**Computer Organization**

**110550126 曾家祐**

**Architecture diagrams:**

**一張含有 圖表 的圖片

自動產生的描述**

**Hardware module analysis:**

* Adder.v：實作 PC = PC+4
* ALU.v：from lab 2 依照指令進行運算
* ALU\_Ctrl.v：根據function code 和 ALUOP code 輸出 ALU\_operation (leftright)
* Decoder.v：根據instructio[31:26]輸出RegDst,RegWrite,ALUOP,ALUSrc
* Instr\_Memory.v：讀取指令
* Mux2to1.v：依照輸入的訊號選擇輸出
* Mux3to1.v：依照輸入的訊號選擇輸出
* Program\_Counter.v：控制進入下一指令的時間
* Reg\_File.v：對register 的輸入輸出
* Shifter.v：運算srl sll
* Sign\_Extend.v：將16bits變成32bit
* Zero\_Filled.v：將16bits變成32bit
* Simple\_Single\_CPU.v：將前面的所有串接起來

**Finished part:**

* Adder.v：單純相加兩個輸入
* ALU\_Ctrl.v：先檢查是不是 addi指令(ALUOP\_I = 000)
  + - 如果不是再依照function code 來輸出ALU\_operation (leftright)和FURslt
* Decoder.v：根據instructio[31:26]輸出RegDst,RegWrite,ALUOP,ALUSrc
  + - (此lab只區分 R format 和ADDI指令)
* Instr\_Memory.v：from TA
* Mux2to1.v：依照輸入的訊號選擇輸出
* Mux3to1.v：依照輸入的訊號選擇輸出
* Program\_Counter.v：from TA
* Reg\_File.v：from TA
* Shifter.v：from lab2
* Sign\_Extend.v：將16bits依照正負(正:0負1)填完左邊32bits
* Zero\_Filled.v：將16bits左邊全部填0成32bits
* Simple\_Single\_CPU.v：依照 diagram 將所有module 連起來

**Problems you met and solutions:**

**一開始不知道要從哪邊下手，後來注意到了architecture diagram 後，知道了這次整個的lab 目標，但還是有些不清楚元件的實作目標再搭配講義內容去了解每個元件的功能、輸入、輸出，之後就知道為什麼要有這些輸入，和輸出原因了。**

**最後再依照diagram 將全部串接起來，也在這個過程將前面的bug順便修復。**

**Summary:**

**這次的lab是要完成一個single cycle cpu，除了lab2 的ALU外，還做了許多額外的小元件。在成功把每個元件做出來之後再依照architecture diagram將完成的元件連起來，讓我更了解整個cpu在不同指令時會經過的流程。**