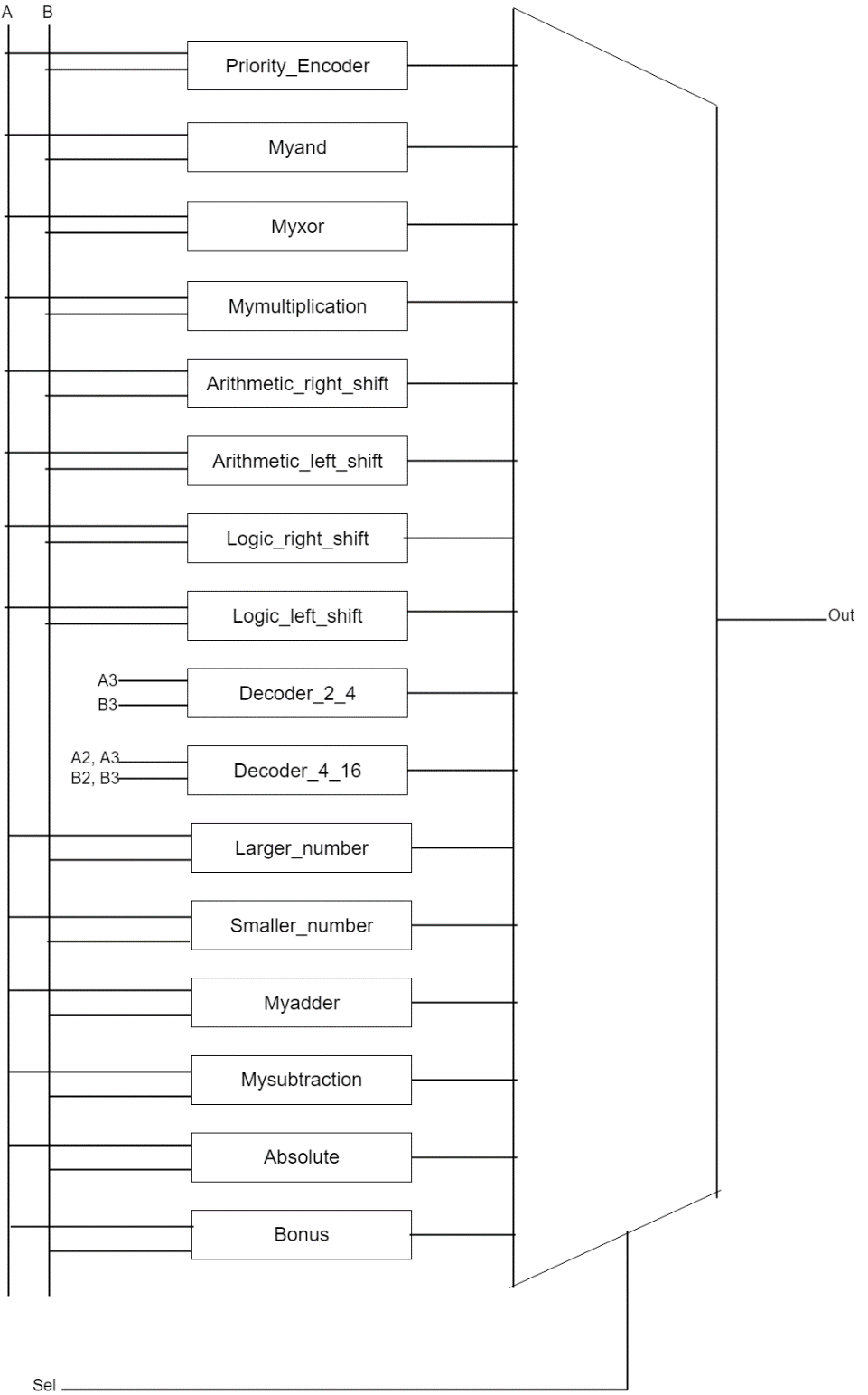
Logic Design

210510210 詹其侁

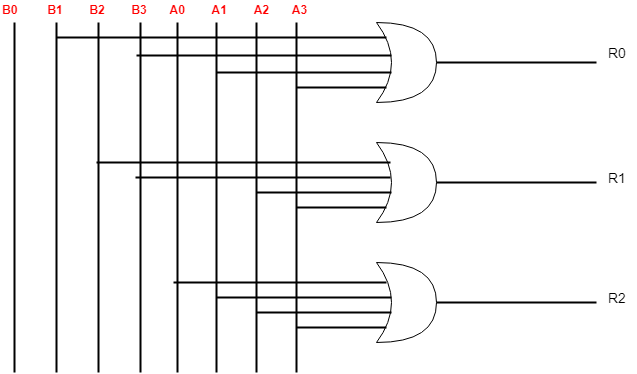
Lab2

**1.Block diagram**

****

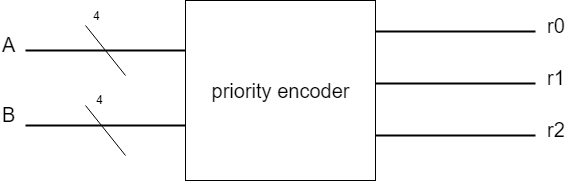
**這是我這次設計的主要架構，裡面有幾個module像是myand, myxor, mymultiplication, Arithmic\_right\_shift, Arithmic\_left\_shift, logic\_right\_shift, logic\_right\_shift都是用verilog內建的語法，所以下面會列出幾個在實作的時候有碰到問題的module，或者是比較複雜的module。**

**2.Simple\_Priority\_Encoder**

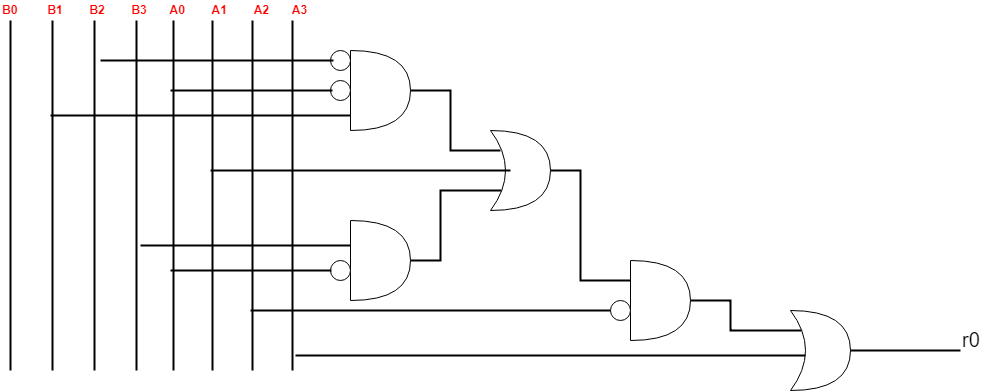
****

**這是我直接用TT做出來的PE，可是這個只針對傳進來的數只有一個1其他都是0的時候有用（根據我觀察的結果），所以我上網查了一下，發現這是簡單優先編碼器。之後我查到解決的辦法就是要把它前面的０也列進去考慮，後面的don’t care就不用管了。下面是改良後的結果。**

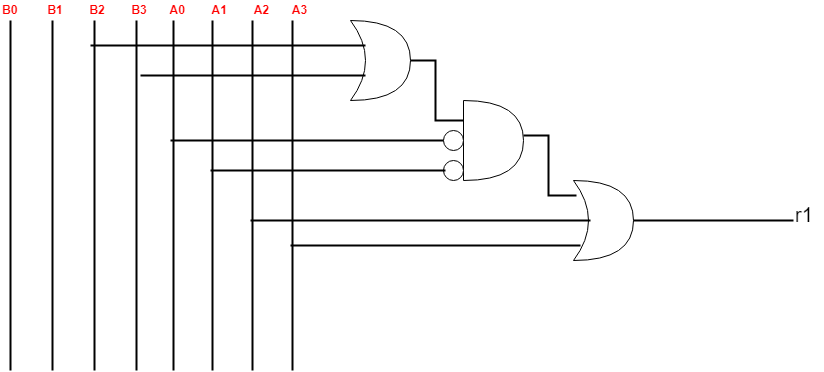
**3.Priority\_Encoder**

****

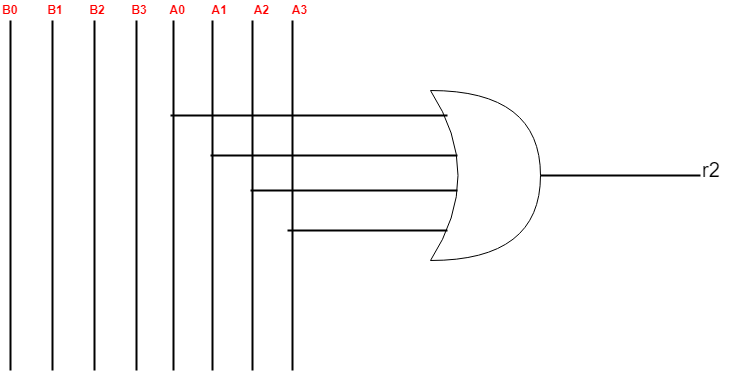
**R0**

****

**R1**

****

**R2**

****

**改良之後的R0, R1, R2是根據前面的R0, R1, R2來改的，拿R0當例子，原本的R0是B1+B3+A1+A3，現在把前面的0都補上去，所以就會變成**

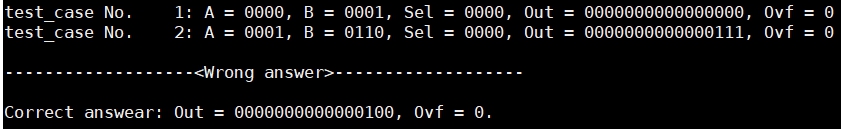
**+ B3 + A1 + A3接著化簡他，將 消掉，剩下 + B3 + A1 + A3**

**最後把提出來 B3 + A1) + A3**

**這樣就可以畫出上面改良版的R0了。**

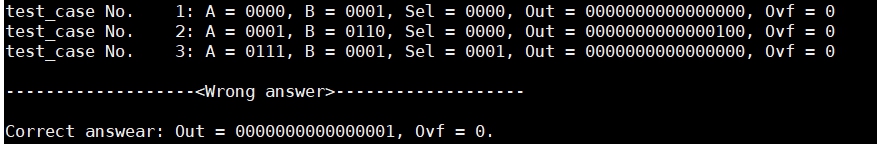
**經過改良後就可以克服掉同時有很多個1輸入進來的狀況，比如說00010110輸入進來就可以顯示出100了。下面兩張圖比較改良前和改良後的差異。**

**改良前**

****

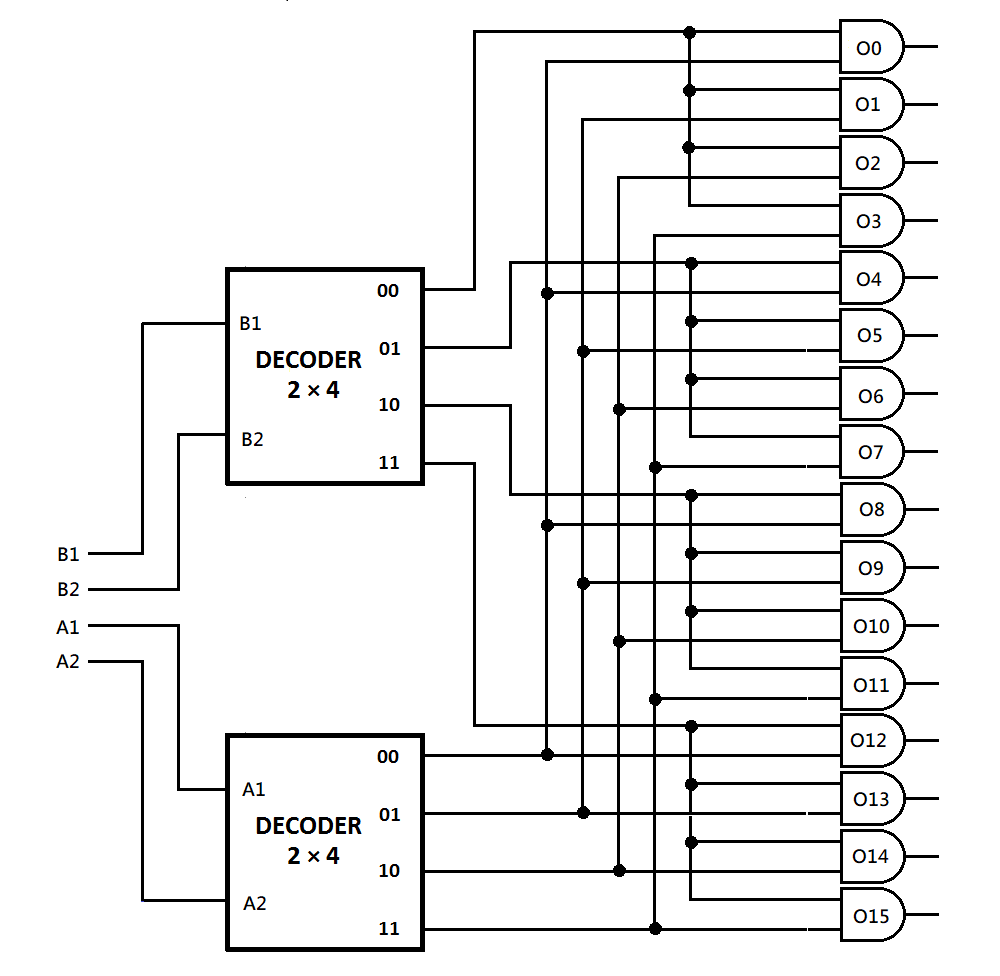
**輸入00010110結果跳出111**

**改良後**

****

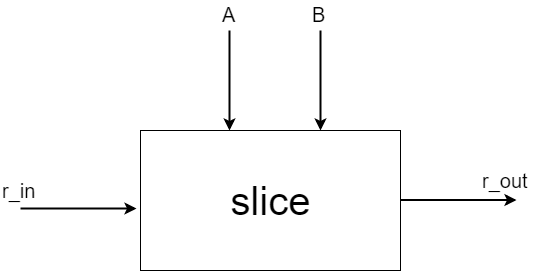
**成功解決這個問題**

**4.Decoder\_4\_16**

****

**Reuse兩個Decoder\_2\_4在輸入的地方，我嘗試用上次助教說的格式，可是還是不太對，所以這次我還是一樣開4個input讓他輸入，可能是我忘記上次正確的格式，所以打錯了。**

**5.Bit\_slice\_larger&smaller**

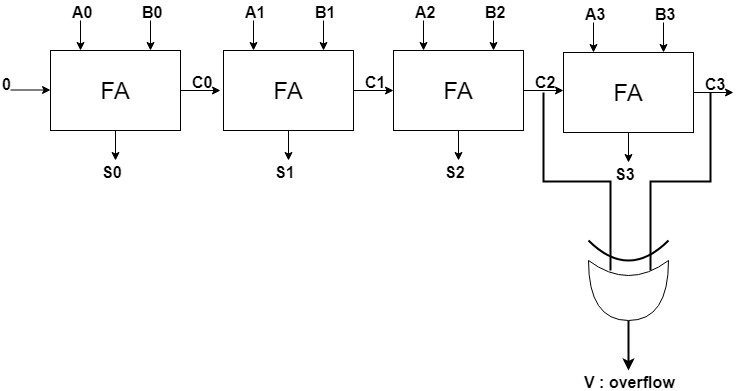
****

**找大找小，權重都是從小到大，剛開始我以為一個是從小到大，另一個就要從大到小，可是後來想過才發現，如果不是將大於、等於、小於，分開判斷的話，都要從小比到大，這樣才不會有他無法判斷的情況。**

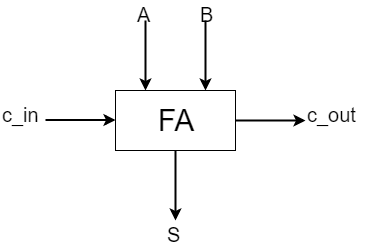
**Case\_larger如果A>=B r\_out=1, r0要傳1進去因為一開始的A跟B都是0。輸出的部分，如果最後的r是1就輸出A反之就輸出B，原因是r=1代表A>=B那不管是大於還是等於，輸出A都沒問題。**

**Case\_smaller如果A<=B r\_out=1, r0要傳1進去因為一開始的A跟B都是0。輸出的部分，如果最後的r是1就輸出A反之就輸出B，原因是r=1代表A<=B那不管是小於還是等於，輸出A都沒問題。**

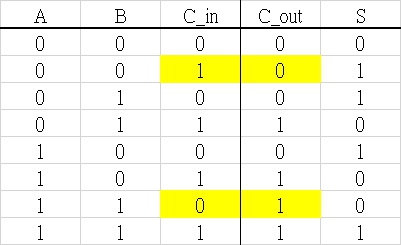
**6.Myadder&Mysubtractor**

****

**Myadder&Mysubtractor\_Bit Slice**

****

**Myadder&Mysubtractor \_truth table**

****

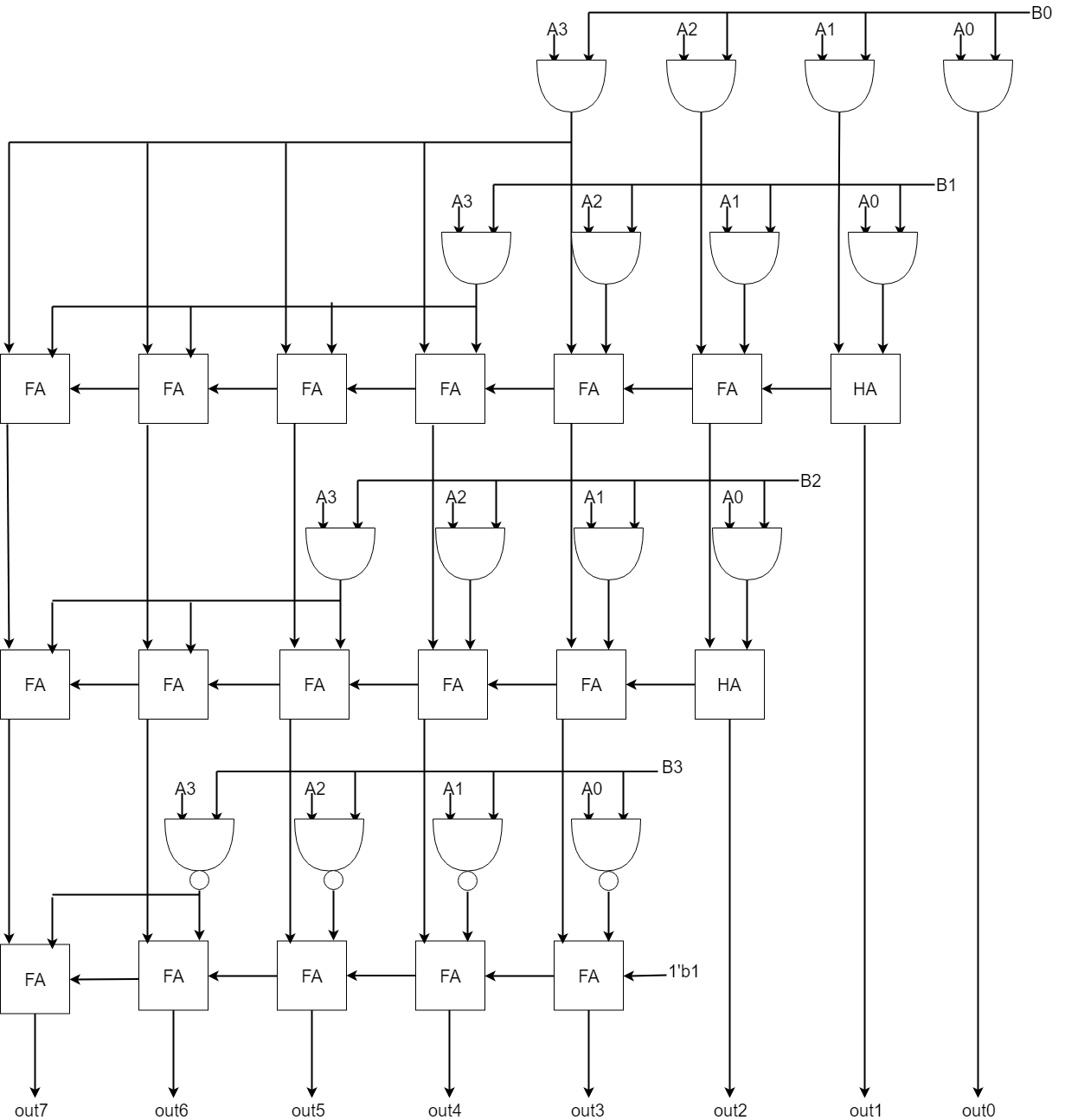
**Add&Sub一開始要傳0進去，因為前面沒有進位。我判斷overflow的地方是在最後一個bit slice的c\_in跟c\_out，兩個不相同，就是overflow。**

**Sub的做法是先將B變成(-B)，reuse myadder將A跟(-B)加起來。**

**7.Absolute**

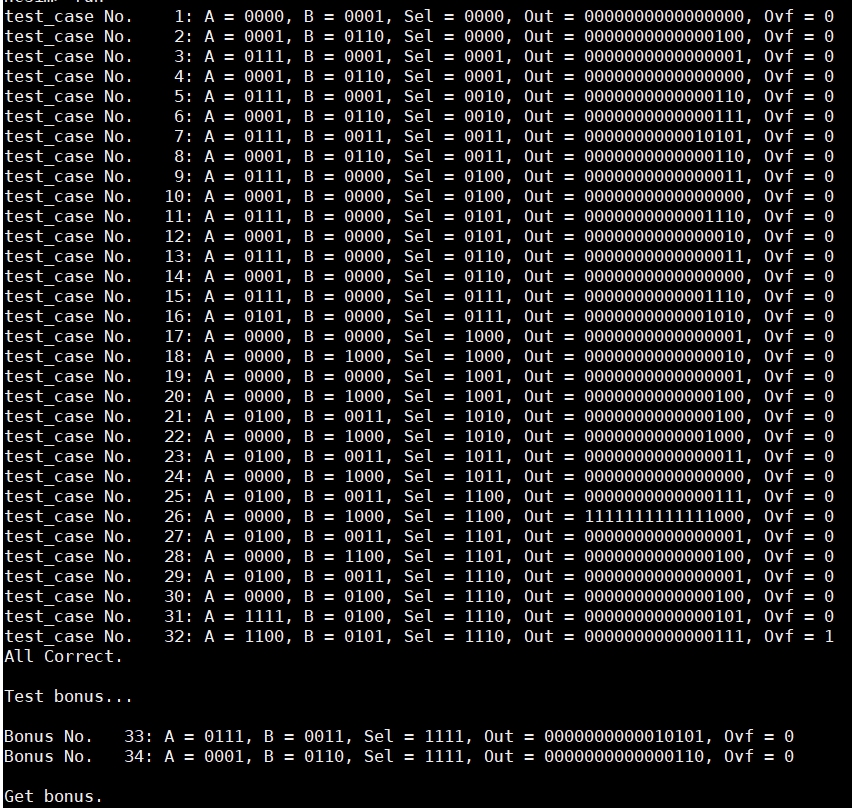
**我先reuse減法，算出來的結果存到一個tmp[3:0]裡面，再用一個wire S2[3:0]存負的tmp，如果tmp[3]是1說明tmp是負的，這時候就把output設成S2，反之就直接把tmp當成output。**

**8.Bonus**

****

**用and gate可以達到相乘的作用之後再用FA和HA把每一項相乘的結果按照權重加起來。最下面有4個nand gate，因為2’s complement的第一位是負的，所以經過nand gate在加1的轉換後，就等價於將A轉成負的之後再跟B相乘，在這裡，如果是兩個正數相乘，那他經過nand gate的轉換，因為B3=0所以結果一定是都是1，在加1就會全部變成0了，所以也不會有錯。**

**9.Ncverilog Simulation Result**

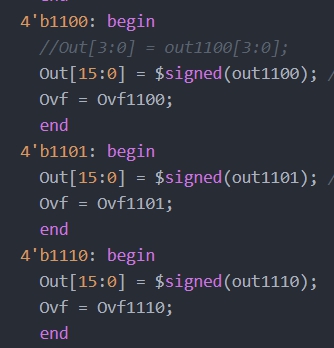
****

**我這次也是寫好一個module後就傳一次，慢慢把他們全部寫完，在做的過程中，mymultiplication讓我想很久，想說要怎麼不用gatelevel來做，因為bonus才會用到，大概花了1個小時，後來在滑助教給的講義”Verilog HDL教學講義”的資料流層次時，看到乘法原來可以直接用，瞬間豁然開朗，下面的shift也順勢寫出來，謝謝助教。**

**10.碰到的問題與解決的方法**

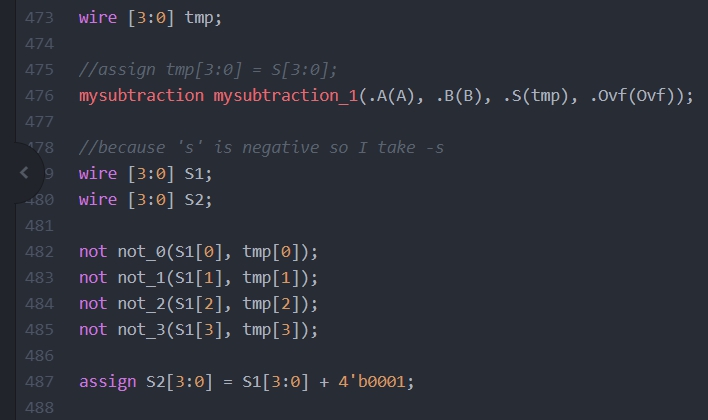
**問題1：在做sign extension的時候，我不想要用if else來判斷他前面的時1還是0。**

**解決方法：後來發現可以強制轉換，下圖程式碼裡面的$signed()就是將括號裡面的wire變成可以自動sign extension的型態，如果等號左邊的矩陣比較長，他就會自動sign extension。**

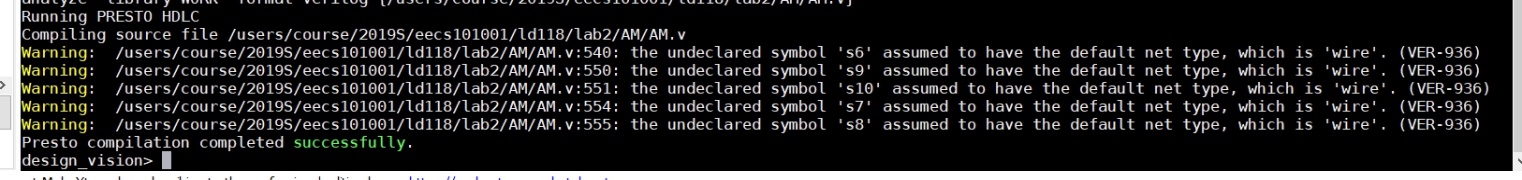
****

**問題２：在做絕對值的module時，我原本是直接傳S進去 mysubtraction，可是這樣會有形態問題，因為我下面有用always所以S得變成reg可是這裡又要傳wire進去。**

**解決方法：在473行我設了一個wire tmp他先記住mysubtraction的結果，下面是用S2記住負的tmp如此一來，便可以解決這個問題。**

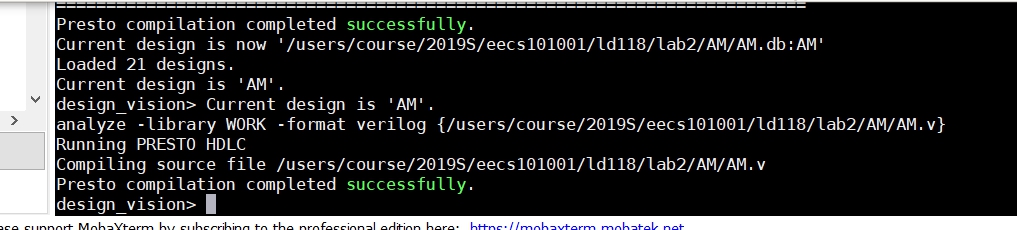
****

**問題3：在合成電路圖的時候遇到的一系列問題**

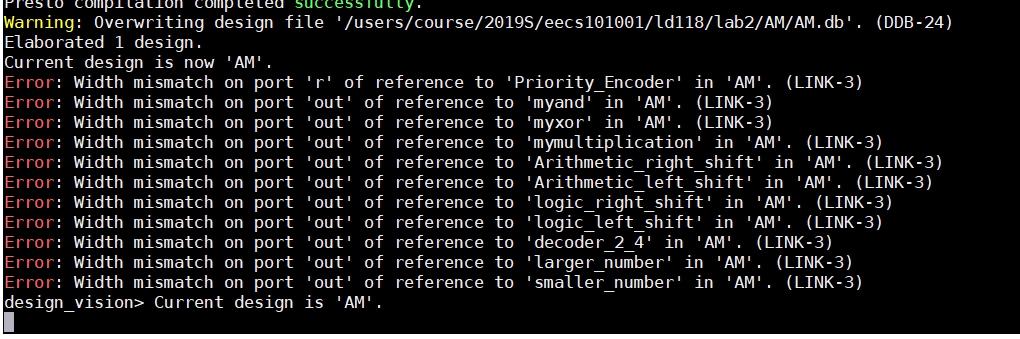
****

**a.警告我的一些wire沒有宣告就使用**

**解決方法：一個對一個的宣告完成。**

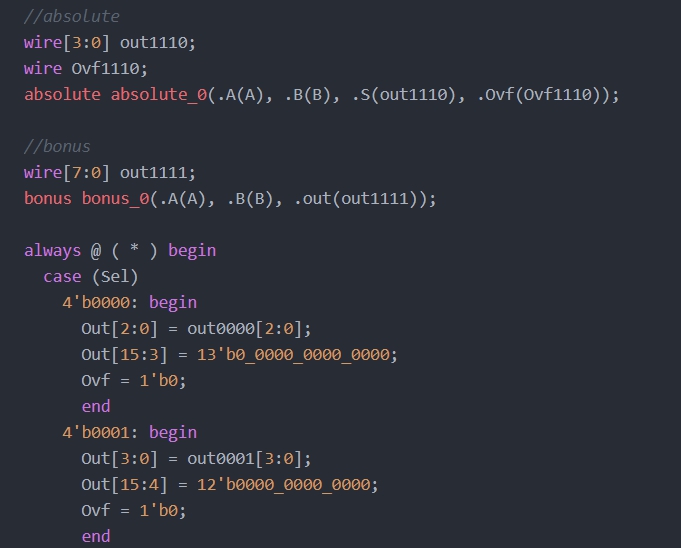
****

**分析成功，本以為結束了。**

****

**b.他又跳出這個警告，後來我上網查了發現是因為我的wire長的接到短的去。**

**解決方法：在主函數額外設out讓他可以接好，如下圖，之後再手動把前面補成0。**

****

**因為實在太長了，所以我就取，宣告的後面一點點，和mux的前面一點點來截圖，但中間都是一樣的，上面依據輸出的out來宣告對應的out(編號)，成功解決(b)的問題。**

**11.結論**

**這次的lab2剛開始看到的時候我完全沒頭緒，也不知道從何下手，因為lab1只有做一個函數而已，lab2突然要做16個，又要把他們融合在一起，剛開始真的很頭大，可是助教很有耐心，在我完成這份報告前總共問了助教3次，第一次是在下課後有先問過助教這次是要做出全部的就好，還是要合在一起。第二次在demo的時候我是先跟助教說我的合成想法，再問助教可不可行。問完後那個假日我就把程式碼打完了，花了整整兩天都在打。第三次就是問一些報告要交的東西，跑出來的結果這樣是不是對的，這些問題。**

**這次lab2讓我學到很多，至少在實作過程中，我碰上的問題比上面列的還要多，很多都很零碎就忘記了，但解決他們都會讓我對verilog更加熟悉。雖然期中考還是考不好，感覺突然要兜一些電路會兜不出來，可是我還是希望能在lab上面努力一點，把該學的學會。**