Welcome to The Hardware Lab!

Fall 2016

Verilog Introduction

Prof. Chun-Yi Lee

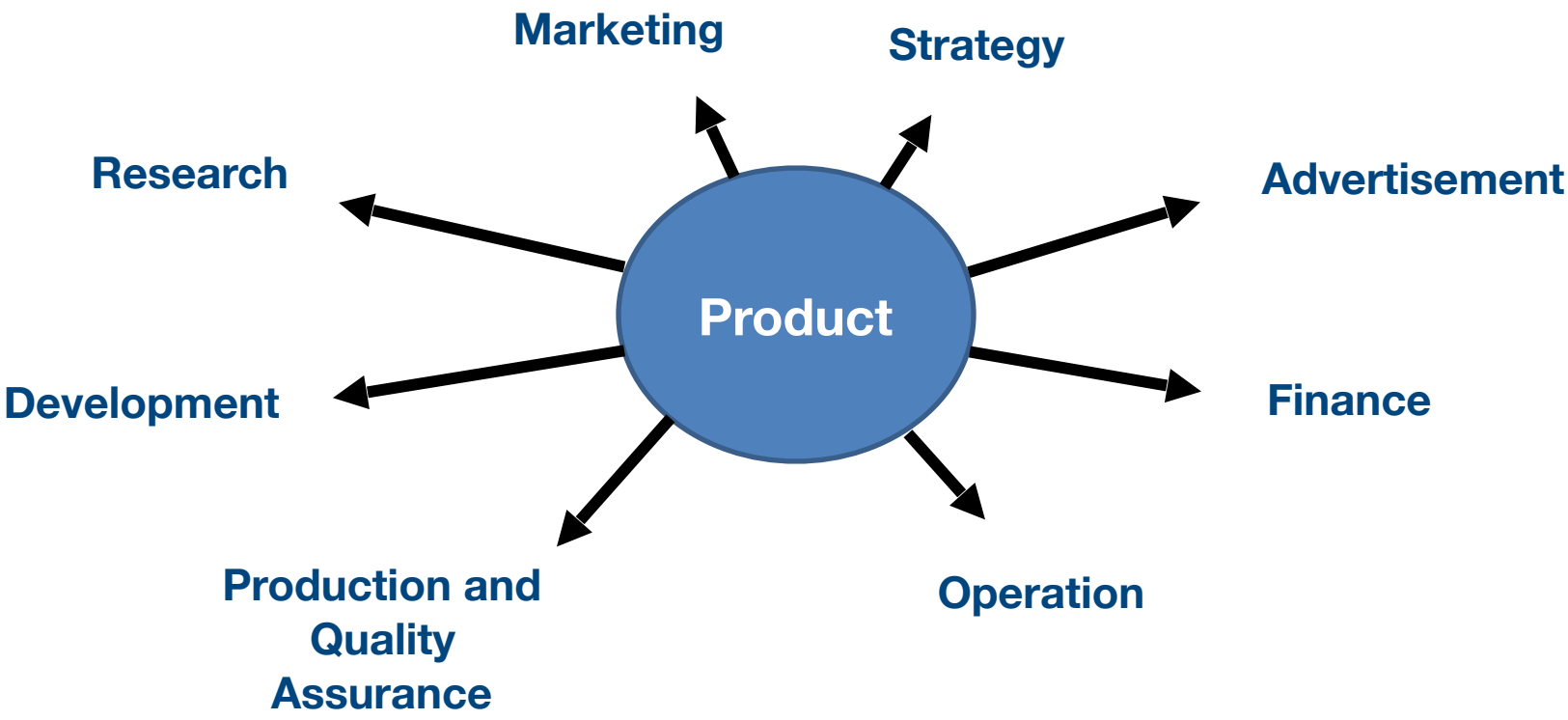
Department of Computer Science
National Sting Hua University

Announcements

- 分組名單 has been uploaded to ILMS
  - Please confirm and check errors
  - https://goo.gl/iDHrZN
- Role playing game begins from **9/27/2015 (Thu)**
  - Team 1, 2, and 3 will make a presentation
  - Please attend the presentation
- Role playing game topics
  - Please remember to post your topic on Google Doc
  - https://goo.gl/yySZxi

Role-Playing Game (Cont'd)

- There are many aspects of a company
  - Different roles in the company responsible for different duties



Agenda

- Verilog lab 1
- Testbench
- Vivado Introduction

*Pictures cited from Marvel.com for education purpose only

Verilog Lab 1

- **5%** of the course score
- Five basic questions (**50%, 10% each**)
  - Demonstration by **9/29/2016 (Thu), in class**
- Lab 1 report (**30%**)
  - Please draw the gate-level circuit of your design
  - Answer the questions listed in the handout
  - Contribution of each team member
  - What you have learned from Lab 1
- Two optional questions (**20%, 10% each**)
  - Challenge it if you have extra time
  - Include in your Lab 1 report if you can make it
  - Describe your design and how it works

Verilog Lab 1 Rules

- Only gate-level description
  - Only basic logic gates are ALLOWED (**and**, **or**, **nand**, **nor**, **not**)
  - **Sorry, no xor & xnor**
- Please **AVOID** using
  - Continuous assignment and conditional operators
  - Behavioral operators (e.g., +, -, &, |, ^, &&, !, ~….., etc.)
  - However, behavioral operators are **ALLOWED** in your **TESTBENCH**
- Demonstrate your work by **waveforms**
  - Edit your own testbench

Verilog Lab 1 Submission

- Demonstration and code submission due date & time:
  5:30pm, 9/29/2016 (Thur)
  This is a HARD deadline

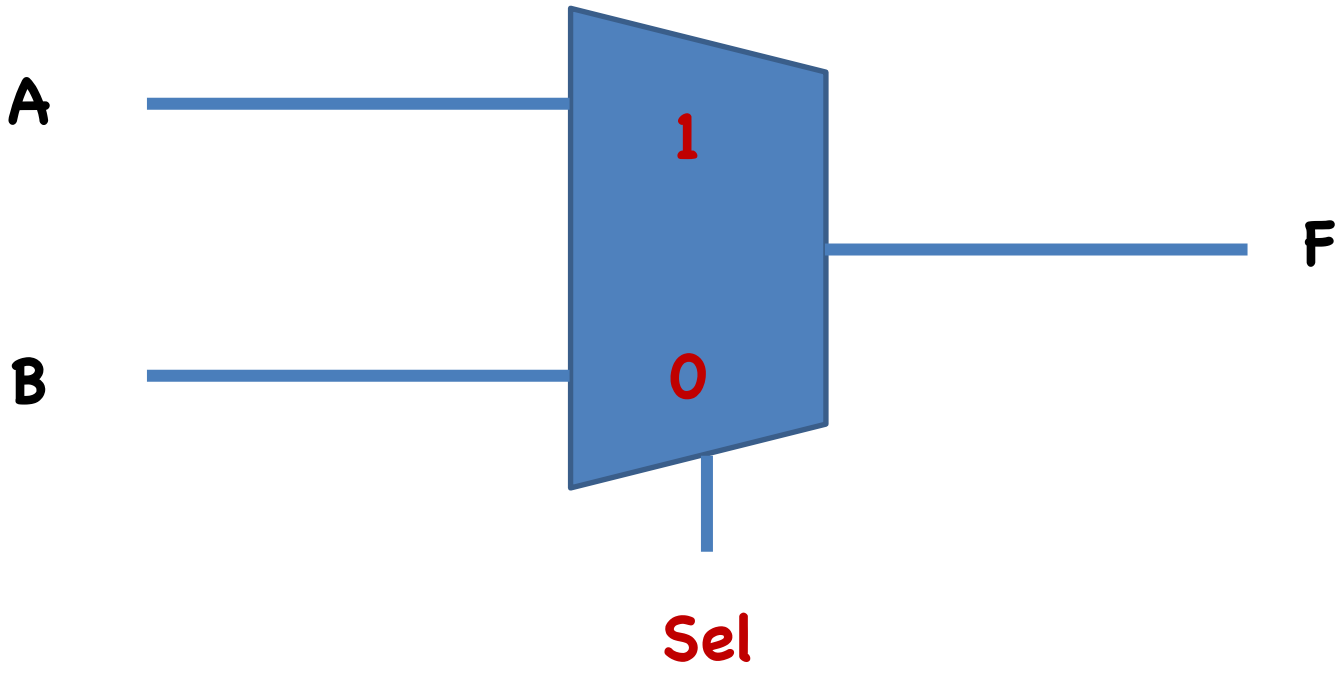
- Lab 1 report submission due date & time:
  5:30pm, 9/29/2016 (Thur)
  This is also a HARD deadline

- Please submit your report and Verilog codes to ILMS
  - You codes have to include your design and testbench
  - Format: **Lab1_Team1_Codes.zip**, **Lab1_Team1_Report.zip**

- We will test your codes by our own testbench

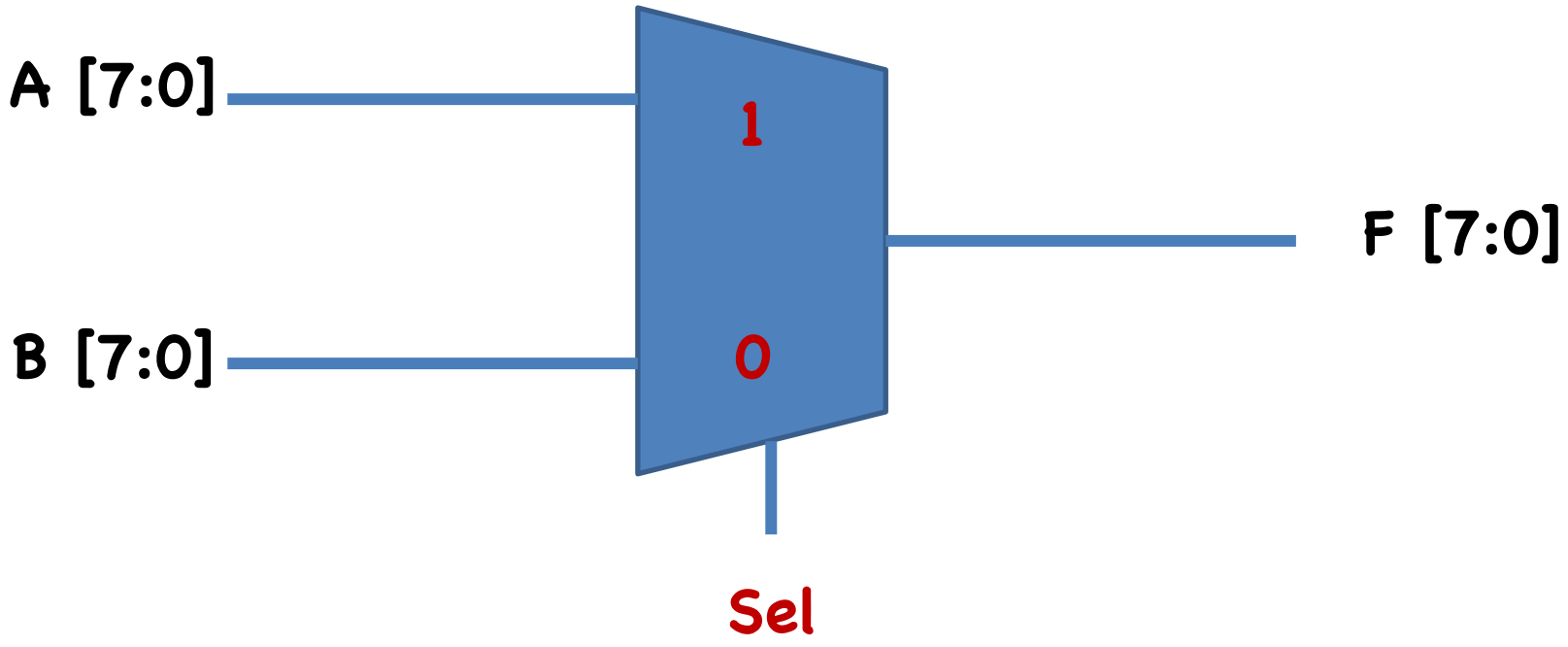
Question 1

- 1-bit **2-to-1 MUX**



Question 2

- 8-bit **2-to-1 MUX**
- Instantiate the **2-to-1 MUX** module from **Question 1**



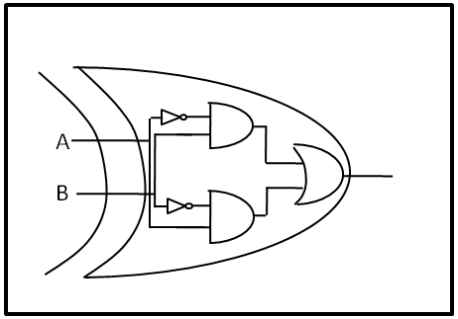
Question 3

- 4 x 16 decoder

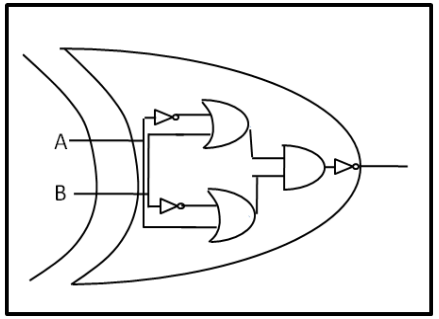
Input Din[3:0]	Output Dout[15:0]	Input Din[3:0]	Output Dout[15:0]
1111	0000_0000_0000_0001	0111	0000_0001_0000_0000
1110	0000_0000_0000_0010	0110	0000_0010_0000_0000
1101	0000_0000_0000_0100	0101	0000_0100_0000_0000
1100	0000_0000_0000_1000	0100	0000_1000_0000_0000
1011	0000_0000_0001_0000	0011	0001_0000_0000_0000
1010	0000_0000_0010_0000	0010	0010_0000_0000_0000
1001	0000_0000_0100_0000	0001	0100_0000_0000_0000
1000	0000_0000_1000_0000	0000	1000_0000_0000_0000

Question 4

- 1-bit **Full Adder**
- Create an XOR module



Design 1

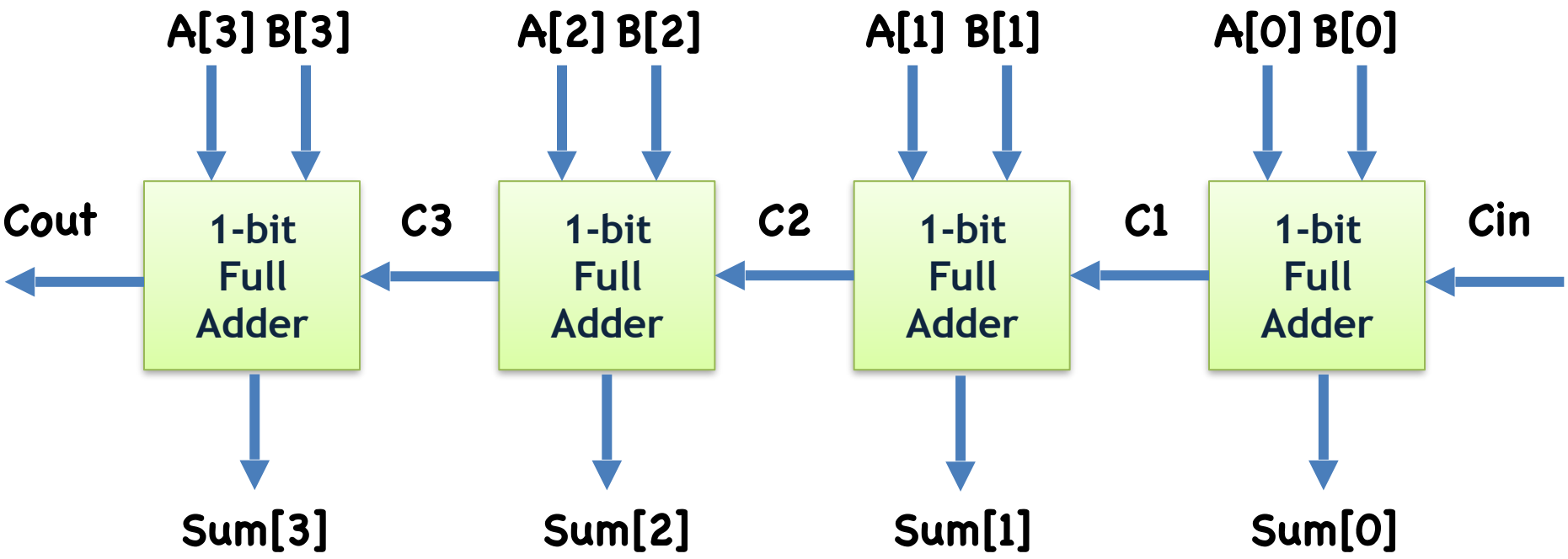


Design 2

- Implement the function of a one-bit full adder
  - Sum = $A \oplus B \oplus C\_in$
  - $C\_out = A \cdot B + A \cdot C\_in + B \cdot C\_in$

Question 5

- 4-bit **Ripple Carry Adder**
- Instantiate the **Full Adder module** from **Question 4**



Optional Question 1

- 3-bit **Comparator**
  - The 3-bits are **unsigned numbers**
  - **No conditional operators, GATE LEVEL only**

A_lt_B	Condition
1'b1	A[2:0] < B[2:0]
1'b0	Otherwise

A_gt_B	Condition
1'b1	A[2:0] > B[2:0]
1'b0	Otherwise

A_eq_B	Condition
1'b1	A[2:0] == B[2:0]
1'b0	Otherwise

Optional Question 2

- 4-bit **Comparator**
  - The 4-bits are **unsigned numbers**
  - **No conditional operators, GATE LEVEL only**

A_lt_B	Condition
1'b1	A[3:0] < B[3:0]
1'b0	Otherwise

A_gt_B	Condition
1'b1	A[3:0] > B[3:0]
1'b0	Otherwise

A_eq_B	Condition
1'b1	A[3:0] == B[3:0]
1'b0	Otherwise

Please Follow The Format Below

Question 1
Module name: Mux_1bit
Input: A, B, Sel
Output: F

Question 2
Module name: Mux_8bits
Input: A, B, Sel
Output: F

Question 3
Module name: Decoder
Input: Din
Output: Dout

Question 4
Module name: FullAdder
Input: A, B, C_in
Output: Sum, C_out

Please Follow The Format Below (Cont'd)

Question 5

Module name: RippleCarryAdder
Input: A, B, C_in
Output: Sum, C_out

Optional Question 1

Module name: Comparator_3bits
Input: A, B
Output: A_lt_B, A_gt_B, A_eq_B

Optional Question 2

Module name: Comparator_4bits
Input: A, B
Output: A_lt_B, A_gt_B, A_eq_B

There is also a template provided on iLMS. Please refer to it

Questions to Answer in the Report

- What are the number of output bits in a 5 input decoder?
- For decoders, how hard is it to verify if the inputs are increased from ***n*** to (***n*+1**)?
- How do you verify that the your circuit is working correctly?
- What is the difference between a Full Adder and a Half Adder?

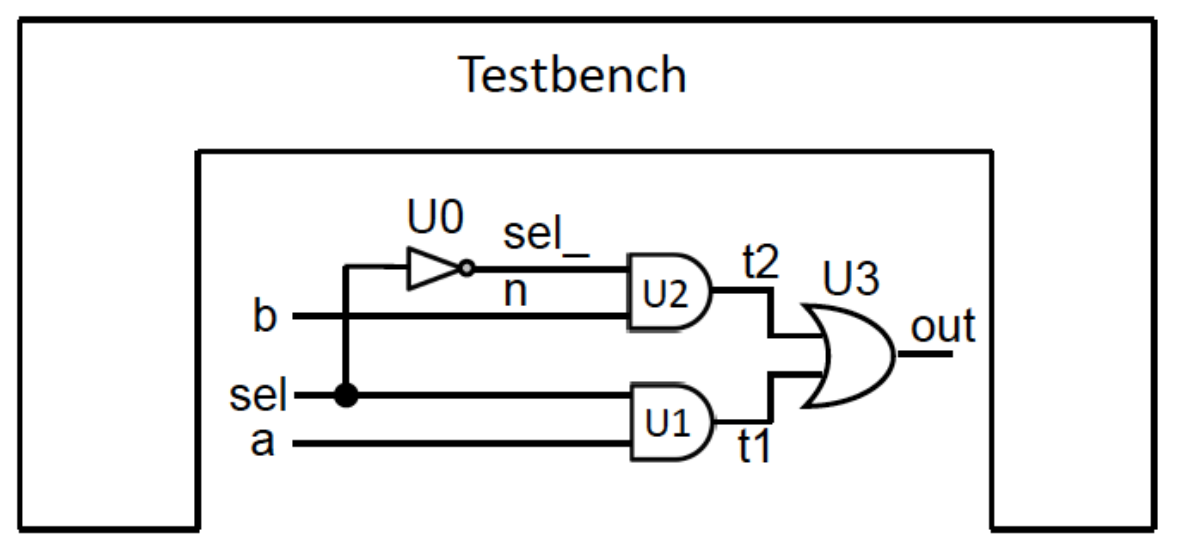
Agenda

- Verilog lab 1
- Testbench
- Vivado Introduction

*Pictures cited from www.engadget.com for education purpose only

Verilog Simulation Framework

- Testbench verifies whether a module is correct or not
- Similar to the main function in C++
- Generate stimulus and check the outputs



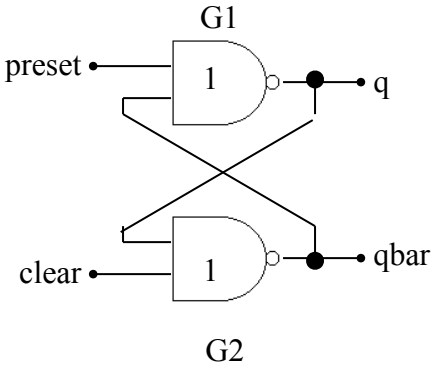
*Pictures cited from Prof. Shih-Chieh Chang's 2010 slides for education purpose only

Verilog Testbench

Design

```
module Nand_Latch_1 (q, qbar, preset, clear);
  output      q, qbar;
  input       preset, clear;

  nand #1     G1 (q, preset, qbar),
              G2 (qbar, clear, q);
endmodule
```



Testbench

```
timescale 1ns / 1ps
module      test_Nand_Latch_1;
 reg        preset, clear;
 wire       q, qbar;

 Nand_Latch_1 M1 (q, qbar, preset, clear);

 always begin
    #20     clear = !clear;
 end

 initial
   begin
              preset = 1'b0;   clear = 1'b1;
    #10
              preset = 1'b1;
 end

endmodule
```

// Simulation Unit / Accuracy
// Testbench module
// Inputs should be declared as reg
// Outputs should be declared as wire

// Instantiate YOUR DESIGN module

// always condition: The description always happens
// The value of clear inverts every 20 ns

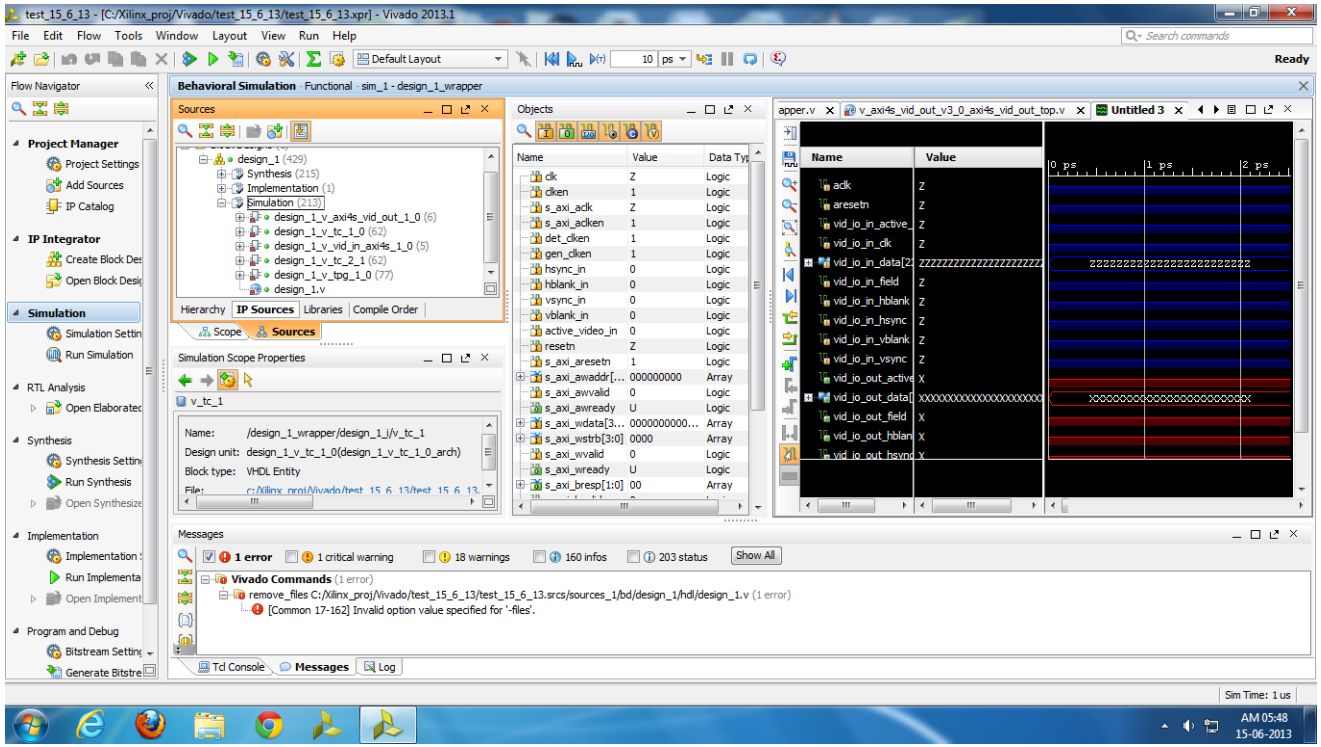
// Initial conditions

// Units of "Simulation Units". In this case, 10ns

*Pictures cited from Prof. Ting-Chi Wang's 2013 slides for education purpose only

Agenda

- Verilog lab 1
- Testbench
- **Vivado Introduction**



*Pictures cited from www.xilinx.com for education purpose only

Vivado Introduction

- Vivado does not support Mac OS (Unfortunately..)
- We will switch to Prof. Wang's slides for Vivado introduction

*Seattle night view taken at LA County Museum of Arts, Los Angeles, CA.
This picture is taken by Chun-Yi Lee himself, who is also a fan of photography