Lab 1

學號: 109062202 姓名:陳禹辰

1. 實作過程

Lab3_1：

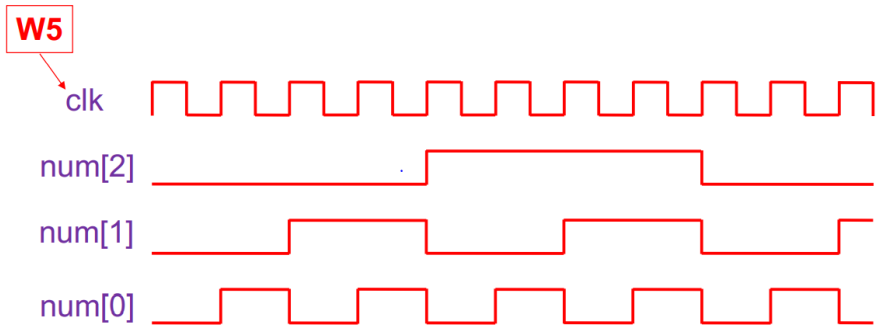
在看完題目之後其實一開始不是很懂 clock_divider 是在做什麼怎麼運作的，後來去翻了老師的上課講義才知道這個東西是在幹嘛的，然後就照著講義給的 template 去打，不同的是講義內的是 3bit，而 lab 內需要的則是有好幾種，所以就設一個 parameter，就可以在呼叫時更改。

```
module clock_divider #(parameter n = 25) (
    input clk,
    output clk_div
);
    reg[n-1:0] num = 0;
    wire [n-1:0] next_num;

    always @(posedge clk) begin
        num = next_num;
    end

    assign next_num = num + 1;
    assign clk_div = num[n-1];

endmodule
```

接著就開始做 lab3_1，我在這部分沒有遇到什麼太大的困難，主要要解決的就是會有兩種閃爍的速度，在 speed = 0 時會比較快而 speed = 1 時則會較慢。我是在額外開兩個 always block 去跑這兩個額外的速度，然後在 posedge 時判斷 speed 是 0 還是 1 而決定要呼叫哪個。

```
always @(*) begin
    if(clk_24 == 1) clk_next_24 = 16'b1111111111111111;
    else clk_next_24 = 16'b0000000000000000;
end
always @(*) begin
    if(clk_27 == 1) clk_next_27 = 16'b1111111111111111;
    else clk_next_27 = 16'b0000000000000000;
end
```

這邊主要要考慮的點是要怎麼判斷怎麼閃爍，因為是要在(100MHz/2^27)下每個 clock 閃爍，所以我就讓他在(100MHz/2^28)下如果 clk 是 1 就亮是 0 就暗，就會符合要求了，但我寫 report 的時候才想到，好像其實判斷每個 posedge 就可以實現了…。



Lab3_2：

3_2 的部分我就遇到蠻多困難的。我先是看完題目後設計我的 FSM，我是設計有四個 state，分別是 rst 用的全亮的 state、flashing state、shifting state、expanding state 這四種，然後完成之後我就直接去跑合成了結果在 generate bitstream 的時候就出錯了，於是我就先把 state 裡面的判斷全部拿掉直接 assign 燈藥量哪幾顆，結果就發現不能直接把 led(output)放在 combinational block 裡面，所以我就額外設了 led_state 跟 led_next 來跑，然後在 sequential block 裡面再把 led_state assign 到 led 上，就發現能成功了。

還有一些問題也造成我在 generate bitstream 的時候造成錯誤，if else 內的東西要對稱或是不能在等號的左右兩邊出現一樣的東西，然後我最後找最久的一個 bug 是我沒有更新到 project 內的 constriants 所以找不到 dir 這個 input 要對應的東西。但也算是讓我更了解在打 verilog 時該注意的東西。

除了 generate bitstream 會失敗之外，我也有遇到一些寫出來之後沒有符合題意的問題，一個是 expanding 該如何操作，這部分我是詢問了朋友的意見所以才打成這樣，我覺得還蠻厲害的，至少我當初是想不到可以這樣寫。我寫 Expanding 的方法就是在 dir = 0 時要向外擴張，所以就把中間的往外移然後在中間補 2 個 1，dir = 1 時則是向內縮減，所以就把中間的兩個 bit 移除掉把兩邊併起來然後在最左跟最右補上 0。

```
if (dir == 0) begin
    led_next[31:16] = {led_state[30:24], 2'b11, led_state[23:17]};
    next_state = expanding;
end else begin
    led_next[31:16] = {1'b0, led_state[31:25], led_state[22:16], 1'b0};
    next_state = expanding;
end
```

Shifting 則是在 dir = 0 時向右移在 dir = 1 時向左移，但是我在 shifting 遇到的問題就是要讓燈不會在 shift 後被吃掉，我處理這個問題的方法是在 led_state 的左右都開一個 16bit 的來儲存移過去兩邊的燈，然後要在 state change 跟 rst 的時候要記得把它歸零，不然可能在下一次進 shifting 時就會有可能不止八個燈。

```
if (led_state[31:16] != 16'b0000000000000000) begin
    if(dir == 0) begin
        led_next = led_state >> 1;
        next_state = shifting;
    end else begin
        led_next = led_state << 1;
        next_state = shifting;
    end
end else begin
    led_next = {16'b0000000000000000, 16'b0000000110000000, 16'b0000000000000000};
    next_state = expanding;
end
```

2. 學到的東西與遇到的困難

遇到的困難在上面都打得差不多了，主要是這次是第一次要燒到板子上，然後每次跑完 synthesis 跟 implementation 結果在最後產生 bitstream 都出錯就很讓人崩潰，我大概重複了好幾十次這個過

程，還多開了一個檔案來試，每次多加一點東西，來確認到底問題出在哪邊，後來發現這樣子寫 code 也比較方便確認哪邊會出問題之類的，其實還蠻方便的，我還問了一堆朋友想要幫忙 debug，結果後來都是發現自己太白癡像是 constraints 沒有更新到之類的。

3. 想對老師或助教說的話

謝謝助教辛苦幫忙 demo 還有老師辛苦上課。

附上笑話：

什麼東西可以攔截電子？

.

.

.

.

.

紅橙黃綠(藍靛紫)