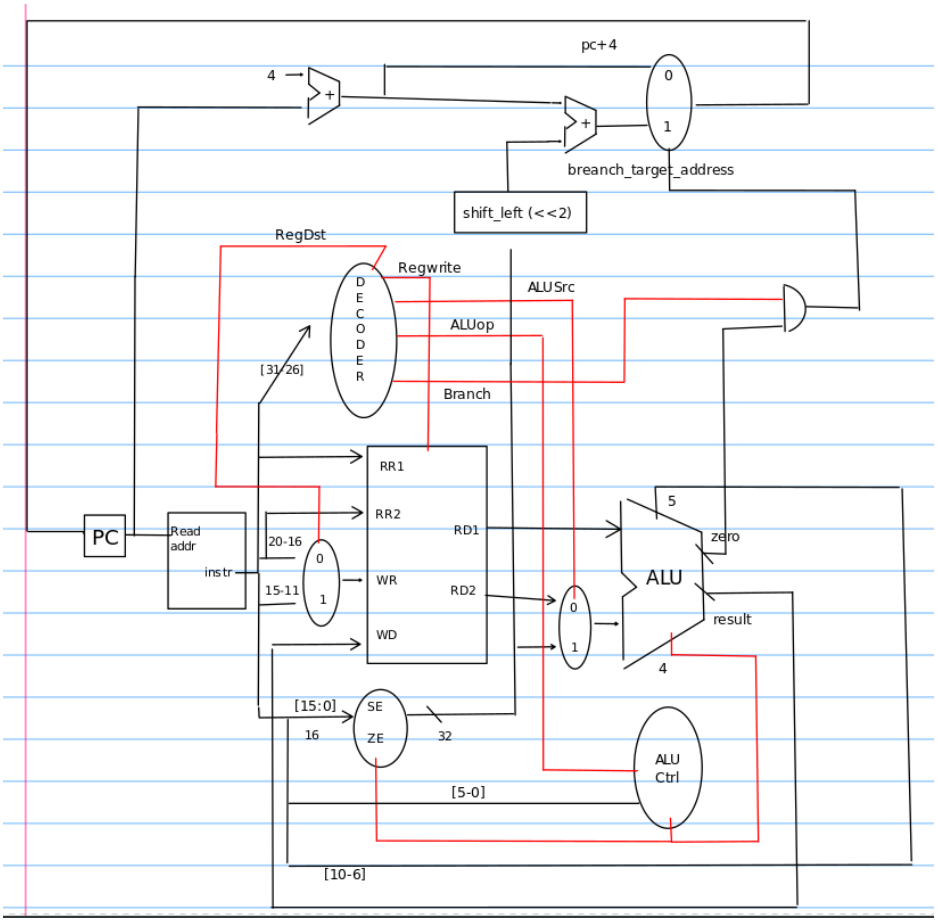
計算機組織LAB2

Architecture diagram

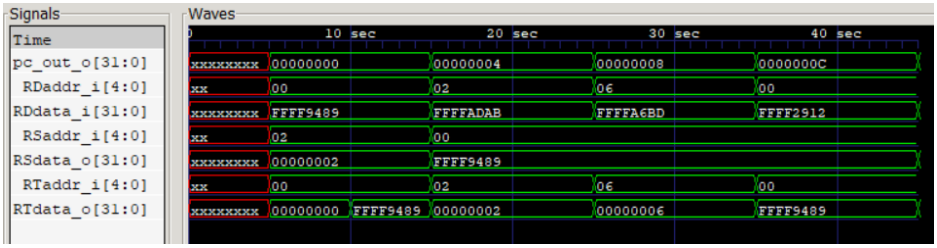


Module description

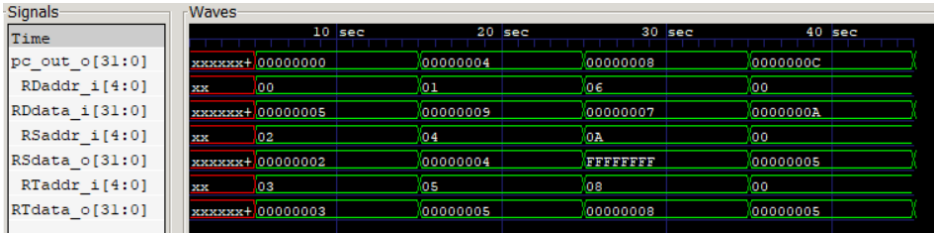
Module	目的
Adder	用於pc+4以及計算branch-target-address
ALU	計算各個所需功能
ALU_ctrl	1.控制ALU現在要做什麼樣的計算
	2.控制SE選擇Sign_Extend或是Zero_extend
Decoder	1.查看OPCode 2.傳值RegDst，ALUSrc等等
	3.根據OPCode傳相對應的ALUop給ALU_control
instr_Memory	讀取指令
MUX_2to1	根據Decoder傳送相對的值
ProgramCounter	下一個指令位置
Shift_Left_Two_32	將要Branch的相對位置往左shift 2bit 之後與PC+4相加就可以得到Branch_Target_Address
Sign_Extend	根據ALU_control做Sign_Extend或是Zero_extend
Simple_Single_CPU	Main file 傳入各個Module所需input以及output

Waveform

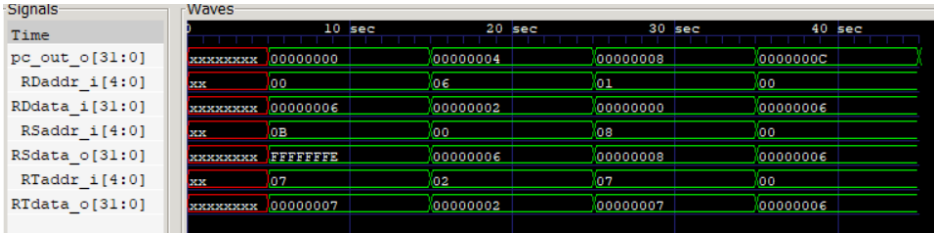
addi



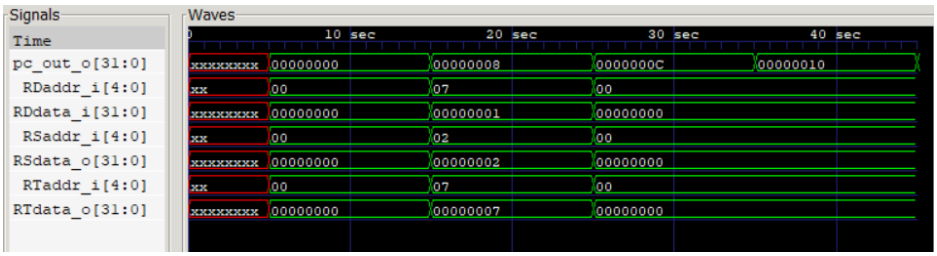
addu



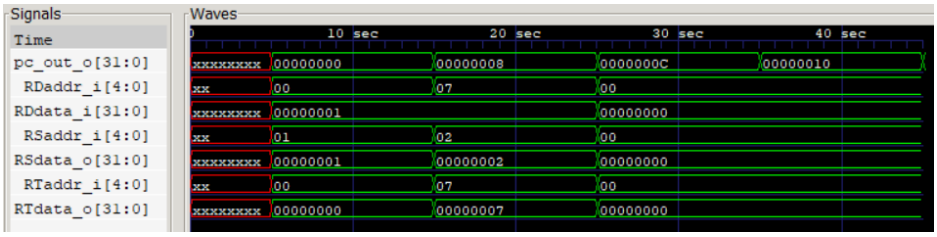
and



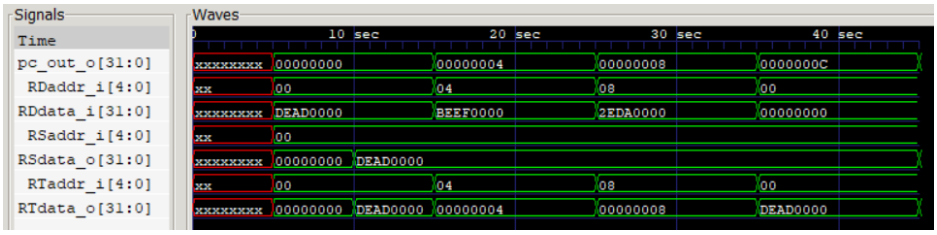
beq



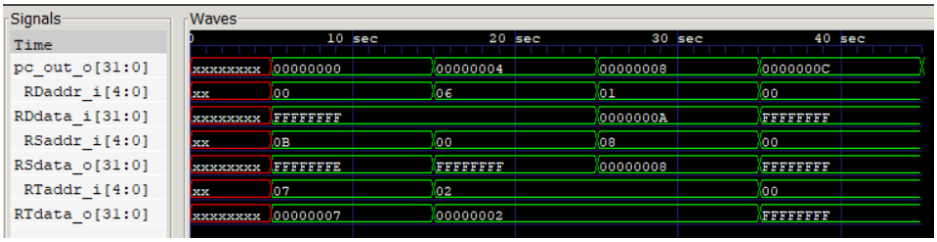
bne



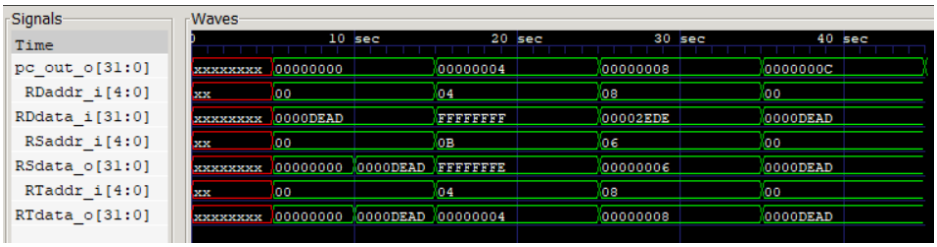
lui



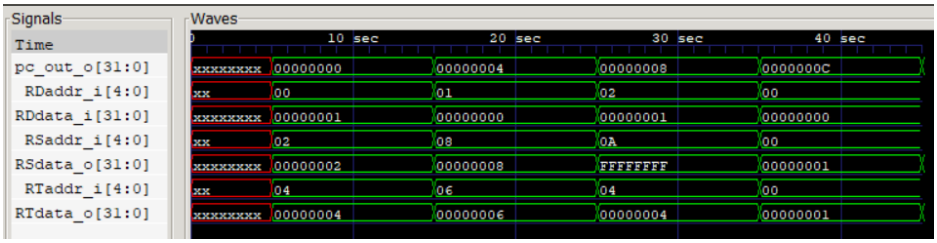
or



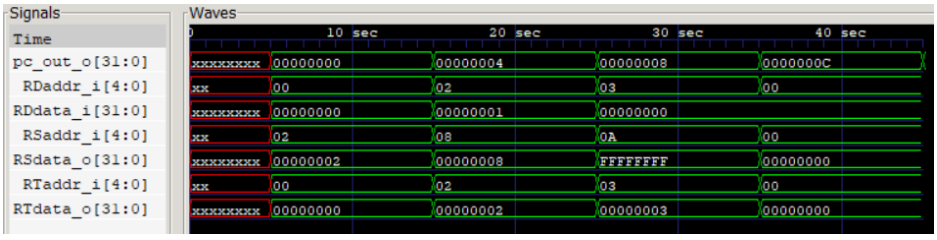
ori



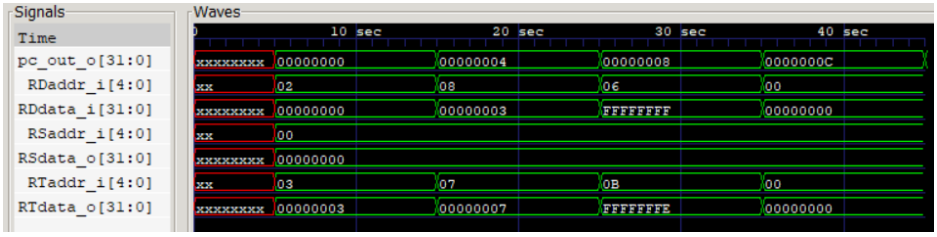
slt



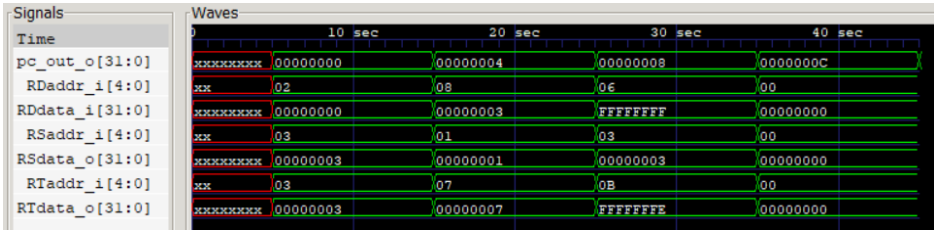
sltiu



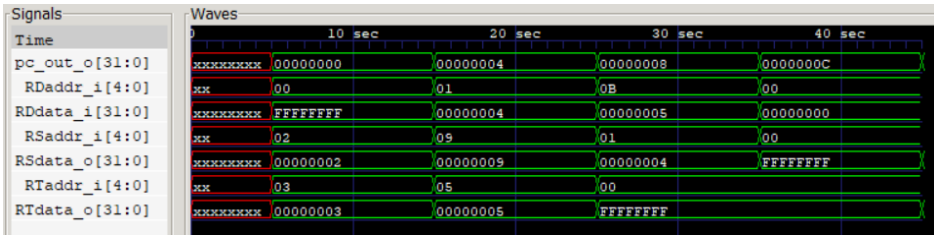
sra



srav



subu



Questions

What is the difference between “input [15:0] input_0” and “input [0:15] input_0” inside the module?

若宣告成[15:0]表示最高位(MSB)index為15，最低位(LSB)index為0
若宣告成[0:15]表示最高位(MSB)index為0，最低位(LSB)index為15

What is the meaning of “always” block in Verilog?

發生了指定的事件(event)時，“always” block 裡的指令就被執行
若表示為always(*) 表示只要有任何變動便會觸發

What are the advantages and disadvantages of port connection by order and port connection by name in Verilog?

Type	Advantage	Disadvantage
by order	允許有port沒有連接 (但傳過去會變成高組態)	要依照parameter的順序給值不可以任意更動
by name	1.不用按照parameter順序填寫	不允許有port沒有連接
	2. 可讀性較佳 (知道目前要傳給哪個參數)	

Contribution

1. 架構:0711282 邱頎霖
2. Code: 0711282 邱頎霖 0711077 柯柏揚
3. Debug以及確認答案： 0711077 柯柏揚

Discussions, Problem Encountered

1. 在使用 >>> 時沒注意到wire性質必須為signed，Google後才發現
2. 因為使用Linux下的Gedit寫Code，沒有變數提示，一度因為大小寫不同沒有結果卡了一段時間，體會到變數命名應該要有一定規則不可以太過隨便
3. 不知道R-type的shamt要怎麼獲得，最後是直接接過去ALU
4. srav的指令格式與一般不太相同(或是我自己認知上的錯誤)

像是測試資料裡頭的 srav r8, r7, r1
對應的機器碼卻是:00000000001001110100000000000111

opcode	rs	rt	rd	shamt	func
0	1	7	8	0	000111

我所認知的應長這樣：

opcode	rs	rt	rd	shamt	func
0	7	1	8	0	000111

不過發現不對回去看測試資料修正後就沒有太大問題了

5. Zero_extend 一開始沒注意到，後來ori一直出錯才發現，最後決定將ALU_control接到Sign_extend判斷是需要Sign_extend還是Zero_extend
6. ALUop與課本不同: 這次給的是3bits，一開始想學課本學的把相同功能歸類到同一ALUop，但後來覺得如果有一對一的對應值Debug較方便
7. 用滑鼠畫架構圖好累XD 而且因為存檔又要修改所以重畫了兩次QQ
8. 雖然報告中沒有規定要附上這些表格，不過還是貼上來希望報告可以完整一點～
助教辛苦了 👍

ALU control(4bit)	功能
0000	AND
0001	OR
0010	ADD
0110	SUB
0111	SLT
0101	SLTiu
1100	NOR
1110	SRA
1111	SRAV
1011	Lui
0011	beq
1001	bne

Type	OPCODE	ALUop	REGdst	ALUSrc	Branch	RegWrite
R	000000(0)	000(0)	1	0	0	1
addi	001000(8)	001(1)	1	1	0	1
sltiu	001011(11)	010(2)	0	1	0	1
beq	000100(4)	100(4)	0	0	1	0
lui	001111(15)	011(3)	0	1	0	1
ori	001101(13)	111(7)	0	1	0	1
bne	000101(5)	110(6)	0	0	1	0