

NTUEE DCLAB

---

**LAB 1: 亂數點名器**

---

*Graduate Institute of Electronics Engineering*  
National Taiwan University

# Outline

---

- Introduction
  - Lab requirements
- Implementation
  - Finite state machine (FSM) and count down control
  - Generating random numbers
  - Reset signal
- Code template
- Simulation and debug
- Report regulations

# Introduction

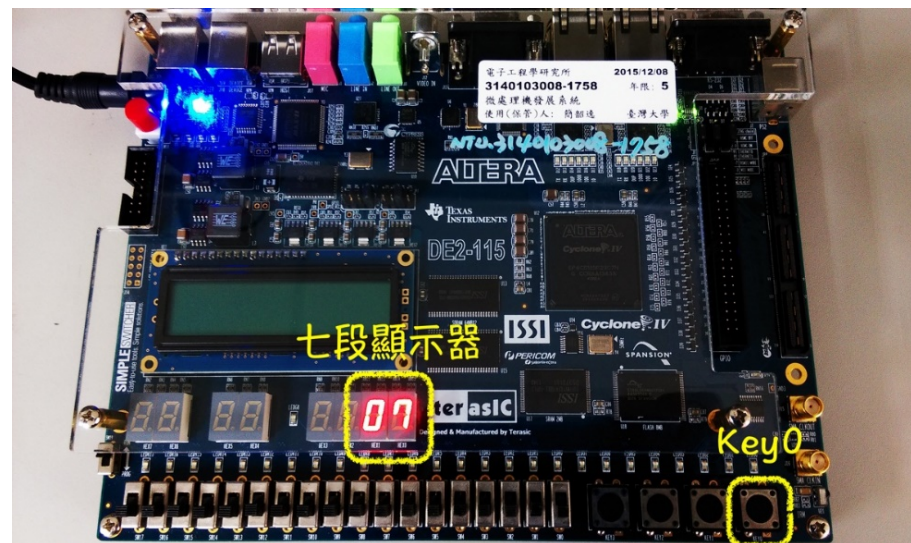
---



<https://www.youtube.com/watch?v=FwTzURTGyvc>

# Lab Requirements

- 按下key1可以reset
- 按下key0可開始點名器運作
  - 隨機產生0~15的亂數
  - 以七段顯示器顯示
  - 數字跳動頻率逐漸變慢
  - 最後停在一個數字上



- Bonus (demo時與report中皆應清楚詳細說明)
  - 跳動中途擷取亂數
  - 記憶前次亂數結果
  - 其他能想到的創意

# Outline

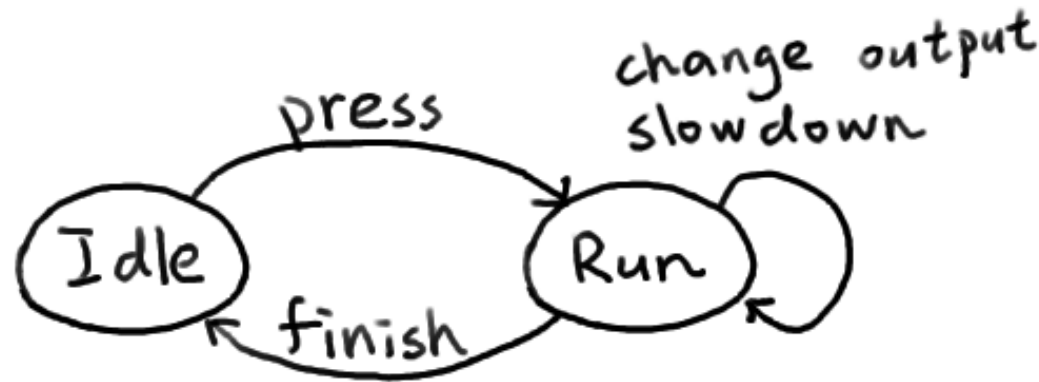
---

- Introduction
  - Lab requirements
- Implementation
  - Finite state machine (FSM) and count down control
  - Generating random numbers
  - Reset signal
- Code template
- Simulation and debug
- Report regulations

# Finite State Machine (FSM) Design

- 簡單範例

- IDLE：等待key0被按下
- RUN：產生亂數並變動輸出切換頻率

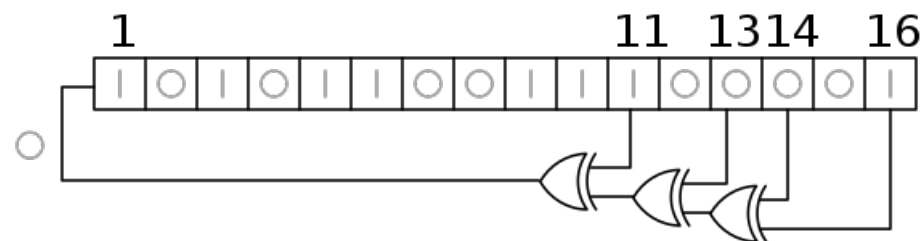


- Think

- 如何變動輸出切換頻率?
  - 用counter計算每經過幾個cycle要輸出 (內建的clock是50MHz)
  - 用更多state來切換停留在一數字上的時間
  - 等等

# Random Number Generation

- 電路上通常都是產生pseudo-random number
  - Linear feedback shift register (LFSR)



- Linear congruential generator

$$X_{n+1} = aX_n + b \pmod{M}$$

- Think
  - 如何在每次按下key0後產生不同的亂數數列?
  - Seed如何產生與設定?

# Reset Signal

---

- 按下key1會產生global reset signal
  - 將所有register設定為初始值
- 寫在sequential block裡面

```
always_ff @(posedge i_clk or posedge i_rst) begin
    if (i_rst) begin
        state_r    <= IDLE;
        counter_r  <= 9'd0;
        o_ans_r    <= 258'd0;
        o_finish_r <= 1'b0;
    end
    else begin
        state_r    <= state_w;
        counter_r  <= counter_w;
        o_ans_r    <= o_ans_w;
        o_finish_r <= o_finish_w;
    end
end
```



# Outline

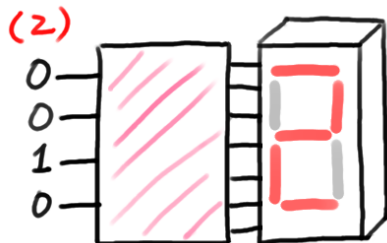
---

- Introduction
  - Lab requirements
- Implementation
  - Finite state machine (FSM) and count down control
  - Generating random numbers
  - Reset signal
- **Code template**
- Simulation and debug
- Report regulations

# Code Template

---

- DE2\_115.qsf
  - Map top-level I/O to FPGA physical I/O
- DE2\_115.sdc
  - Timing constraints
- DE2\_115.sv
  - Top module mapped to FPGA
- Debounce.sv
  - Stabilize key press glitch
  - Provide 1-clock-pulse keydown/keyup signal
- SevenHexDecoder.sv



# Code TODO

---

- Add your code to [Top.sv](#)
  - always\_comb
  - always\_ff

```
1  module Top (  
2      input      i_clk,  
3      input      i_rst_n,  
4      input      i_start,  
5      output [3:0] o_random_out  
6  );  
7  
8  // please check out the working example in lab1 README first  
9  
10 endmodule
```

# Outline

---

- Introduction
  - Lab requirements
- Implementation
  - Finite state machine (FSM) and count down control
  - Generating random numbers
  - Reset signal
- Code template
- **Simulation and debug**
- Report regulations

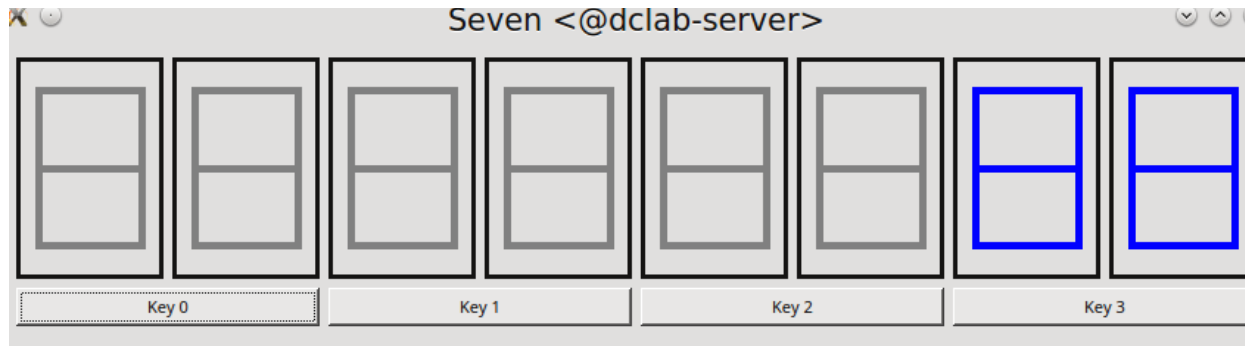
# Run Testbench on Server

---

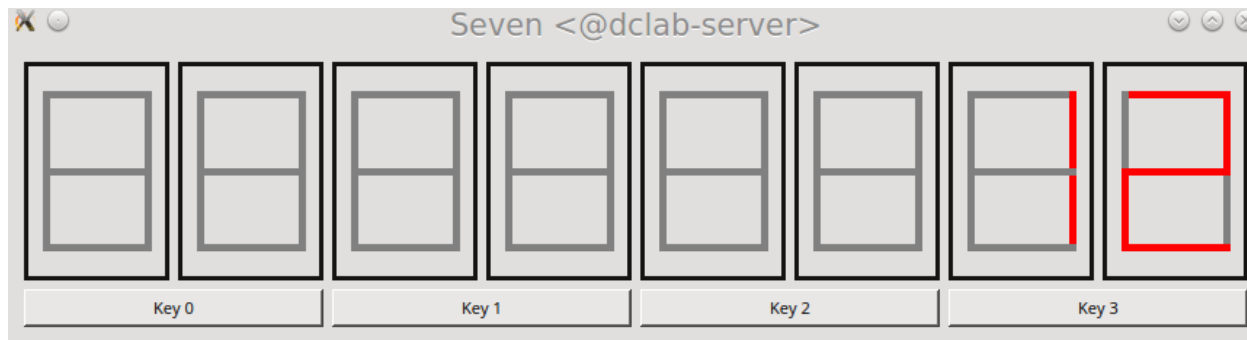
- Login to DCLab server
- Keep the provided directory structure
- Type “tool 2” to enable *ncsim* (for simulation) and *nWave* (for viewing waveforms)
- Change the directory to lab1/sim/
- Type “make -f ../../Makefile Top” to run GUI simulation

# Testing GUI

- Blue lines mean something is wrong



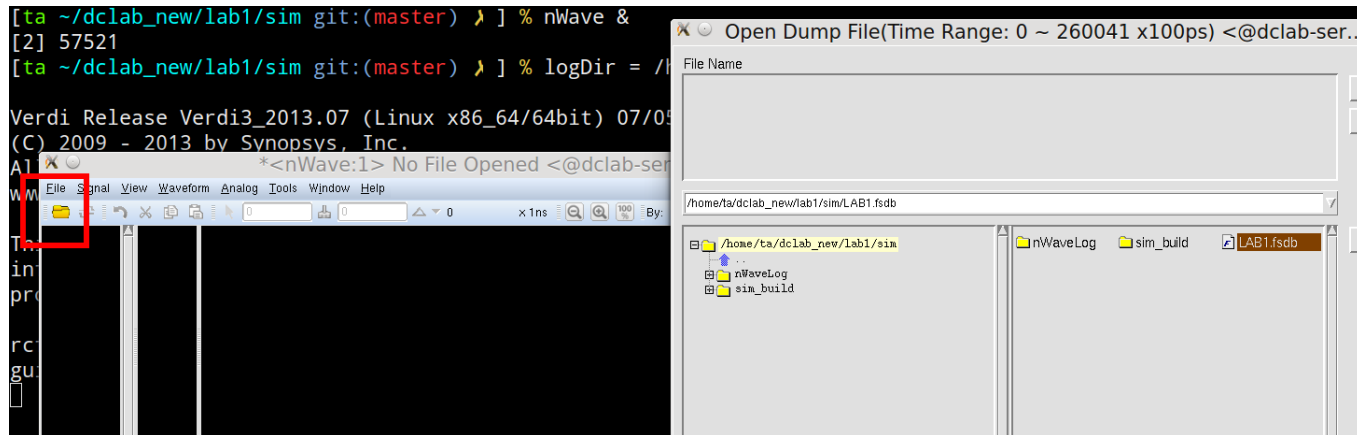
- Red lines mean correct output



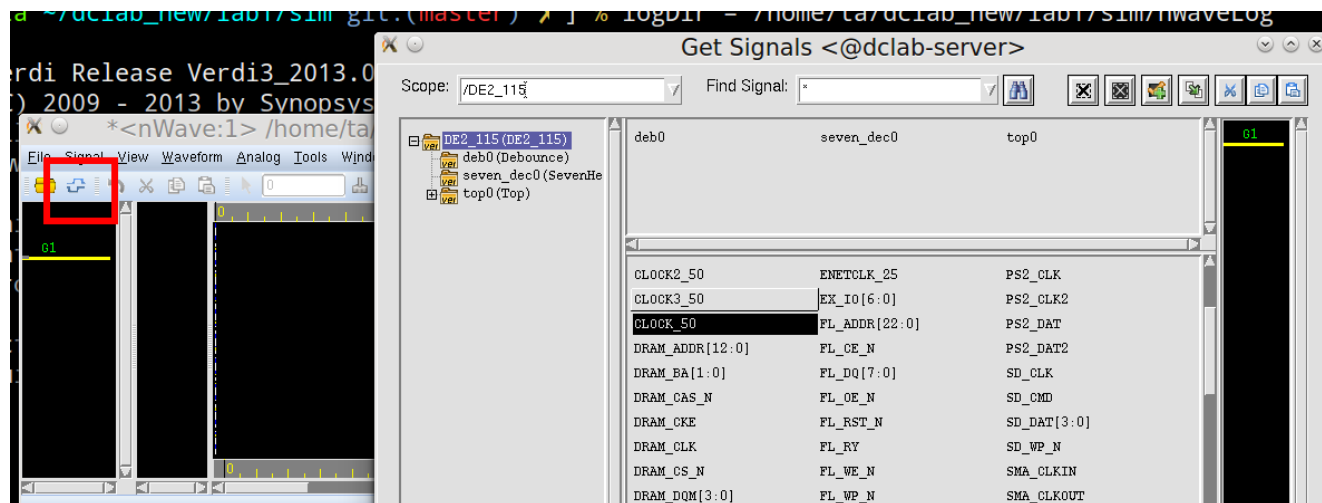
注意：Simulation用的clock頻率跟實際FPGA上的50MHz不同，因此跳動頻率效果看起來會不同

# Debug with Waveform

- Type “nWave &” and open the waveform file



- Choose the signals you wish to observe



# Outline

---

- Introduction
  - Lab requirements
- Implementation
  - Finite state machine (FSM) and count down control
  - Generating random numbers
  - Reset signal
- Code template
- Simulation and debug
- **Report regulations**



# Report Regulations

---

- 內容應包含
  - 使用所需器材與架設方式
  - 使用方式與詳細步驟
  - 實作設計技術細節與巧思
  - 碰過的問題或挑戰與解決方式
- 一組交一份，以pdf檔繳交
- 命名方式：teamXX\_lab1\_report.pdf
  - Ex: team01\_lab1\_report.pdf
- 繳交期限：demo當天午夜
  - 遲交每三天\*0.7

# Questions?