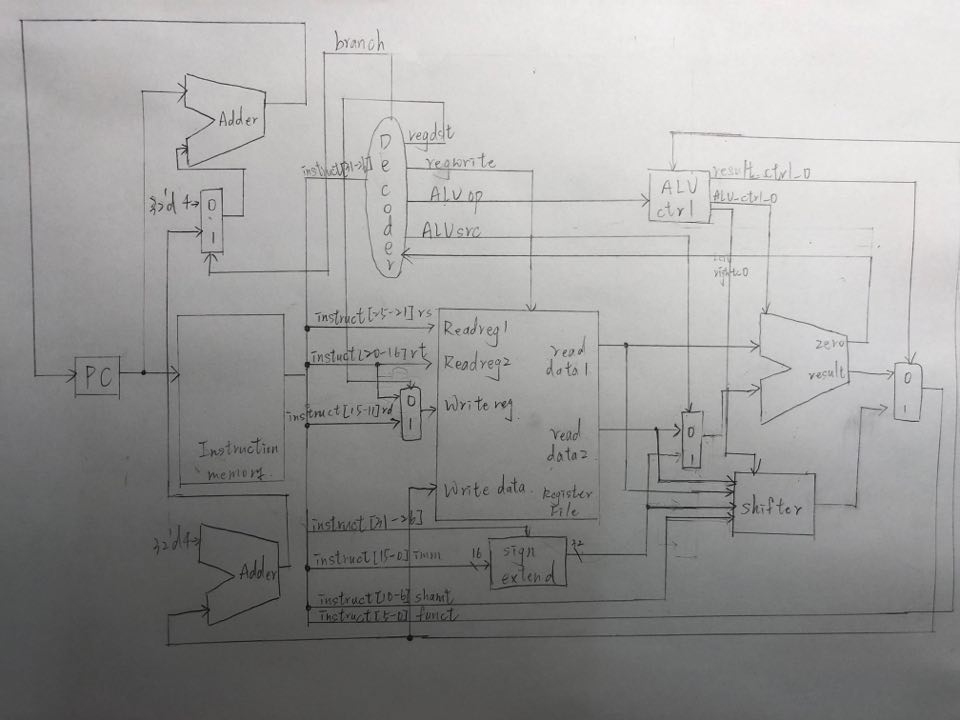
**Computer Organization Lab 2 Report**

Simplified Single-cycle CPU

* Architecture diagram:



* Module description:

PC: 指出下一個instruction的地址給instruction memory

Instruction memory: 根據地址給出instruction

Register file: 根據instruction的rs rt 地址輸出data，將結果data 寫入rd

Adder: 將兩個輸入相加然後輸出

Decoder: 依據Opcode，給出相關資訊讓我們得到正確的ALU結果正確的，以及依據zero給出branch的結果

ALU\_ctrl: 依據decoder的ALUop以及 R-type 的funct知道要對ALU下何種指令，還有對shifter輸入什麼值

ALU: 功能有加 減AND OR STL，STL對signed 跟unsigned都可以，輸出有兩個一個就是上面方法的結果，另一個是zero檢查結果的每一個bit是不是都是零，用在beq bne

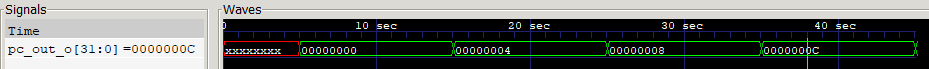
Shifter: 輸入有五個:要如何shift的方法，sign extend，read data1，read data2，shamt，可以因應bne beq sra srav lui 所有情況

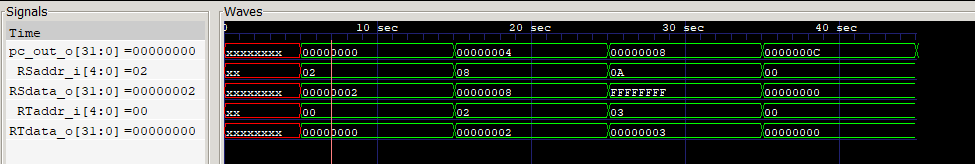
Sign extend: 裡面有分成zero extend跟sign extend前者是不管怎樣都將16bit前面加0，後者是根據第16bit去決定要加0還是1

MUX\_2to1:將兩個輸入根據有一個決定值去決定給下一個module要用什麼值

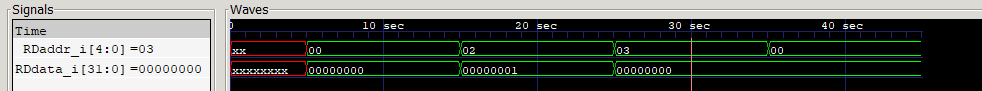
* Waveform

The output of program counter module



rs and rt register selection input and data output

rd register selection input and data input



* Questions
  + Input [15:0] input\_0是把input\_0的MSB設定成input\_0[15] LSB設定成input\_0[0]，而input [0:15]則是把input\_0的MSB設定成input\_0[0] LSB設定成input\_0[15]
  + always block的意思是一直監視著program裡面的值一有發生變化就會立即改變always裡面值的reg
  + by order: 優點是很直覺，根據module的輸入輸出順序去填就可以

缺點是如果module很多的話要記得很多連接的順序是很難的

By name: 優點是不用可慮順序，根據module的名稱去填就可以，缺點是比較不直覺沒有按照順序

* Contribution

我負責sign\_extend ALU MUX，另一位負責decoder alu\_ctrl shifter adder至於bug fixing 跟case generation都是由我們兩個人一起找出的，但應該也沒有說一個module是誰去完成的，我們兩個人都會對對方的module去做提問以及覺得怎麼樣寫可以更好，最後最終的Architecture diagram是由我們兩個一起想過，通過test找出哪裡設計錯誤，最後成功的結果，前面的寫各個module只是最剛開始的分工。

* Discussions, problem encountered and miscellaneous

這次的LAB是第一有接這麼多不同的module，剛開始覺得很難要寫很多module，後來才覺得原來每一個module所做出的工作其實沒有很難，但全不接再一起卻能有如此強大的功能，這也是第一次程式作業是兩個人一起，感覺到了分工合作的強大吧，自己不用全部完成一件事，反倒是了解別人幫自己做的事，自己也幫別人做的事，最後還蠻快的就完成。