

Lab6 Report

6-1. 24/12 AM/PM Clock

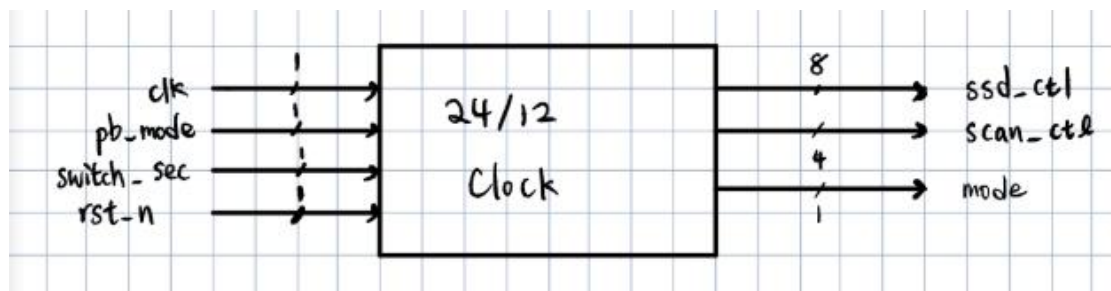
●Design Specification

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_mode,(讓時鐘可以切換 24/12 時制)
- switch_sec(讓時鐘可以切換時:分/xx:秒)

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)
- Mode(24 時制不亮/12 時制亮)



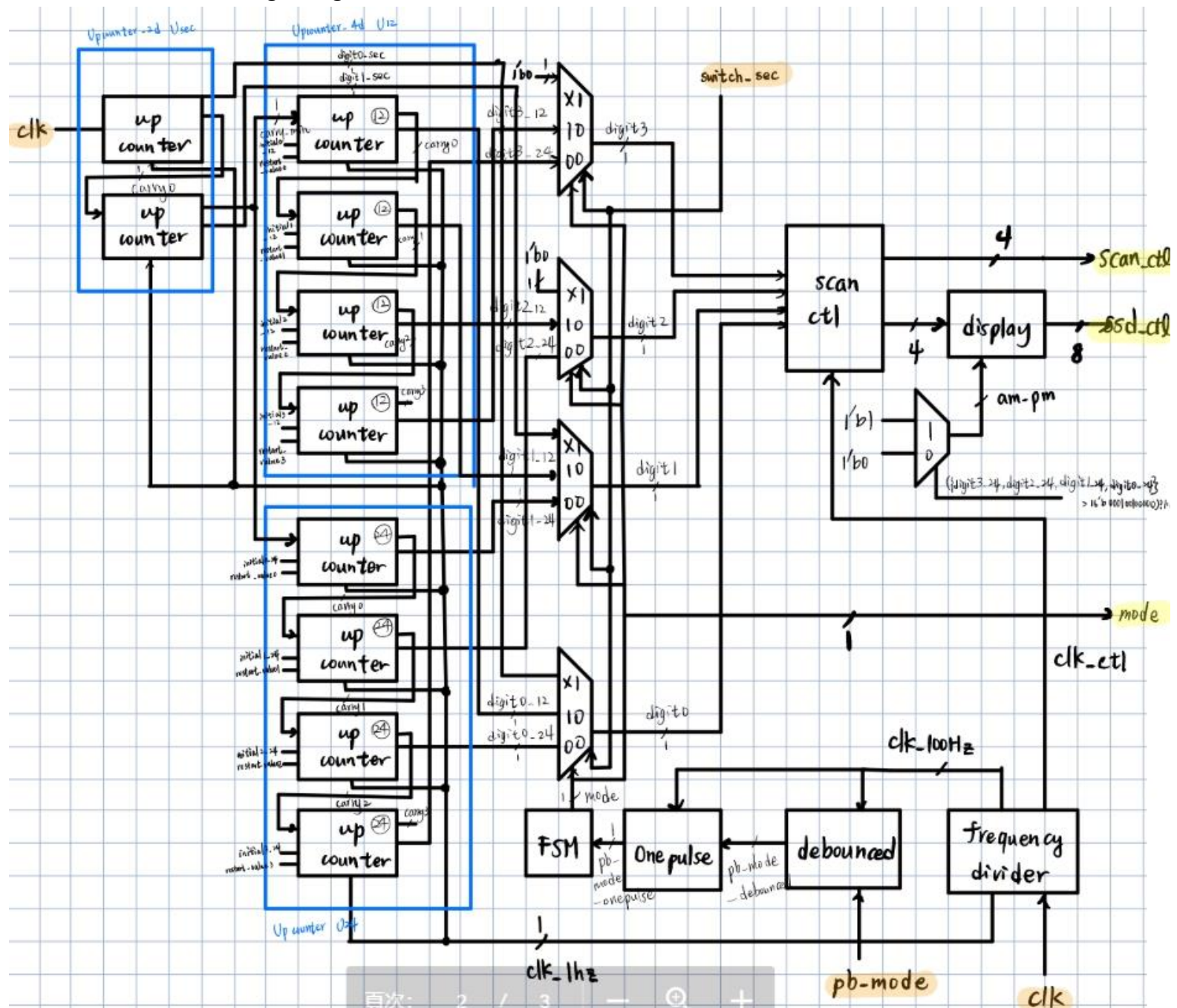
●Design Implementation(1.1)

1.Outline

這個實驗主要是要做出一個可以 24/12 時制切換的時鐘，另外還要加上一個 switch 來切換時:分 xx:秒的按鍵，會使用到的功能大多之前都用過了，只是要重新組合成另外一個有用的新功能。

這個實驗與之前實驗主要不同的地方是，之前大多是利用 Down Counter 來做實驗，但是這次是 Up counter，所以要額外多改一些符號還有條件，詳細不同會在下方說明。

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。

(圖的字太小抱歉，後面會放大解釋)

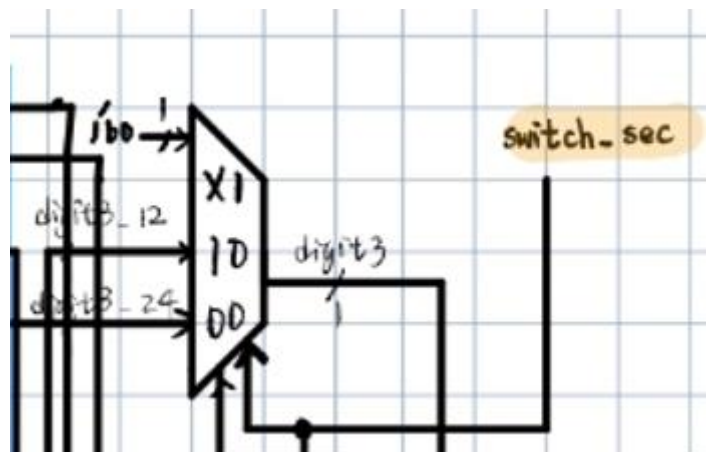
(橘色螢光筆為 input、黃色螢光筆為 output)

※圖中是以正常時鐘為標準，但是我為了 demo 方便，所以我在 verilog 中把 `clk_1Hz` 改成 `clk_100Hz`，才可以快速看出變化。

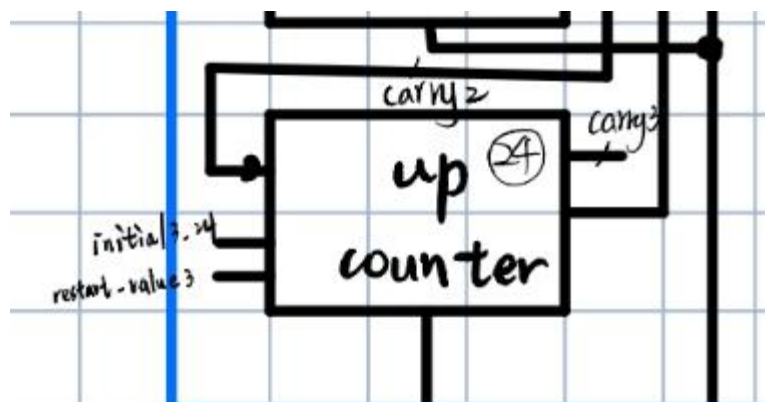
▲Up counter*3

我主要是利用到三個 Up Counter 同時計時來完成這個實驗，一個是 2_bits 的給秒數計時，另外兩個 4_bits 的分別給 12 時制/24 時制計時，再利用 mux 來分別決定七段顯示器上面要顯示的是什麼結果，而 mux 的決定條件是利用由 pb_mode 控制的 fsm 狀態還有 switch_sec，當沒開啟 switch_sec 時，一開始 mode 預設是=0(24 時制的狀態)，若按下 pb_mode 則可以切換成 mode=1，也就是 12 時制的狀態。再把所有結果接到 scan_ctl、display 顯示出結果。

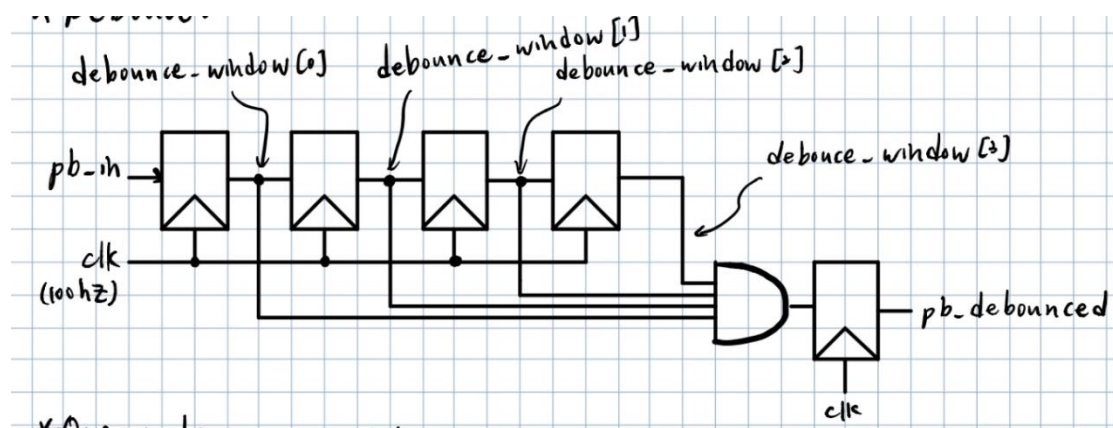
另外秒數的部分是利用 switch 來控制顯示秒數的功能，我直接在 top module 中利用一組 if-else 來控制，可以由右圖中的 mux 中的控制條件為 x1 顯示看出，當 switch 開啟的時候，不管現在的 mode(控制 24/12 時制)是什麼狀態，都會顯示秒數的狀態。



還有因為是時鐘的緣故，不能直接像是之前的作業一樣，把 initial 還有歸零時的數值設成一樣，(12 時制)也就是說在 12 點 59 分要進位時不能回到 00 點 00 分，而是要變成 01 點 00 分繼續開始，所以除了 initial value 之外還要再另外設一個 reset_value 讓時鐘能夠成功進位到正確的數值。

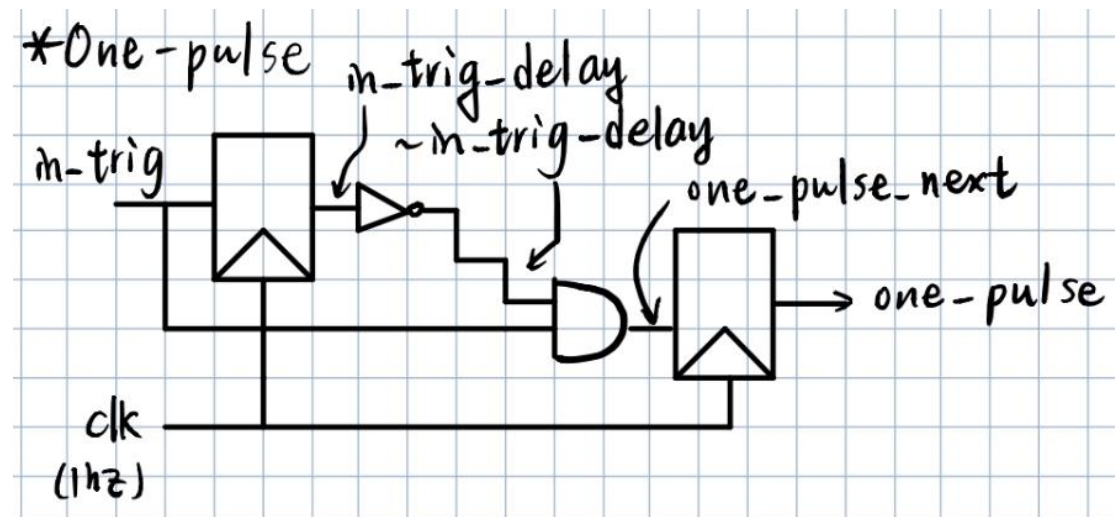


▲Debounce

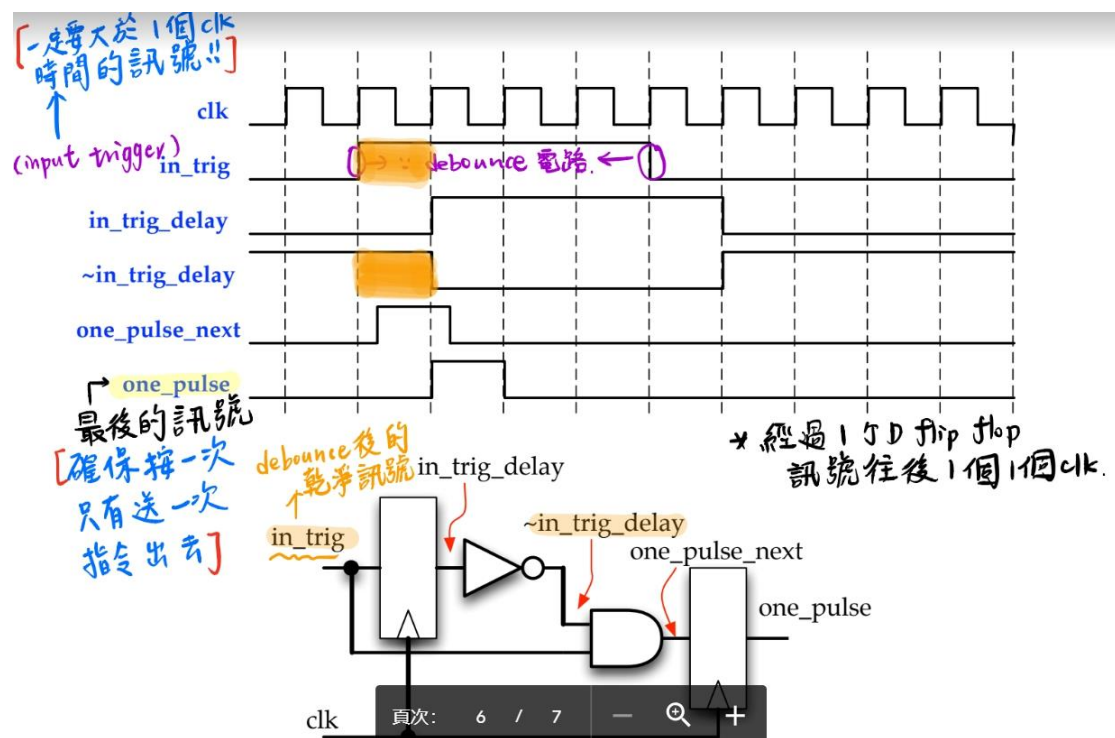


↑ 上圖是利用四個 D-flip flop 來改善按下按鈕會震盪的問題，如果把經過四個 D-flip flop 的結果全部 and 起來，就要當所有值(debounce_window[3:0])全部都等於 1 才會有一個 step 出來，後來再接上一個 D-flip flop 是為了要讓訊號更穩定，因為可以讓 pb_debounced 的結果與 clk 排隊。另外值得一提的是這些 D-flip flop 接的 clk 是經過 clock generator 出來的 100hz，因為與按鈕震盪的頻率比較接近，可以比較準確的判斷按鈕的 trigger 與否。

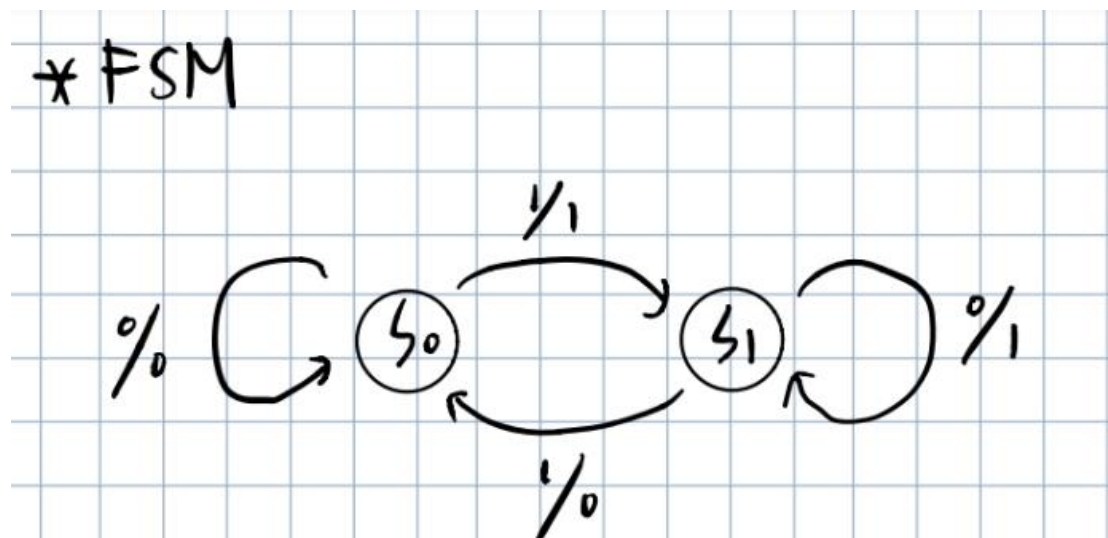
▲One-pulse



↑ 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)



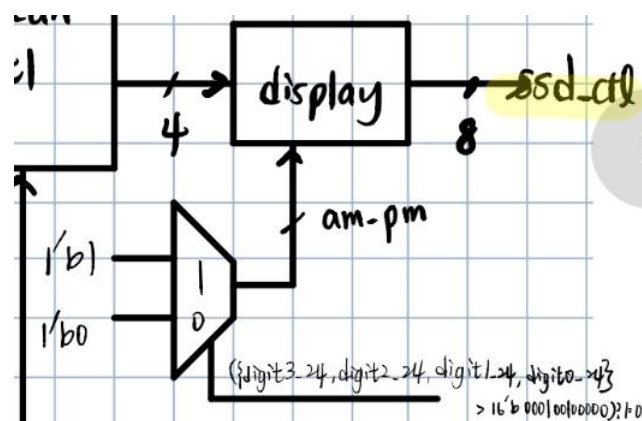
▲FSM



↑利用 FSM 來控制 Upcounter，S0 是 24 時制的狀態、S1 則是 12 時制的狀態，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)

▲判斷 PM/AM

我利用 24 時制的四位數字當作 MUX 的判斷條件，如果大於 12:00 的時候，就會讓傳入 display 的 am_pm 訊號為 1，讓小黑點一起亮起，代表現在是 PM 的狀態。



↓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_mode | pb_rst | Switch_sec | Mode |
|-----|---------|--------|------------|------|
| Pin | U18 | U17 | T1 | L1 |

● Discussion

在打這個實驗的時候遇到以下幾個問題：

1. 原本 Up counter 接的是 1Hz，但是我為了 demo 方便所以我把頻率接成 100Hz，但一開始我只改了秒數的 Up counter，而忘記改了分與小時的 Up counter，所以我做出來就只有秒數一直瘋狂進位，但是分鐘的 Up counter_4d 沒有接到 carry_min 的訊號，所以不為所動。

<sol:> 把所有的頻率都接成 clk_100Hz 即可讓訊號順利傳遞。

2. 我原本寫的 Verilog 檔案不知道為什麼一直不能執行，可以跑 Bitstream 但是不能灌入板子內。

<sol:> 我重新弄一個新的檔案，並把所有的內容都複製過去就可以成功執行了。

3. 我原本寫出要判斷 PM/AM 的敘述是用

```
assign pm_counter_mode =  
({digit3_24,digit2_24,digit1_24,digit0_24}>16'd1200)?1:0;
```

來做判斷，但是不知道為什麼實際板子跑出來到凌晨 5:00 就會點點就會亮了。

<sol:> 改成

```
assign pm_counter_mode =  
({digit3_24,digit2_24,digit1_24,digit0_24}>16'b0001_0010_0000_0000)?1:0;
```

就可以成功執行了，但是我還是不知道為什麼……上課的時候請問助教或教授。

6-2. Year/Month/Date function(no leap years)

● Design Specification

Input:

clk, (接上板子的原本的震盪頻率 W5)

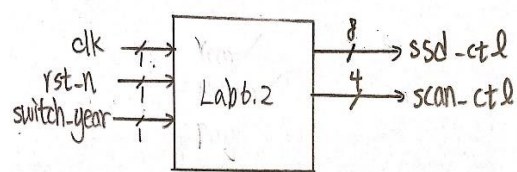
rst_n, (控制整個功能的開關)

switch_year (控制顯示年或是月份/日期)

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)

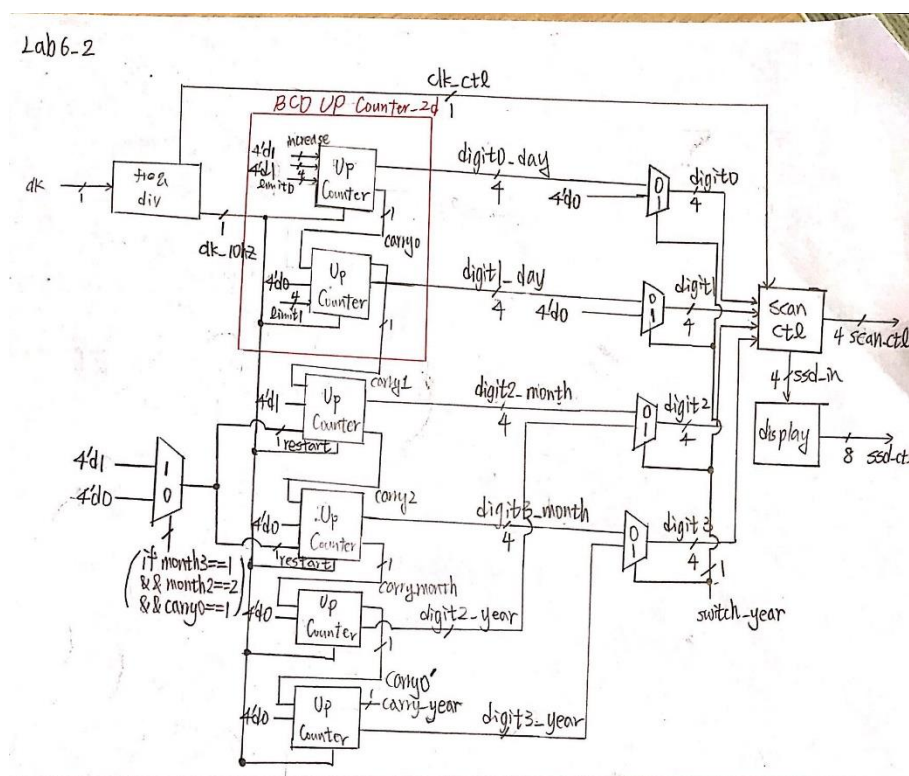


● Design Implementation

1. Outline

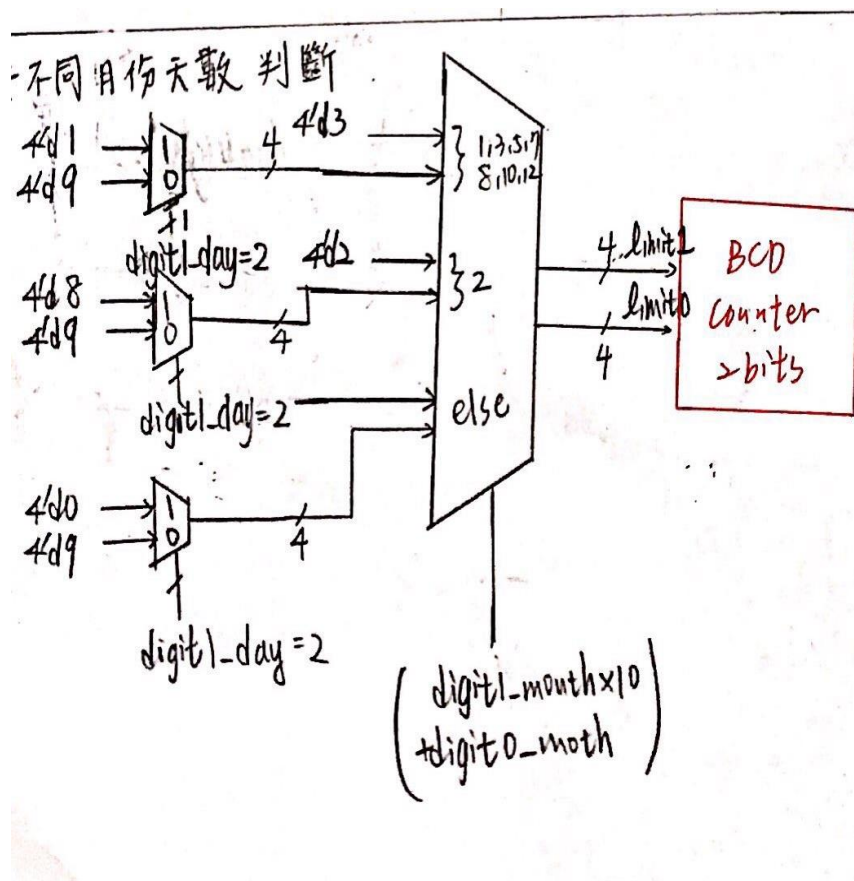
這一題主要是要製作一個能表現年分、月份、還有日期的 counter，雖然大部分的觀念不同，但是因為不同月份，而會有不同的日期，造成在判斷上面相對比較困難，所以基本上所有的小功能 module 之前都寫過了，在這邊就只特別說明新的功能。

2. Logic Diagram



我利用一個 4bits 的 BCD-upcounter 來表現月份和日期，然後在把月份進位後的 carry_month 接去一個 2bits 的 BCD-upcounter 來表現年分再去經過一個 mux 來判斷要輸出的 digit，判斷條件是由一個 switch_year 來判斷，只要推下去，顯示的會是年分；沒推則是月份還有日期。

值得一提的有這個判斷月份的 mux，如下圖，因為上面的圖畫不下，所以我另外寫在這邊。



我是利用 case 的語法來判斷當月份是 31 天、30 天亦或是 28 天，達到相對應的月份時，給 counter 個位數還有十位數的 limit 值，也就是說當 counter 數到 limit 值然後又要進位的時候，就會回去原本的 initial 值了。另外還有要特別注意的地方是要在最左方再加上一個小 mux 來讓日期在尚未進位到 2x 錢的時候 limit 仍然是原本的大小，才不會讓 counter 在只有個位數或是只有 1x 就在 8 或 9 進位。

↓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 |

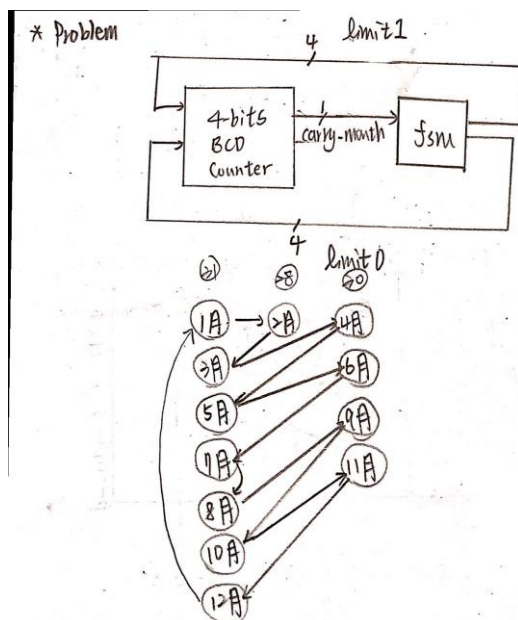
| I/O | ssd_ctl[3] | ssd_ctl[2] | ssd_ctl[1] | ssd_ctl[0] |
|-----|------------|------------|------------|------------|
| Pin | W4 | V4 | U4 | U2 |

| I/O | Switch_year |
|-----|-------------|
| Pin | T1 |

● Discussion

在打這個實驗的時候是我最痛苦的一次…一直找不到哪裡錯誤，直到整個重弄好幾次，快死掉了……

我原本想利用像是上一題的 fsm 方法，來判斷月份的日期數，而 fsm 的 input 是 counter_4d 的十位數日期的 carry 也就是說，我原本想說只要日期達到 limit 值要進位的時候，就會讓 fsm 也一起到達下一個狀態，但是不管怎麼弄都會跑出一樣的錯誤，我利用左下圖來解釋。

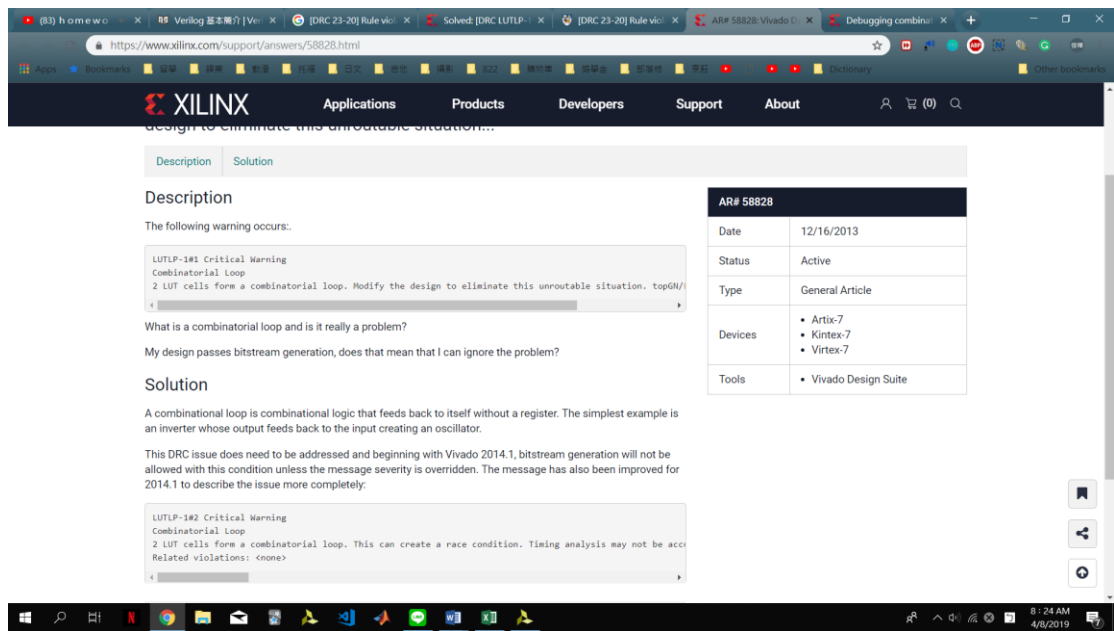


這個是我原本的想法，想說利用 FSM 來改變不同月份的 limit 值，但是問題是出在這是一個 combinational logic 的設計，不可以把後面的 output 在接回去前面 module 的 input，而且我不管怎麼樣在另外設變數阿，或是把變數設在不同的 module 裡面，最後都還是會出現這個問題，所以說我最後才利用 case 直接寫死，才不會有這樣的問題。

另外一個小問題就是 verilog

裡面不接受全形的字母，我之後也要注意。

●Reference



design to eliminate this unroutable situation...

Description

The following warning occurs:

```
LUTLP-1#1 Critical Warning
Combinatorial Loop
2 LUT cells form a combinatorial loop. Modify the design to eliminate this unroutable situation. topGN/
```

What is a combinatorial loop and is it really a problem?

My design passes bitstream generation, does that mean that I can ignore the problem?

Solution

A combinational loop is combinational logic that feeds back to itself without a register. The simplest example is an inverter whose output feeds back to the input creating an oscillator.

This DRC issue does need to be addressed and beginning with Vivado 2014.1, bitstream generation will not be allowed with this condition unless the message severity is overridden. The message has also been improved for 2014.1 to describe the issue more completely:

```
LUTLP-1#2 Critical Warning
Combinatorial Loop
2 LUT cells form a combinatorial loop. This can create a race condition. Timing analysis may not be acc
Related violations: <none>
```

| AR# 58828 | |
|-----------|---------------------------------------|
| Date | 12/16/2013 |
| Status | Active |
| Type | General Article |
| Devices | • Artix-7 • Kintex-7 • Virtex-7 |
| Tools | • Vivado Design Suite |

去找自己問題解決辦法時候找的網站，但是最後還是重寫才解決，應該是那子的寫法就不能被接受，所以不管怎麼改都是徒勞無功。

6-3. Year/Month/Date/Hour/Min/Sec function(with leap years)

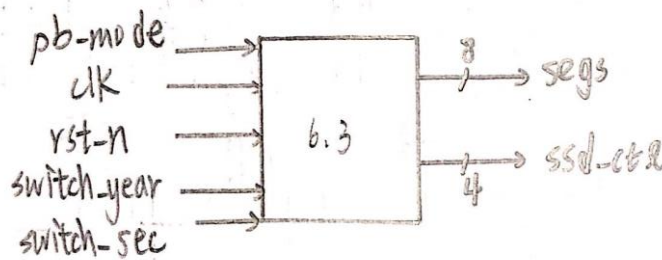
●Design Specification

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_year(控制顯示年或是月份/日期)
- switch_sec(控制顯示秒數)
- pb_mode(控制顯示時間為 24 或 12 時制)

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)



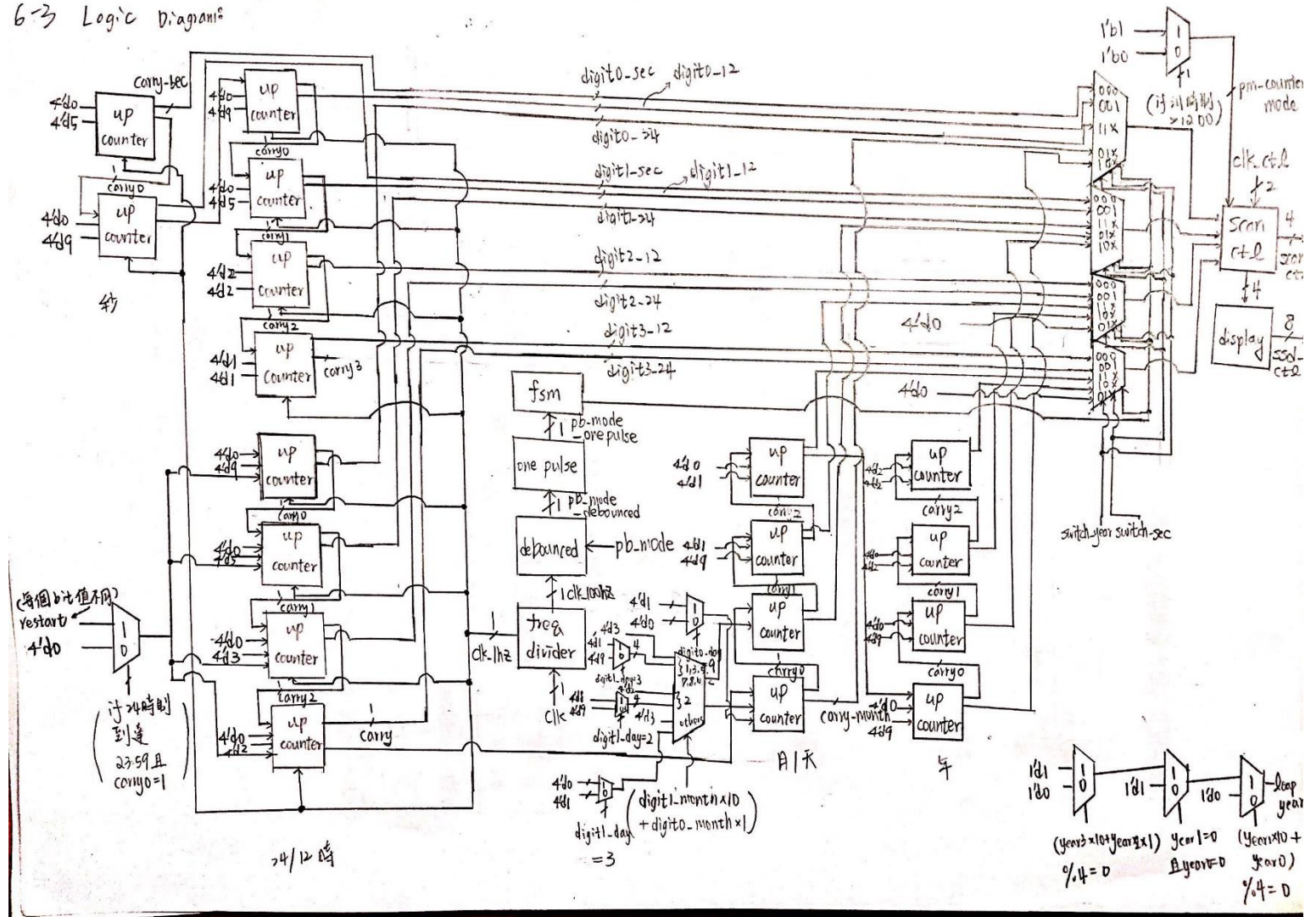
● Design Implementation

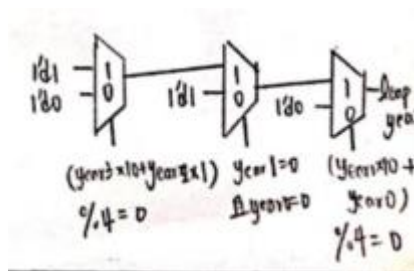
1. Outline

這題主要要把一二題的功能組合起來，另外還要加上閏年的概念，所以要加上判斷是否為閏年的 mux。

2. Logic Diagram

6-3 Logic Diagrams





上圖是判斷是否為閏年的 mux，先判斷是否被 4 整除，再判斷是否被 100 整除，最後判斷是否被 400 整除，再接回去月的 module 當作判斷條件。

↓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 | Switch_year | Switch_second | Pb_mode |
|-----|-------------|---------------|---------|
| Pin | U1 | T1 | U18 |

● Discussion

這題花的時間比較少，因為經過之前兩題的前車之鑑。

閏年除了被 4 整除外，還有被 100、400 整除的規則，但由於必須邊注意有沒有 2/29 號，但又要處一年份，所以也不能條太快，因此可能會需要用到調整 clk 速度的方法，才可以比較好檢查到是否有完成題目的要求。

● Conclusion

這次的實驗真的讓我非常頭疼，因為 6.2 的實驗讓我失去了好幾個晚上，一直不斷想新辦法、重跑 bitstream，在灌到板子上，但是結果還是一樣，直到沒辦法跑去問學霸才得到答案，如果我上學期有好好學的話，可能就不會遇到這種困難了，看來理論跟實際操作真的還是有很大的差別，唯有良好的理論基礎才可以讓自己更深刻的瞭解實作的方法，不行了我要趕快去看別科了……

