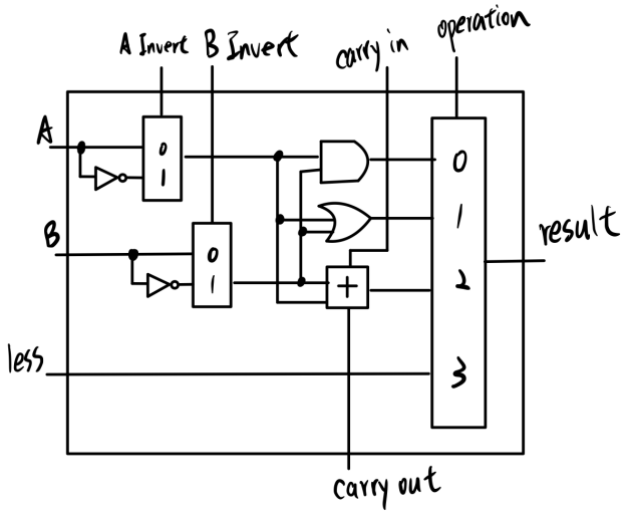
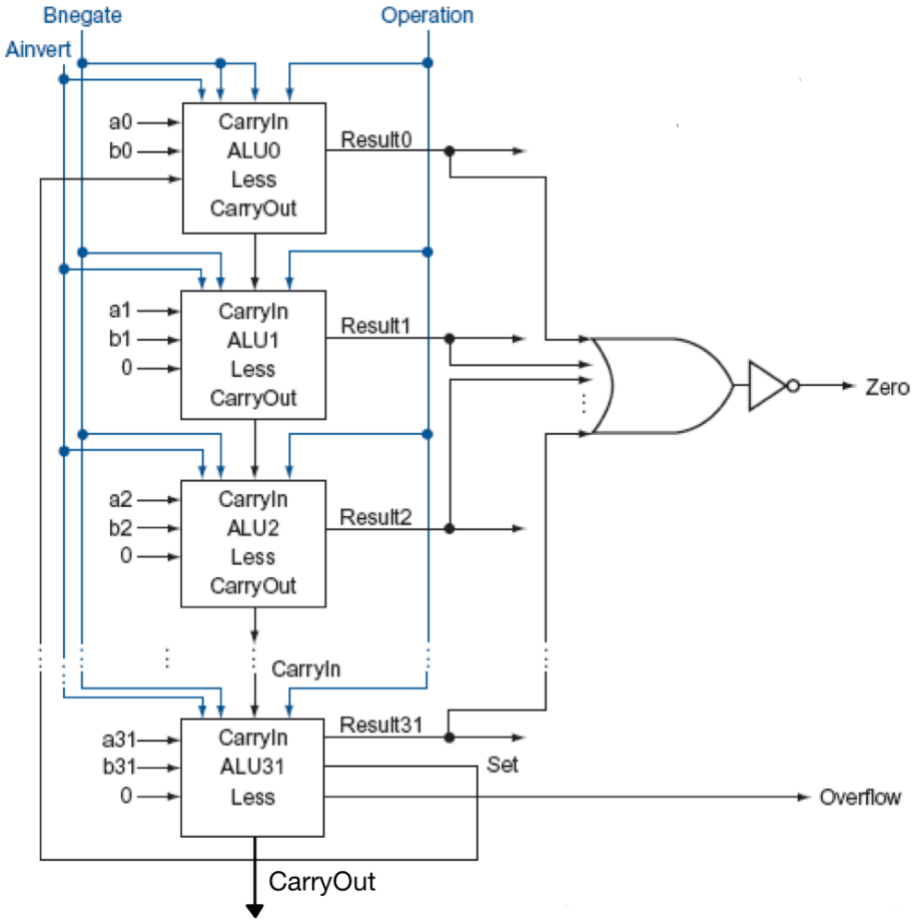
Computer Organization

Architecture diagrams:

For Single ALU :



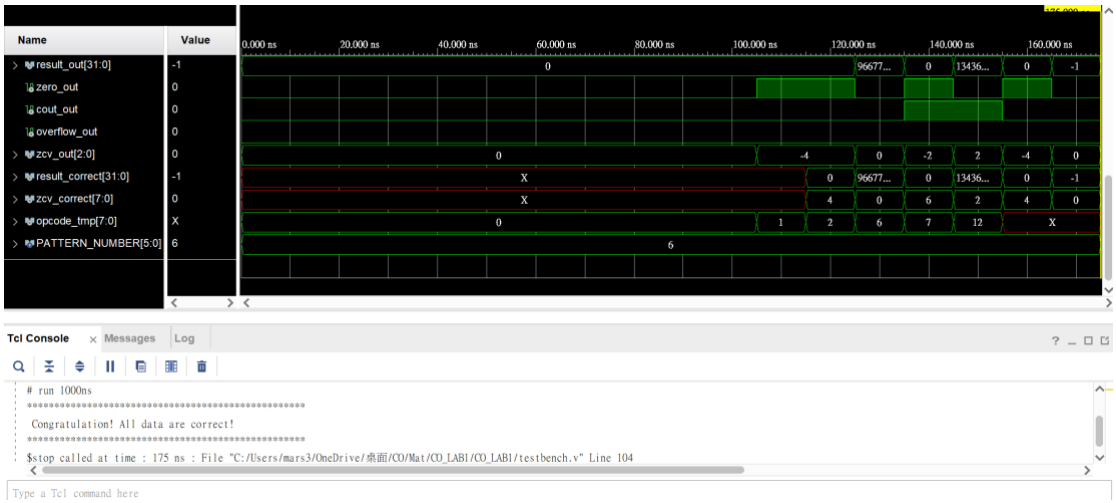
For 32-bit ALU :



Hardware module analysis:

我單一個 ALU 的設計法與課本的設計方法相同，而 32-bit 的 ALU 則是參考課本設計方法並使用一些我自己覺得比較好的實作方式，像是我並沒有額外設計專門給第一或最後一個 bit 的 ALU，我將最後一個 bit 的 overflow 情況在 alu.v 另外處理他，以及最終的 output result, carry out 都是在 ALU module 外另外處理，我覺得這樣比較不會讓 single ALU 的 module 設計太複雜，程式也更好懂。

Experiment result:



上圖是我這次跑出來的波形圖，與我預期的相符合，在檔案中提供的測資也都正確，沒有出現大量未知的數值之類的。而 console 顯示出來的也符合說明文件中提及的「Congratulation All data are correct!」。

Problems you met and solutions:

我在這次 Lab 遇到的問題是我的 result 輸出結果都跟答案一模一樣，但 ZCV 卻不知道會什麼會有一些測資跟答案不同，有時候是 carry out 不同，有時候是 overflow 不同，後來的解決方法是我發現我並沒有在最一開始將 zero, carry out, overflow 這些 output 得 register 初始化，導致輸出的值並不穩定，後來在 always 內最前方將 zero, carry out, overflow 初始化成 0 問題就解決了。

Summary:

我覺得從一個單一的 ALU 做起有讓我更清楚整個 ALU 的架構，像是要怎麼用 2 bit 的 operation control 去區分全部的 operation，以及各個 operation 相對應的 result, carry out 的輸出等等，對後續做 32-bit 的 ALU 都很有幫助。有了前面 single ALU 的基礎，在設計 32 bit ALU 的時候就可以比較把精力放在處理每個 ALU 之間的 carry 以及最後一個 bit 的 overflow 的狀況。