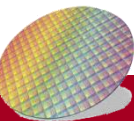


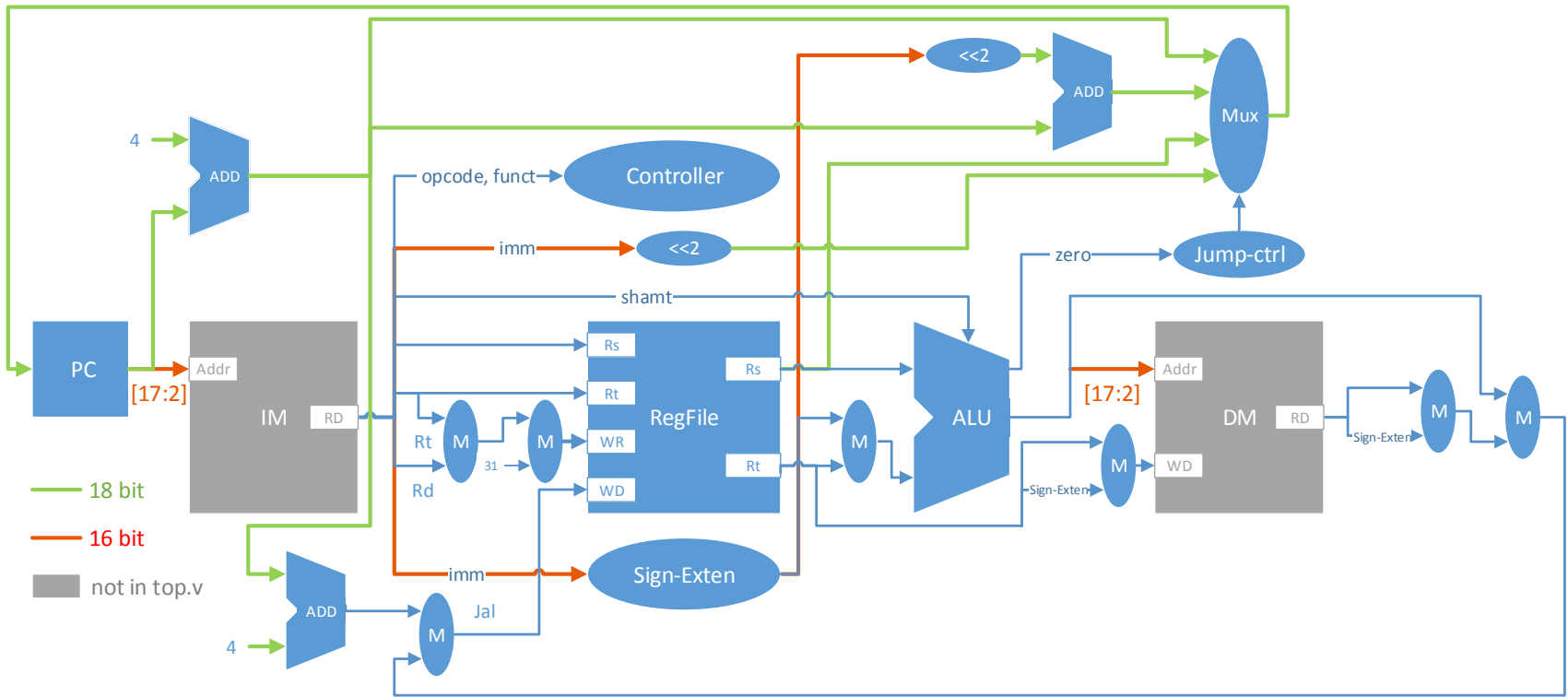


成功大學

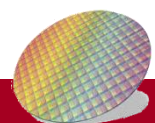
National Cheng Kung University

Computer Organization Homework 2 - Single-cycle CPU Design





Reading: Chapter 4.1 ~ 4.4





R-type

Assembler Syntax

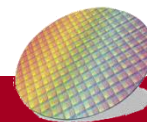


R-type Instruction Machine Code Format



opcode	Mnemonics	SRC1	SRC2	DST	funct	Description
000000	nop	00000	00000	00000	000000	No operation
000000	add	\$Rs	\$Rt	\$Rd	100000	$Rd = Rs + Rt$
000000	sub	\$Rs	\$Rt	\$Rd	100010	$Rd = Rs - Rt$
000000	and	\$Rs	\$Rt	\$Rd	100100	$Rd = Rs \& Rt$
000000	or	\$Rs	\$Rt	\$Rd	100101	$Rd = Rs Rt$
000000	xor	\$Rs	\$Rt	\$Rd	100110	$Rd = Rs \wedge Rt$
000000	nor	\$Rs	\$Rt	\$Rd	100111	$Rd = \sim(Rs Rt)$
000000	slt	\$Rs	\$Rt	\$Rd	101010	$Rd = (Rs < Rt) ? 1 : 0$
000000	sll		\$Rt	\$Rd	000000	$Rd = Rt \ll shamt$
000000	srl		\$Rt	\$Rd	000010	$Rd = Rt \gg shamt$
000000	jr	\$Rs			001000	$PC=Rs$
000000	jalr	\$Rs			001001	$R[31] = PC + 8 ;$ $PC=Rs$

Figure 1. R-type MIPS instructions





I-type

Assembler Syntax

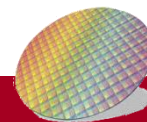


I-type Instruction Machine Code Format



opcode	Mnemonics	SRC1	DST	SRC2	Description
001000	addi	\$Rs	\$Rt	imm	$Rt = Rs + imm$
001100	andi	\$Rs	\$Rt	imm	$Rt = Rs \& imm$
001010	slti	\$Rs	\$Rt	imm	$Rt = (Rs < imm) ? 1 : 0$
000100	beq	\$Rs	\$Rt	imm	If($Rs == Rt$) $PC = PC + 4 + imm$
000101	bne	\$Rs	\$Rt	imm	If($Rs \neq Rt$) $PC = PC + 4 + imm$
100011	lw	\$Rs	\$Rt	imm	$Rt = Mem[Rs + imm]$
100001	lh	\$Rs	\$Rt	imm	$data = Mem[Rs + imm]$ $Rt = data[15:0] \leftarrow \text{Sign-extend } 16\text{bits}$
101011	sw	\$Rs	\$Rt	imm	$Mem[Rs + imm] = Rt$
101001	sh	\$Rs	\$Rt	imm	$data \leftarrow Rt[15:0] \text{ Sign-extend } 16\text{bits}$ $Mem[Rs + imm] = Rt$

Figure 1. I-type MIPS instructions





J-type

Assembler Syntax

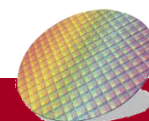


J-type Instruction Machine Code Format



opcode	Mnemonics	Address	Description
000010	j	jumpAddr	PC = jumpAddr
000011	jal	jumpAddr	R[31] = PC + 8 ; PC = jumpAddr

Figure 1. J-type MIPS instructions





General rules for deliverables

- Complete this homework INDIVIDUALLY.
- When submitting your homework, compress all files into a single **zip** file, and upload the compressed file to **Moodle**.

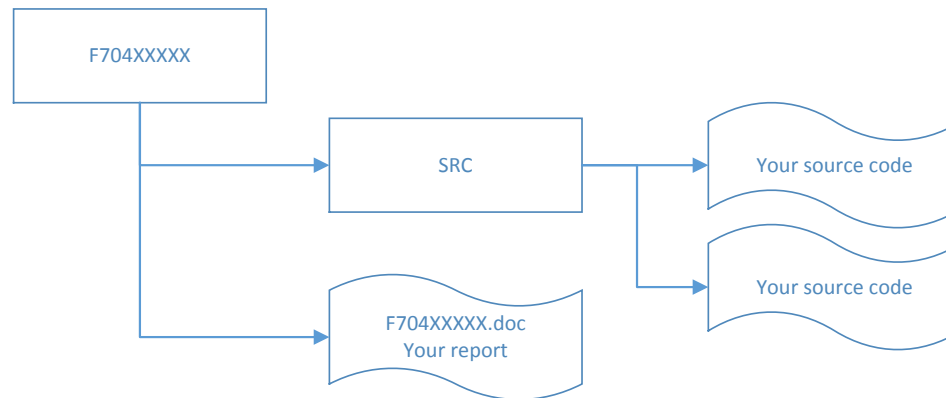
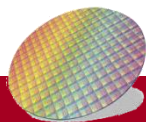


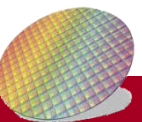
Figure 1. File hierarchy for homework submission





Homework requirement

- Complete the Single cycle CPU that can execute all the instructions from the MIPS ISA section.
 - top.v (your single-cycle cpu)
 - Controller.v
 - Regfile.v
 - ALU.v
 - PC.v
 - Jump_Ctrl.v

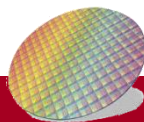




Homework requirement-2

- Verify your CPU with the benchmark and take a snapshot (e.g. Figure 6)

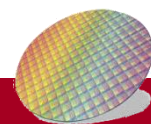
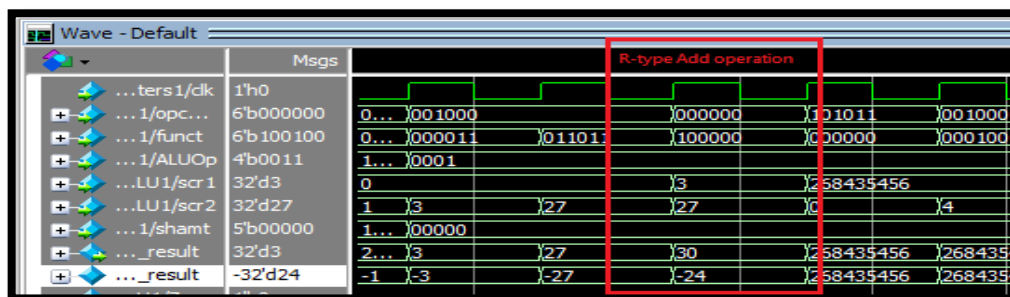
```
VSIM 16> run -all
# [ testfixture1.v ] Rtype test START !!
# =====
#
# \(^o^)/ The Rtype result of DM_data is PASS!!!
#
# =====
# [ testfixture1.v ] Itype test START !!
# =====
#
# \(^o^)/ The Itype result of DM_data is PASS!!!
#
# =====
# [ testfixture1.v ] Jtype test START !!
# =====
#
# \(^o^)/ The Jtype result of DM_data is PASS!!!
#
# =====
#
#
# Single Cycle CPU
# *****
# **                               **
# ** Congratulations !!           **
# **                               **
# ** Simulation PASS!!            **
# **                               **
# **                               **
# **                               **
# *****
# student ID :
#
#
```





Homework requirement-3

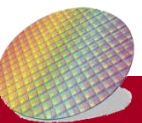
- Take snapshot





Homework requirement-4

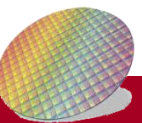
- Final Report
 - a. Complete the project report. The report template is provided.
 - b. If your CPU datapath is different from Figure 5, submit the Verilog files of your CPU design and explain how it works.





常見問題 - Outline

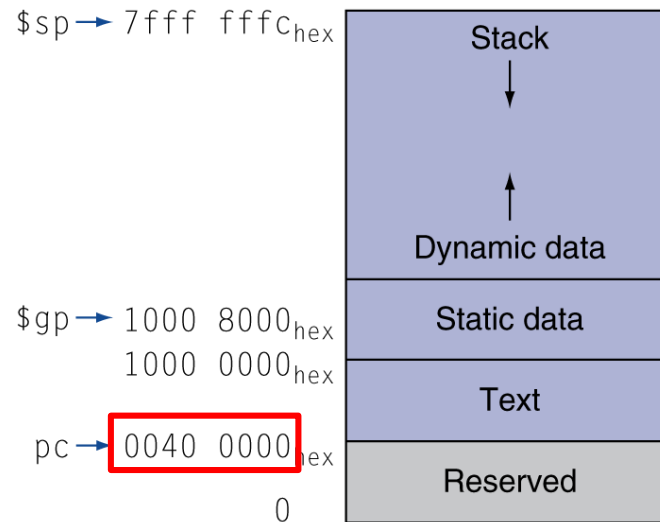
- Q: 為甚麼PC是 18 bits ?
IM & DM 的Address 是 16 bits?
- Q: 為甚麼沒有ALU control ?
- Q: 請問Jump_ctrl的功能為何?
- Q: 是不是所有藍色的線都是 32bits?
- Q: 作業2中的ADD如何實現?
可以加入其他.v檔嗎?
- Q: 無法讀取tb1資料夾下的data檔 該怎麼辦?



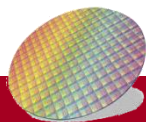


常見問題

- Q: 為甚麼PC是 18 bits ?
IM & DM 的Address 是 16 bits?
- A: 方便電路實現
IM & DM 的 data 為 32bits, 故Address長度為 $18 \gg 2 = 16$ bits



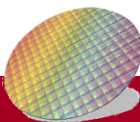
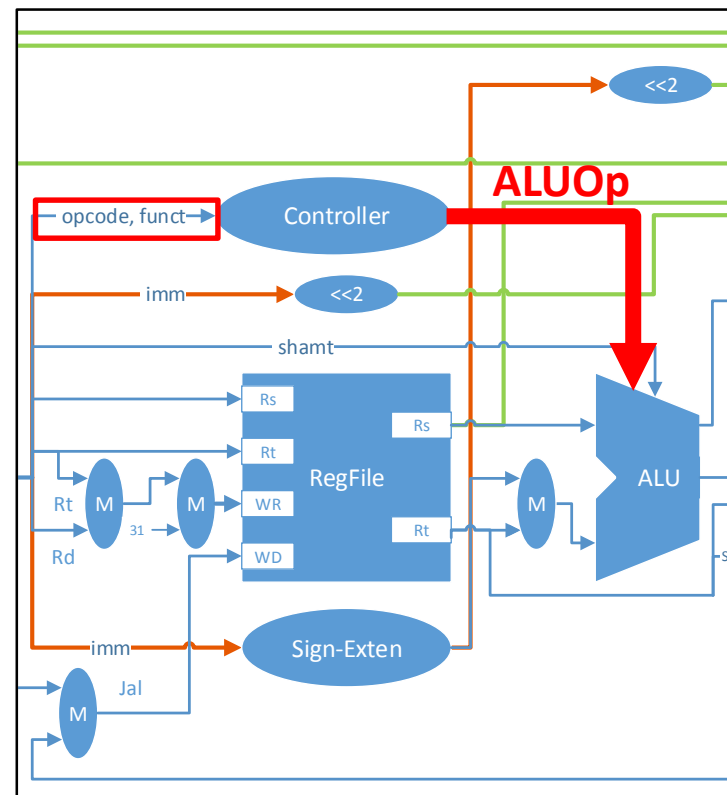
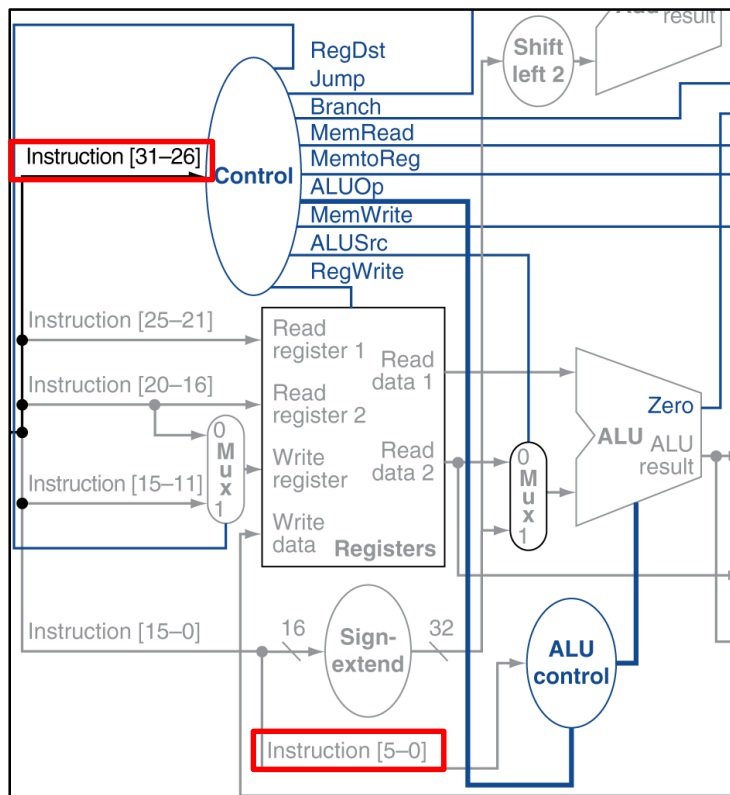
只取後18 bits
PC = 0x0_0000





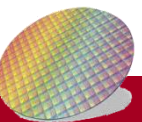
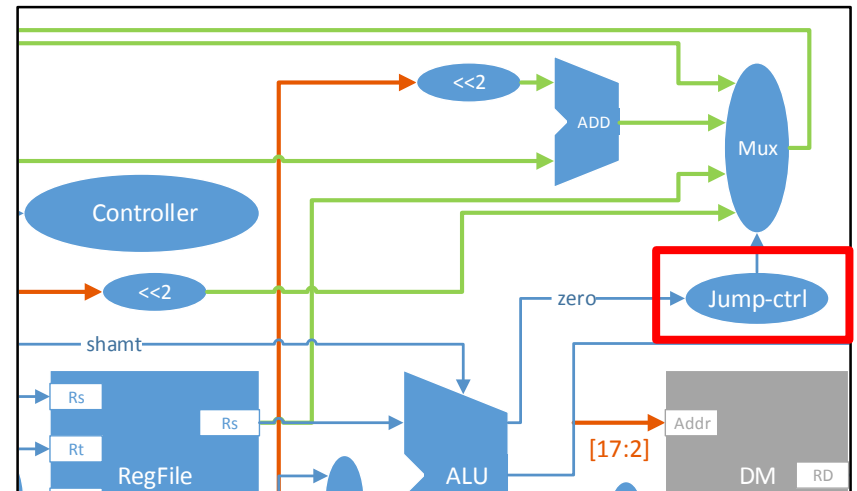
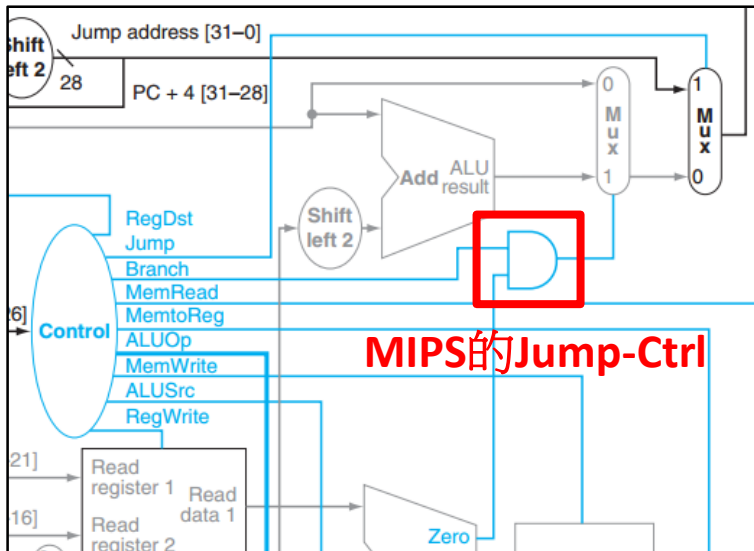
常見問題

- Q : 為甚麼沒有ALU control ?
- A : 在Controller裡面實現



常見問題

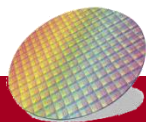
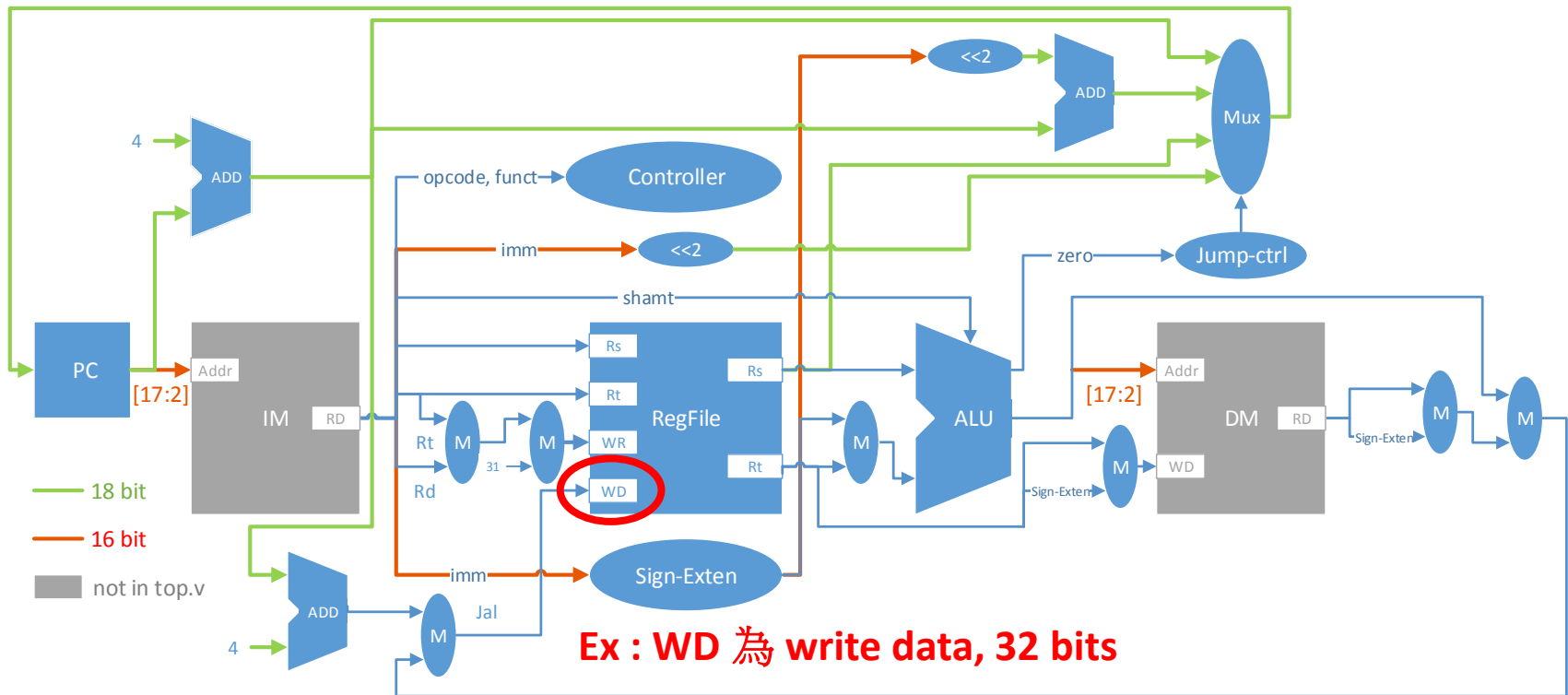
- Q : 請問Jump_ctrl的功能為何?
- A : 比較方便管理Jump行為





常見問題

- Q: 是不是所有藍色的線都是 32bits?
- A: 不一定 可以從module輸入端來判斷位元

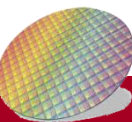
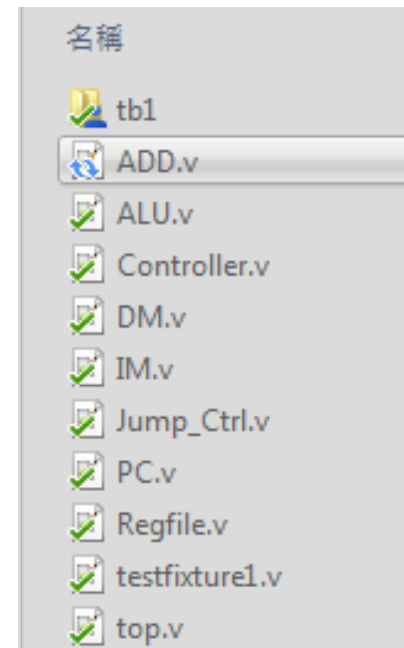




常見問題

- Q : 作業2中的ADD如何實現？
可以加入其他.v檔嗎？
- A : 在top.v中使用assign語法 or 加入ADD.v
可以

```
top.v x  
1 assign C = A + B;
```





常見問題

- Q : 無法讀取tb1資料夾下的data檔 該怎麼辦?
- A : 請將tb1資料夾放在建project的資料夾底下

```
# ** Warning: (vsim-7) Failed to open readmem file "./tb1/IM_data.dat" in read mode.
#
# No such file or directory. (errno = ENOENT) : [REDACTED]
# Time: 0 ps Iteration: 0 Instance: /cpu_tb
# ** Warning: (vsim-7) Failed to open readmem file "./tb1/golden_DM.dat" in read mode.
#
# No such file or directory. (errno = ENOENT) : [REDACTED]
# Time: 0 ps Iteration: 0 Instance: /cpu_tb
```

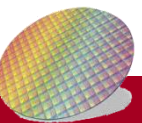
Layout NoDesign

ColumnLayout AllColumns

Project目錄位置

Project - C:/Users/[REDACTED]/740XXXXX/SRC/HW2

Name	Status	Type	Order	Modified
DM.v	✓	Verilog	3	12/02/2016 05:38:54 ...
ALU.v	✓	Verilog	1	01/21/2017 08:49:13 ...
IM.v	✓	Verilog	4	01/20/2017 11:56:15 ...
Regfile.v	✓	Verilog	6	01/21/2017 11:16:57 ...
top.v	✓	Verilog	0	02/10/2017 07:45:07 ...
PC.v	✓	Verilog	5	02/10/2017 10:52:53 ...
Controller.v	✓	Verilog	2	01/23/2017 11:12:57 ...
testfixture1.v	✓	Verilog	7	01/17/2017 03:52:50 ...



Q & A time

