NTUEE DCLAB

LAB 1: 亂數點名器

Graduate Institute of Electronics Engineering

National Taiwan University

Outline

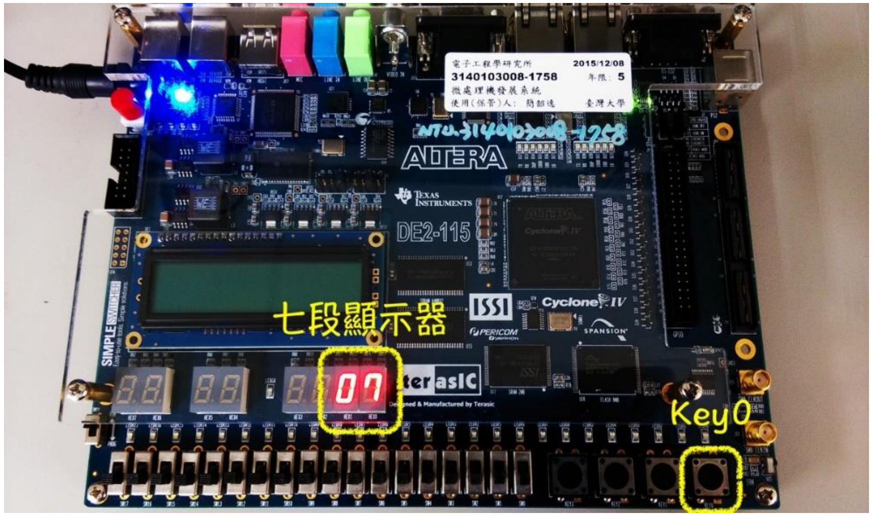
- Introduction
  - Lab requirements
- Implementation
  - Finite state machine (FSM) and count down control
  - Generating random numbers
  - Reset signal
- Code template
- Simulation and debug
- Report regulations

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



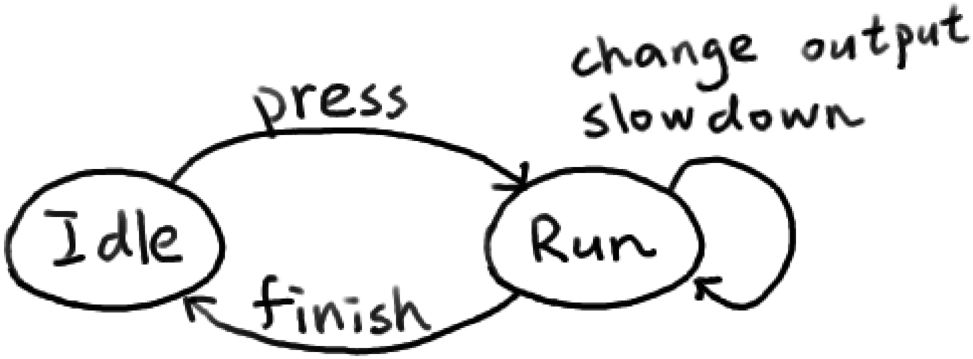
- Bonus (demo時與report中皆應清楚詳細說明)
  - 跳動中途擷取亂數
  - 記憶前次亂數結果
  - 其他能想到的創意

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

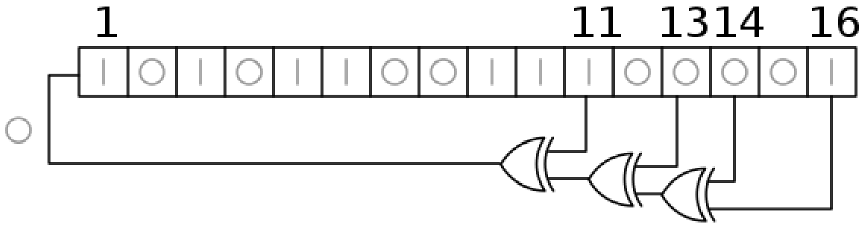
- 簡單範例
  - IDLE：等待key0被按下
  - RUN：產生亂數並變動輸出切換頻率



- 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)



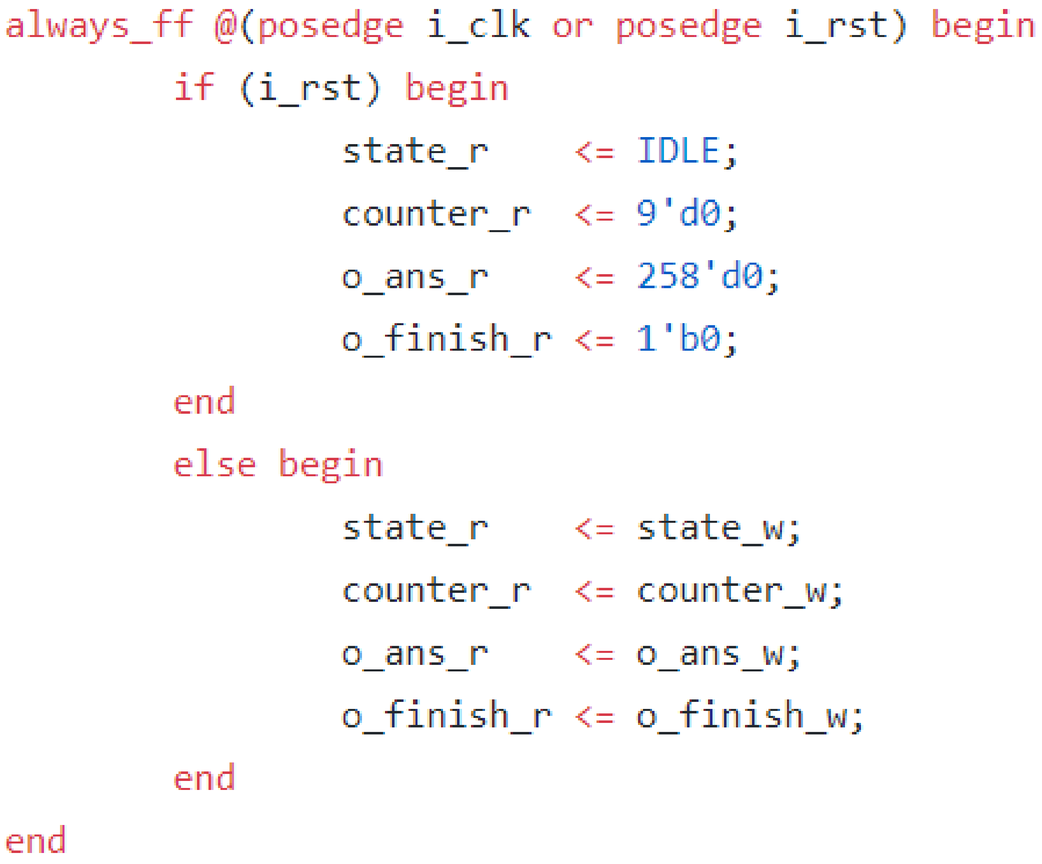
  - Linear congruential generator

$$X_{n+1} = aX_n + b \ \left(\mathrm{mod\ \ M}\right)$$

- 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



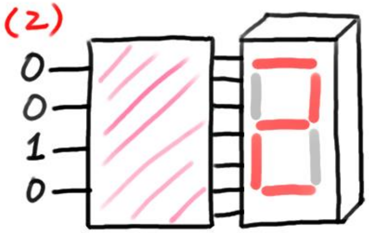
```
always_ff @(posedge i_clk or posedge i_rst) begin
        if (i_rst) begin
                state_r    <= IDLE;
                counter_r  <= 9'd0;
                o_ans_r    <= 258'd0;
                o_finish_r <= 1'b0;
        end
        else begin
                state_r    <= state_w;
     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

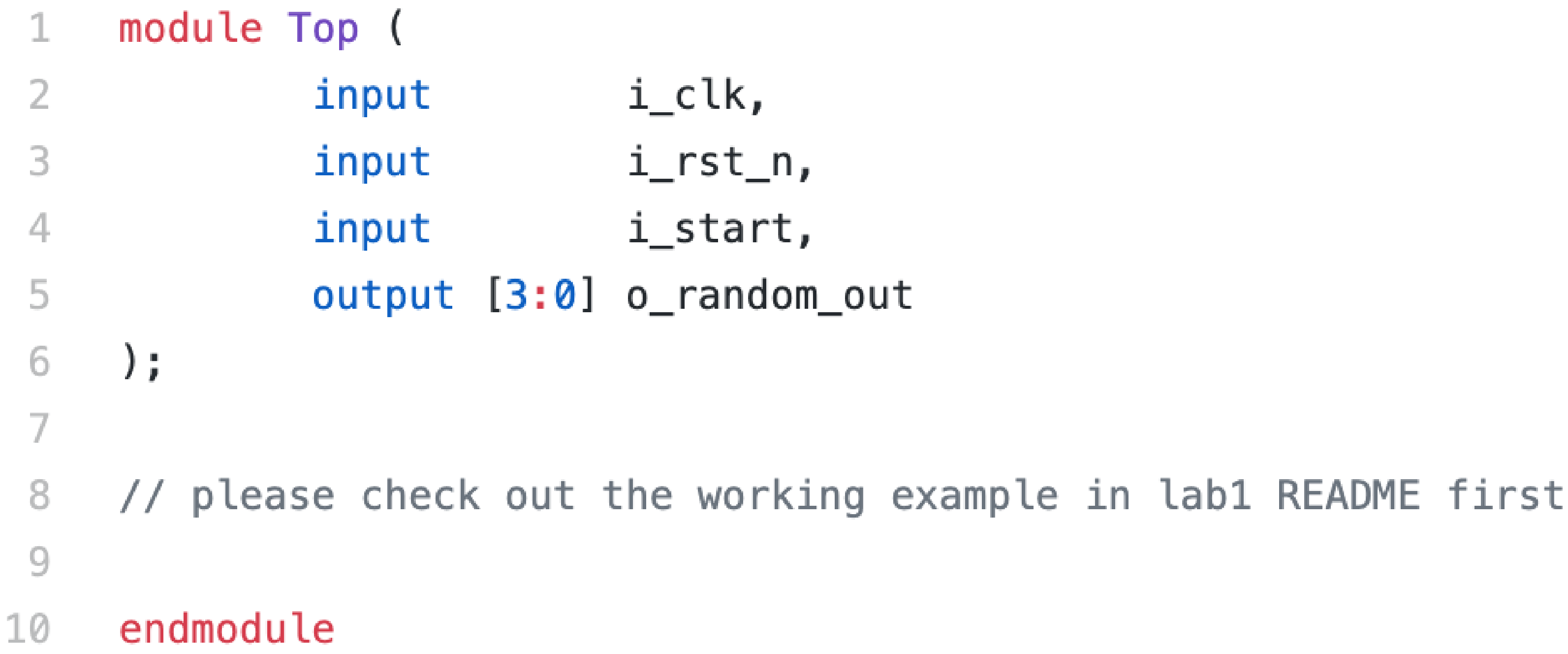
Code Template

- 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



```systemverilog
module Top (
        input        i_clk,
        input        i_rst_n,
        input        i_start,
        output [3:0] o_random_out
);

// please check out the working example in lab1 README first

endmodule
```

Outline

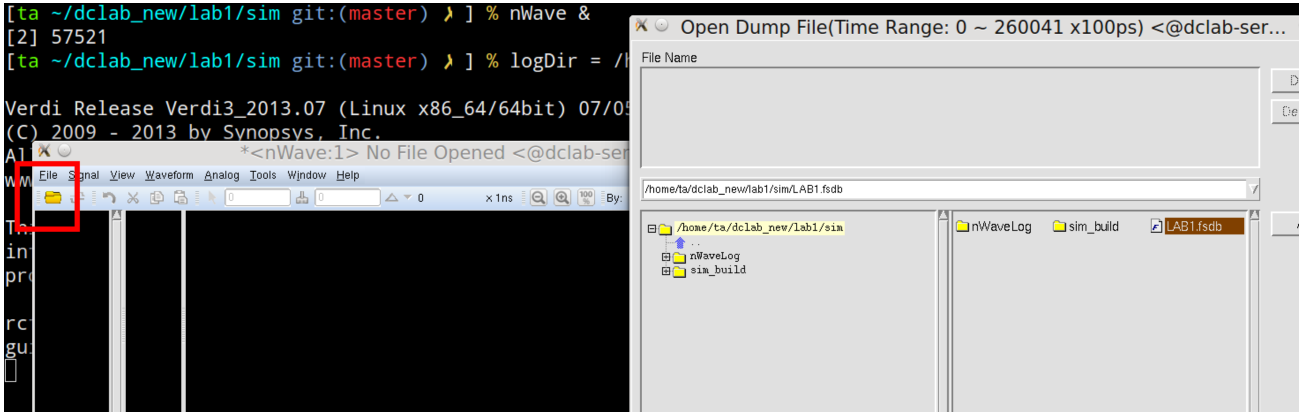
- Introduction
  - Lab requirements
- Implementation
  - Finite state machine (FSM) and count down control
  - Generating random numbers
  - Reset signal
- Code template
- Simulation and debug
- Report regulations

Run Testbench on Server

- Login to server
- Keep the provided directory structure
- Source the EDA tools (for ncverilog, Verdi)
- > cd lab1/sim/
- > ncverilog tb_Top.sv ./../src/Top.sv +access+rw
- > nWave &

Debug with Waveform

- Type “nWave &” and open the waveform file



```
[ta ~/dclab_new/lab1/sim git:(master) ⅄ ] % nWave &
[2] 57521
[ta ~/dclab_new/lab1/sim git:(master) ⅄ ] % logDir = /h

Verdi Release Verdi3_2013.07 (Linux x86_64/64bit) 07/05
(C) 2009 - 2013 by Synopsys, Inc.
Al
ww

Th
in
pr

rc
gu
```

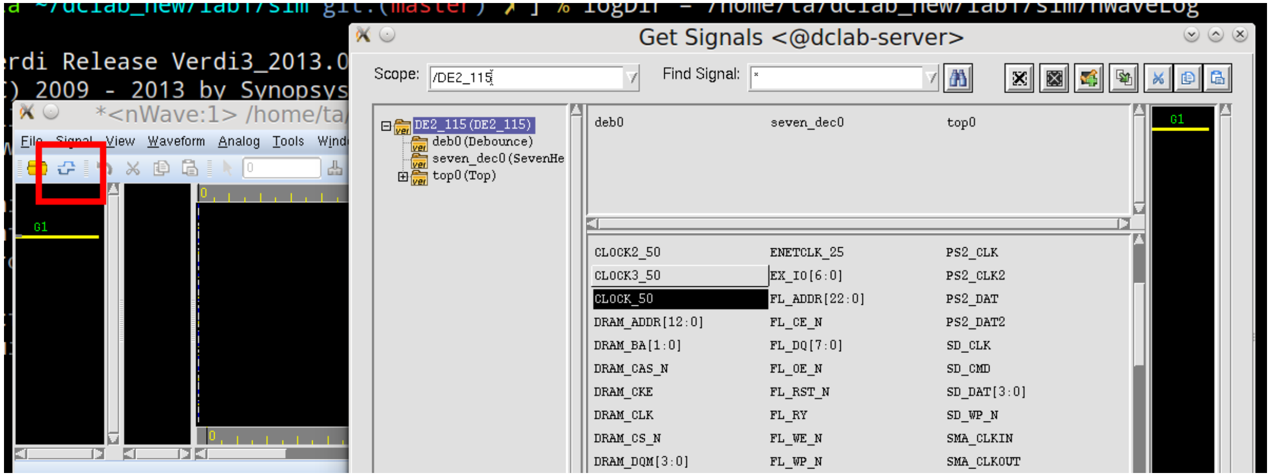
*<nWave:1> No File Opened <@dclab-ser
File
Signal
View
Waveform
Analog
Tools
Window
Help
x 1ns
By:
Open Dump File(Time Range: 0 ~ 260041 x100ps) <@dclab-ser...
File Name

/home/ta/dclab_new/lab1/sim/LAB1.fsdb

```
⊟ /home/ta/dclab_new/lab1/sim
    ..
  ⊞ nWaveLog
  ⊞ sim_build
```

nWaveLog   sim_build   LAB1.fsdb

- Choose the signals you wish to observe



```
a ~/dclab_new/lab1/sim git:(master) ⅄ ] % logDir = /home/ta/dclab_new/lab1/sim/nWaveLog
rdi Release Verdi3_2013.0
) 2009 - 2013 by Synopsys
```

Scope: /DE2_115 Find Signal: *

```
⊟ DE2_115(DE2_115)
    deb0(Debounce)
    seven_dec0(SevenHe
  ⊞ top0(Top)
```

```
deb0              seven_dec0        top0

CLOCK2_50         ENETCLK_25        PS2_CLK
CLOCK3_50         EX_IO[6:0]        PS2_CLK2
CLOCK_50          FL_ADDR[22:0]     PS2_DAT
DRAM_ADDR[12:0]   FL_CE_N           PS2_DAT2
DRAM_BA[1:0]      FL_DQ[7:0]        SD_CLK
DRAM_CAS_N        FL_OE_N           SD_CMD
DRAM_CKE          FL_RST_N          SD_DAT[3:0]
DRAM_CLK          FL_RY             SD_WP_N
DRAM_CS_N         FL_WE_N           SMA_CLKIN
DRAM_DQM[3:0]     FL_WP_N           SMA_CLKOUT
```

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

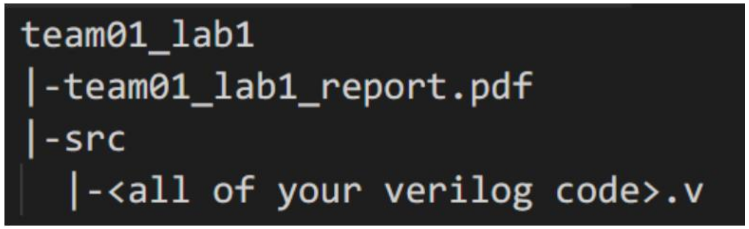
Report Regulations

- 內容應包含
  - File Structure
  - System Architecture (需標示Data Path)
  - Hardware Scheduling (FSM or Algorithm Workflow)
  - Fitter Summary截圖
  - Timing Analyzer截圖
  - 遇到的問題與解決辦法，心得與建議

- 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

Submission Rules

- 繳交檔案架構



```
team01_lab1
|-team01_lab1_report.pdf
|-src
  |-<all of your verilog code>.v
```

  - 將 teamXX_labX 資料夾包成一個 zip 後，上傳到實驗室 NAS 各組的 submission 資料夾
  - src 資料夾內的 Verilog 可自行命名，只要在 report 中有說明層級架構即可
  - 繳交期限：**demo 日當天 23:59 前**

- **若未遵守繳交格式會酌情扣分**

Questions?