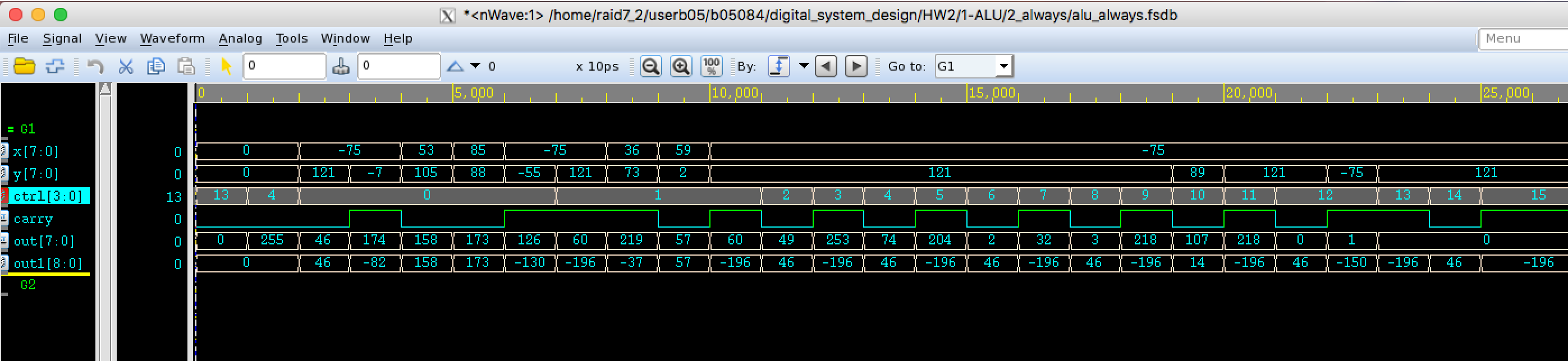
1. ALU

testbench:

除了0000和0001有比較複雜的有號數加減法，其他的指令都是直觀的移位、邏輯運算，因此不需要太多測資驗證，只需一到兩項測資，而在測試加減法時，正負溢位和不會溢位的運算都須考慮，才能驗證carry的正確性。



what I found:

大致上沒有太大的問題，只有在add和sub的carry遇到了一些問題，要用邏輯運算算出carry其實沒有那麼直觀，所以我多設了out1[8:0]，直接擴充一個bit進行加減，再把out1[8]給carry。

2. register

testbench:

因為題目要求隨機產生測資，所以使用$random，在第一輪時將register全部寫入新值，並讀取測試結果，第二輪時則先讀取，並將讀取到的值invert後一一寫入，最後再讀取測試結果。

What I found:

我在測試時，發現結果和答案差了一個cycle，因為register和tb都是自己寫的，所以會不知道哪個才是對的，要看過nWave後，才確認是tb不對。

另外碰到的另一個問題是read的讀取時間和條件，雖然我寫的tb看不出影響，但卻會造成下一題的錯誤。題目中的synchronously並不是指和CLK同步，而是輸入指令時，可同步輸出結果。

3.simple calculator

what I found:

從tb觀察，從輸入到輸出的時間並沒有給足夠的cycle，而且寫入只能在posedge進行，所以每個module如何連接非常重要。Data\_in經過mux再經過ALU都是combinational logic，在posedge時寫入，同時可以讀出存入的值。我遇到的第一個問題是我誤以為read必須要在posedge或negedge讀取，後來才知道不用，另一個問題則是在接線時要用wire還是reg，mux到ALU用reg來選擇接收bus\_x或data\_in，而ALU連到register的線則使用wire。