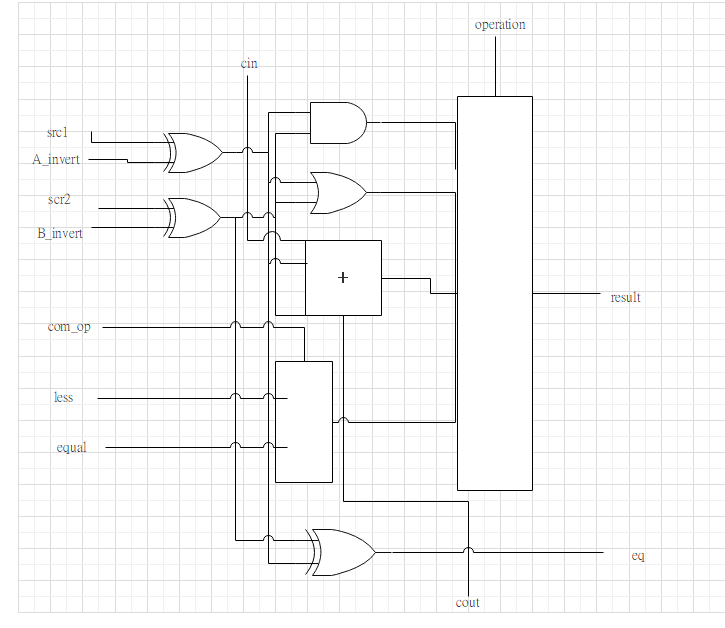
Lab1

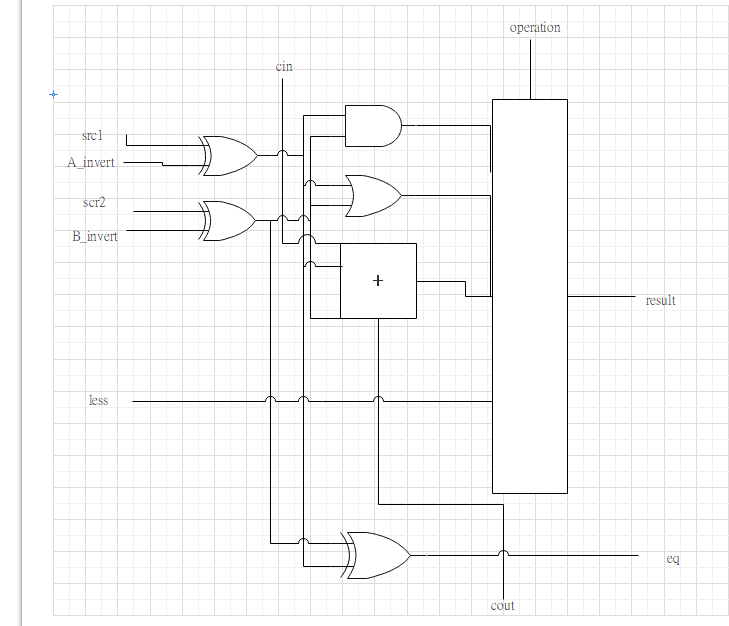
0710028 陳敬諺

1. Diagram

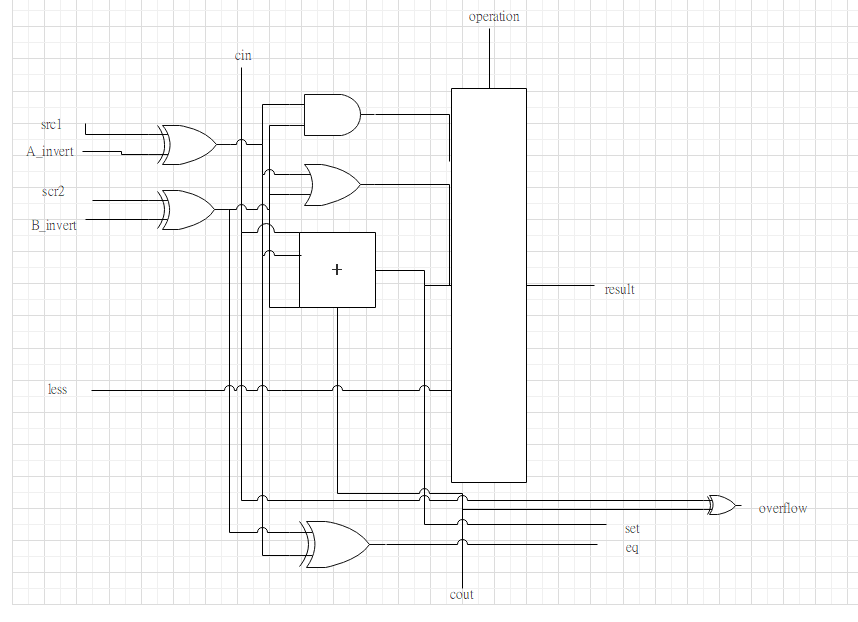
alu\_TOP0:



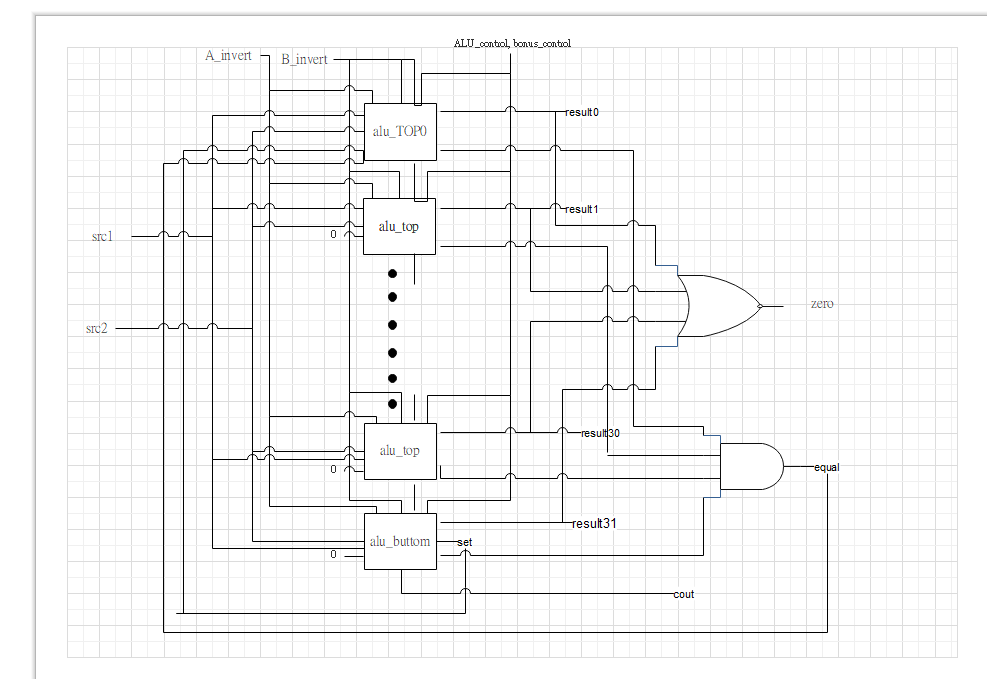
alu\_top:



alu\_buttom:



alu:



1. Detail

1-bit alu分成三個部分解釋:

alu\_top (for result[1]~[30]):

m1,m2: 讀取scr1,scr2是否有invert的結果

op1: m1,m2做AND運算

op2: m1,m2做OR運算

op3: m1,m2,cin做XOR運算，及result[n]做ADD或SUB的結果

op4: SLT,SGT,SLE,SGE,SEQ,SNE的結果，由於除result[0]以外，其餘皆為0，所以我選擇直接與less連接，簡化電路

eq: 比較sr1,scr2是否相同

cout: 計算m1,m2,cin若有兩個1以上則進位

alu\_buttom (for result[31]):

與alu\_top相比，這邊多了以下功能:

set: 表示scr1是否小於scr2，方法是利用scr1-scr2，當scr1-scr2 < 0，代表alu\_buttom的計算結果為1，可直接作為set值

overflow: 表示運算結果是否出現滿溢，以cin,cout做XOR來判斷

alu\_TOP0(for result[0]):

因為SLT,SGT,SLE,SGE,SEQ,SNE的output只有result[0]會有影響，所以需要另外做判斷，利用K-map可得到equal,less與com\_op(bonus\_control)的關係式。

alu:

實際完成32-bits alu，另外還有以下功能

zero: 對所有的result做NOR運算

equal: 對所以1-bit alu的eq做AND運算，並做完alu\_TOP0的equal輸入

1. command

iverilog –o bonus.vvp testbench.v alu.v alu\_TOP0.v alu\_top.v alu\_buttom.v

1. problems & solution

整個過程中比較棘手的問題主要是less的判斷，原本是單純想從高位往低位數比較，不過後來覺得過於麻煩才發現用減法判斷是最好實現的方式。

1. lesson learnt

之前有修過數位電路實驗，算是再次複習吧，不過還是花了不少時間。