Lab07_Electronic_Clock_II_report

106061146 陳兆廷

Experiments:

1. 題目要求一個碼表，可以計時、凍結當前 lap 時間，因此需要一個有 enable 功能且有分、秒的 clock、一個 FSM 來計算當前的 state、一個除頻器，以及一個大 module 和 SSD 顯示功能。(圖一)

   在 clock module 裡面，需要有輸出的各位分、秒、輸入的 en、clk、rst，而內容很簡單，就如同上次的時鐘內容，在 enable 時開始計時即可(圖二)。

   在 FSM 裡面，輸入要利用兩個按鈕(in1、in2)來控制開始，還有一個 clk，輸出值則有判斷是否要計時的 count_enable(接上 clock 的 en)、凍結畫面的 freeze_enable、是否重製的 reset_enable(接上所有的 rst)，而其所有的 state 有如圖(四)四種，state diagram 如圖(五)。

   最後用大的 display module 將各個小 module 合併，輸入值為 in1、in2 兩個 botton 以及 clk，輸出值為控制 SSD 的 ssd_ctl 還有一個 8bit 的 SSD，就大功告成。

圖(一)

```
output [`BCD_BIT_WIDTH-1:0]sec0,
output [`BCD_BIT_WIDTH-1:0]sec1,
output [`BCD_BIT_WIDTH-1:0]min0,
output [`BCD_BIT_WIDTH-1:0]min1,
input en,
input clk,
input rst
```

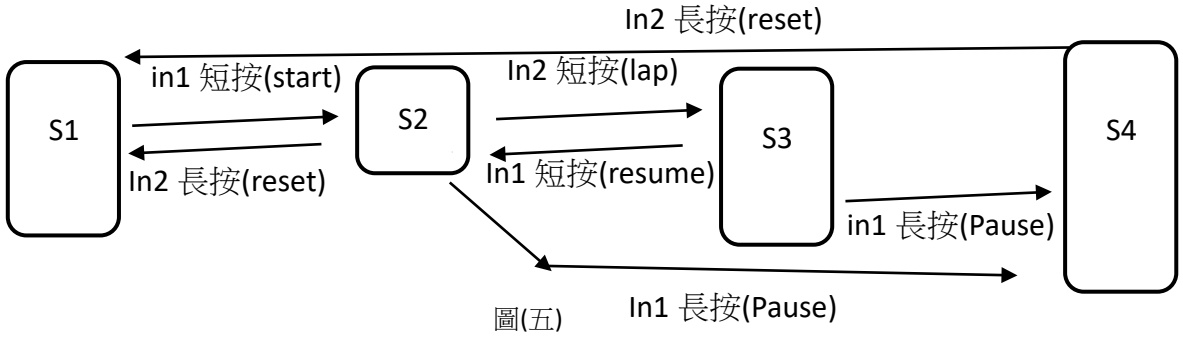
圖(二)

```
module FSM2(
    output count_enable,
    output freeze_enable,
    output reset_enable,
    input in1,
    input clk,
    input in2
    );
```

圖(三)

```
`define STAT_DEF 3'b000
`define STAT_COUNT 3'b001
`define STAT_FREEZE 3'b010
`define STAT_STOP 3'b011
```

圖(四)



圖(五)

   這個作業遇到的難題是 FSM 的撰寫，在 lap 計算時如何讓時鐘凍結在那個畫面且又繼續計時。約花了我 2 個小時。

2. 這題和 bonus 是一樣的，因此我合在一起寫。

   題目要求以一個 I/O switch 控制 setting，可以自訂時間的倒數計時器(至 23:59)以及碼表兩種，可以用 3 個按鈕來控制。

   因此我對第一題稍作修改，在 clock 裡面新增 hr0、hr1、led，並新增一個 setting，當 setting == 1 時，用兩個輸入值 min_enable、hr_enable 設定初始值，當進入倒數計時模式、開始計時前可以設定初始的時間，且在倒數至 0 時 0 分 0 秒時 led 會變成 1，如圖(六)，接著再依照 setting 來控制正數或是

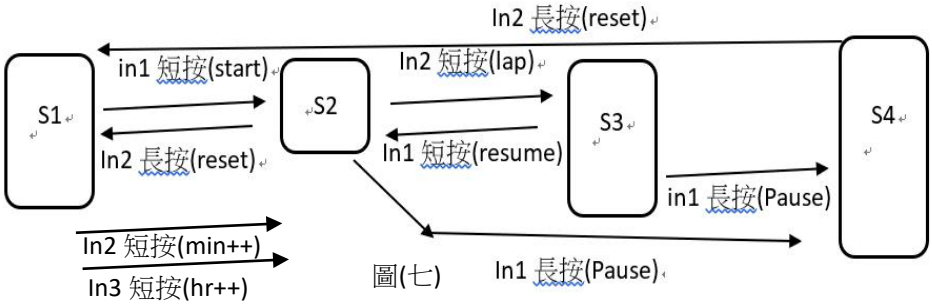
倒數計時，分別寫出碼表及倒數計時器。

在 FSM 裡，我新增幾條線和幾個輸出，在 reset 時(S1)，當 setting 是在倒數計時時，in2 及 in3 分別代表 min_enable 及 hr_enable，傳入 clock 設置初始值，而其餘便如同第一題，因為在 clock 裡已經寫好依照 setting 來正數或是倒數，只要 count_enable == 1 就會動作。(圖七)

最後在 display module 裡面，只要新增輸入 in1、in2、in3、setting，新增輸出 led，並控制當時間小於一小時時顯示分及秒，時間大於一小時時顯示時及分，再輸入至 SSD 輸出即可。

```
module clock(
    output [`BCD_BIT_WIDTH-1:0]sec0,
    output [`BCD_BIT_WIDTH-1:0]sec1,
    output [`BCD_BIT_WIDTH-1:0]min0,
    output [`BCD_BIT_WIDTH-1:0]min1,
    output [`BCD_BIT_WIDTH-1:0]hr0,
    output [`BCD_BIT_WIDTH-1:0]hr1,
    output led,
    input setting,
    input in3,
    input in2,
    input en,
    input clk,
    input rst
);
```

圖(六)



圖(七)

這題在構想時，一度多用了幾個 state，原本想在 FSM 控制正數或是倒數，所以多了 state_stopwatch 和 state_timer，又多一個 state_setting，後來搞了很久，發現在 clock 裡面控制兩種方法最為簡單，也可以直接及時輸出時、分、秒的各位數字，再修正了許多 Bug，像是無法重新設定等，耗費約三個小時。

這次的 lab 又讓我再次練習各種 state 及 FSM 的應用。