EECS 2070 02 Digital Design Labs 2019

Lab 3

學號：107062107　姓名：王領崧

1.前言

第一次實作 FPGA 板的 lab，分成四個部分，從最基礎的調頻器開始做起，接著是單一個 LED 燈的左右移動，後面加入另一組 3 個 LED 燈以不同頻率各自移動，最後則是當一個 LED 燈接觸到 3 個的時候，會產生碰撞換方向的效果。

2.實作過程

clock_divider

主要做法是利用 counter 方式，隨著原始的 posedge clk 進行相加，然後產生出的新的 clk 數值則是取自於 next_num 的 MSB。因為會需要兩種不同頻率，所以宣告 parameter 賦予 n 值，這樣後面使用 module 時，可以用#的方式，彈性的調整 n 值。

```
module clock_divider(clk, clk_div);
    parameter n = 26;
    input clk;
    output clk_div;

    reg [n-1:0] num;
    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
```

lab03_1

第一個作業為簡單的跑馬燈，分成兩個 always block 下去實作。首先利用調頻器弄出一個新的頻率 clk_div，接著當 posedge clk_div，開始進行各種參數的判斷，如果 rst==1，則進行初始化，將 LED 燈亮為最左邊的，當 enable 的時候，led 就會等於 next_led，反之則 hold 住原本的值，類似 DFF 的概念。而另一個 always block 則是進行 combantional logic，依據不同的 dir，來決定 next_led 是現在 led 的左移還是右移。

```
clock_divider cd0(.clk(clk), .clk_div(clk_div));

always @(posedge clk_div) begin
    if (rst == 1) begin
        led = {1'b1, 15'b0};
    end
    else begin
        if (en == 1) begin
            led = next_led;
        end
        else begin
            led = led;
        end
    end
end
```

```
always @(*) begin
    if (dir == 1) begin
        next_led = {led[14:0], led[15]};
    end
    else begin
        next_led = {led[0], led[15:1]};
    end
end
```

lab03_2

lab3_2 分成 4 個 always block 以及 4 條 led，前面兩個 always block 是兩種不同 posedge clk trigger，一個是/2^23，一個是/2^26，而我把兩組 led 分成兩個不同頻率下去運算，因此一個 always block 會有兩條 led，裡頭參數判斷與 3_1 的模式大同小異，只有在 disable 的部分，要根據 dir，給予目前 led 的位置。第三個 always block 為 combinational logic 的部分，依據判斷來決定左移還是右移，只不過同組但不同頻率的 led，是依照當下在板子上跑的 led 來決定，而不是依上一個 cycle 來 shift。最後一個 block 則是最終 led 的產出，當在 enable 的時候，根據 dir 的不同，將兩組 led 做 or 的邏輯運算。

```
always @(posedge clk_div23) begin
    if (rst == 1) begin
        led1_23 = {1'b1, 15'b0};
        led3_23 = {3'b111, 13'b0};
    end
    else begin
    if (en == 1) begin
        led1_23 = next_led1_23;
        led3_23 = next_led3_23;
    end
    else begin
        led1_23 = (dir == 1) ? led1_26 : led1_23;
        led3_23 = (dir == 1) ? led3_23 : led3_26;
    end
    end
end

always @(posedge clk_div26) begin
    if (rst == 1) begin
        led3_26 = {3'b111, 13'b0};
        led1_26 = {1'b1, 15'b0};
    end
    else begin
        if (en == 1) begin
            led3_26 = next_led3_26;
            led1_26 = next_led1_26;
        end
        else begin
            led3_26 = (dir == 1) ? led3_23 : led3_26;
            led1_26 = (dir == 1) ? led1_26 : led1_23;
        end
    end
end
```

```
always @(*) begin
    if (dir == 1) begin
        next_led1_23 = {led1_26[14:0], led1_26[15]};
        next_led3_26 = {led3_23[14:0], led3_23[15]};
        next_led1_26 = {led1_26[14:0], led1_26[15]};
        next_led3_23 = {led3_23[14:0], led3_23[15]};
    end
    else begin
        next_led1_23 = {led1_23[0], led1_23[15:1]};
        next_led3_26 = {led3_26[0], led3_26[15:1]};
        next_led1_26 = {led1_23[0], led1_23[15:1]};
        next_led3_23 = {led3_26[0], led3_26[15:1]};
    end
end

always @(posedge clk) begin
    if (rst == 1) begin
        led = {3'b111, 13'b0};
    end
    else begin
        if (en == 1) begin
            led = (dir == 1) ? led1_26 | led3_23 : led3_26 | led1_23;
        end
        else begin
            led = led;
        end
    end
end
```

lab03_3

lab3_3 分成 4 個 always block 以及 3 條 led，除了基本一個燈(led1)與三個燈的 led(led3)，多了一個三燈中間的判斷 LED(led3_middle)。三燈與判斷的 LED 就是利用基本的跑馬燈模式，一個 always block 當 DFF，一個用來做 shift，因為都是同一個方向，實作起來更加容易。碰撞的部份，則是先利用 led1 和 led3 做 AND，如果他們有碰撞交疊到，那麼結果就不為 0，接著再與 led3_middle 做 AND 判斷，如果不為 0 的話，代表碰撞點在中間，那就依照 dir 來左移或右移兩格，如果為 0，代表只是碰撞到兩邊的話，就只要左移或右移一格，並且把 dir=~dir，如果 led1 和 led3 AND 結果為 0 的話，那麼 led1 就直接等於 next_led1，dir 也不需改變，為一般跑馬燈移動的模式，而最後再把 led1 OR led3 的結果傳至 led 即可。

```
always @(posedge clk_div23) begin
    if (rst == 1) begin
        led1 = {1'b1, 15'b0};
        dir = 1'b1;
    end
    else begin
        if (en == 1) begin
            if ((led1 & led3) != 16'd0) begin
                if ((led1 & led3_middle) != 16'd0) begin
                    if (dir == 1) begin
                        led1 = {led1[13:0], led1[15:14]};
                        dir = ~dir;
                    end
                    else begin
                        led1 = {led1[1:0], led1[15:2]};
                        dir = ~dir;
                    end
                end
                else begin
                    if (dir == 1) begin
                        led1 = {led1[14:0], led1[15]};
                        dir = ~dir;
                    end
                    else begin
                        led1 = {led1[0], led1[15:1]};
                        dir = ~dir;
                    end
                end
            end
            else begin
                dir = dir;
                led1 = next_led1;
            end
        end
        else begin
            led1 = led1;
            dir = dir;
        end
    end
end
```

```
always @(posedge clk_div26) begin
    if (rst == 1) begin
        led3 = {3'b111, 13'b0};
        led3_middle = {2'b01, 14'b0};
    end
    else begin
        if (en == 1) begin
            led3 = next_led3;
            led3_middle = next_led3_middle;
        end
        else begin
            led3 = led3;
            led3_middle = led3_middle;
        end
    end
end

always @(*) begin
    next_led3 = {led3[0], led3[15:1]};
    next_led3_middle = {led3_middle[0], led3_middle[15:1]};
    if (dir == 1) begin
        next_led1 = {led1[0], led1[15:1]};
    end
    else begin
        next_led1 = {led1[14:0], led1[15]};
    end
end

always @(posedge clk) begin
    if (rst == 1) begin
        led = {3'b111, 13'b0};
    end
    else begin
        if (en == 1) begin
            led = led1 | led3;
        end
        else begin
            led = led;
        end
    end
end
```

3.學習到的東西與困難

第一次使用 FPGA 板子，覺得十分新奇，寫了這麼久的 verilog，終於看到實體結果的呈現。不過因為沒有 testbench 的原因，加上 FPGA 跑出的結果不是那麼固定，在 debug 上面變得十分吃力，常常毫無頭緒到底是哪個環節出了差錯，亦或者是整個架構都是有問題的，也藉此深刻體會到事前規劃的重要性，不然以之前東補西補的方式，肯定會完成得十分艱難。另外，我在 reg 變數的使用需要再細心一點，有時候會在兩個 always block 給值，導致我常常有 multiple-driven 而合成失敗的問題。

4.想對老師和助教說的話

希望以後都是至少兩個禮拜再一次的 lab，不然以我打 lab 的速度，真的會吃不消。