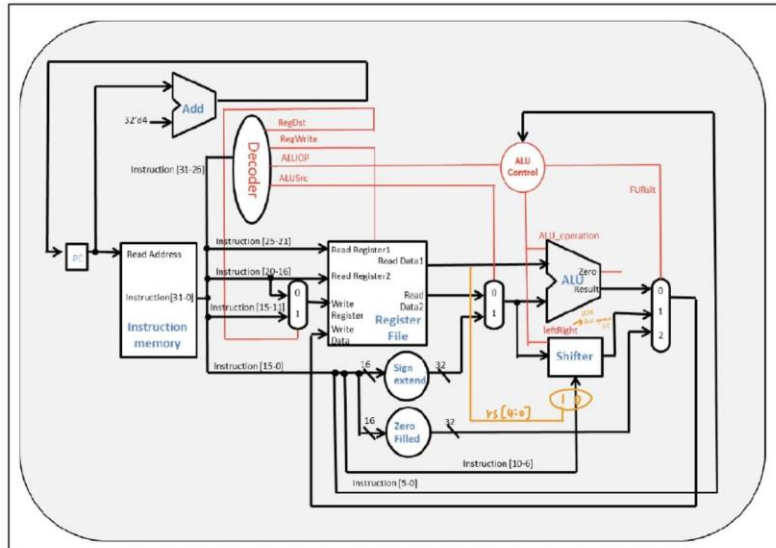


Computer Organization

Architecture diagrams:



Hardware module analysis:

使用助教給的檔案來實行。

- Adder：用來實作 $PC \rightarrow PC + 4$ 。
- Program_Counter：控制甚麼時候進下一個指令。
- Instr_Memory：讀指令。
- Mux2to1：
 - Mux_Write_Reg：控制要寫入的檔案位址。
 - ALU_src2Src：選擇要送入 ALU 的第二筆 data。
 - For_shifter(加分題新增)：選擇要送入 Shifter 的 shamt。
- Mux3to1：選擇要傳哪個運算完的資料進暫存器。
- Reg_File：讀檔案。
- ALU：做運算。
- ALU_Ctrl：根據 Decoder 送出來的 ALUop 與 function field 作分析去決定 ALU 要做哪種運算。
- Decoder：用指令的 op 部分作分析送出控制訊號。
- Shifter：讓跟 shift 有關的指令在此作運算。
- Sign_Extend：把數字的位元擴增到 32 位。
- Zero_Filled：此次的指令沒有用到。
- Simple_Single_CPU：結合上述。

Finished part:

全部的要求皆完成。

Program_Counter、Instr_Memory、Reg_File 沒有動到，全使用助教給的檔案。

Adder：單純相加。

Decoder：用 op 去判斷，如果是 R type 將 RegDst_o 設為 1、ALUSrc_o 設為 0，I type 相反。RegWrite_o 因為這次的指令全部都需寫入檔案，直接設為 1。R type 將 ALUOp_o 設為 2、I type 設為 1。

ALU_Ctrl：依據 ALUOp_i 與 funct_i 定出 ALU_operation_o、FURslt_o、choose_v_o，ALU_operation_o 由附錄制定，FURslt_o 用來控制結果要從 ALU 還是 Shifter 還是 zero_filled 出來(此次未用)，我定 0 為 ALU、1 為 Shifter。choose_v_o 是我新增的控制訊號，將 SLLV 與 SRLV 指令設為 1，其餘為 0。

ALU：參考附錄完成。

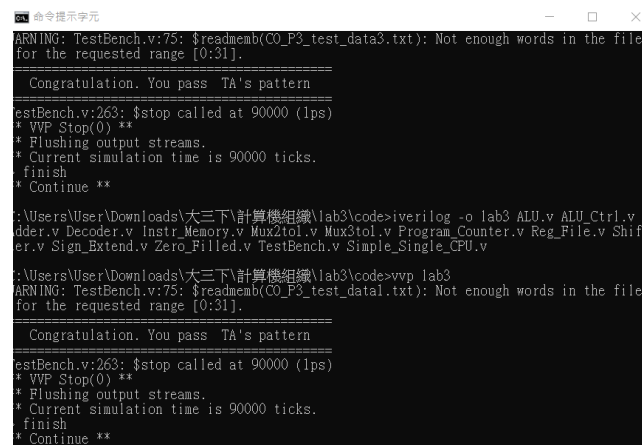
Zero_Filled：填充 0。

Sign_Extend：用最高位來將 16 位元擴充到 32 位元。

Shifter：以上次 lab 的內容實作。

MUX2to1、MUX3to1：單純利用控制訊號去決定要送出哪個輸入的值，重點在 Simple_Single_CPU 怎麼設置去利用它。

Simple_Single_CPU：依據上面的 Diagram 去填入該使用的值。



```
命令提示字元
WARNING: TestBench.v:75: $readmemb(CO_P3_test_data3.txt): Not enough words in the file
for the requested range [0:31].

=====
Congratulation. You pass TA's pattern
=====
estBench.v:263: $stop called at 90000 (lps)
* VVP Stop(0) **
* Flushing output streams.
* Current simulation time is 90000 ticks.
  finish
* Continue **

C:\Users\User\Downloads\大三下\計算機組織\lab3\code>iverilog -o lab3 ALU.v ALU_Ctrl.v
Adder.v Decoder.v Instr_Memory.v Mux2to1.v Mux3to1.v Program_Counter.v Reg_File.v Shif
er.v Sign_Extend.v Zero_Filled.v TestBench.v Simple_Single_CPU.v

C:\Users\User\Downloads\大三下\計算機組織\lab3\code>vvp lab3
WARNING: TestBench.v:75: $readmemb(CO_P3_test_data1.txt): Not enough words in the file
for the requested range [0:31].

=====
Congratulation. You pass TA's pattern
=====
estBench.v:263: $stop called at 90000 (lps)
* VVP Stop(0) **
* Flushing output streams.
* Current simulation time is 90000 ticks.
  finish
* Continue **
```

Problems you met and solutions:

中間有點像是鬼打牆，錯了但找不出來，不知道怎麼下手就改寫了 code 好幾遍，後來發現問題出在中間有一個小地方的一個字母打錯字，以及 ALU 有點怪怪的。一開始在想 ALU_Ctrl 沒有 leftright 的 output 不知道怎麼辦，後來想到可以由 ALU_operation 的其中一位當作區別送入 Shifter。加分題原本是打算把 choose_v 訊號與 rs、shamt 都送入 Shifter 讓它一起判斷，後來覺得會讓 Shifter 複雜化，於是多用了一個 MUX2to1 先確定好誰要當 shamt。

Summary:

這次的作業是要完成一個簡單的 Simple Cycle CPU，將架構中各個小單元完成各自的功能並連接起來。

一開始看到覺得很難都沒頭緒，但想到開頭後就開始知道它應該要是甚麼模樣，對 CPU 的架構更加熟悉了，做出來很感動，是個有用的練習。但因對 Verilog 不夠熟練，花了很久的時間在上面，也花了很久在構思與 debug，遇到了很多很奇怪的問題，一度崩潰，幸好最後有做出來。