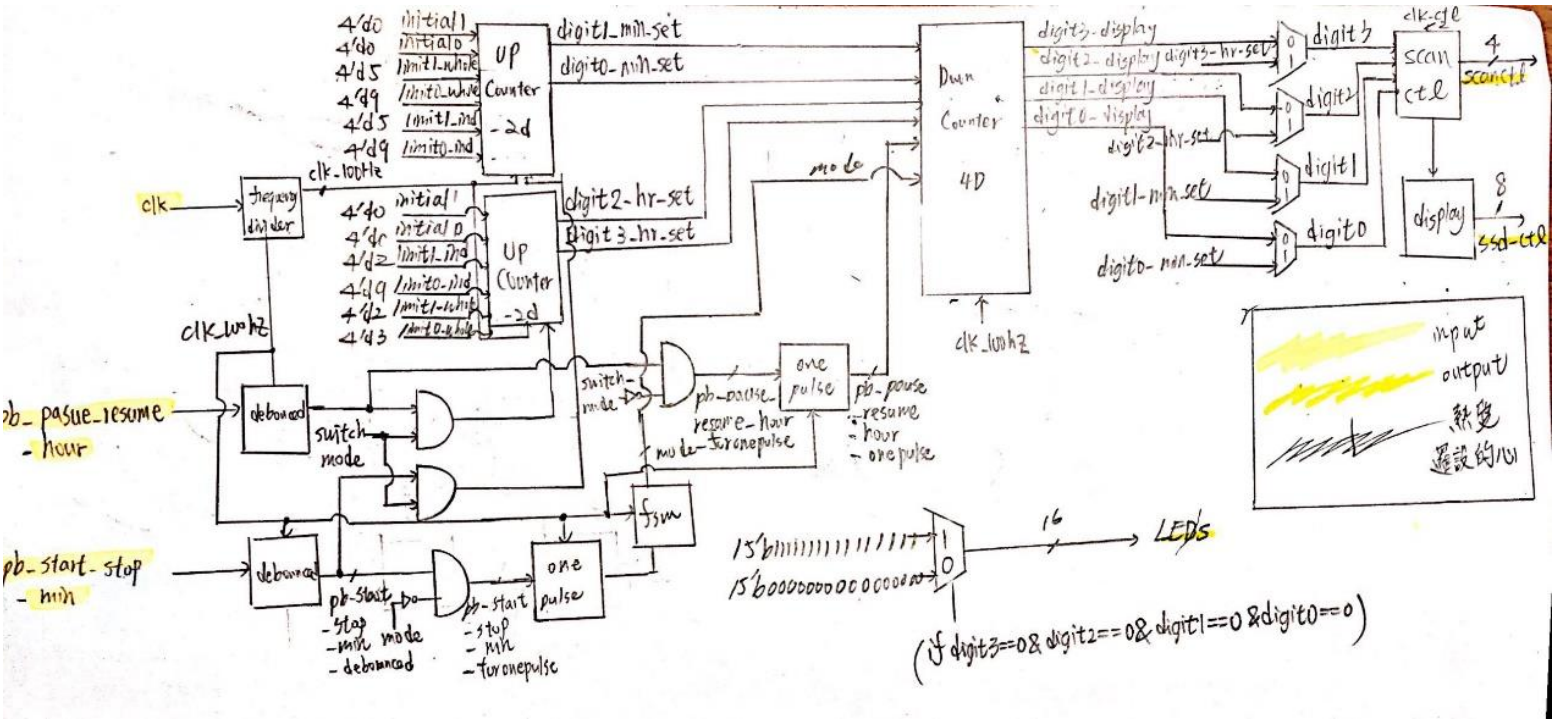
●Design Implementation

1.Outline

這一題就比較時用一點點了，就是一個可以設定 downc counter 的起始值，就像是那種烤蛋糕時候要用的倒數計時器。

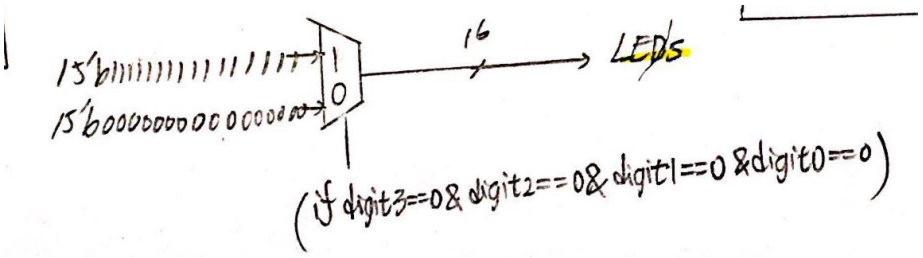
這題並沒有運用到新的 module，就是 module 要用的比較多比較繁雜，另外值得一提的還有在按鈕方面的我就沒有利用長按短按了，我利用當時 mode 的狀態來分別按鈕的功能，我覺得這樣會讓多功能的按鈕每個功能完美分開，不會彼此互相影響，是目前比較好的解決方式。

2.Logic Diagram



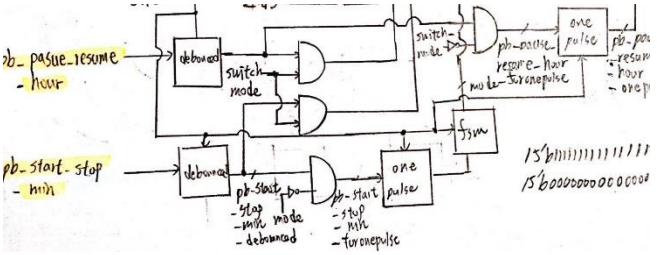
↑ 上圖是整個系統的邏輯圖，沒看過的小功能會在下面解釋。

▲LEDs



當所有 digit 都是零的時候會讓 LEDs 全亮。

▲把按鈕功能分開的方式



因為這次多功能按鈕的多功能是分別在不同 mode 上時產生的，也就是說我要控制是 hr/min 的時候是在 setting 模式;我要控制的是 stop/start 的時候是在 down counting 模式，所以我不需要使用 7-1 那樣

的一個 Dflipflop 來延遲創造多功能效果，我是利用 and 來把 pb 的訊號和 mode 弄再一起，也就是說當兩個條件都符合=1 的時候才會有效果，就不會有這種情況:當我在 setting 的時候結果 counter 已經偷偷在數了；或是我在控制 down counter 的時候 setting 也偷偷在跑了。

這是我想到的方法，不過不知道這樣好不好，因為這個讓我想到上個學期馬教授說到的 driving 問題，好像是類似的觀念，把 clk 和其他東西 AND 再一起當作控制條件會讓 driving 的力量變弱。但是我這個情況並不是和 CLK AND 再一起，而已經是 pb_debounced 了，不知道會不會發生那種接太多讓原本方波變得有點像是 exponential 的圖形產生，之後會去問教授。(不過這樣打比較不會讓多功能的效果混在一起)

↓以下是 I/O 接腳

I/O	segs [7]	segs [6]	segs [5]	segs [4]	segs [3]	segs [2]	segs [1]	segs [0]	Clk	Rst_n
Pin	V14	U14	U15	W18	V19	U19	E19	U16	W5	R2

I/O	ssd_ctl[3]	ssd_ctl[2]	ssd_ctl[1]	ssd_ctl[0]
Pin	W4	V4	U4	U2

I/O	Pb_pause_resume_hour	pb_start_stop_min	Switch_mode
Pin	W19	T17	T1

I/O	LEDs[15]	LEDs[14]	LEDs[13]	LEDs[12]	LEDs[11]	LEDs[10]	LEDs[9]	LEDs[8]
Pin	L1	P1	N3	P3	U3	W3	V3	V13

I/O	LEDs[7]	LEDs[6]	LEDs[5]	LEDs[4]	LEDs[3]	LEDs[2]	LEDs[1]	LEDs[0]
Pin	V14	U14	U15	W18	V19	U19	E19	U16

●Discussion

這題目雖然不算非常困難，但是在 debug 的時候還是遇到了蠻多困難的，很多都是雖然 module 是重複利用的，可以直接引入，但是因為每個題目的功能不一樣，還是要有一些微調，像是有需要 initial vale，因為會回復的時候值不一樣。還有我每次想法都不一樣，結果回去用就忘記我之前在幹嘛了，結果都砍掉重練……

所以我就乾脆所有東西名子都取很長很長，可以從上面的 pb 名子看出來，

這樣不管我什麼時候回來看都知道當時的我在想什麼東東。

7-3. Integrate 7-1, 7-2

●Design Specification

Input:

clk,(接上板子的原本的震盪頻率 W5)

rst_n, (控制整個功能的開關)

pb1_rst_n,(有 rst_n 的效果，按下去基本其他效果都沒有用了)

pb2_start_hour,

(在顯示模式可以 stop/start，在設定模式可以改變 hour)

pb3_reset_min,(在顯示模式可以 reset，在設定模式可以改變 min)

pb4_mode_lap,(長按可以改變模式，短按是 lap 的效果)

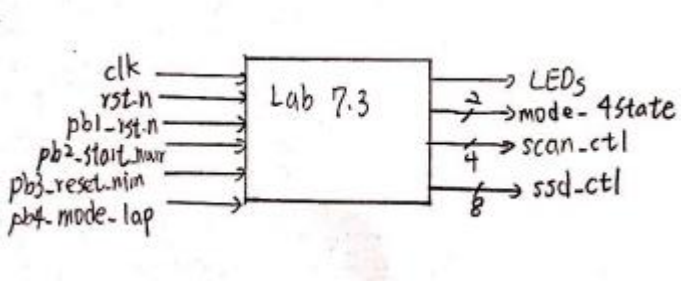
Output:

[7:0]segs,(傳給七段顯示器的 output)

[3:0] ssd_ctl,(控制每一個七段顯示器的亮與暗)

LEDs,(顯示都是零的時候會亮，只有一個燈而已)

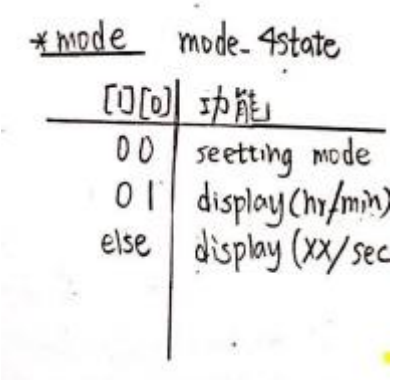
[1:0]mode_4state(讓你知道現在的狀態是什麼，可以進行什麼操作)



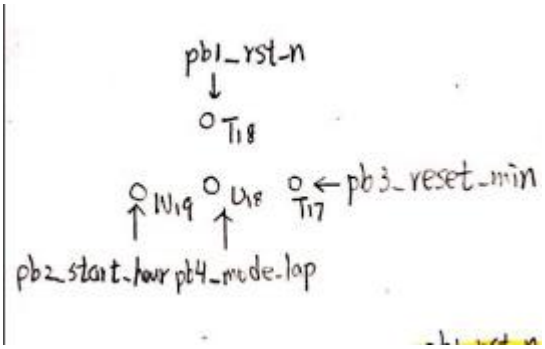
●Design Implementation

1.Outline

基本上就是把前兩個實驗的功能合再一起，把所有 module 塞再一起，但是比較困難的部份應該是要怎麼把有限的按鈕賦予合適的功能(還有寫結報)。

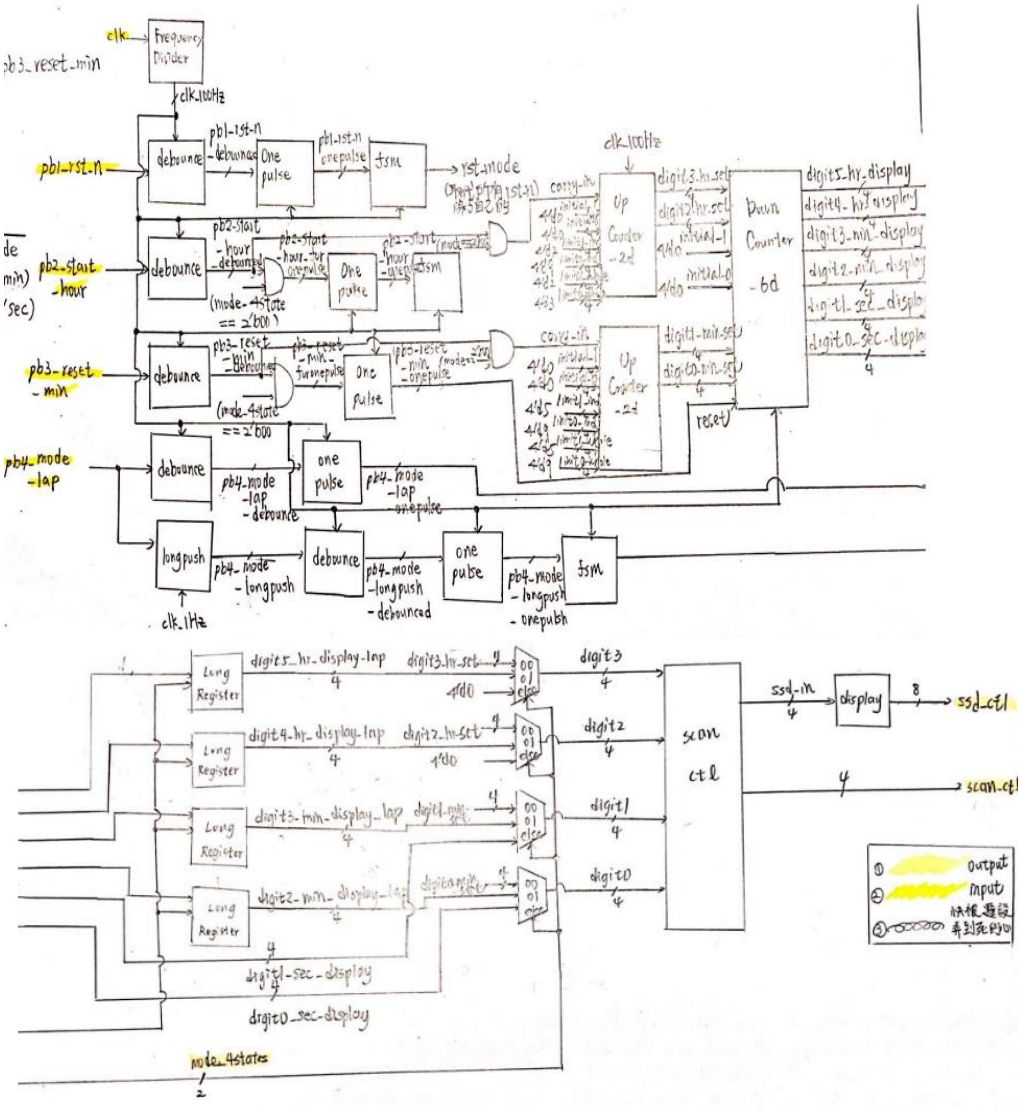


←左圖是 mode 對照當前的功能，1 的時候會亮，在 00 的時候可以利用兩個按鈕來設定小時還有分鐘，在 01 的時候可以顯示小時/分 display，其他時候就是顯示當前秒數。



←左圖是按鈕功能的對照圖，這樣比較清楚比較直覺。

2.Logic Diagram



↑ 上圖是整個系統的邏輯圖，因為功能 module 比較多，所以分成兩張圖來表示，下面那張是連接上面那張的延伸。

(為了簡潔，rst_n(在這邊是 rst_mode)就沒有畫進去了，還有一些之前 clk 的接線也省略，有畫出比較重要的)

↓以下是 I/O 接腳

I/O	segs [7]	segs [6]	segs [5]	segs [4]	segs [3]	segs [2]	segs [1]	segs [0]	Clk	Rst_n
Pin	V14	U14	U15	W18	V19	U19	E19	U16	W5	R2

I/O	ssd_ctl[3]	ssd_ctl[2]	ssd_ctl[1]	ssd_ctl[0]
Pin	W4	V4	U4	U2

I/O	Pb1_rst_n	Pb2_start_hour	Pb3_reset_min	Pb4_mode_lap
Pin	T18	W19	T17	U18

I/O	Mode_4states[1]	Mode_4states[0]	LEDs
Pin	E19	U16	L1

●Discussion

這個實驗沒有遇到什麼問題，因為就只是把上面兩個合在一起，比較大的問題可能是有時候太多 module 會迷失自己，突然忘記自己當時在幹嘛。

●Conclusion

因為大部分的 module 都是用之前寫的，所以現在比較考驗如何使用這些工具，並連接他們。而且從原本只用了數個 module 到現在 7.3 用了快 25 個 module。不知道我 final 會不會超過 100 個 module……邏輯設計實驗的各個部份的作業真的蠻繁雜的，雖然不是說難到做不出來，但是要想得很全面，還有要畫邏輯圖來讓其他人了解自己的設計，花的時間真的不是普通的多，我可以漸漸體會到我未來的人生了