



# Lab2

## Serial Multiplier in Verilog

助教：蔣宗廷

# 課程目的

- 複習 Verilog，設計硬體乘法器
- 使用 Icarus Verilog Simulator 完成設計驗證

# Verilog 簡介

- 在 Verilog 中我們建構各個模組(module)，採「由上而下」階層方式設計硬體
- 利用測試平台(Testbench)驗證設計的功能是否符合需求
- 能夠描述多種層次電路，例如：描述模組功能的行為層次(Behavioral level)、描述邏輯閘連接形式的邏輯層次(Gate level)等

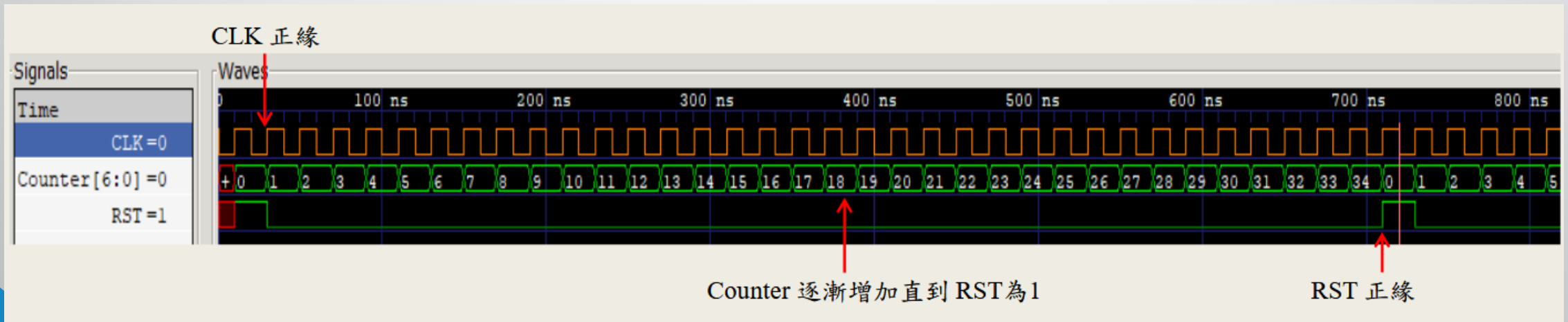
# Verilog架構-Module(1/2)

```
1 //以前者(1ns)為單位，以後者(1ps)的時間，查看一次電路的行為
2 `timescale 1ns/1ps
3
4 //宣告module名稱,輸出入名稱
5 module lab( 自訂Module名稱
6     CLK,
7     RST,
8     in_a,
9     in_b,
10    Product,
11    Product_Valid
12 );
13 // in_a * in_b = Product
14 // in_a is Multiplicand , in_b is Multiplier
15
16 //定義port, 包含input, output 定義 input port
17 input CLK, RST;
18 input [15:0] in_a; // Multiplicand
19 input [15:0] in_b; // Multiplier
20 output [31:0] Product;
21 output Product_Valid; 定義 output port
22
23 reg [31:0] Mplicand; //被乘數
24 reg [15:0] Mplier; //乘數
25 reg [31:0] Product;
26 reg Product_Valid; 宣告reg變數作為儲存空間reg
27 reg [5:0] Counter ; [15:0]表16-bit之reg
28 reg sign; //isSigned
```

# Verilog 架構-Module(2/2)

```
29
30 //Counter
31 always @(posedge CLK or posedge RST)
32 begin
33     if(RST)
34         Counter <= 6'b0;
35     else
36         Counter <= Counter + 6'b1;
37
38 end
```

硬體行為描述



# Verilog架構-Testbench

```
1  `timescale 1ns/1ps
2  `include "lab2.v" Include欲測試的檔案
3
4  //`define SEED 120
5  module tb_lab2();
6
7  reg signed [15:0] in_a;
8  reg signed [15:0] in_b;
9  reg             CLK;
10 reg             reset;
11
12 wire signed [31:0] out;
13 wire             out_valid;
14 //省略部分變數
15 lab m1(CLK, reset, in_a, in_b, out, out_valid);
16
17 initial begin
18
19     //$dumpfile("lab2.vcd"); // gtkwave
20     //$dumpvars;
21     CLK = 1'b0;
22     #20 reset = 1;
23     temp_a = 16'd3;
24     temp_b = 16'd9;
25     #20 reset = 0;
26     /*中略*/
27     #20 reset = 0;
28     #700 $finish;
29
30
31 end
```

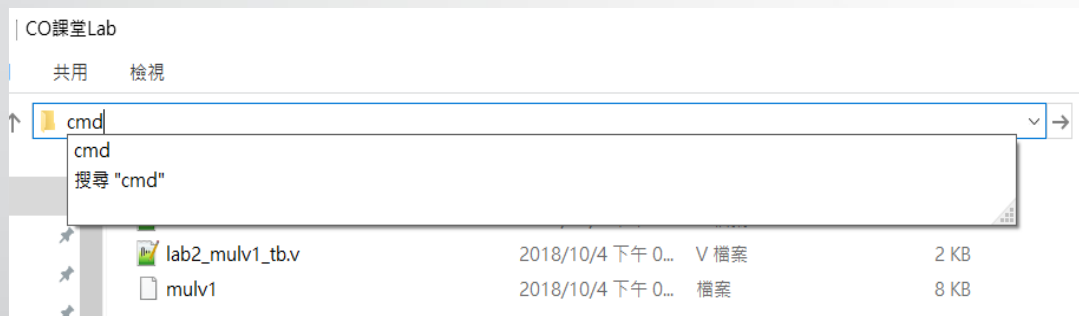
宣告為wire(接線)，作為  
硬體傳遞I/O用途

初始化區域：  
僅在硬體通電後，執行一次

# 範例練習

- 在本實驗中同學將透過 lab2.v 的範例，藉由 Icarus Verilog 進行編譯、模擬驗證
- 使用 Icarus Verilog 的 iverilog、vvp 兩個指令，進行編譯及模擬

## 1. 在程式檔案路徑欄位打開命令提示字元



## 2. 輸入以下指令進行編譯：

- iverilog -o mulv lab2\_mulv1\_tb.v
- vvp mulv

```
C:\Windows\System32\cmd.exe
Microsoft Windows [版本 10.0.17134.285]
(c) 2018 Microsoft Corporation. 著作權所有，並保留一切權利。

C:\Users\ASUS\Desktop\COLab02\CO課堂Lab>iverilog -o mulv lab2_mulv1_tb.v

C:\Users\ASUS\Desktop\COLab02\CO課堂Lab>vvp mulv

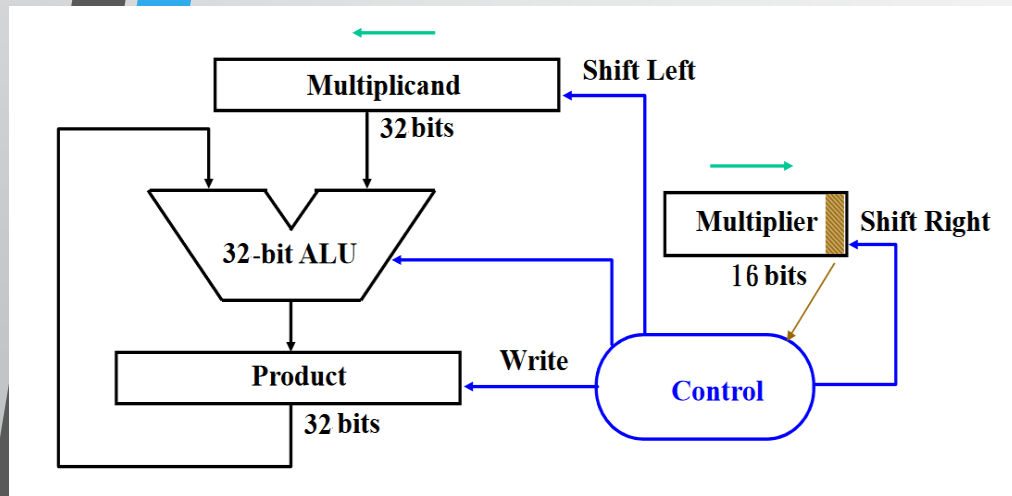
//////////////////
// Successful //
//////////////////
doing    3 *      9 ...

your answer is          27,
correct answer is       27

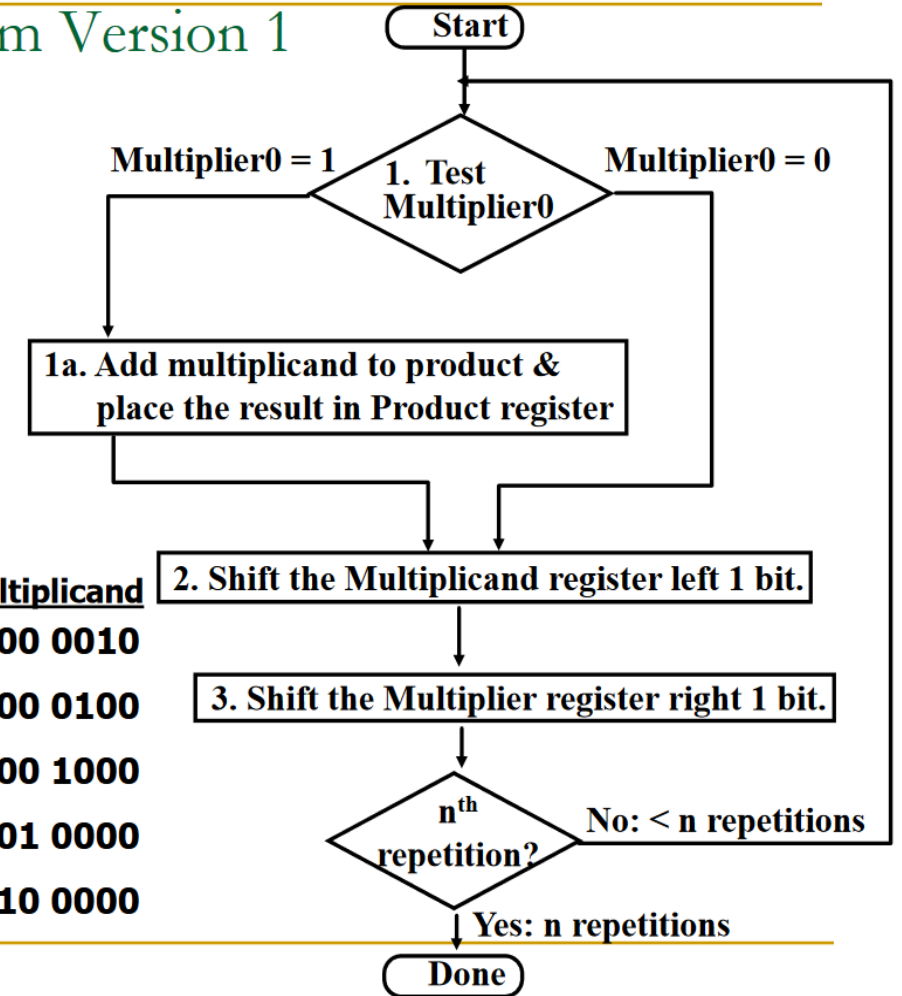
//////////////////
// Successful //
//////////////////
doing   1111 *   1001 ...

your answer is    1112111,
correct answer is  1112111
```

# Serial Multiplier



## Multiply Algorithm Version 1



Product	Multiplier	Multiplicand
0000 0000	0011	0000 0010
0000 0010	0001	0000 0100
0000 0110	0000	0000 1000
0000 0110	0000	0001 0000
0000 0110	0000	0010 0000
0000 0110		
0000 0110		

0000 0110

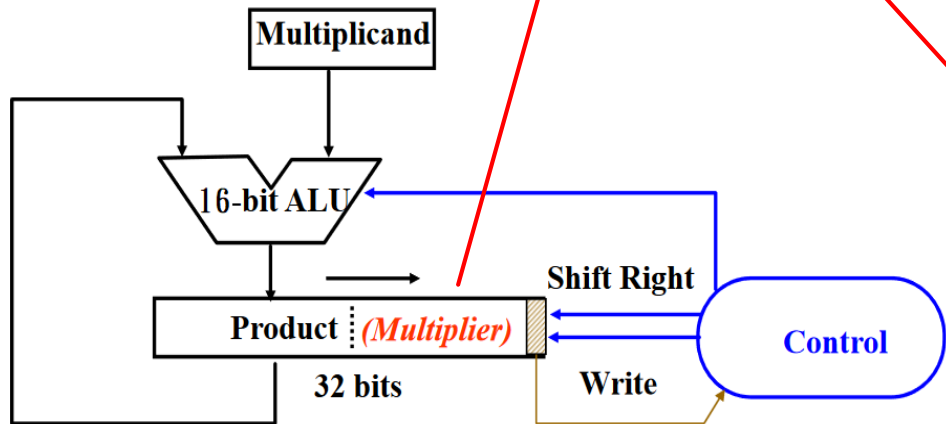


# 課堂作業

- 依照範例練習之流程跑模擬 lab2.v、lab2\_mulv1\_tb.v，並demo給助教看。
  - iverilog -o mulv lab2\_mulv1\_tb.v
  - vvp mulv
- 修改 lab2.v 使其可以在 lab2\_mulv1\_tb.v 做負數運算。
  1. 判斷 in\_a 及 in\_b 是否為負數，若為負數則對其做二補數轉換。  
二補數轉換：if in\_b[15] == 負數，則 Mplier <= ~in\_b + 1'b1; (in\_a同理)。
  2. 若 in\_a, in\_b 一正一負則結果為負，需要將運算結果做二補數轉換。  
對 in\_a, in\_b 做 XOR 判斷運算結果的正負號：sign = in\_a[15] ^ in\_b[15]; (reg sign)  
利用 sign 決定結果是否需要做二補數轉換：if sign == 負數，則結果 <= ~(結果-1'b1)

# Optimized Serial Multiplier

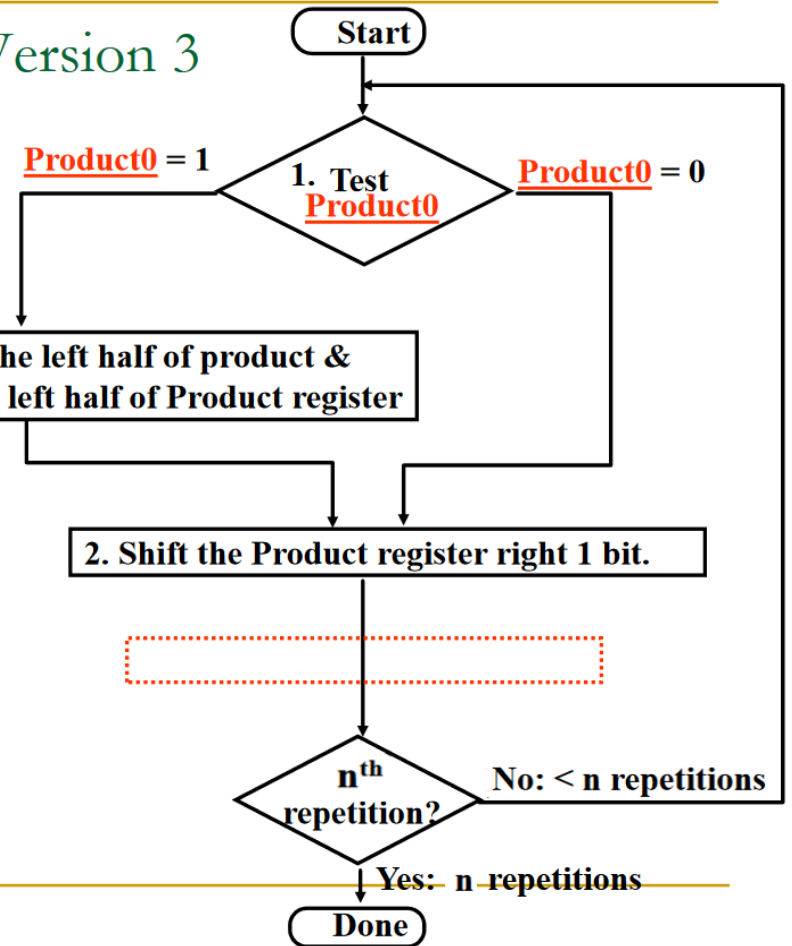
注意一開始這裡的Product要先加上Multiplier



## Multiply Algorithm Version 3

Multiplicand Multiplier  
0010 0011

	Product	Multiplicand
	0000 0011	0010
1:	0010 0011	0010
2:	0001 0001	0010
1:	0011 0001	0010
2:	0001 1000	0010
1:	0001 1000	0010
2:	0000 1100	0010
1:	0000 1100	0010
2:	0000 0110	0010
	0000 0110	0010



# 回家作業

- 修改範例程式“lab2.v” 使用PPT p.10 頁的 Optimized Serial Multiplier 方法在不更動 Testbench 程式的情況下測試是否正確，使其能執行V3版本的 signed 以及 unsigned 乘法運算。
- testbench：“lab2\_mulv3\_signed.v” 以及 “lab2\_mulv3\_unsigned.v” 分別對應模擬測試乘法器功能。

## 作業上傳格式：

- 將三個.v檔壓縮為一個.zip壓縮檔上傳  
“學號-mulv3.v”  
“lab2\_mulv3\_signed.v”  
“lab2\_mulv3\_unsigned.v”
- 以學號命名  
(補充:請記得修改testbench內的include檔名)

# 評分標準

- 課堂實作 40%
- 回家作業 60%
  - 執行lab2.\_mulv3\_signed.v測試功能正確(1/3)
  - 執行lab2.\_mulv3\_unsigned.v測試功能正確(1/3)
  - 前10名上傳,且前兩點皆正確(1/3)
  - 逾期補交九折計算
  - 未交不予計分計算
- 繳交期限：11/7 23:59

# 附錄

- Notepad++安裝方法
- Icarus Verilog安裝方法

# 安裝 Notepad++



- <https://notepad-plus-plus.org/download/v7.5.8.html>  
點選空紅框處下載解壓縮

## Download 64-bit x64

- Notepad++ Installer 64-bit x64: Take this one if you have no idea which one you should take.
- Notepad++ zip package 64-bit x64: Don't want to use installer? Check this one (zip format).
- Notepad++ 7z package 64-bit x64: Don't want to use installer? 7z format.
- Notepad++ minimalist package 64-bit x64: No theme, no plugin, no updater, quick download and play directly. 7z format.
- SHA-1/MD5 digests for binary packages: Check it if you're paranoid.

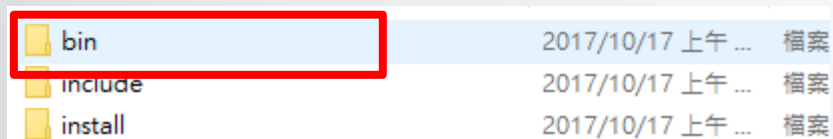
- 點選notepad++.exe執行

license.txt	2016/12/27 上午 ...	文字文件	17 KB
notepad++.exe	2018/7/23 上午 0...	應用程式	3,333 KB
readme.txt	2016/12/27 上午 ...	文字文件	2 KB

# 安裝 Icarus Verilog

- 在本次實驗課，同學將使用 Icarus Verilog 的 iverilog、vvp、gtkwave 來模擬及觀測 8-bit adder 的執行結果和波形

## 1. 將附檔解壓縮後打開bin資料夾



## 2. 依序安裝執行檔：iverilog.exe、vvp.exe、gtkwave.exe



Mac 安裝方式

<http://easonchang.logdown.com/posts/649863>

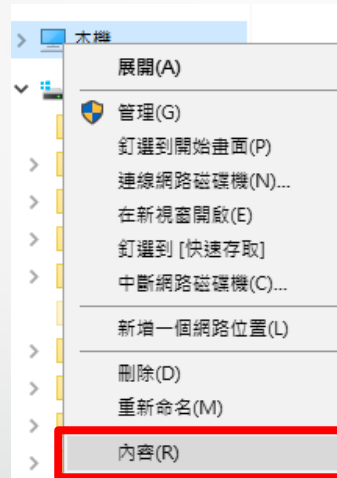
# 設定環境變數 (1/2)

- 避免同學將程式全放在bin資料夾編譯、執行，請同學依照下面步驟操作：

## 1. 打開檔案總管



## 2. 在本機圖示點擊右鍵，選擇內容



## 3. 點擊進階系統設定



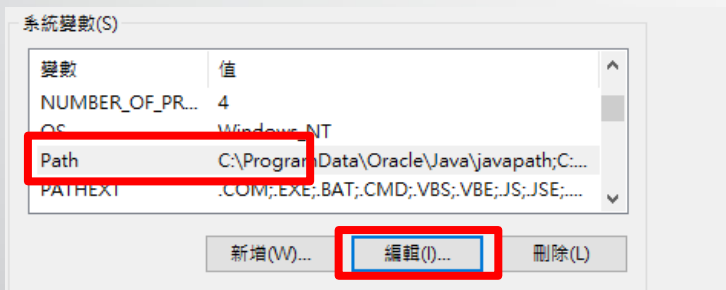


# 設定環境變數 (2/2)

## 4. 點擊環境變數



## 5. 點擊path並按下編輯



## 6. 新增並輸入bin資料夾路徑，按下確定

※路徑為iverilog與gtkwave下的bin資料夾，

助教已將資料放置同處，同學只需新增一個環境變數

