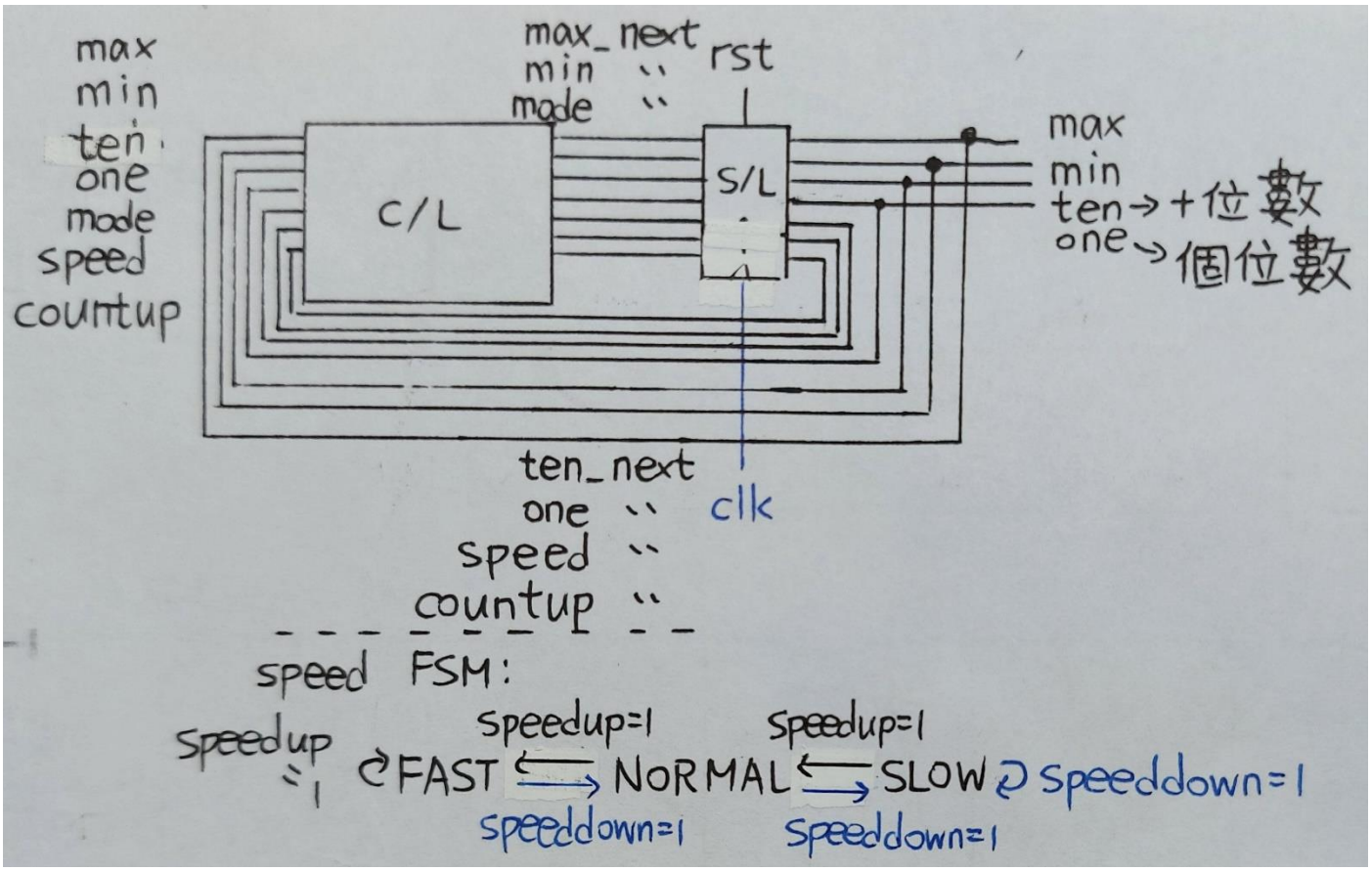
Lab 4

學號: 109062318 姓名: 簡弘哲

1. 實作過程

Lab4-1:



Ten 負責記錄十位數，one 則記錄個位數(實際上可以用一個長度為 2 的陣列來記錄會更為簡潔)，要留意數到 boundary number(0,9)時十位數的變換，下圖中的 code 有特判 boundary case(00,99)，如果數到極限的話就維持住。

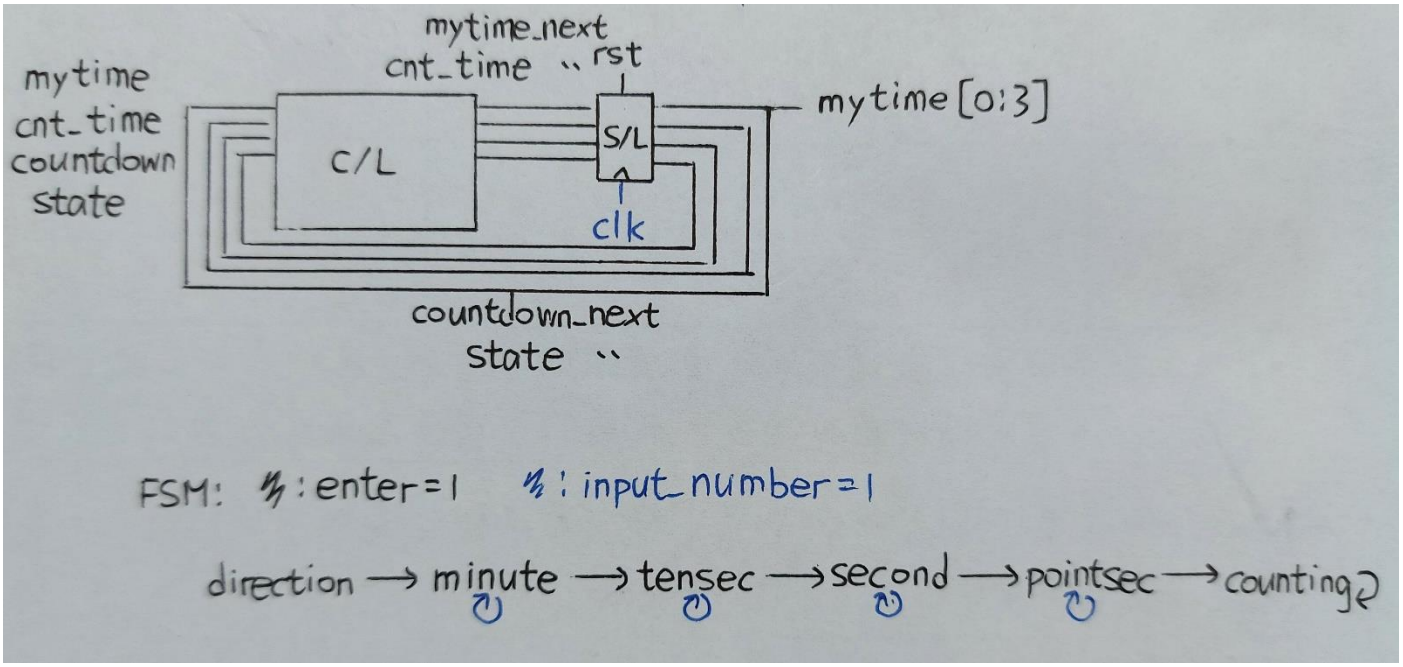
```
if(countup) begin
    if(ten==9 && one==9) begin   //99
        max_next=1;min_next=0;
        ten_next=9;
        one_next=9;
    end else begin  //normal increment
        max_next=0;min_next=0;
        if(one==9) begin
            one_next=0;
            ten_next=ten+1;
        end else begin
            one_next=one+1;
            ten_next=ten;
        end
    end
end else begin
    if(ten==0 && one==0) begin   //00
        min_next=1;max_next=0;
        ten_next=0;
        one_next=0;
    end else begin  //normal decrement
        min_next=0;max_next=0;
        if(one==0) begin
            one_next=9;
            ten_next=ten-1;
        end else begin
            one_next=one-1;
            ten_next=ten;
        end
    end
```

後來發現 max,min 不需要設 max_next,min_next 這兩個變數，分別用一句 assign 並檢查目前的十位數及個位數就可判斷是否需要亮起。

Mode 記錄現在是 start 還是 pause，在數秒的時候需要檢查現在的 mode 有沒有等於 start。

我是利用 fsm 去控制 speed 的速度並分為 3 個 state，分別是 FAST,NORMAL,SLOW。(事實上也可以利用 if 判斷以及一些對 speed+1、-1 的動作就可達成目的，會更為簡單，只不過當時設計時沒有想到這點)

Lab4-2:



Mytime 是目前用來顯示在 7-segment 的時間

Cnt_time 用來記錄碼表需要數到的終點，如果是 countdown 則 cnt_time 是 0000，否則的話就是使用者所設定的時間

State(3bits)用來記錄目前所在的狀態(總共 6 個)，分別是

設定 direction(direction),

設定 minute(minute),

設定 10 秒(tensec),

設定 1 秒(second),

設定 0.1 秒(pointsec)

以及數秒(counting)

圖中的黑色箭頭代表按下 enter 按鈕，而藍色箭頭則代表按下 input_number 按鈕(在 direction state 應補上指向自己的箭頭，代表按下 countdown 按鈕切換 start/pause)

數秒部分的實作:

藉由比較 mytime(現在時間)跟 cnt_time(終點)來判斷是否數到終點，數秒的動作是用巢狀 if 並檢查 boundary case(0,9)，array index 0 指的是分鐘、1 是 10 秒、2 是 1 秒、3 是 0.1 秒。

```
if(mytime[0]!=cnt_time[0] || mytime[1]!=cnt_time[1] || mytime[2]!=cnt_time[2] || mytime[3]!=cnt_time[3]) begin
    if(countdown) begin
        if(mytime[0]==0 && mytime[1]==0 && mytime[2]==0 && mytime[3]==0) begin
            mytime_next[0]=0;
            mytime_next[1]=0;
            mytime_next[2]=0;
            mytime_next[3]=0;
        end else begin
            if(mytime[3]==0) begin   //ex.1:11.0=>1:10.9
                mytime_next[3]=9;
                if(mytime[2]==0) begin //ex.1:10.0=>1:09.9
                    mytime_next[2]=9;
                    if(mytime[1]==0) begin //ex.1:00.0=>0:59.9
                        mytime_next[0]=0;
                        mytime_next[1]=5;
                    end else begin  //ex.0:10.0=>0:09.9
                        mytime_next[1]=mytime[1]-1;
                    end
                end else begin  //ex.0:05.0=>0:04.9
                    mytime_next[2]=mytime[2]-1;
                end
            end else begin  //ex.0:00.5=>0:00.4
                mytime_next[3]=mytime[3]-1;
            end
        end
```

、

Bonus:

```
module clock_divider_100ms #(parameter n = 25'd1000_0000)(
    input clk,
    output clk_div
);
    reg [24:0] num = 0;
    wire [24:0] next_num;

    always @(posedge clk) begin
        if(next_num>=n+1) begin
            num<=0;
        end else begin
            num<=next_num;
        end
    end

    assign next_num = num + 1;
    assign clk_div = (num == n) ? 1 : 0;
endmodule
```

已知板子的 clk 1s 會 100M 上下，那經過 0.1s 就已經 10M 上下了，所以 counter 就數 10M，數到 10M 的時候就代表已經過了 0.1s，超過 10M 就把 num 重設為 0。

在 num==10M 的時候將 clk_div 輸出 1，num!=10M 的時候就維持 0。

因為 num=10M 只會維持一個 clk cycle，clk_div 也是，所以我的 clock divider 有把 onepulse 的效果一同做進去。

2. 學到的東西與遇到的困難

學到的東西:

1.最好不要有 clk,rst 之外的 posedge 信號例如 en,dir

2.如何設計 0.1s 精確的 clock divider

3.為了顯示時間而使用了陣列與 for loop，以及了解陣列[MSB:LSB]跟[LSB:MSB]的差異

遇到的困難:

1.

在寫 4-2 時把獨立的功能分成很多 always block，但這些 always block 的=左手邊有共同的變數導致 multi net driven，後來把=左手邊有相同變數的 always block 合成一個大的 block 就解決了。

2.

在 4-2 每一個 state 都設一個用來顯示 7-segment 的陣列，導致需要管理每一個時間的狀態而難以 debug。

顯示 7-segment 設 1 個(mytime)

Number setting state 設 2 個(goal,set_time)

counting state 設 2 個(countup_time,countdown_time)

3.

clk 的速度沒調好導致按下按鈕完全沒反應或是不靈敏，以及忘記在 combinational block 裡忘了寫

if(…_onepulse)導致按鈕不會 work

4.

4-2 一開始不知道如何往上數，後來想到要 maintain 一個變數記錄數到的終點，如果是往下數那終點就是 0，反之就是使用者設定的數字，在進入 counting state 前設定好。如下面兩圖:(mytime 為現在時間,cnt_time_next 則是要數到的終點)

```
end else if(state==POINTSEC) begin
    if(input_number_1pulse) begin
        if(mytime[3]==9) begin   //pointsec has reach its max
            mytime_next[3] = 0;
            cnt_time_next[3] = 0;
        end else begin
            mytime_next[3] = mytime[3]+1;
            cnt_time_next[3] = (countdown) ? 0 : mytime[3]+1;
        end
    end else if(enter_1pulse) begin
        state_next = COUNTING;
        for(i=0;i<4;i=i+1) begin
            mytime_next[i] = (countdown) ? mytime[i] : 0;
        end
    end
end else if(state==COUNTING) begin
```

```
end else if(state==MINUTE) begin
    if(input_number_1pulse) begin
        if(mytime[0]==1) begin //minute has reach its max 1
            mytime_next[0] = 0;
            cnt_time_next[0] = 0;   //set the goal
        end else begin
            mytime_next[0] = mytime[0] + 1;
            cnt_time_next[0] = (countdown) ? 0 : mytime[0]+1;  //set the goal
        end
    end else if(enter_1pulse) begin
        state_next = TENSEC;
    end
end else if(state==TENSEC) begin
    if(input_number_1pulse) begin
        if(mytime[1]==5) begin //tensec has reach its max 5
            mytime_next[1] = 0;
            cnt_time_next[1] = 0;
        end else begin
            mytime_next[1] = mytime[1] + 1;
            cnt_time_next[1] = (countdown) ? 0 : mytime[1]+1;
        end
    end else if(enter_1pulse) begin
        state_next = SECOND;
    end
end else if(state==SECOND) begin
    if(input_number_1pulse) begin
        if(mytime[2]==9) begin //second has reach its max 9
            mytime_next[2] = 0;
            cnt_time_next[2] = 0;
```

3. 想對老師或助教說的話

覺得 lab 一次比一次難、花更多心力與時間