**Computer Organization, Spring 2019**

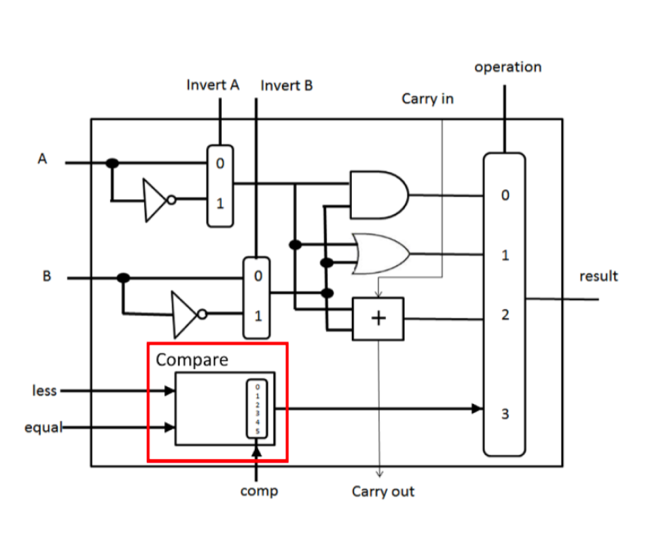
**Lab 1: 32-bit ALU**

**Name: 王耀德**

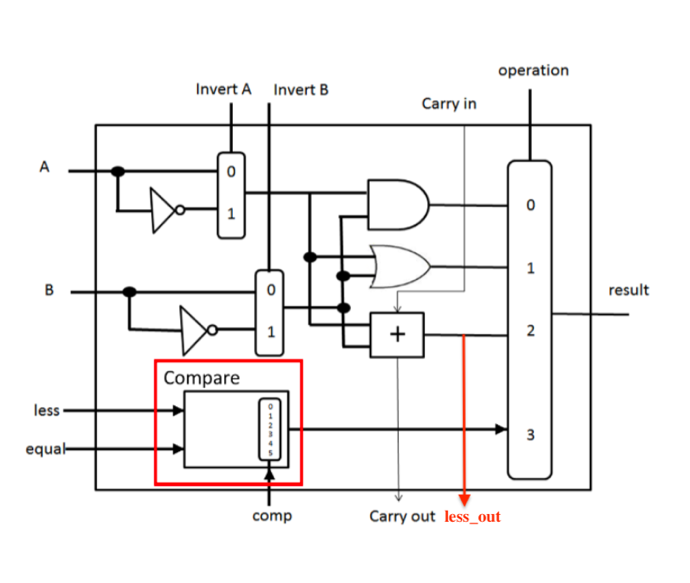
**Student ID: 0716055**

(1) My architecture diagram

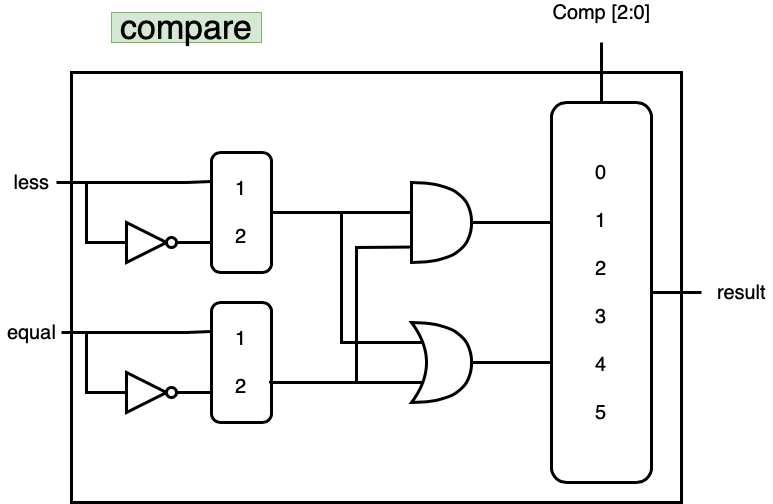
1.alu\_top:



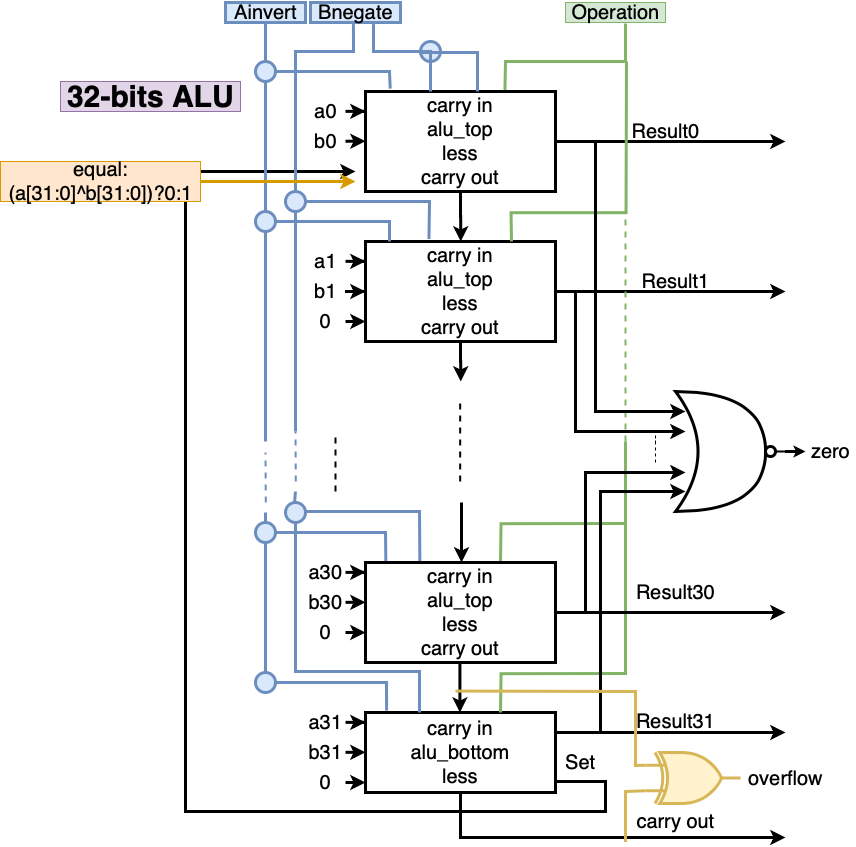
2. alu\_bottom:



3. compare:



4. 32-bit alu:



(2) Detailed description of the implementation

這次作業主要包含四個module，分別是1 bit 的alu\_top, alu\_bottom、compare，並組成最後的32-bit alu。先在alu\_top中做好and, or, adder，搭配invert就可以完成題目的前六種功能。

而slt則是將減法差值的最後一位，也就是差值的正負傳入作為less，然後與equal進入compare做判斷，所以alu\_bottom多出一個output就是減法的輸出，傳入第一個alu\_top作為less。比較要注意的是slt的result只有在第一位，所以其他位數的slt result都輸出0即可。

在32-bit alu中，前三十一個使用alu\_top，最後使用alu\_bottom，最後會得到32-bit 的result跟carry out，並把結果拿來判斷zcv。Cout是最後位carry out，Overflow是把第30跟31個alu的carry out做xor判斷，如果不同則overflow。Zero是將所有result 做nor，代表只有result全零zero才會為1。

(3) Command for compiling your source codes   
iverilog -o bonus.vvp testbench.v alu.v alu\_top.v alu\_bottom.v compare.v

(4) Problems encountered and solutions

已經很久沒寫verilog了，花很多時間想module要怎麼用，還有register跟wire的用法，這部分是看編譯出來的錯誤訊息慢慢修正。除此之外，一開始測試時減法一直有部分錯誤，中間都差一位數，後來是檢查alu input的欄位有一格13填成12，細心檢查、修改就解決了。再來是第六筆測資一直失敗，檢查SLT的部分也沒有寫錯，後來發現是equal的地方沒有寫好，最後修正equal的判斷，就完成lab1了。

(5) Lesson learnt (if any)

學到alu的組成，還有熟悉verilog，對之後的lab應該會更容易上手。