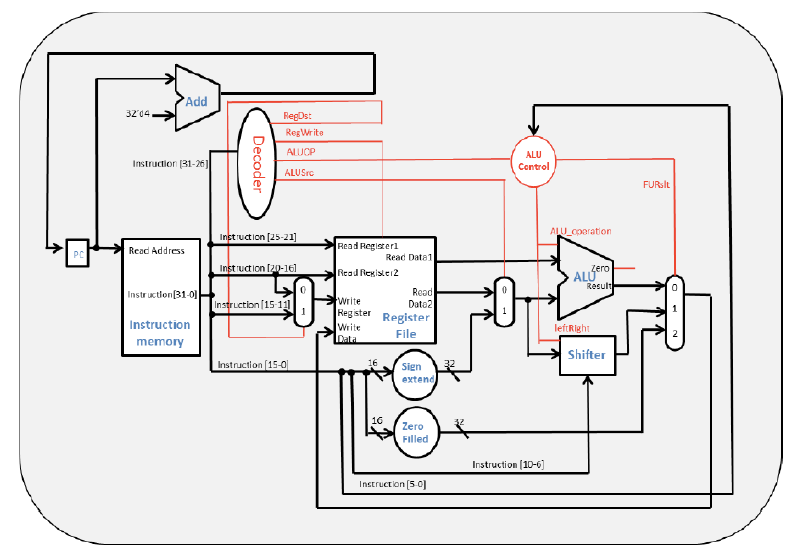
**Computer Organization**

**Architecture diagrams:**

****

**Hardware module analysis:**

**1. Adder**

將兩個32 bits的in put的資料直接相加，實作上直接用assign相加即可。

**2. Decoder**

Input instr\_op，再根據instruction set中的Op field對應出不同的output (RegWrite, ALUOp, ALUSrc, RegDst)。

**3. ALU\_Ctrl**

根據Decoder給出的ALUOp作為input，再input funct，根據instruction field中的Function Field對應給出不同的output (ALU\_operation, FURslt)。

**4. MUX2to1**

根據input的1bit的select\_i，選擇輸出為input1或input2。

**5. MUX3to1**

根據input的2bit的select\_i，選擇輸出為input1,2或3。

**6. Sign\_extend**

將16 bits的data拓展成32 bits的數據，將0-15位元的資料設為原本input的值，並將16-31位元的資料設為input data最高位(第15位元)的值。

**7. Zero\_filled**

將16 bits的data拓展成32 bits的數據，將0-15位元的資料設為原本input的值，並將16-31位元的資料設為0。

**8. ALU\_1bit**

此module為上次的作業內容。根據input (a, b invertA, invertB, op, cin, less)給出ALU計算過後的output。實作上a, b為1 bit的data，依據op及invertA, invertB決定做何種運算。

**9. ALU**

此module為上次的作業內容。利用32個ALU\_bit module去計算32bits 的ALU結果。實作上利用generate及for迴圈做出1到31位元的ALU，第0位元的cin及less需另外接線。

**10. Shifter**

根據leftRight判斷shift left or right。

**11. Simple\_Single\_CPU**

將所有module統整起來，依照architecture diagram接線，做出simple single cycle CPU。

**Finished part:**

完成了所有module並通過助教提供的三個測資

**Problems you met and solutions:**

在實作Decoder及ALU\_Ctrl的時候，搞反operation和function的6 bits input，中間判斷上又寫錯一些東西，結果就大爆炸，花了一些時間搞懂spec中的instruction set的內容，最後才修正成功。

**Summary:**

我覺得最複雜的點應該就是判斷op及funct的data並給出對應個output，要了解每一步每一個module每一條線代表的功用，最後在simple\_single\_CPU中接線，接線只要按照圖上的內容一條一條接就好，挺有趣也挺麻煩的。