

# 數位電路實驗 Lab1 Report--點名控制器

第一組

B06901160 翁挺瑋

B06901177 劉 凡

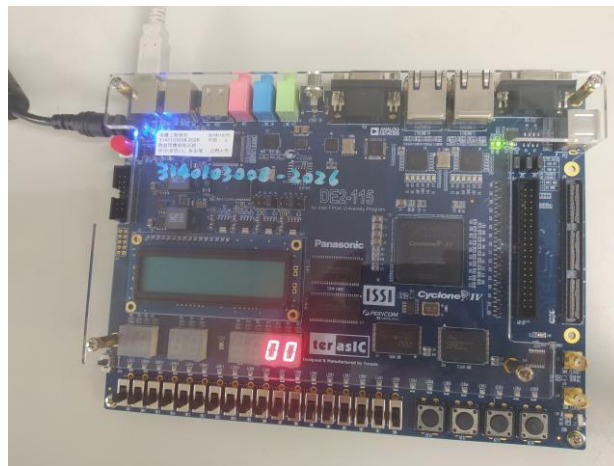
B06901028 林炯聿

## 一、所需器材及架設方式

所需器材：Altera DE2-155 FPGA

軟體工具：工作站、Quartus II

架設方式：

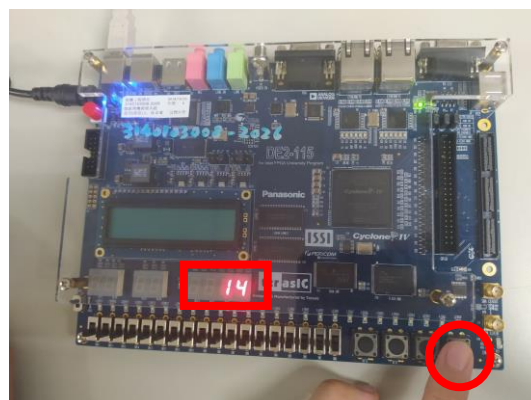


## 二、實作功能

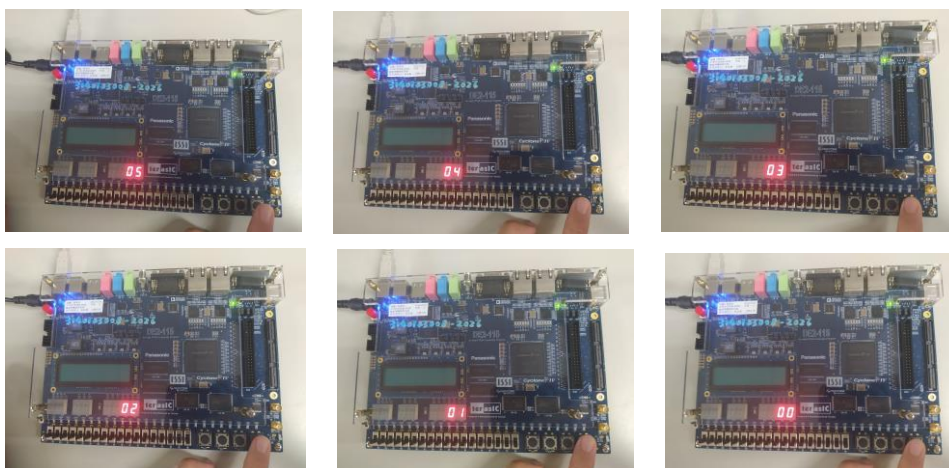
- 基本實作功能：按下 key1 reset；按下 key0 隨機產生 0~15 的亂數，數字跳動逐漸變慢，最後停在一個數字上。
- Bonus：跳動中可以隨時按 key0 使當前數字倒數至 0，若隨機跳動 7 秒後沒有按下 key0，則該數字開始閃爍，閃爍期間按下 key0 可以使該數字開始倒數至 0，任何倒數過程中按下 key0 都可以重新開始隨機產生亂數，任意時間按下 key1 都可以 reset。

## 三、使用步驟

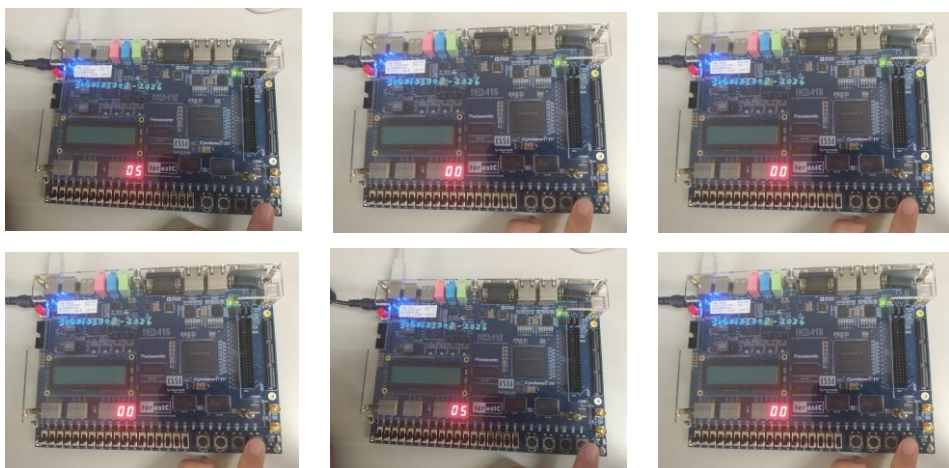
1. 連接 USB 和電源供應器到 FPGA 版上，按下 FPGA 紅色按鈕開機。
2. 按下 i\_start[key0]，開始產生 0~15 的隨機亂數。



3. 執行途中按下 key0 可以由該數字倒數至 0。



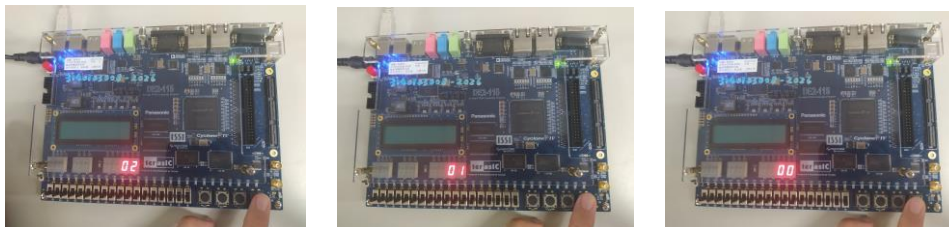
4. 若隨機跳動過程中沒有按下 key0，則 7 秒後該隨機數字開始閃動。



5. 閃動過程中按下 key0 開始倒數至 0。



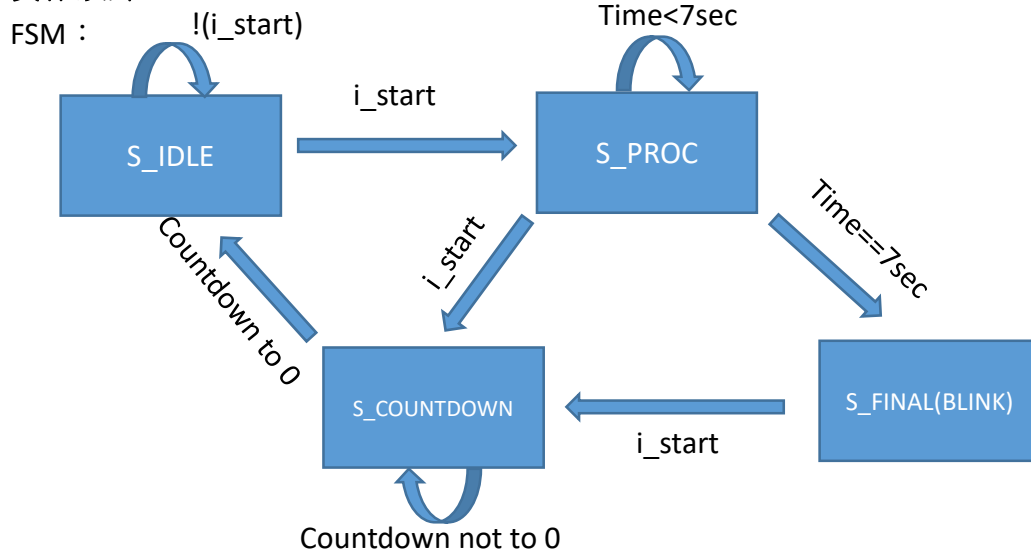
- 6.



7. 倒數過程中任意時間按下 key0 會重複步驟 2。  
8. 任意時間按下 reset[key1]回到 00 狀態。



#### 四、實作設計



#### 五、實作原理

##### 1. 產生亂數：

使用 linear congruential generator 的公式： $X_{n+1} = (AX_n + B)(\text{mod } M)$

Choose  $A = 2$ 、 $B = 4\text{-bit seed}$ 、 $M = 16$

```

44      seed_w = (seed_r < 4'd15)?seed_r+2:1;
45      $display("%d",seed_r);
46      state_w = S_IDLE;
47      o_random_out_w = o_random_out_r;
48      if (i_start) begin
49          state_w = S_PROC;
50          o_random_out_w = seed_r;
51          counter_w = 32'd0;
52          B_w = seed_r;
53      end
  
```

第 44 行為 seed 產生方式：seed 隨著 clock 一直在 1~15 跳動並且保持為奇數，可以使每次隨機產生的亂數不同，同時任意奇數和 16 互質也可以達到分布最亂的效果。

##### 2. 調整速度：

```

0      begin
1          congruent_w = (A*congruent_r-8,r)316;
2          o_random_out_w = congruent_r;
3          state_w = S_IDLE;
4          $display("%d",o_random_out_w);
5      end
1000000: begin
11      congruent_w = (A*congruent_r-8,r)316;
12      o_random_out_w = congruent_r;
13      state_w = S_IDLE;
14      $display("%d",o_random_out_w);
15  end
2000000: begin
21      congruent_w = (A*congruent_r-8,r)316;
22      o_random_out_w = congruent_r;
23      state_w = S_IDLE;
24      $display("%d",o_random_out_w);
25  end
6000000: begin
61      congruent_w = (A*congruent_r-8,r)316;
62      o_random_out_w = congruent_r;
63      state_w = S_IDLE;
64      $display("%d",o_random_out_w);
65  end
10000000: begin
101      congruent_w = (A*congruent_r-8,r)316;
102      o_random_out_w = congruent_r;
103      state_w = S_IDLE;
104      $display("%d",o_random_out_w);
105  end
15000000: begin
151      congruent_w = (A*congruent_r-8,r)316;
152      o_random_out_w = congruent_r;
153      state_w = S_IDLE;
154      $display("%d",o_random_out_w);
155  end
  
```

指定出現時間。

### 3. 閃爍：

```
S_FINAL:begin
    state_w = S_FINAL;
    counter_w = counter_r+1;
    if (i_start) begin
        state_w = S_COUNTDOWN;
        counter_w = 32'd0;
        o_random_out_w = temp_random_out_r;
    end
    if(counter_r==20000000) begin
        o_random_out_w = 4'd0;
    end
    else if(counter_r==40000000) begin
        o_random_out_w = temp_random_out_r;
        counter_w = 32'd0;
    end
end
end
```

記住當前數字並在該數字與 00 之間做切換。

### 4. 倒數：

```
counter_w = counter_r+1;
state_w = S_COUNTDOWN;
if(o_random_out_r==0) begin
    temp_random_out_w = 4'd0;
    o_random_out_w = 4'd0;
    state_w = S_IDLE;
    counter_w = 32'd0;
end
else if(counter_r==20000000) begin
    o_random_out_w = o_random_out_r-1;
    state_w = S_COUNTDOWN;
    counter_w = 32'd0;
end
```

記住當前數字，每 2000000 個 clock cycle 做一次減一。

## 六、碰到問題與解決方式

第一次的作業算是入門實作的感覺，簡單版的兩個 state 就可以完成，我們擴充到 4 個 state 想要練習看看更多功能的實作，過程中也遇到了很多的 bug，隨著變數及行數越來越多，寫 code 的過程也要越來越仔細，練習做 bonus 的時候有更佳的了解 system verilog 的架構以及完成多個 state 的寫法，想信之後會越來越好的。