

## 邏輯系統實驗

### Lab 9 PWM 呼吸燈

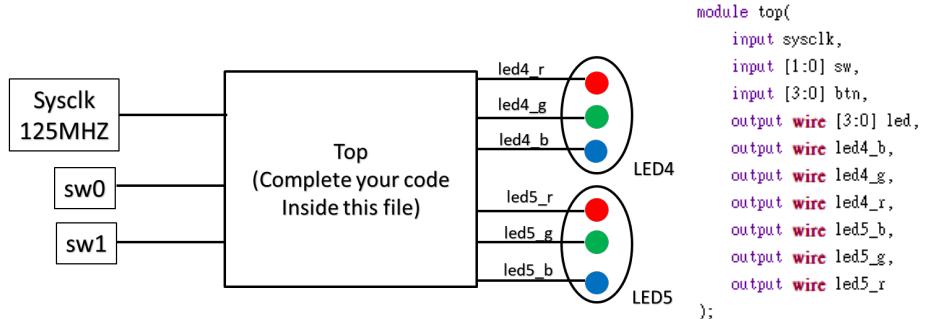
2022/03/02

第 5 組	
組員姓名	學號
林珮玉	E24084096
廖本恩	E24102179
蘇冠誠	E24084143

# 基礎題：

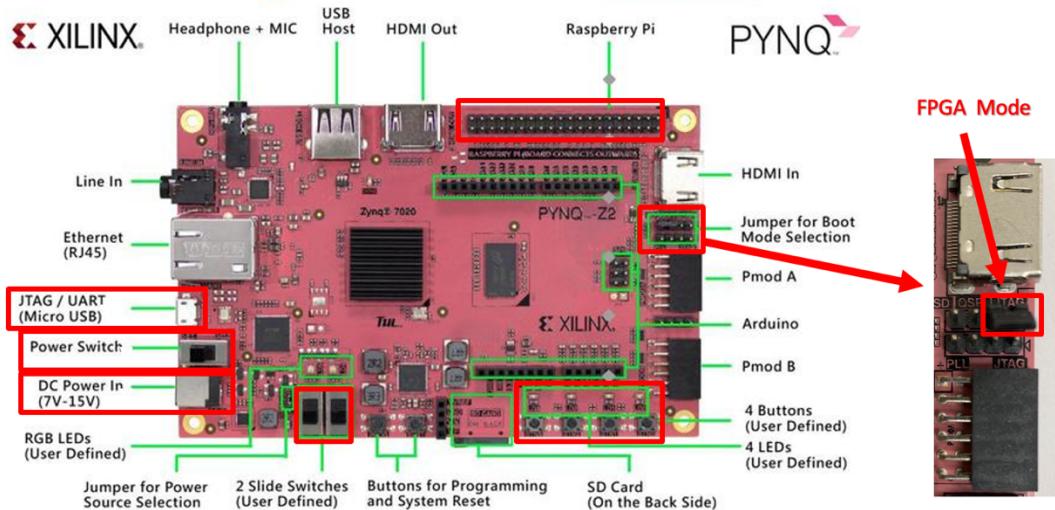
## 1. 簡述題目：

產生一組 clock divider 之波型，並使用開發板 PYNQ-Z2 實現。



## PYNQ-Z2

Set Jumper to FPGA boot mode



## 2. 實現方式：

```
dff_div.v
module dff_div(
    input n_rst,
    input trigger,
    output reg div
);

always @ (posedge trigger or negedge n_rst) begin
    if (!n_rst) begin
        div <= 1'b0;
```

```

    end else begin
        div <= !div;
    end
end

endmodule

```

### top.v

```

`timescale 1ns / 10ps

module top(
    input sysclk,
    input [1:0] sw,
    input [3:0] btn,
    output wire [3:0] led,
    output wire led4_b,
    output wire led4_g,
    output wire led4_r,
    output wire led5_b,
    output wire led5_g,
    output wire led5_r
);

    wire [2:0] color_led4; // [r,g,b]
    wire [2:0] color_led5; // [r,g,b]
    wire n_rst, n_rst0, n_rst1, rst0, rst1;
    wire lightR0, lightR1, lightG0, lightG1, lightB0, lightB1;
    wire light;
    wire [31:0] div_clk;

    wire [7:0] phase0, phase1_r, phase1_g , phase1_b;
    wire [7:0] duty0, duty1r, duty1g , duty1b;
    wire phase_trig0, phase_trig1r, phase_trig1g, phase_trig1b;
    wire duty_trig0, duty_trig1r, duty_trig1g, duty_trig1b;

    // The negative trigger reset
    assign n_rst = sw[0] | sw[1];
    assign n_rst0 = sw[0];
    assign n_rst1 = sw[1];

```

```

assign rst0 = ! n_rst0;
assign rst1 = ! n_rst1;

ripple_div rd0(.n_rst(n_rst), .sysclk(sysclk), .div(div_clk));
phase_counter p0w(.rst(rst), .enable(1'b1), .phase_shift(8'd0), .div_clk(div_clk), .duty_trig(duty_trig0), .phase_trig(phase_trig0));
phase_counter p1r(.rst(rst), .enable(1'b1), .phase_shift(8'd0), .div_clk(div_clk), .duty_trig(duty_trig1r), .phase_trig(phase_trig1r));
phase_counter p1g(.rst(rst), .enable(1'b1), .phase_shift(8'd31), .div_clk(div_clk), .duty_trig(duty_trig1g), .phase_trig(phase_trig1g));
phase_counter p1b(.rst(rst), .enable(1'b1), .phase_shift(8'd63), .div_clk(div_clk), .duty_trig(duty_trig1b), .phase_trig(phase_trig1b));

LUT_sin LS0( .phase_idx(phase0), .data(duty0));
LUT_sin LS1R( .phase_idx(phase1_r), .data(duty1r));
LUT_sin LS1G( .phase_idx(phase1_g), .data(duty1g));
LUT_sin LS1B( .phase_idx(phase1_b), .data(duty1b));

// white
pwm pwm0(.rst(rst), .enable(1'b1), .duty_up(duty_trig0), .phase_up(phase_trig0), .phase(duty0), .out(light));
pwm pwm1r(.rst(rst), .enable(1'b1), .duty_up(duty_trig1r), .phase_up(phase_trig1r), .phase(duty1r), .out(lightR1));
pwm pwm1g(.rst(rst), .enable(1'b1), .duty_up(duty_trig1g), .phase_up(phase_trig1g), .phase(duty1g), .out(lightG1));
pwm pwm1b(.rst(rst), .enable(1'b1), .duty_up(duty_trig1b), .phase_up(phase_trig1b), .phase(duty1b), .out(lightB1));

assign {led4_r, led4_g, led4_b} = {light, light, light} & {sw[0], sw[0], sw[0]}; //color_led4;
assign {led5_r, led5_g, led5_b} = {lightR1, lightG1, lightB1} & {sw[1], sw[1], sw[1]}; //color_led5;

assign led[0] = btn[0];
assign led[1] = btn[1];
assign led[2] = btn[2];
assign led[3] = btn[3];

endmodule

```

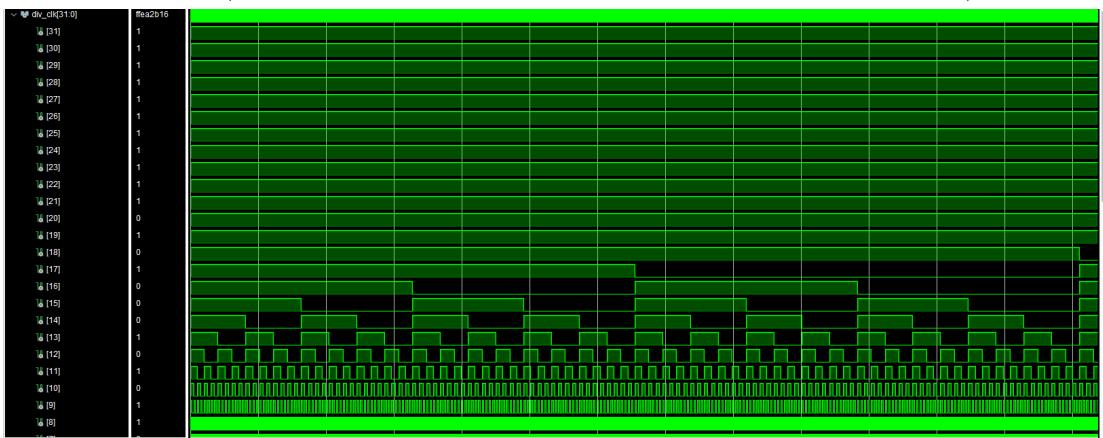
### 3. 結果呈現

甲、波形產生之目的與方法：

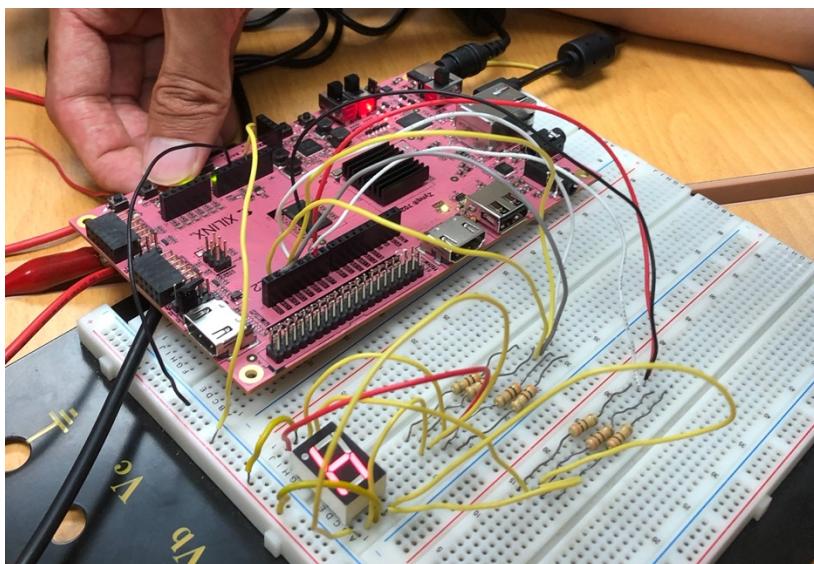
由於 125MHZ 的 Sysclk 頻率太高，利用以下公式以 Clock Divider 將頻率降至適合操作的頻率。

$$\text{divided clock frequency} = \frac{125\text{MHz}}{2^n}$$

乙、波型截圖（當時做出來太開心結果忘記截圖，先用助教的哦 QQ )



開發版 LED4 及 LED5 之成果照片



## 心得

### 組員一 林珮玉 E24084096

實驗前一天在預習時發現這次要打的 code 有好幾份、程式有點複雜，還要做硬體實現，心想這次完蛋了明天應該會做不完。不過實驗當天助教有上傳一份新的檔案，經過助教精心修改過的 code 真的簡單好多！只剩下挖空參數要帶，至於 FPGA 的部分只要照著指示大抵就可以完成了。不過我們這次卻壓線才做完，因為我們忘記設定 Vivado 某些初始化功能，導致程式 run 超級久還 run不出來，幸好最後有助教的提醒才讓實驗如期完成。心得是雖然這次實驗只是現場可程式化邏輯閘電路的雞毛蒜皮（非常之基礎），程式撰寫的部分也需要自己花時間理解與練習，但整體來說還是一個很棒的體驗，學到很多，謝謝助教！

## **組員二 廖本恩 E24102179**

這次實驗室第一次碰開發板，相比之前只能看到一堆波形圖，能看到實際的物品在動更有成就感。之前在高中也有碰過 Arduino，但是能對板子的操作就少很多，不像開發板可以自己定義 input，output，各種腳位等；也看到 verilog 可以當作 general purpose 的程式來實現不同需求。

## **組員三 蘇冠誠 E24084143**

實驗從這部份開始就是我生疏的領域，這部分我沒幫什麼忙但也幫不了什麼忙，所以我只能靜靜地看著他們把程式要求的一些參數補上，以及照流程跑開發板，還對跑了一個多小時都沒跑好（也就是很有問題）的燒錄軟體發個牢騷，一次實驗就這麼過去了