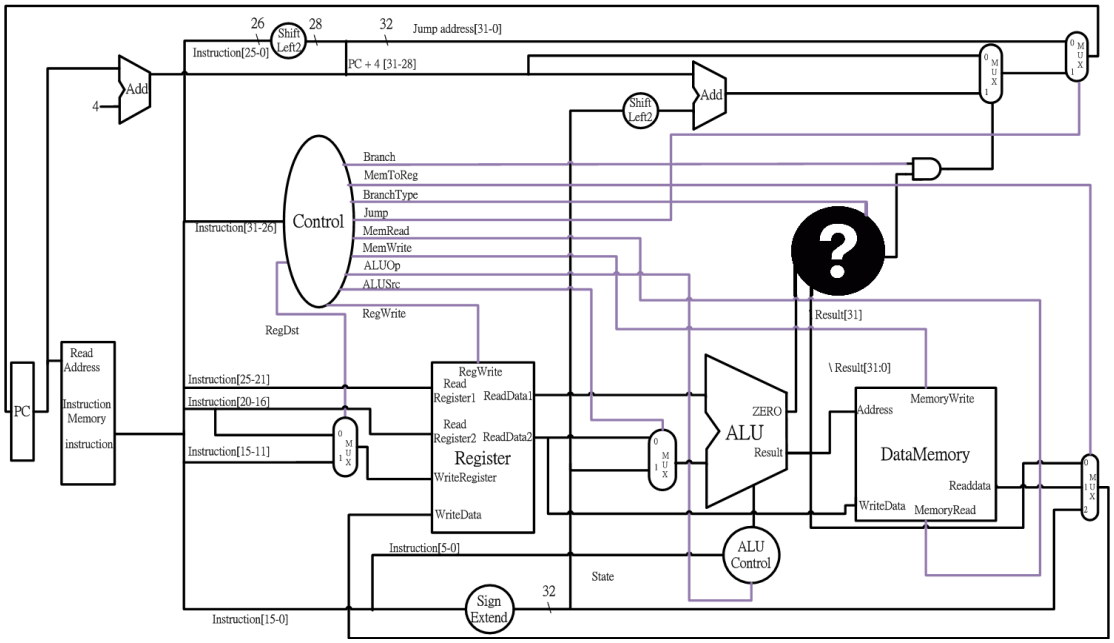
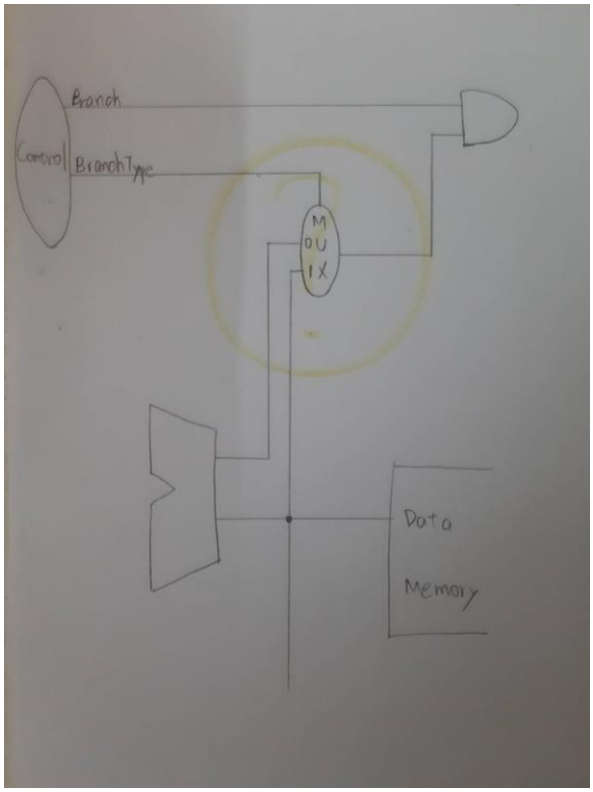
Computer Organization

0613144_葉之晴、0613149_翁羽亮

Architecture diagram:



?部分如下



Detailed description of the implementation:

1. **control unit's output signals**

   (1)Branch_o：

   當分支指令為「not equal」狀態時，分支指令為 0

   (2)Jump_o：

   當 jump 是 j 或 jal 指令時，為 0

   當 jump 是 jr 指令時，為 2

   當 jump 其他狀況時，為 1

   (3)MemRead_o：

   如需讀取記憶體，便會使 MemRead 為 1(例：lw)

   (4)Mem_Write_o：

   如需寫入記憶體，便會使 MemWrite 為 1(例：sw)

   (5)MemtoReg：

   當指令為 lw 時，MemtoReg 為 1

   當指令為 sw 時，MemtoReg 為 2

   當指令其他時，MemtoReg 為 0

2. **Signals for jump address**

   jump_addr_o[1:0] = 2'b00; // jump_addr_o 是 jump 指令的目標位址

   jump_addr_o[28-1:2] = im_out[26-1:0];//im_out 是指令記憶體的輸出

   jump_addr_o[32-1:28] = adder1_sum[32-1:28];//adder1_sum 是 PC+4

   PC+4 的訊號也需要儲存至暫存器 Reg[31]，才可儲存 jump address

3. **MUX**

   (1) 2-to-1 MUX for the PC source:

   (a)選擇輸入訊號 = (Branch 的 MUX 輸出)和(Branch)

   (b)source1 = PC+4

   (c)source0 =branch 指令的位址+(PC+4)

   (2) 3-to-1 MUX for the PC source:

   (a)選擇輸入訊號 = Jump_o

   (b)source0 = jump_addr_o

   (c)source1 =前方 MUX 的輸出

   (d)source2 =Reg[Rs]

   (3) 3-to-1 MUX determining which data will be written into register:

   (a)選擇輸入訊號 = MemtoReg_o from control signal

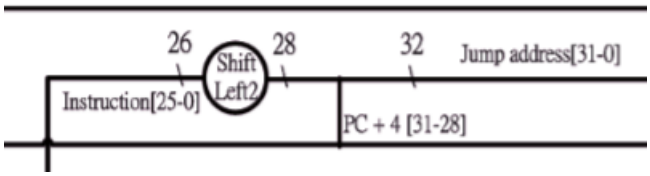
(b)source0 = result of ALU

(c)source1 = output of data memory

(d)source2 = output of sign extension

Problems encountered and solutions:

1. 跳躍位址有必要 Shift Left 2 嗎?如圖



所以將其替換成

jump_addr_o[1:0] = 2'b00;

jump_addr_o[28-1:2] = im_out[26-1:0];

jump_addr_o[32-1:28] = adder1_sum[32-1:28];

發現結果是一樣的，所以其實應該可以省略

Lesson learnt (if any):

在不需要**Shift Left 2** 的狀況下該怎麼實現**jump address**

Jump 指令要怎麼從不同地方(如：暫存器)指派到 PC