**Lab 5: Timers**

**106061218 李丞恩**

**Pre-lab**

**Design Specification**

(1) I/O ports

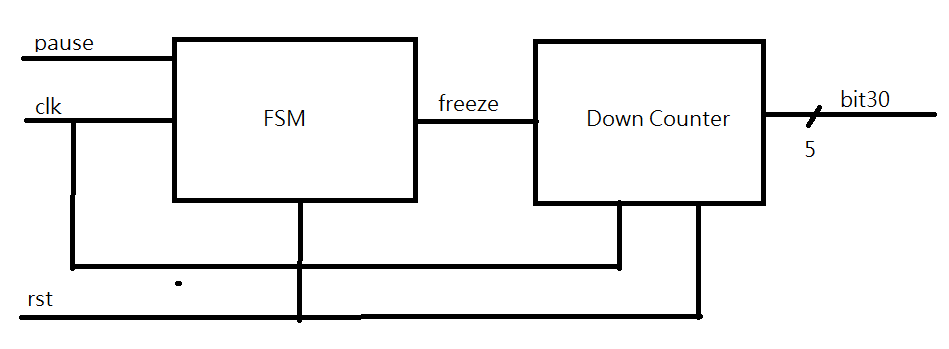
input rst reset或開關

input clk 全域時脈

input pause 按鈕

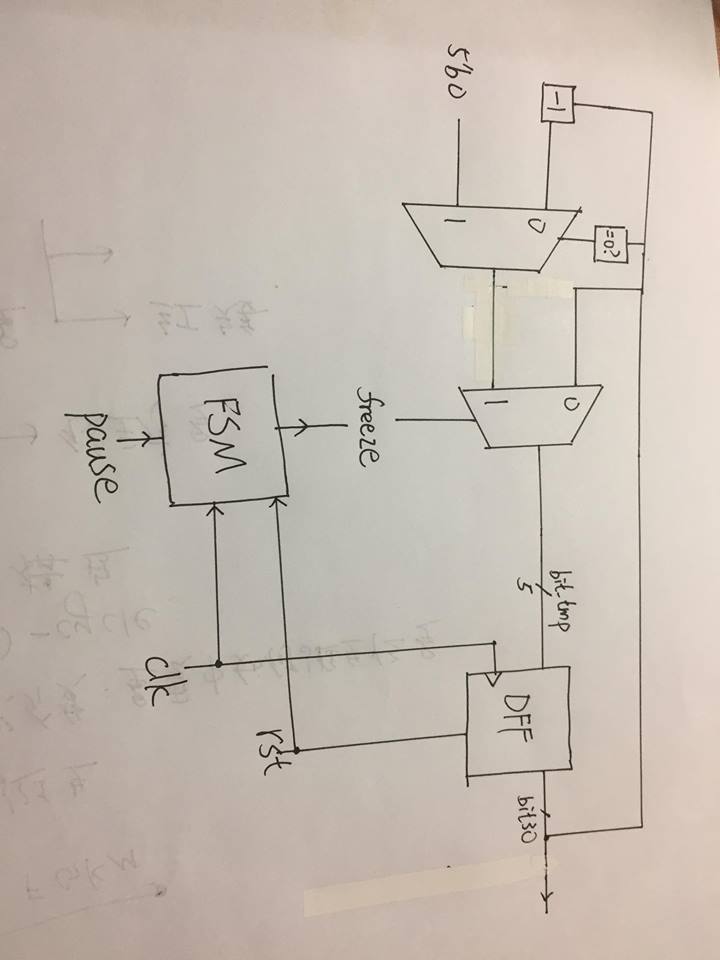
output reg [4:0] bit30 即30秒倒數計時

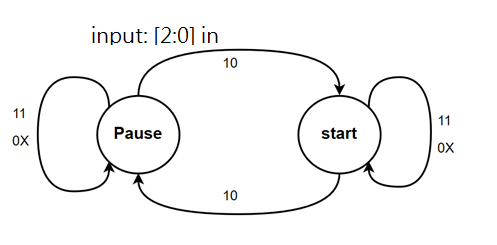
(2) Block Diagram



**Design Implementation**

(1) Logic Diagram





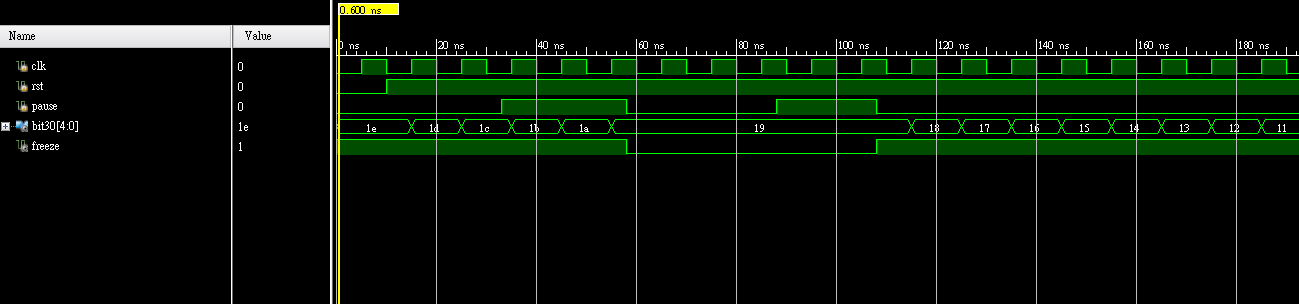
(2) Verilog codes

這次我的code在原本的檔案內有打註解，在此就不另外說明。

(3) Verification

Test Bench 跑出來的結果如圖示：

由圖可知控制暫停與否的freeze訊號為0時，counter就會停止作用。



**2. 30-second down counter with pause function.**

**Design Specification**

(1) I/O ports

Input:

Clk: global clock

Rst: 接在button上，採正緣觸發。

pb\_in: 接在另一個button上，負責暫停。

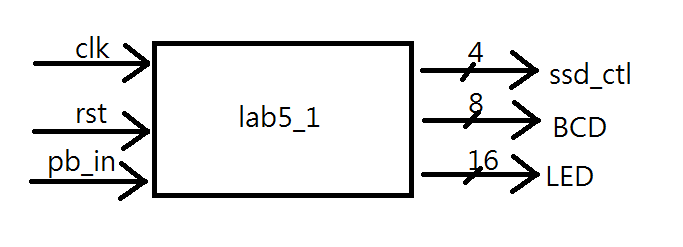
output

[7:0] BCD

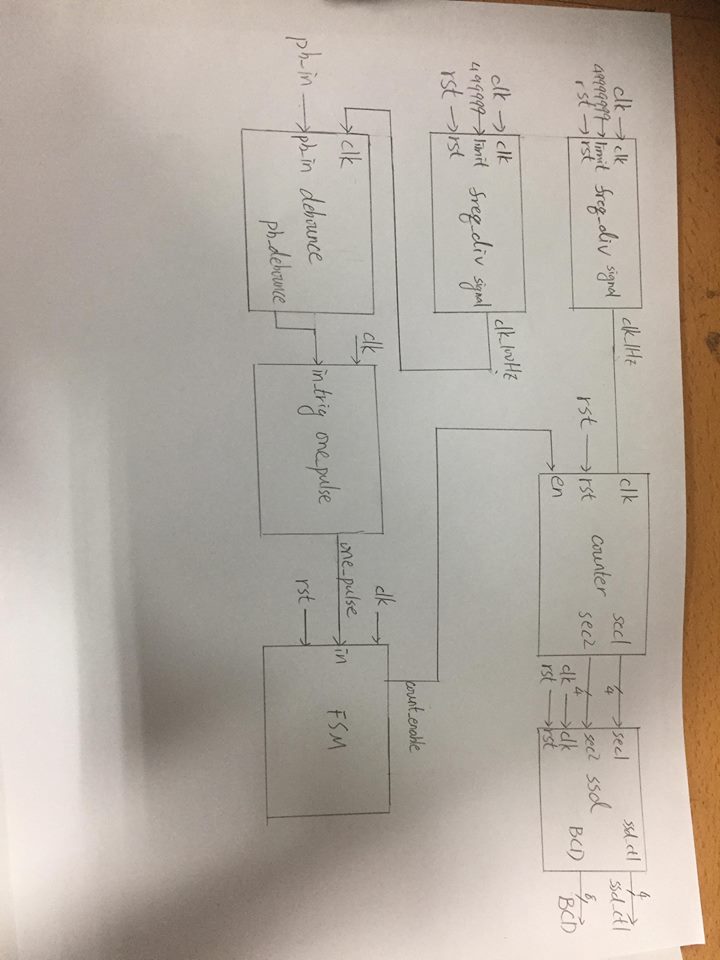
[3:0] ssd\_ctl

[15:0] LED: 倒數到0的時候全部變0

(2) Block Diagram



**Design Implementation**

****

Logic Diagram

每個module的功用如下：

Freq\_div：普通的除頻器。藉由給予不同的上限值來產出不同頻率的clock。在這個lab中，產生Counter用的1Hz的clock和debounce要用的100MHz的clock。

Debounce：老師給的範例程式，負責把按下去的雜訊處理乾淨。

One\_pulse：老師給的範例程式，負責產生脈衝丟給FSM

FSM：只有兩種state：倒數與暫停。轉換的依據是經由按壓按鈕的訊號經以上兩個module處理後所產生的脈衝。

Counter：非常普通的30秒下數器，但有把FSM的state接進來達成按一次按鈕以暫停的功能。

Ssd：負責把剩餘秒數顯示在七段顯示器上。

**3. The same function as 2. Instead of using two push buttons**

**Design Specification**

(1) I/O ports

Input:

Clk: global clock

pb\_in: 接在另一個button上，負責暫停。

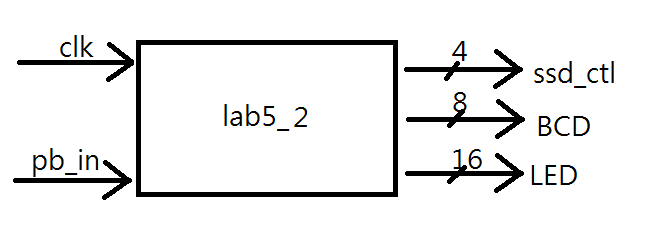
output

[7:0] BCD

[3:0] ssd\_ctl

[15:0] LED: 倒數到0的時候全部變0

(2) Block Diagram



**Design Implementation**

這一題大抵與上一題相同，最大的相異之處在於把input中的reset拿掉了。每個module的功用如下：

Freq\_div：普通的除頻器。藉由給予不同的上限值來產出不同頻率的clock。在這個lab中，產生Counter用的1Hz的clock和debounce要用的100MHz的clock。

Debounce：老師給的範例程式，負責把按下去的雜訊處理乾淨。

One\_pulse：老師給的範例程式，負責產生脈衝丟給FSM

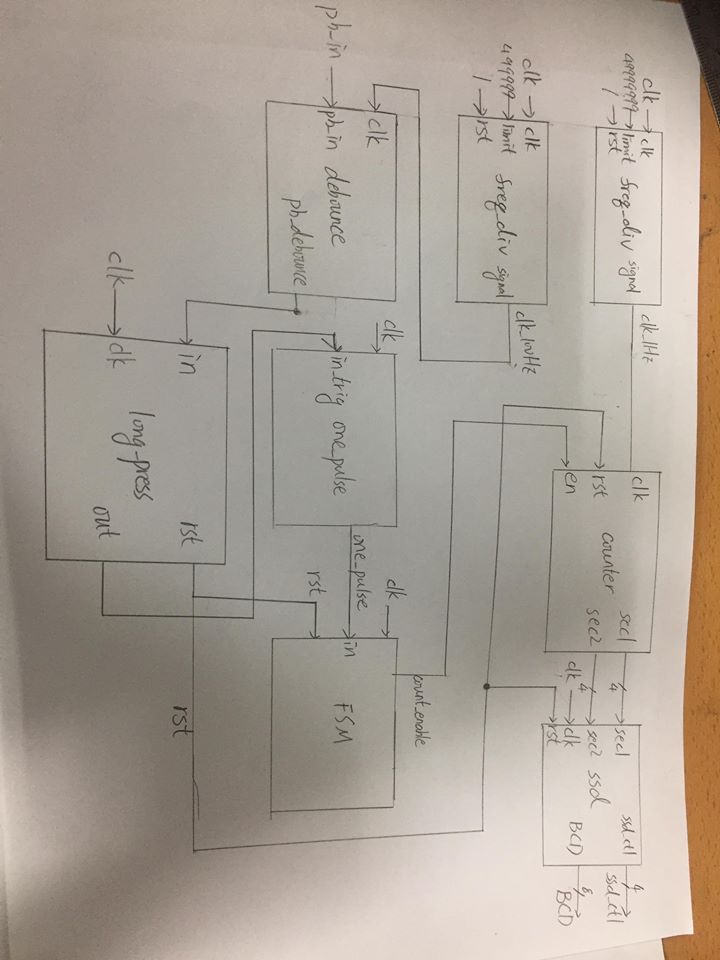
FSM：只有兩種state：倒數與暫停。轉換的依據是經由按壓按鈕的訊號經以上兩個module處理後所產生的脈衝。

Counter：非常普通的30秒下數器，但有把FSM的state接進來達成按一次按鈕以暫停的功能。

Ssd：負責把剩餘秒數顯示在七段顯示器上。

Long\_press：reset的產生器，同時也是一個上數器。Reset的預設值是0，只要經過debounce的信號為1，上數器就會一直上數，到一定值後就會把reset變成1。

Logic Diagram:

****

**Discussion**

　　寫prelab時遇到很多問題。首先，counter完全不倒數，後來發現原因是若倒數結束時把全部的暫態令成0要寫在FSM傳出來的freeze信號控制的if條件句下，而不能額外寫成一個組合邏輯。另外就是對top module的概念不熟悉導致配線錯誤，致使counter完全不聽freeze信號的指揮而暫停。以後再遇到這種需要寫很多module的電路，應該要把每個module獨立寫完，並寫一個top module做組裝。組裝時應注意要把所有input output都確實接到正確的信號，不能有所偏差。

後來嘗試寫了這個lab寫了兩個禮拜，我就對關於push\_button的一切徹底絕望了。

直到兩個月後的今天，我，變強了。於是我把之前寫的東西重新接過一遍，結果發現只要我把reset改成正緣觸發，一切就沒事了，原來如此。

**Conclusion**

從禮拜一弄到禮拜四 QQ

寫電路的順序：先把各個獨立的module寫好，再從top module把所有module組裝成一個電路。

不只key board，原來push button也要用正緣觸發的reset，難怪學長整個學期的lab都是posedge reset。

**References**

《Verilog硬體描述語言數位電路》，鄭信源，儒林圖書公司(2016)。

另外也要感謝熱心助人的李佳璋同學、陳永憲同學、丁文淵同學。