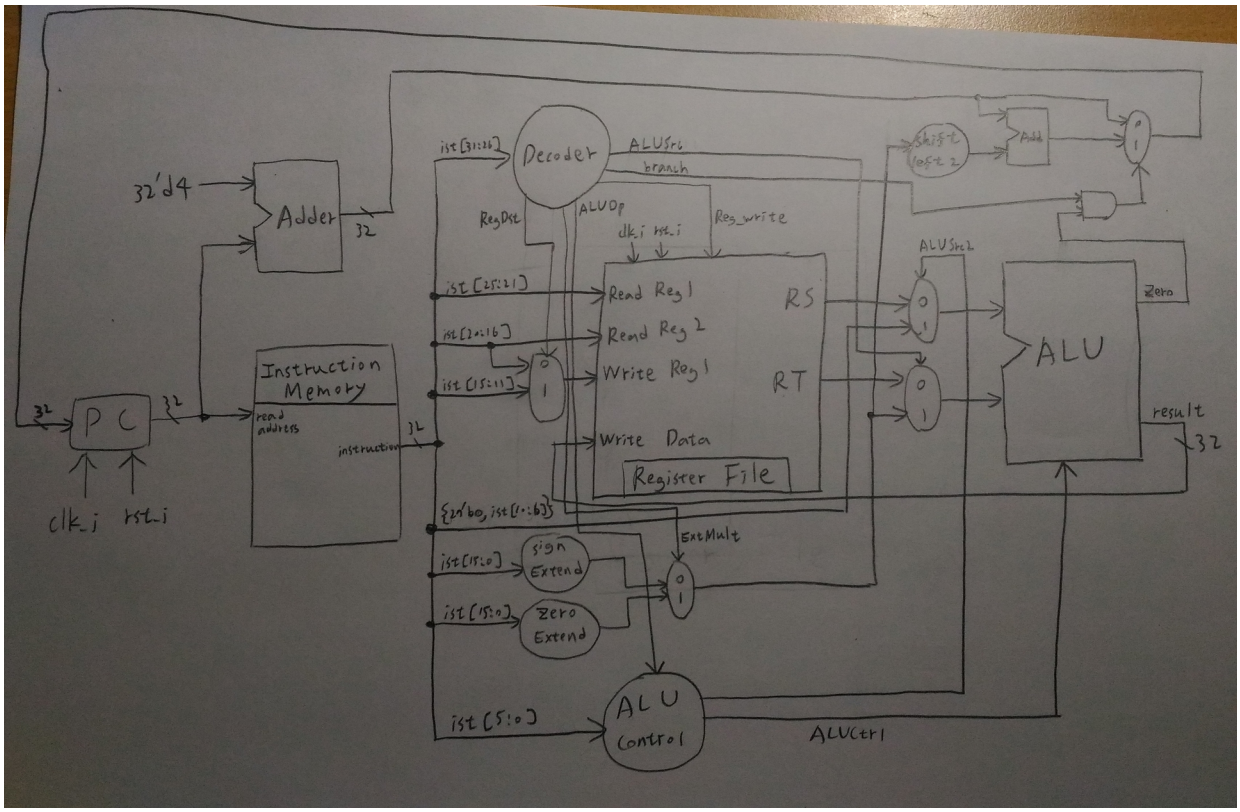
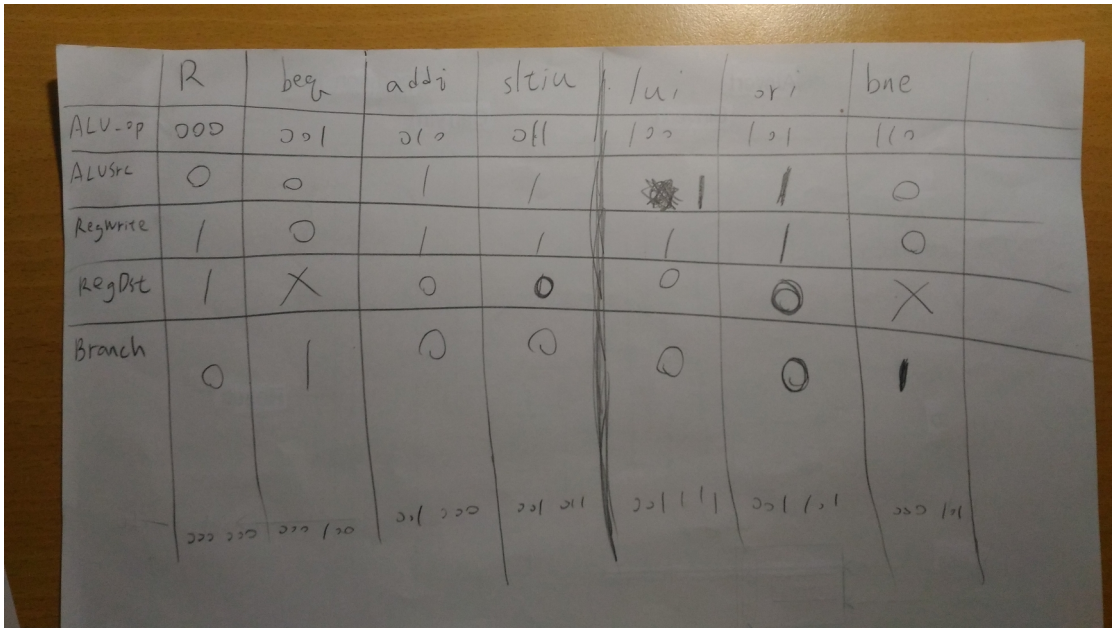
Computer Organization

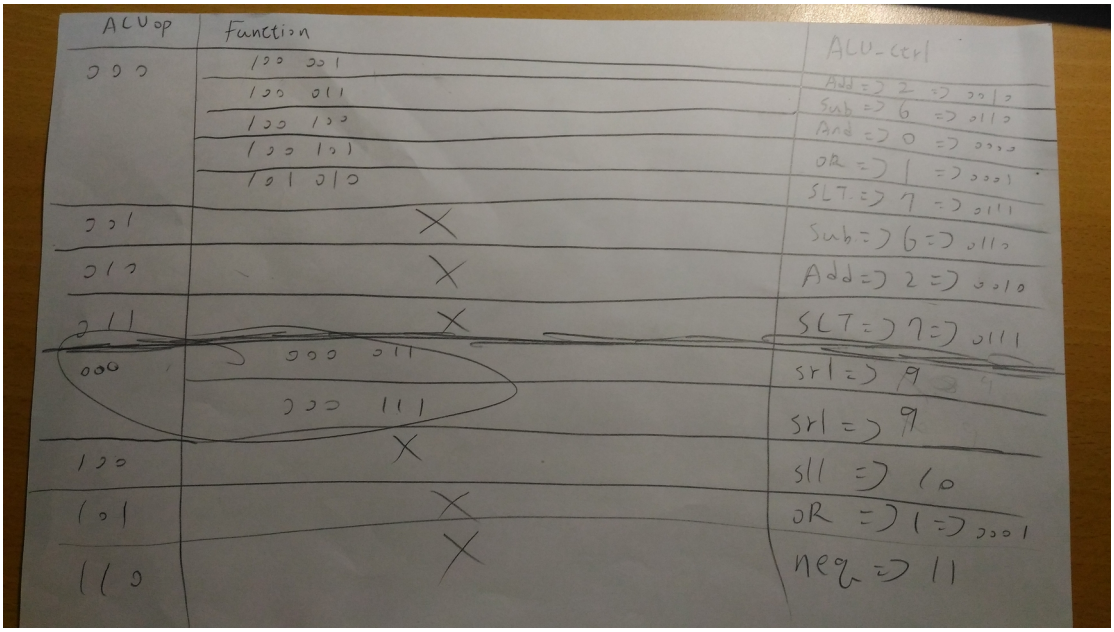
Architecture diagram:



Detailed description of the implementation:

我認為這次作業最重要的點在於 Decoder 以及 ALU_Ctrl 的配置，因此畫了兩張表格，讓自己能對照著來實作





依照這兩張表格就能很清楚的完成 Decoder 以及 ALU_Ctrl 這兩個 module，而其他的 module 依照各個功能來實作後，最後用 Simple_Single_CPU 來把所有 module 接起來就完成了

Problems encountered and solutions:

遇到的問題其實滿多的

一、一開始對於 Decoder 的每項輸出並不是那麼的熟悉，花了好一些時間才了解整體運作方式

二、Basic instruction 的最後，在接 Simple_Single_CPU 時卡了一段時間，也強迫自己好好的全部理解過一次之後才寫出來

三、Basic instruction 其實並不是太難，跟著已經附上的架構圖實作之後很快就可以出來了，但 Advance instruction 卻花了我許多時間，因為要做的每個指令格式都十分特殊：

SRA 不看 RS，只看 RT 和 shamt 來運算，讓我必須修改 ALU 前的 input，並加入 multiplexer 和從 ALU_Ctrl 接出來的 signal 才能完成

SRAV 比較特別的是讓我了解要先轉換 signed variable 才能>>>，這邊因為已經有提示了，所以沒花太多時間

LUI 也沒花太多時間，不過原因是因為 verilog 可以直接利用邏輯的表示法來做左移 16 位的運算，如果要實際修改 ALU 的話，想必是要花更多時間的

ORI 有提到必須做 zero extension，因此又讓我多加了一個 multiplexer 和從 Decoder 接一條 signal 出來才做完這部份的功能

而最後的 bne，這部份讓我要多創一個 register 叫作 set，把 zero 接到 set，再利用 set 做跟 beq 相反的事情才完成

Lesson learnt (if any):

一、Single Cycle CPU 的架構
二、Decoder 發出的 signal 分別做了哪些事情
三、一些之前沒用過的 verilog 語法
四、可以簡單的修改 Single Cycle CPU 的能力