Lab 6

110060027 朱豐蔚

I. exp_1 (electronic clock)

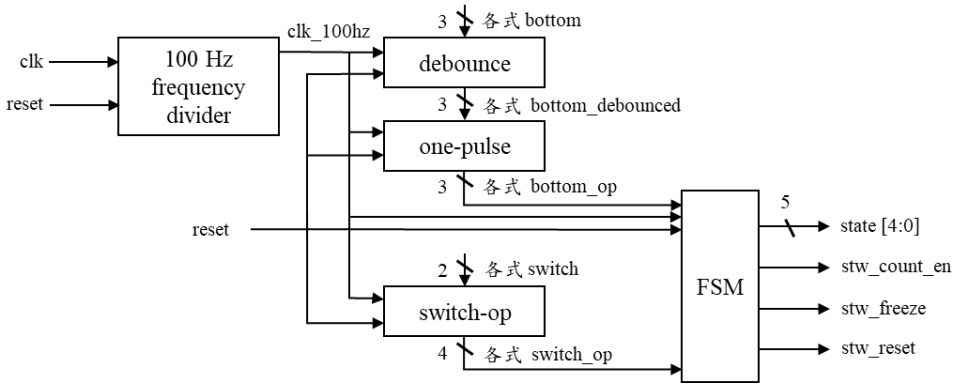
Design Specification

Input: buttom_left, buttom_mid, buttom_right, clk, reset, switch_alarm, switch_mode_left, switch_mode_right.

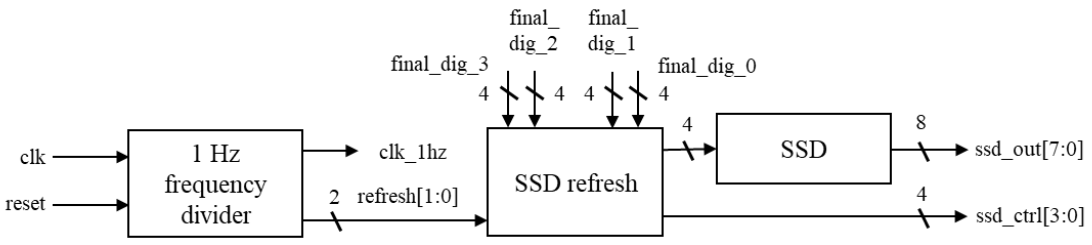
Output: stw_work, led_alarm[7:0], ssd_ctrl[3:0], ssd_out[7:0], led_display[2:0], led_set [3:0].

Design Implementation

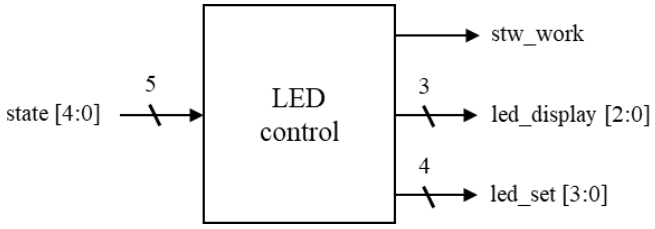
Block diagram:



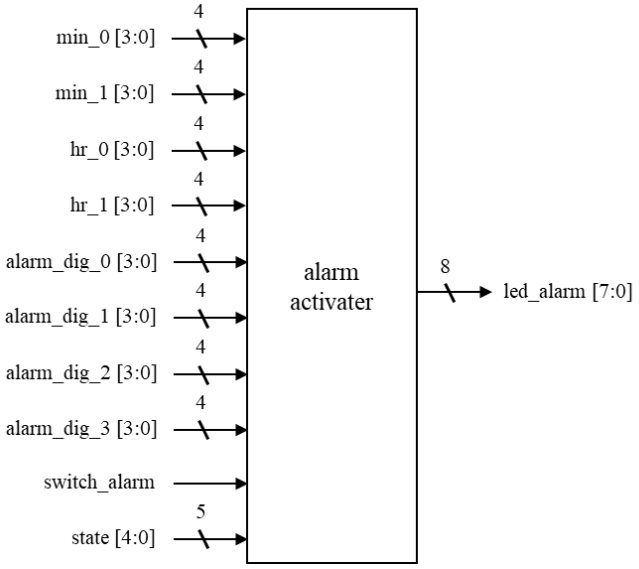
100 Hz frequency divider	將 1 億 hz 的 clk，透過設立 500K 的 counter 將其轉成 100 hz 的 clk_100hz 訊號，以供後續 debounce、one-pulse 使用
debounce	使用 windows 的設計處理雜訊，以防止按鈕失靈或誤觸
one-pulse	將 debounce 訊號轉成一 clk 週期的脈衝訊號，以提供給 FSM 使用
switch-op	將 switch 的 turn on、turn down 以類似 one-pulse 的延誤設計，將其轉成一 clk 週期的脈衝訊號，以提供給 FSM 使用
FSM	控制所有 state 彼此之間的切換，以輸出 state 狀態給後續進行各功能的切分，並輸出有關與控制 stopwatch 的訊號 stw_count_en、stw_freeze、stw_reset



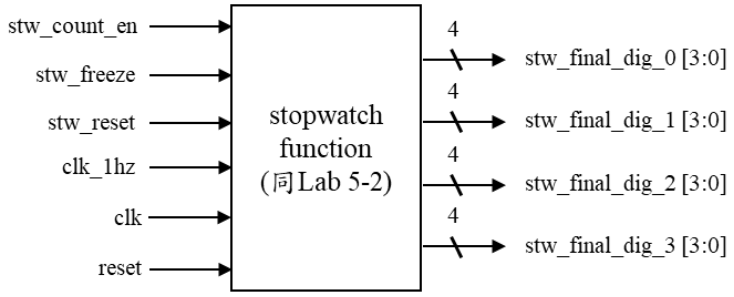
1 Hz frequency divider	將 1 億 hz 的 clk，透過設立 50M 的 counter 將其轉成 1hz 的 clk_1hz 訊號，並取其 counter 的第 17、16 位元作為 refresh [1:0]，以提供 SSD 四個數字的輪轉更新用
SSD refresh	透過 refresh 訊號，輪替輸出四個輸入數字與控制 SSD 顯示哪個位置的 ssd_ctrl 訊號
SSD	將 BCD 轉成七段顯示器的輸出，使其能輸出供肉眼辨別的羅馬數字



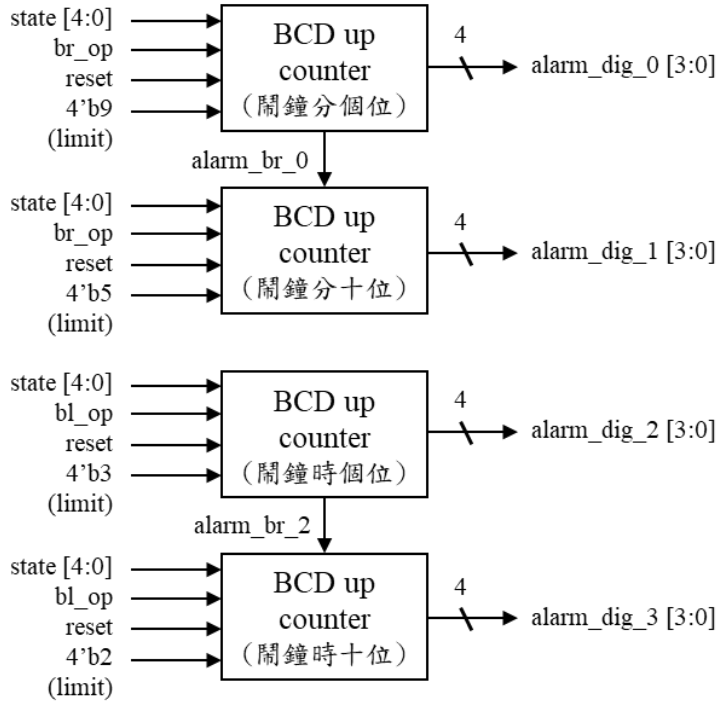
LED control	依據 state 的狀態，控制該亮那些燈(不包含鬧鐘的燈)，在設定模式下(state[4:3] == 2'b01)，就讓 led_set 根據設定的項目分別亮起；在顯示模式下(state[4:3] == 2'b00)，就讓 led_display 根據顯示的項目分別亮起；在 stopwatch 模式下(state[4:3] == 2'b10)，就讓 stw_work 亮起



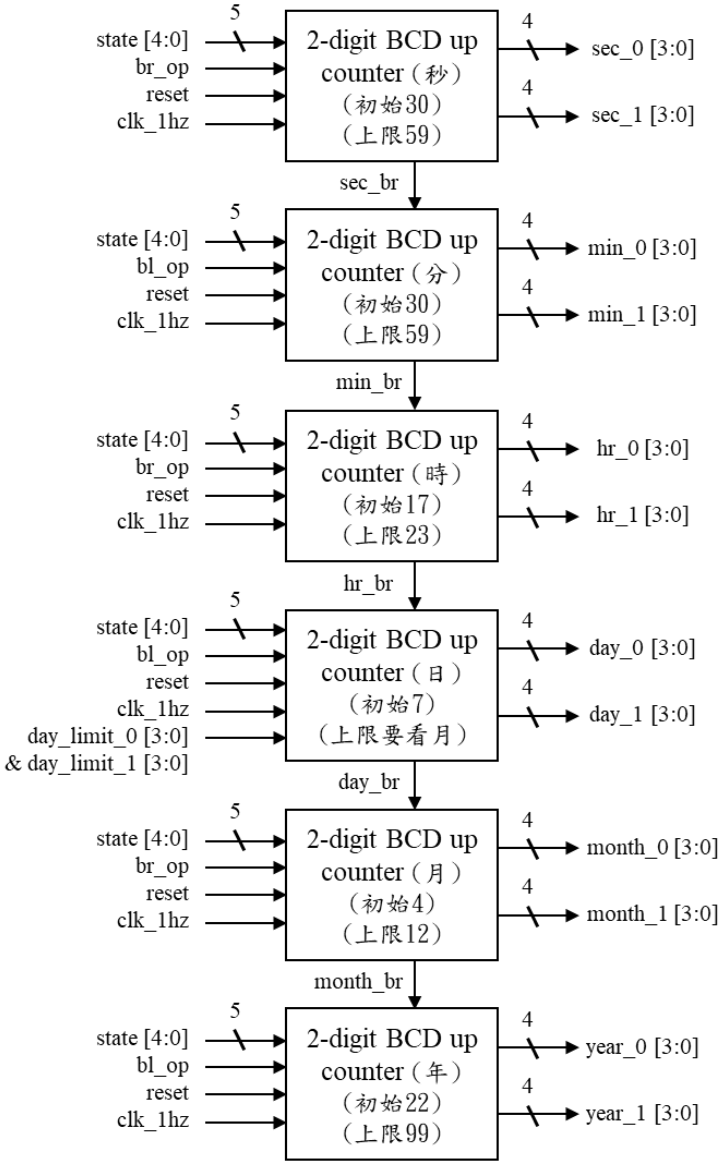
alarm activater	若小時的十位數、個位數與鬧鐘小時的十位數、個位數相同，且分鐘的十位數、個位數與鬧鐘分鐘的十位數、個位數相同，且 switch_alarm 是打開的狀態，就使鬧鐘的燈亮起 注意：在 stopwatch 模式下(state[4:3] == 2'b10)，鬧鐘功能要關閉



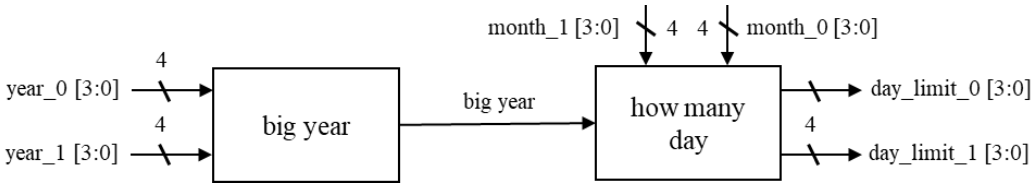
Stopwatch function (同 Lab 5-2)	同 Lab 5-2，設立 4 個 counter 處理分鐘、秒的十位數、個位數，並另外用 4 個 flip flops 處理 freeze 功能，以達到 stopwatch 的目的



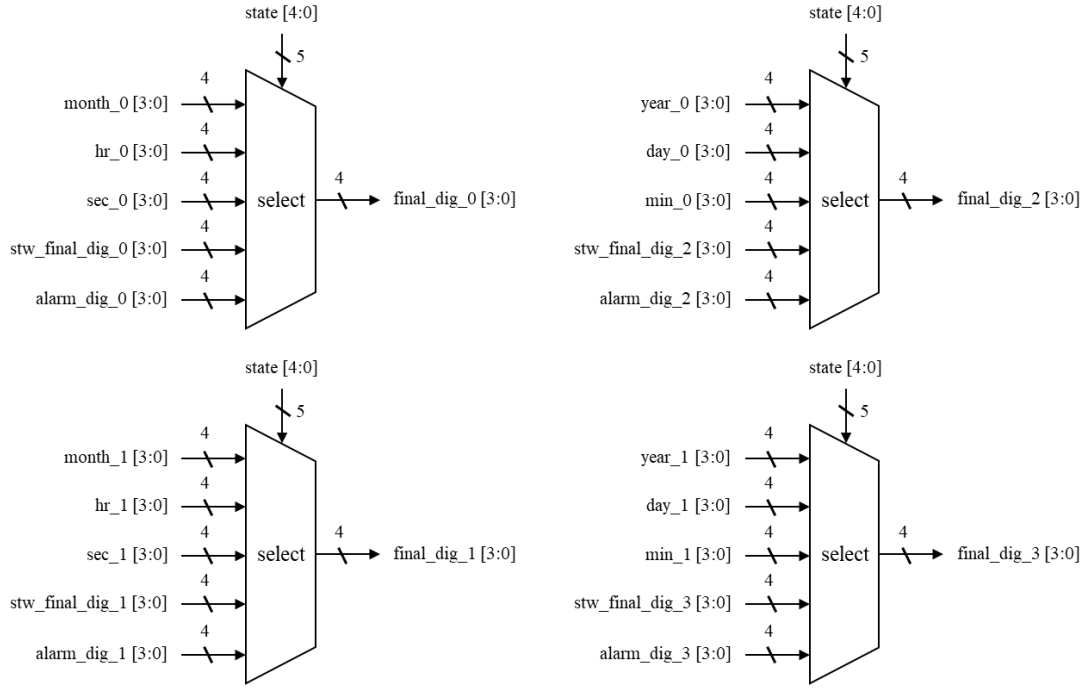
BCD up counter	在 counter 中，需要定義初始值、上限、enable 訊號、與進位訊號，當在鬧鐘的設定模式(state[4:0] == 5'b01000)下，counter 才會被 enable，此時能夠透過右邊按鍵去控制鬧鐘的分鐘數、用左邊按鍵去控制鬧鐘的小時數 注意：鬧鐘的初始值為 17：31，故要讓 4 位數字的初始值由上至下分別設定為 1、3、7、1。上限值如圖中所示



2-digit BCD up counter	類似上述 counter，但在此使用 2 digit 的版本，以方便進行初始值、上限值的設定，當在鬧鐘的顯示模式(state[4:3] == 2'b00)下，此時 clk 訊號設定為 clk_1hz，而在鬧鐘的設定模式(state[4:3] == 2'b01)下，此時 clk 訊號設定為右邊按鍵與左邊按鍵，即在設定模式下，時間是不會繼續數的，此點符合日常的使用習慣 注意：時鐘的初始值為 22 年 04 月 07 日 17 時 30 分 30 秒，故要讓 6 個 counter 的初始值由上至下分別設定為 30、30、17、07、04、22。上限值除了日期的上限會隨月份、閏年有所改變(由後續模組進行設定)，其餘皆與圖中所示相同

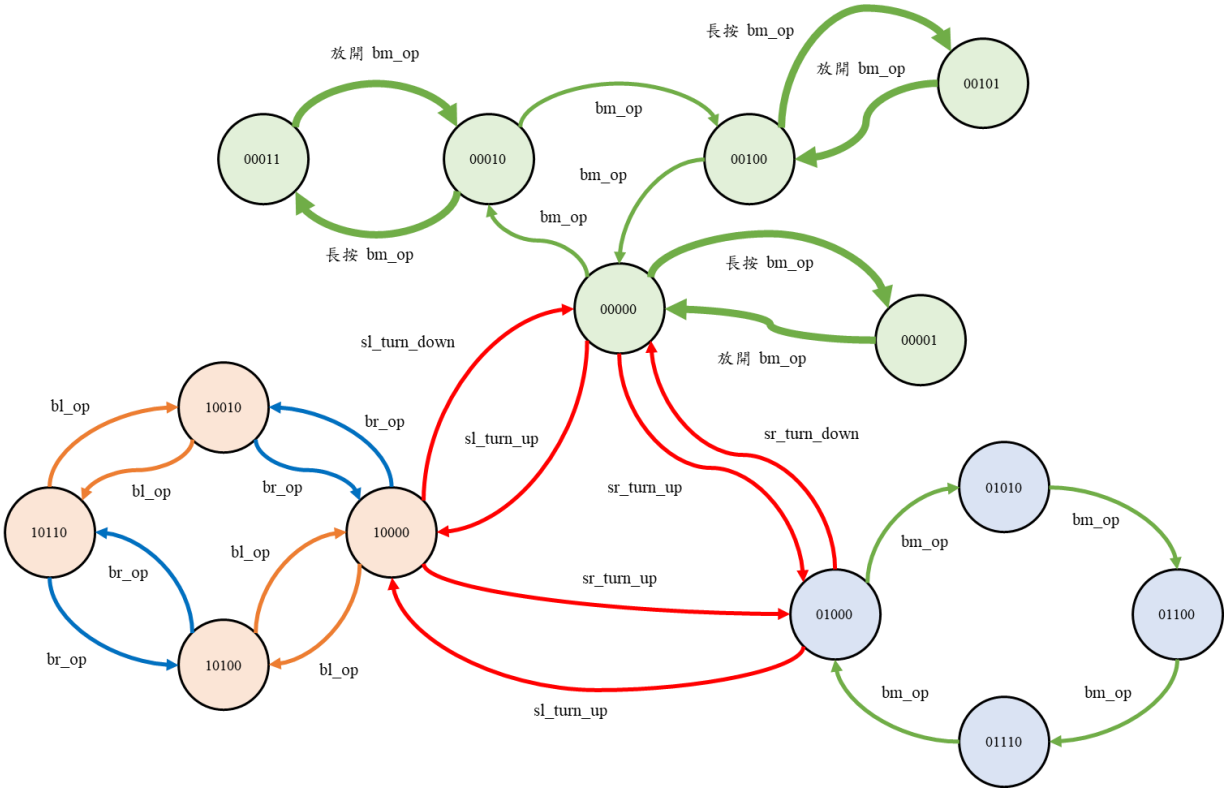


big year	觀察年的個位數與十位數的關係，若十位數的最後一位元(year_1[0])是 1，代表十位數為奇數，此時個位數為 2 或 6 (year_0[1:0] == 2'b10)即滿足閏年條件；若十位數的最後一位元(year_1[0])是 0，代表十位數為偶數，此時個位數為 0 或 4 或 8 (year_0[1:0] == 2'b00)即滿足閏年條件，在閏年時，使輸出 big_year = 1 注意：在本次 Lab 設計中，並未考慮逢百不閏、逢 400 又閏的情況
how many day	透過 case 與閏年與否，判斷當月有幾天，作為天數的上限控制



select	設計一個 5 選 1 的 select 模組，select 的訊號由 state 決定，當為顯示年月或設定年月模式時(state[4:0] == 5'b00000 or 5'b01010)，使最終輸出為年與月的 counter 訊號；當為顯示日時或設定日時模式時(state[4:0] == 5'b00010 or 5'b01110)，使最終輸出為日與時的 counter 訊號；當為顯示分秒或設定分秒模式時(state[4:0] == 5'b00100 or 5'b01100)，使最終輸出為分與秒的 counter 訊號；當為設定鬧鐘模式時(state[4:0] == 5'b01000)，使最終輸出為鬧鐘的 counter 訊號；當為 stopwatch 模式時(state[4:3] == 2'b10)，使最終輸出為 stopwatch 的 counter 訊號

Design flow:



考慮此題所需要的 state，如上圖設計，淺綠色背景的 state 為顯示模式，其中以三個 state(5'b00000、5'b00010、5'b00100)分別作為顯示年月、顯示日時、顯示分秒的主 state，彼此透過中央按鍵切換，而我另外創立三個副 state(5'b00001、5'b00011、5'b00101)作為各自的長按顯示鬧鐘功能。在長按顯示鬧鐘、放開又回到原狀態的功能上我下了很大的心思，我設想了兩種方式，一是如上述再多設立三個 state，二是設立 record 變數去紀錄長按前的模式，使放開後能透過 if 來決定 state 的變化方向。在本次 Lab 中，我並未嘗試過第二種寫法，有待未來嘗試。

回到 FSM 上，淺藍色背景的 state 為設定模式，其中以四個 state(5'b01000、5'b01010、5'b01100、5'b01110)分別作為設定鬧鐘、設定年月、設定日時、設定分秒的 state，彼此透過中央按鍵切換。

最後，淺橘色背景的 state 為 stopwatch 模式，同 Lab 5-2 的 FSM，此功能需要 4 個 state 與右邊按鍵、左邊按鍵鍵來分別控制是否數、是否 lap 的切換，而長按左邊按鍵要有 reset 的功能，如前述 module 中 FSM 的描述，我透過 stw_count_en、stw_freeze、stw_reset 的設計來完成 stopwatch 功能。

而顯示模式、設定模式、stopwatch 模式彼此的切換則是使用 switch_op(分為 turn on 與 turn down)的訊號來完成。

而除了 FSM 以外的其餘設計細節，皆於上述 block diagram 的 module 介紹中提及。

I/O pins assignment:

I/O	clk	reset	bottom_left	bottom_right	bottom_mid	switch_alarm	switch_mode_left	switch_mode_right
LOC	W5	T1	W19	T17	U18	R2	V16	V17

I/O	ssd_ctrl[3]	ssd_ctrl[2]	ssd_ctrl[1]	ssd_ctrl[0]
LOC	W4	V4	U4	U2

I/O	s_o[7]	s_o[6]	s_o[5]	s_o[4]	s_o[3]	s_o[2]	s_o[1]	s_o[0]
LOC	W7	W6	U8	V8	U5	V5	U7	V7

I/O	led_set[3]	led_set[2]	led_set[1]	led_set[0]	led_display[2]	led_display[1]	led_display[0]	led_alarm[0]
LOC	L1	P1	N3	P3	U3	W3	V3	V13

I/O	led_alarm[1]	led_alarm[2]	led_alarm[3]	led_alarm[4]	led_alarm[5]	led_alarm[6]	led_alarm[7]	stw_work
LOC	V14	U14	U15	W18	V19	U19	E19	U16

Discussion:

這次的 Lab 對我來說是一次很大的挑戰，也是第一次讓我們著手進行那麼大量的模組牽線與撰寫，除了考驗我們對每個模組的使用掌握度外，更考驗著我們對於那麼多模組彼此的串聯使否邏輯清晰與了解透徹，才能達成這次的實驗。

在設計中，有兩個點是我思考最久的，第一是 2 digit counter 的初始值、上限值撰寫問題、第二是閏年與天數上限的問題。

在原先，其實我是使用 2 個單位元的 BCD counter 作為 2 digit counter 的撰寫方式，但由於在日與月的撰寫中，必須考慮到沒有 0 月或 0 日存在，故需要在進位後的值上做考慮，除此之外，更要考量 reset 後的初始值，此兩者彼此更可能會互相干擾，於是在後期，我便改為直接設立 2 digit counter 來統一化各式 counter 以解決 counter 互相不統一初始值、上限值、進位後值的問題，而這樣的寫法也幫助了我在進行進位的寫法上更為簡潔與快速。這次的經驗也讓我了解到了統整化的重要性，透過小模組更進一步的推疊，就能夠形成大模組，以應付更複雜、更多樣的各種情況。

而關於閏年與天數上限的問題，我在如何判斷閏年上設想了許久，在原先，我本是打算使用 case 硬處理各式年份對應到的閏年情況，但後來，透過思考後，我認為 4 年一閏的規律應該會與 2 的平方有關係，才進而發現十位數與各位數與閏年之間的關係，進而解決掉閏年的問題。這次經驗也讓我思考，若今日遇到不是 4 的規律的問題，例如 3 次、5 次，是否也有通解，但我認為當不是 2 的次方時，找到其規律的可能性就極為低。

Conclusion

我認為這次 Lab 最大的收穫，是有了在很複雜的情況、很多的模組下，能夠控制並串聯各個模組的經驗，我想未來一定會遇到更多同樣複雜，或更複雜的情況，但只要按部就班，先從 FSM 下手，再分別進行各項功能的撰寫，就能省去許多 debug 的時間與問題，也能更有計畫性的完成 project。

此外，在進行撰寫時所遇到的各種問題，如初始值、上限值，或是閏年的判斷，也都是在訓練我們遇到問題時的解決思考靈活性，這點是最為重要的，在遇到困難時，從各個方面下手，尋找解決方式，我喜歡這樣的挑戰與嘗試，也熱愛於其中。

感謝老師與助教安排這次的 Lab，讓我能在經驗與創意上都有明顯的突破，並在完成 Lab 的剎那收穫滿滿的成就感與感動。

References

FSM 編撰方式: (from 第六週上課講義)
Freeze 功能編撰方式: (from 第五週上課講義)