Computer Organization Lab3

0816146 韋詠祥

Architecture diagrams

以 Lab 2 的程式碼為基礎，增加了 `sw/lw`、`jal/jr` 等指令的支援

因為有些地方有三種可能的輸入，因此從之前的 2x1 MUX 選擇器擴充為 4x1 MUX 選擇器

這次作業規格給的架構圖看起來要實作 `jal/jr` 會有困難，因此多加了兩條線、兩個控制訊號進來，如下圖紅色標記所示

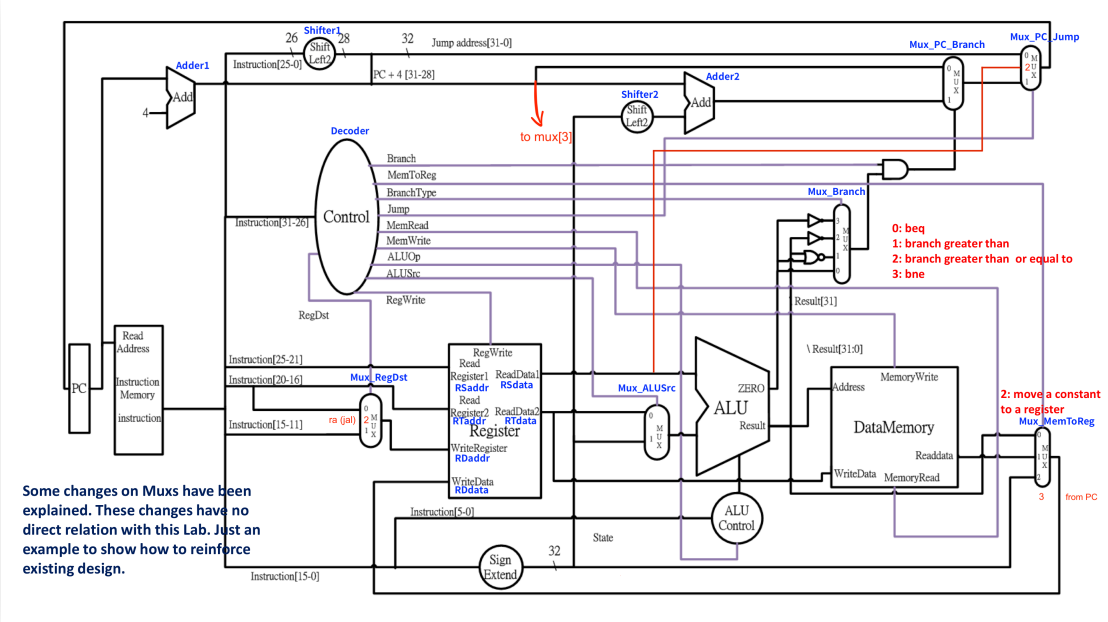

Fig 1. 架構圖，修改自作業規格

Hardware Module Analysis

之前老師說過實際寫 CPU 時不會用 K-map 等方法化簡，大多時候都是用直接使用真值表，不過上次 Lab 2 的 opcode 比較簡單，當時使用幾個邏輯運算就產生好輸出了。但這次的 `Decoder` 輸入變多、輸出也變兩倍，因此改用 switch case 的方式產生

這次的 CPU 本體從上次的 110 行增加到 160 行，主要增加了 `Data Memory` 模組及 `BranchType`、`Jump`、`MemToReg`、`MemRead`、`MemWrite` 等控制訊號

Result

正確實作所有模組以後，成功讓兩筆測試資料的輸出與 spec 上的答案相符

```
[2:03:05] sean :: Sean-MBPR  ➜  Repos/NCTU-109B-Comp-Org/Lab3-Single-Cycle-CPU (master*) » make
cat testcase1.txt | tr -d ' ' | awk '/./' > testcase.txt
perl -pe 's/(reg \[32\-1:0\] Instr_Mem \[0:)\d+\-1\];/${1}'"$(cat testcase.txt | wc -l | tr -d ' ')"'-1];/' -i Instr_Memory.v
iverilog -o cpu ALU.v ALU_Ctrl.v Adder.v Data_Memory.v Decoder.v \
                Instr_Memory.v MUX_2to1.v MUX_4to1.v ProgramCounter.v Reg_File.v \
                Shift_Left_Two_32.v Sign_Extend.v Simple_Single_CPU.v testbench.v
./cpu
VCD info: dumpfile debug.vcd opened for output.
PC =         60
# Data Memory
         1,          2,          0,          0,          0,          0,          0,          0
         0,          0,          0,          0,          0,          0,          0,          0
         0,          0,          0,          0,          0,          0,          0,          0
         0,          0,          0,          0,          0,          0,          0,          0
# Registers
R0  =          0, R1  =          1, R2  =          2, R3  =          3, R4  =          4, R5  =          5, R6  =          1, R7  =          2
R8  =          4, R9  =          2, R10 =          0, R11 =          0, R12 =          0, R13 =          0, R14 =          0, R15 =          0
R16 =          0, R17 =          0, R18 =          0, R19 =          0, R20 =          0, R21 =          0, R22 =          0, R23 =          0
R24 =          0, R25 =          0, R26 =          0, R27 =          0, R28 =          0, R29 =          0, R30 =        128, R31 =          0
cat testcase2.txt | tr -d ' ' | awk '/./' > testcase.txt
perl -pe 's/(reg \[32\-1:0\] Instr_Mem \[0:)\d+\-1\];/${1}'"$(cat testcase.txt | wc -l | tr -d ' ')"'-1];/' -i Instr_Memory.v
iverilog -o cpu ALU.v ALU_Ctrl.v Adder.v Data_Memory.v Decoder.v \
                Instr_Memory.v MUX_2to1.v MUX_4to1.v ProgramCounter.v Reg_File.v \
                Shift_Left_Two_32.v Sign_Extend.v Simple_Single_CPU.v testbench.v
./cpu
VCD info: dumpfile debug.vcd opened for output.
PC =        132
# Data Memory
         0,          0,          0,          0,          0,          0,          0,          0
         0,          0,          0,          0,          0,          0,          0,          0
         0,          0,          0,          0,         68,          2,          1,         68
         2,          1,         68,          4,          3,         16,          0,          0
# Registers
R0  =          0, R1  =          0, R2  =          5, R3  =          0, R4  =          0, R5  =          0, R6  =          0, R7  =          0
R8  =          0, R9  =          1, R10 =          0, R11 =          0, R12 =          0, R13 =          0, R14 =          0, R15 =          0
R16 =          0, R17 =          0, R18 =          0, R19 =          0, R20 =          0, R21 =          0, R22 =          0, R23 =          0
R24 =          0, R25 =          0, R26 =          0, R27 =          0, R28 =          0, R29 =        128, R30 =          0, R31 =         16
```

Fig 2. 實驗結果

從波形圖來看，各時刻的輸出值也與人工計算的結果相同

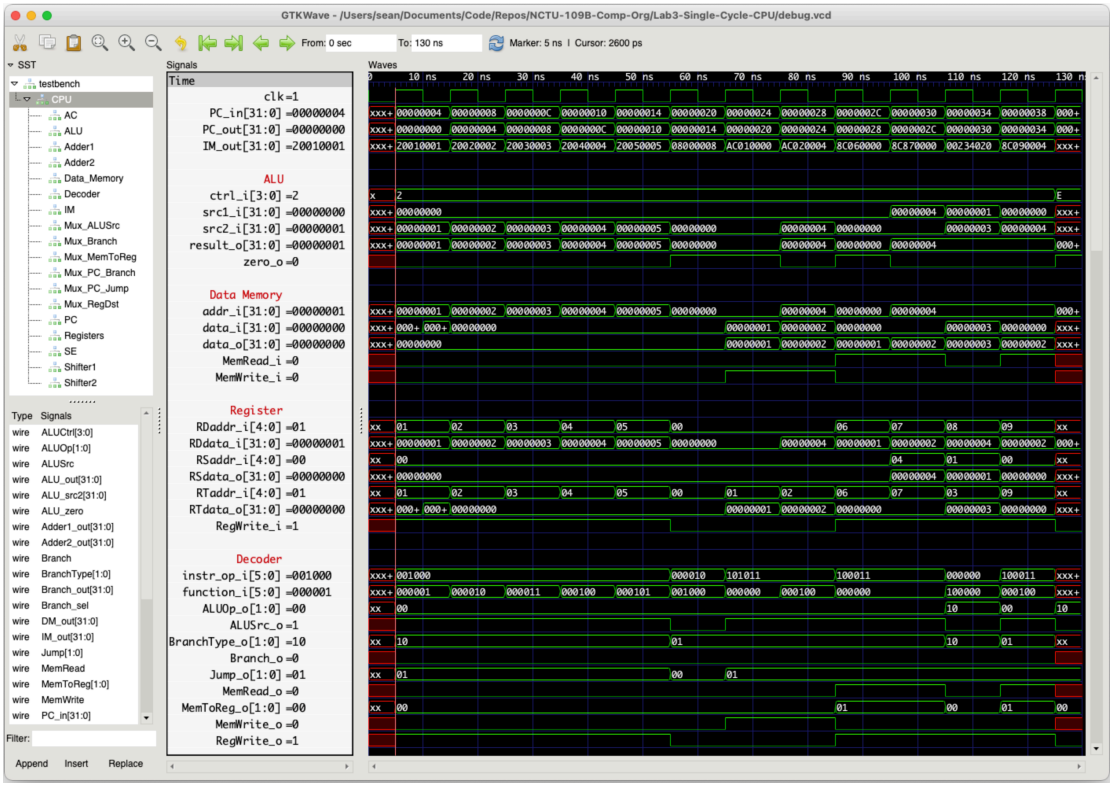

Fig 3. 測資 1 波形圖

Summary

跟上次 Lab 2 的基本功能比起來，這次 Lab 3 雖然多了四五個指令，但就必須用更複雜的判斷來達成，由於一開始使用邏輯運算來產生控制訊號，一開始的實作一直出現奇怪的問題，在除錯的過程中我更熟悉每個控制訊號要長怎樣、模組與模組之間的訊號會如何變化

因為只有對特定的測試資料測試，也還沒寫單元測試、整合測試相關腳本，所以在成功讓 Lab 3 兩筆測試資料得到正確答案後，準備寫這份報告時才發現 Lab 2 的 `slti` 從好的變成壞的了，很擔心以後也會發生類似狀況，或許下次 Lab 4 支援的指令更多時會來做這件事

Reference

MIPS Converter: https://www.eg.bucknell.edu/~csci320/mips_web/

MIPS CPU implemented in Verilog: https://github.com/jmahler/mips-cpu/