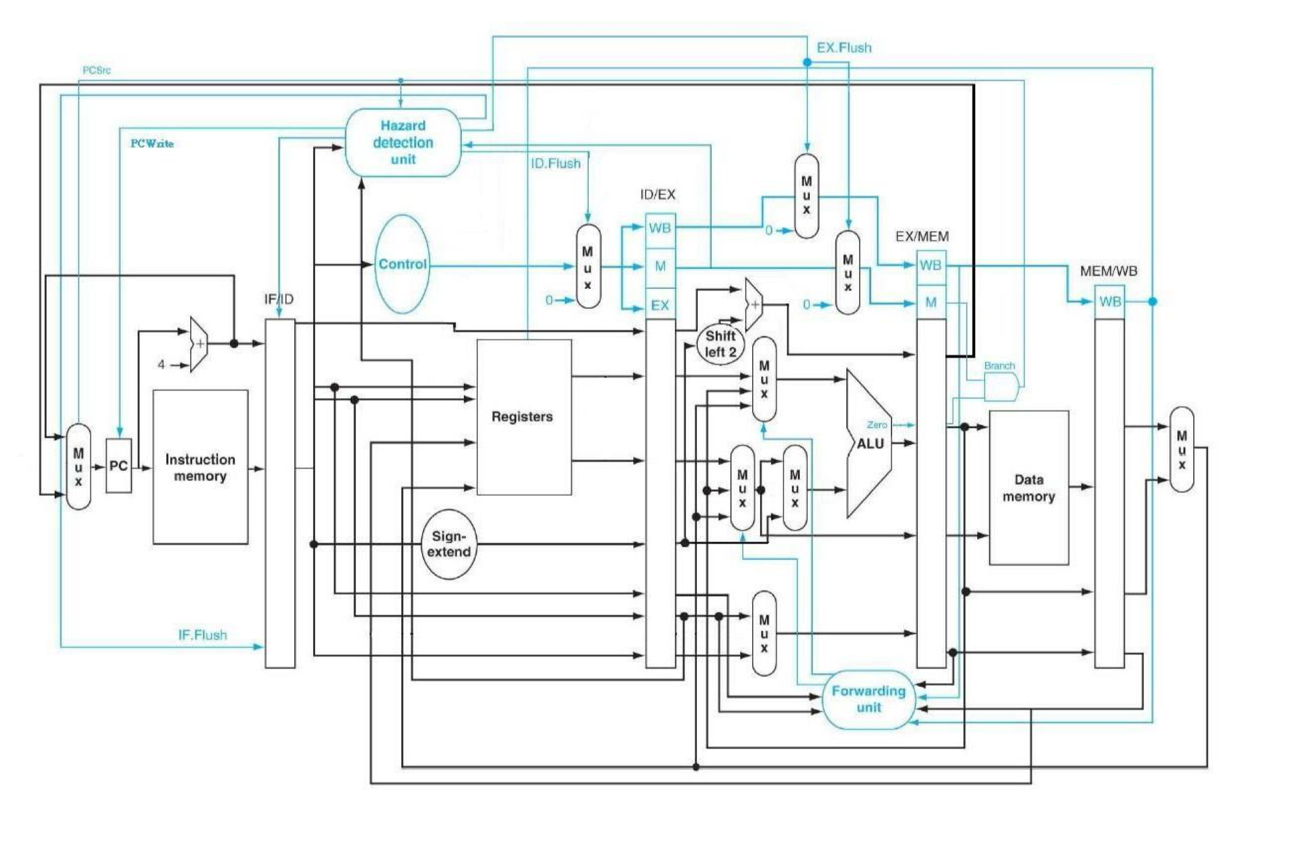
**Computer Organization Lab5**

**0516022 吳禹璇**

Source code and the note:

我沒有覺得什麼地方是特別重要的，這次純粹是從之前寫過的每一次lab抓一些用過的東西出來用，再加一些新的功能(例如forwarding還有hazard)，最後照著圖接的而已，所以如果要說重要，應該就是我之前寫的code都很健康也都能使用吧。

My architecture:



Hardware module analysis:

雖然是沿用之前的code，不過這次是有改比較多地方。

例如一開始的ProgramCounter，加入了一個PCWrite的input，主要是判斷有沒有hazard發生，一旦發生了就要把它設成0，下一個指令在讀的時候才不會繼續更新pc。

圖上有好幾個地方的mux的input變成3個，因為懶得再寫新的mux\_3to1，所以我把之前寫的mux\_4to1再拿來用。

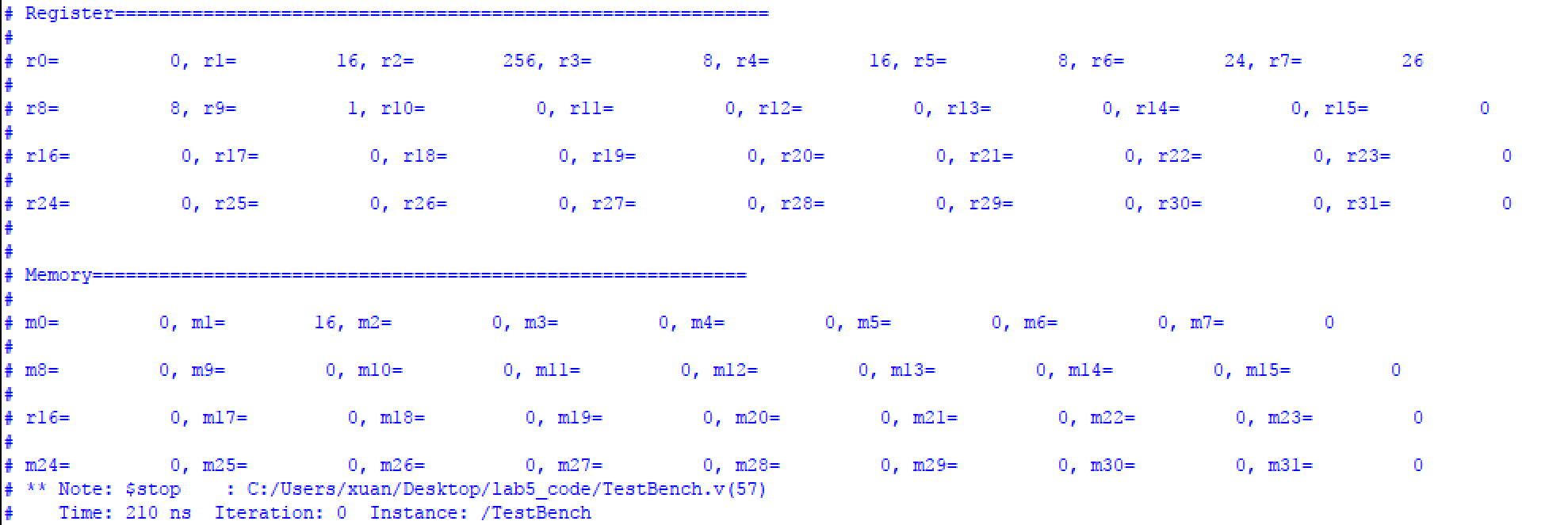
上次lab我印象中是因為沒用到branchtype跟jump，所以把他們的判斷從decoder拿掉，結果現在又要用上，所以重新把它們加回去了，branchtype主要是負責分辨是那四種之中的哪一種。

新寫的地方就是forwarding跟hazard這兩個檔案，雖然完全要自己寫，不過因為第四章的投影片裡面都已經有寫出來了(ch4 p.78 & p.83)，所以在寫的時候只要先搞清楚input跟output，再把投影片上的判斷式加上去就好了。

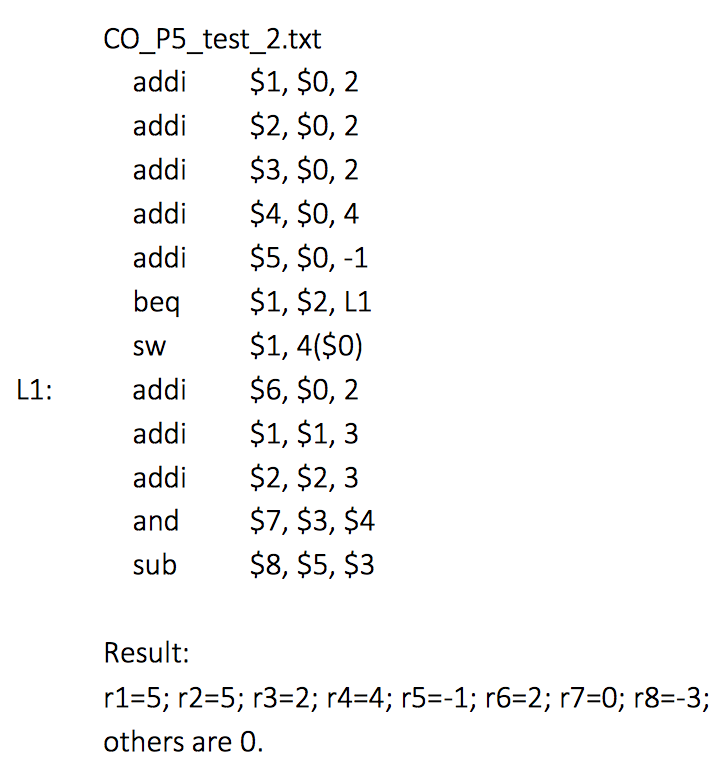
另外我還有寫一個pipe\_reg\_if\_id的檔案，主要是為了判斷if.flush以及if\_id\_stall的module。

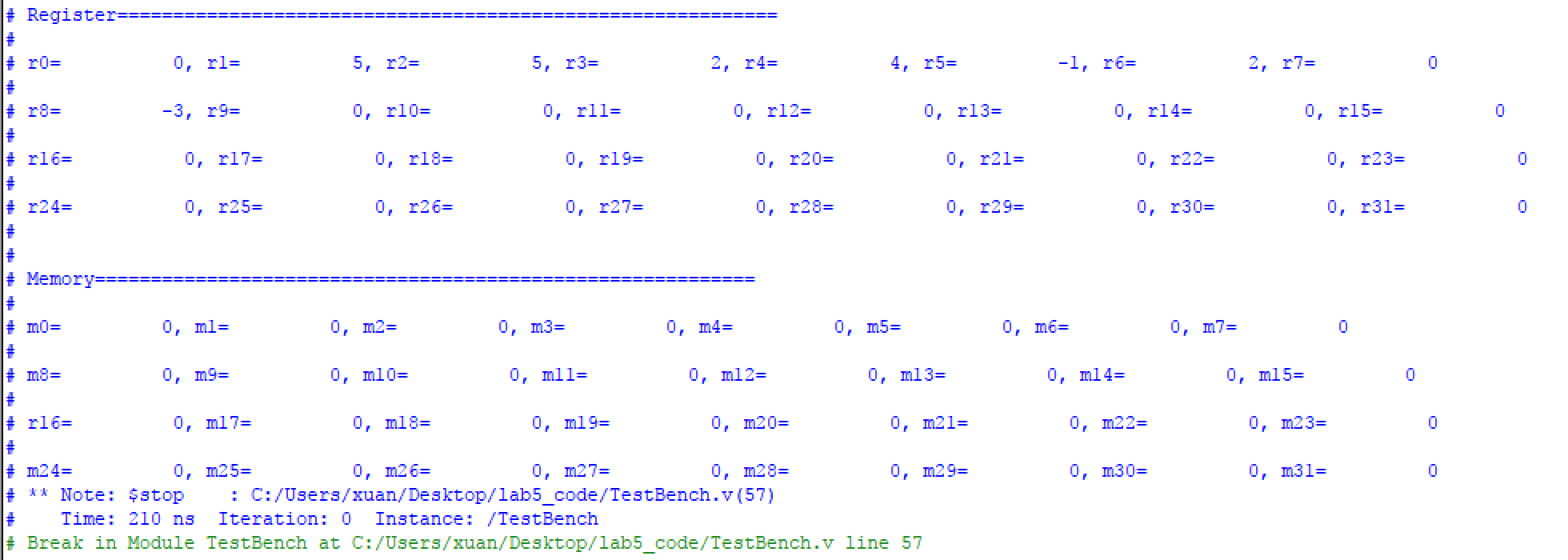
Finished part:

CO\_P5\_test\_1 :



CO\_P5\_test\_2 (I made it by myself) :





Problems you met and solutions:

其實越到後面的lab，雖然應該算是越來越複雜，但是因為越來越熟悉module的設計模式，還有接線的模式，常用的語法也就那幾種，所以不太會遇到什麼大問題，唯一就是要注意逗號啊end啊endcase啊這些，不要自己忘記或不小心刪掉，不過同時debug速度也變快了，因為error或warning也越快看懂了，所以整體算是完成的挺快的。

Summary:  
 上次寫完lab4就覺得這次應該是要做forwarding的部分，果不其然，另外還有hazard，因為這邊的概念算是期末考的範圍，所以在寫的時候算是又幫我複習了一次概念，覺得算是一舉兩得吧，也很開心這學期的作業都結束了。