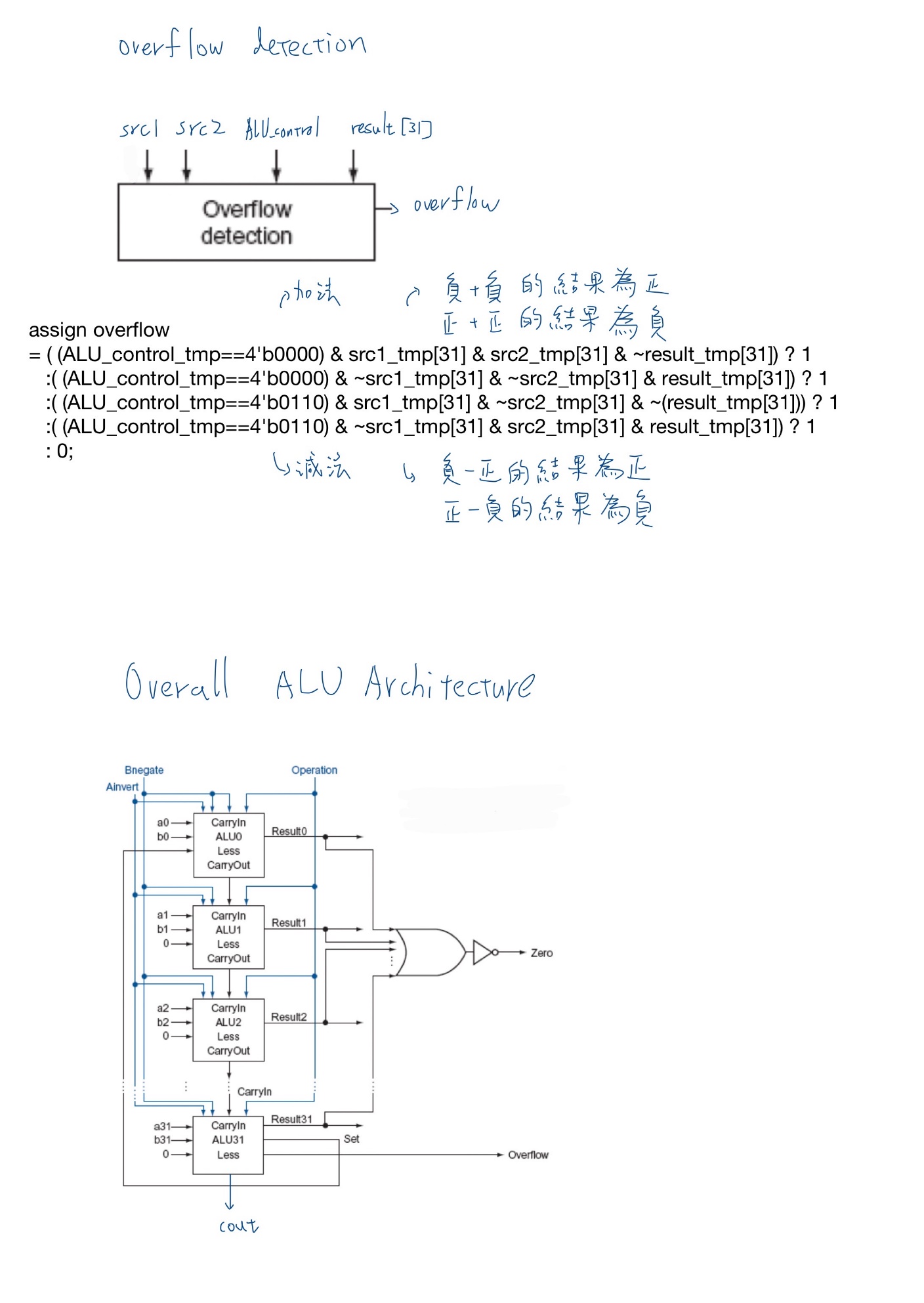
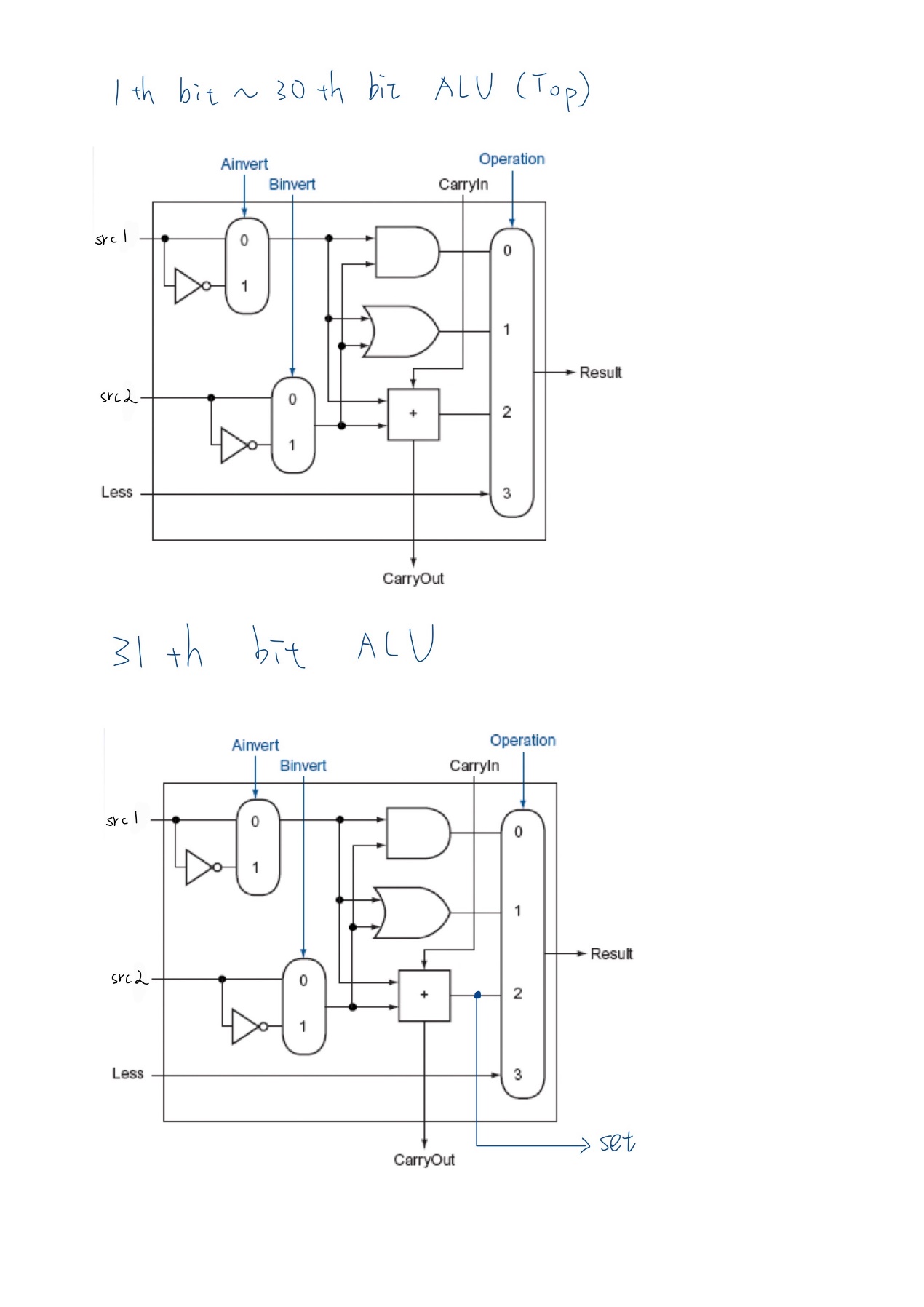
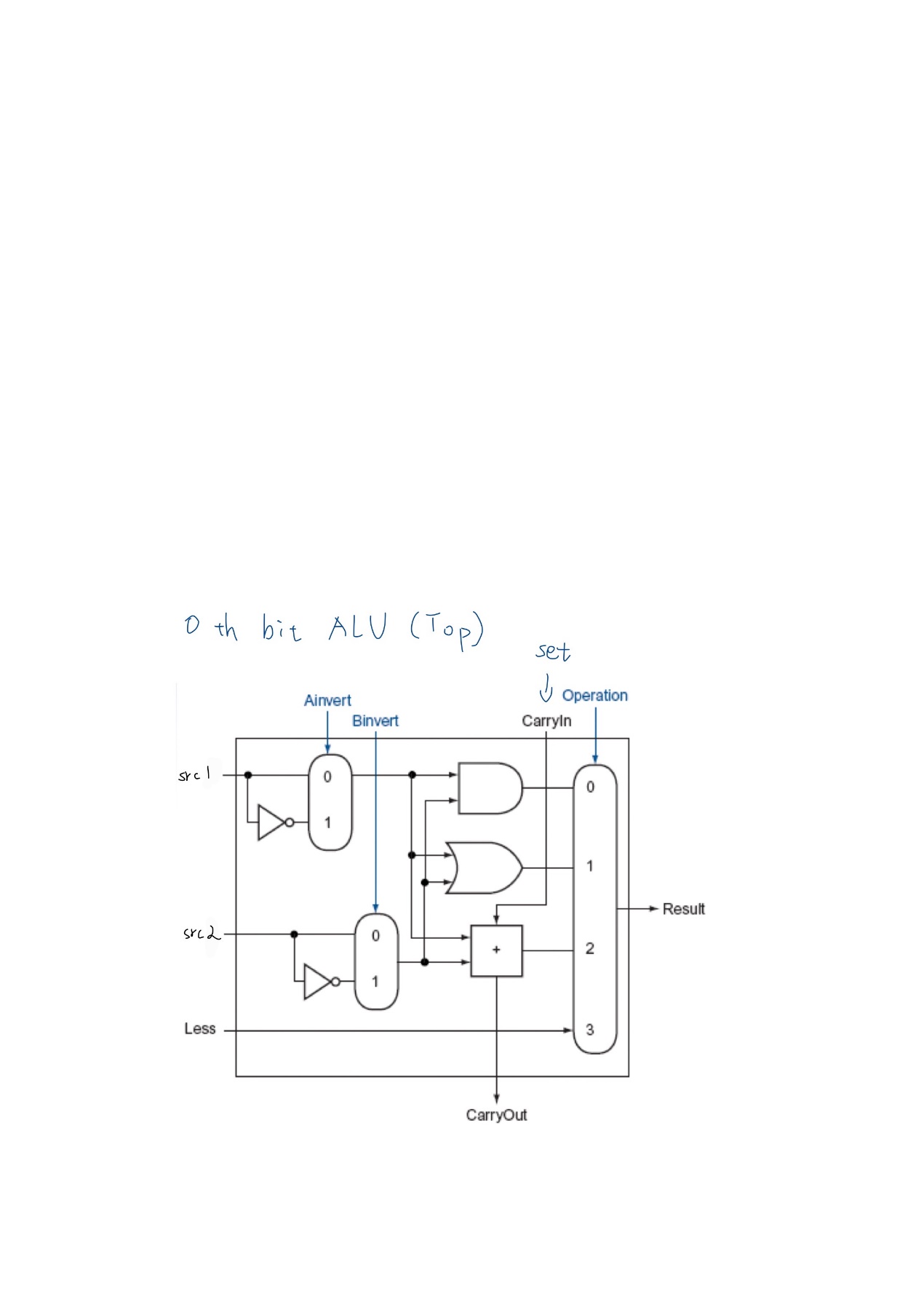
**Computer Organization**

**0819823陳子祈 LAB1**

**Architecture diagrams:**

****

**Hardware module analysis:**

1. **ALU**

**32-bit ALU可用devide and conquer來解決，也就是將問題拆成1個bit的ALU(devide)，解決1個bit的ALU(conquer)，並串連所有1-bit的ALU module(combine)，其詳細作法如下:根據ALU\_control的輸入來決定呼叫1-bit的ALU需要用到甚麼operation及src1、src2需不需要反向的A\_invert、B\_invert，再將src1、src2、less、operation、A\_invert、B\_invert、carry\_in等參數輸入到1-bit的ALU，然後將1-bit的ALU的輸出carry\_out接到下一級的ALU，1-bit的ALU也會輸出result，依此類推直到倒數第二個bit的ALU。因為set要接到第0個bit的ALU，所以最後一個bit的ALU長得有點不一樣，會多輸出set。這邊可以使用generate for的方式呼叫module，這樣就不用一個一個bit呼叫module，不用怕接錯也比較容易讀。**

**把所有的result都算完之後，再來輸出zero、cout、overflow。Zero就是看result所有bit是不是都是0，cout就是最後一個bit的ALU的cout。Overflow的設計可以參考上一頁的說明。**

1. **ALU\_top**

**ALU\_top就是除了最後一個bit之外其他所有的1-bit ALU，負責將src1、src2、less、operation、A\_invert、B\_invert、carry\_in等參數算出result與cout。先根據A\_invert、B\_invert看要不要將src1、src2變號，將結果放到s1、s2，再根據operation的不同，result與cout有各自的算法:**

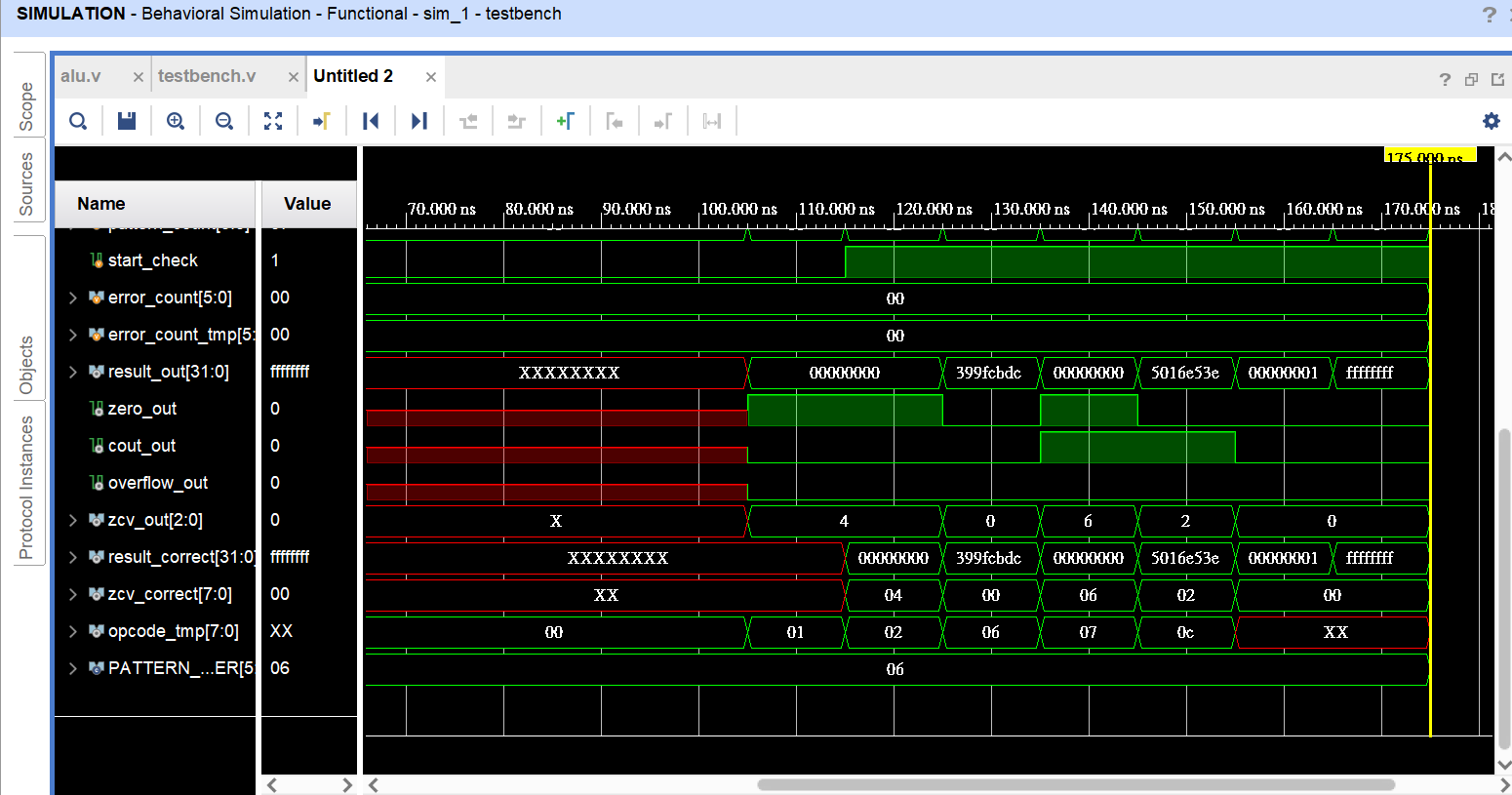
|  |  |  |
| --- | --- | --- |
| **Operation** | **result** | **cout** |
| **And** | **s1 & s2** | **0** |
| **Or** | **s1 | s2** | **0** |
| **ADD** | **s1 ^ s2 ^ cin** | **(s1&s2) | (s1&cin) | (s2&cin)** |
| **Set less than** | **less** | **(s1&s2) | (s1&cin) | (s2&cin)** |

1. **ALU\_last**

**如同前述所說，ALU\_last會比ALU\_top多一個set的輸出，set其實就是s1 ^ s2 ^ cin。Result與cout算法如下:**

|  |  |  |
| --- | --- | --- |
| **Operation** | **result** | **cout** |
| **And** | **s1 & s2** | **0** |
| **Or** | **s1 | s2** | **0** |
| **ADD** | **s1 ^ s2 ^ cin** | **(s1&s2) | (s1&cin) | (s2&cin)** |
| **Set less than** | **less** | **0** |

**Experiment result:**



**result\_out與result\_correct結果相同，也就是我的輸出與預期輸出結果相同。另外雖然我的zcv早一點出現，但check correct的時間是zcv\_correct有訊號的時候才會開始，所以我的結果測出來是對的。**

**Problems you met and solutions:**

**問題1: Verilog語法極不熟悉**

**解決: 透過不斷上網查資料，慢慢了解一些verilog的語法，雖然還不成熟，但已經盡力了。遇到比如呼叫module應寫在always前面還是後面、output要寫在always前面還是後面、wire跟reg的運用等問題，後來終於解決。**

**問題2: 結果總會早一個週期出現**

**解決: 本來我是直接把ouput接每個bit的module，每次result跟zcv都會早一個週期出現，後來改成進入always才讀取src1、src2，才終於解決這個問題。**

**Summary:**

**本次LAB是我第一次真正寫verilog，所以過程中遇到很多大大小小的問題，花了好多時間上網查、問學長才終於解決，希望可以將這次寫32 bits ALU的過程當成一次經驗，以後不會這麼害怕碰硬體描述的東西了。**