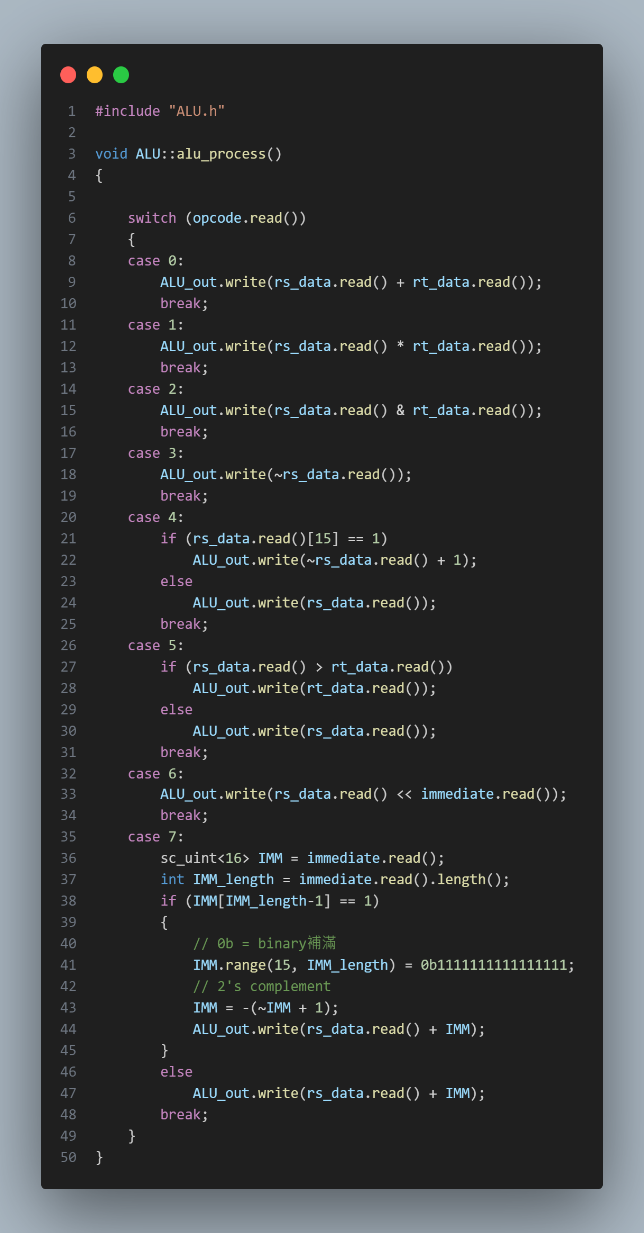
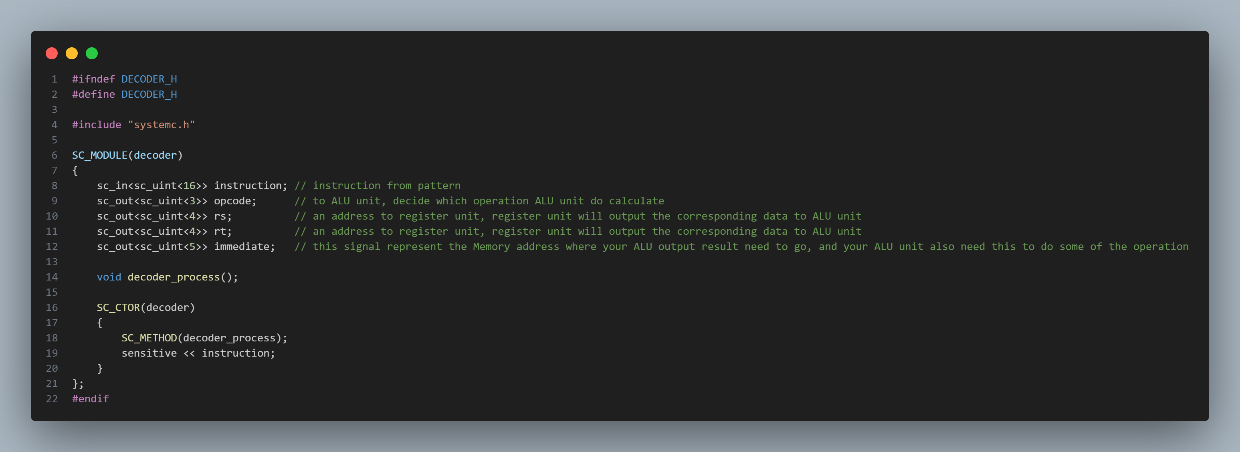
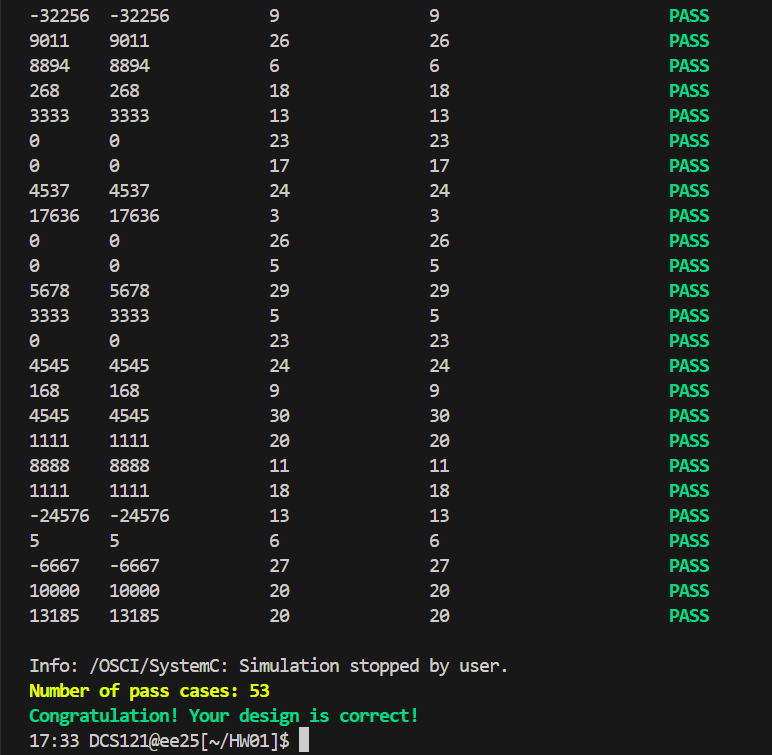
**Hw1\_109511314\_鄭旭恩**

***Central Processing Unit (CPU) - ALU & ID unit***

**1.Sketch:**

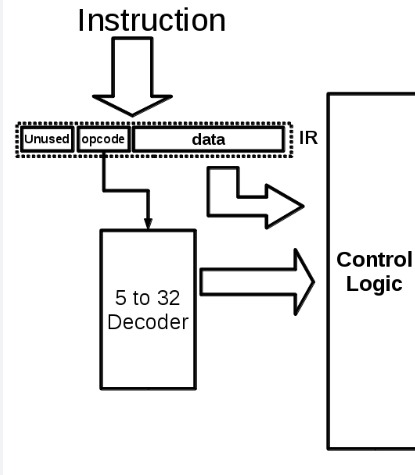
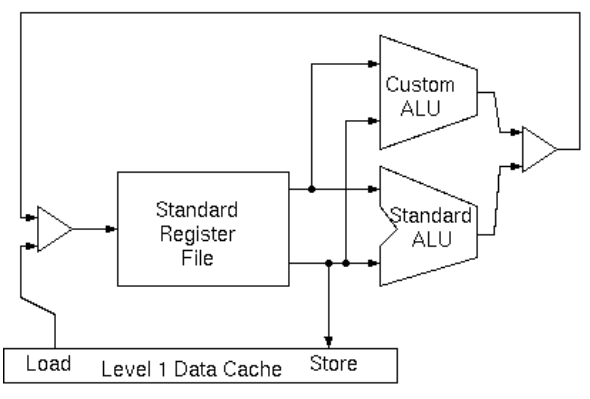
****

**2.Result:**

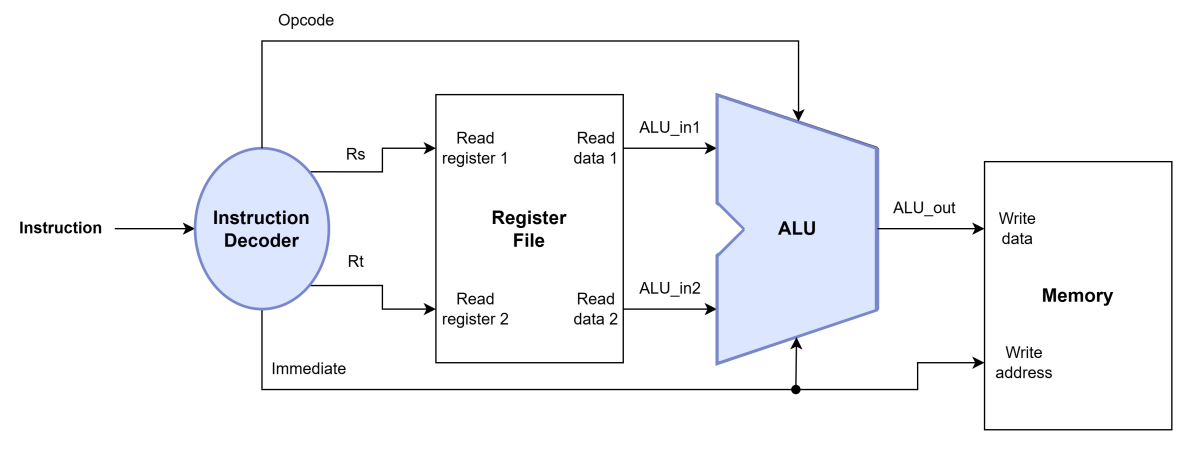
****

**3.Think:**

而ID是負責在前期負責解析及識別指令，讓處理器可以準確執行指令，並根據指令的要求動態配置資源。透過使用地址，可以更有系統地組織和管理資料，同時也有助於使系統設計更加簡潔。這種設計方式有助於提高系統的效率和可擴展性，同時也使得程式碼更易於理解和維護。如右下圖。

這次實驗中所用到的ALU和ID都是在設計CPU以及GPU等等運算單元時很重要的基本構造。ALU負責執行各種算術和邏輯運算，包括加法、減法、位元運算等。ALU非常重要因為其直接影響CPU或GPU等等的處理效能以及運算能力。如左上圖。



這次實驗的block diagram如上圖，在這份作業中，需要設計一個16位元的CPU，並設計一個算術邏輯單元（ALU）和一個指令解碼器（ID）單元。每個指令將根據指令集體系結構（ISA）執行不同的操作。最終，將結果將被存儲在特定的記憶體位置。

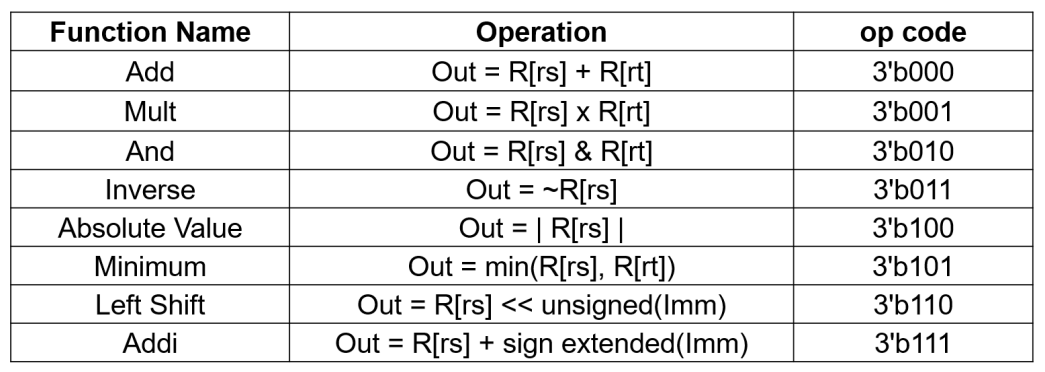
Rs和Rt表示特定register的地址。使用4位元的register address，共有16個可用register。每個寄存器可以容納16位元的數據。指令解碼器（ID）單元必須將相關位元分配給register file，使算術邏輯單元（ALU）能夠訪問其操作所需的數據。

Immediate表示一個記憶體地址。一個5位元的即時值表示有32個不同的記憶體塊可用。在你的架構中，ID單元需要將相關位元分配給這個記憶體，從而使ALU能夠正確執行指令。

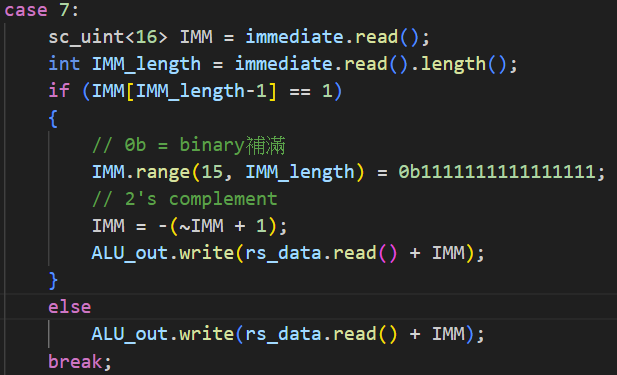
**4.Problem finding:**

這次實驗中，在設計ALU及ID的時分別都遇到一些需要解決的問題，如以下。

***ALU***

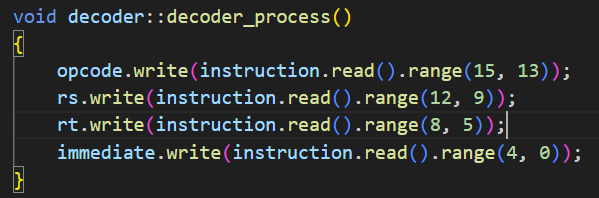
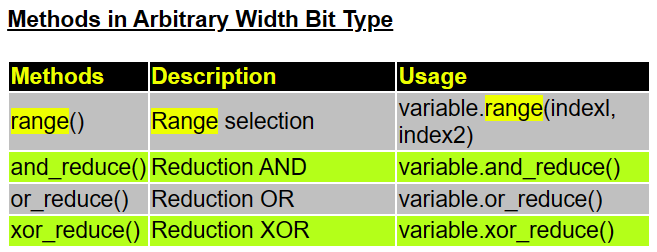
******

如上圖所示，此為在ALU中所要運行的程式。在前四個op code中並沒有遇到甚麼問題，因為都為基本的運算語法，同時也非常直觀。然而從第五個op code開始後開始需要用Behavior去描述其邏輯運算。尤其在op code的index[7]的時候。試過許多方式處理immediate的值，最後選擇先分類MSB是1還是0。若為1則先找出immediate的長度，並將剩餘的值做sign extension，接著做two’s complement。以immediate 24 =(11000)為例，轉成-(00111+1)=-(01000)，便輸出-8，再將其加上rs\_data。若MSB為0則直接加上rs\_data便可，如下圖。



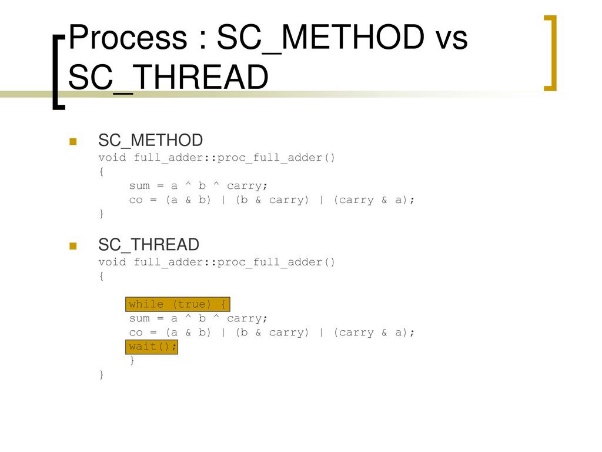
***ID***

再將instruction分成opcode、rs、rt、immediate的時候，我所用到systmec中的.range()函數，直接把instruction特定range的值取出寫入對應到的資料中。如下圖所示。



***SC\_METHOD v.s SC\_THREAD***

由於CPU為持續重複運算，只要有新的instruction進入，便會執行。因此在SC\_CTOR時，可以直接用SC\_METHOD，如同verilog的always語法。若要用THREAD則需要用while(true)迴圈包起來，並在結束時加上wait()。這也是我一開始在打程式不小心忽略的地方。



**5.Suggestion:**

Well done overall.

**6.Reference:**

* [SystemC DataTypes Part III (asic-world.com)](https://www.asic-world.com/systemc/data_types3.html)
* 
* [Chapter 3 : Combination and Sequential Circuits Modeling - ppt download (slideplayer.com)](https://slideplayer.com/slide/13258671/)
* [Part II CST SoC D/M Slide Pack 6 (Tools/Tech/Eng): Conservation Cores Approach (cam.ac.uk)](https://www.cl.cam.ac.uk/teaching/1213/SysOnChip/materials/sg6tte/zhpf2ee9a4a1.html)
* [IAY0340-Digital Systems Modeling and Synthesis (ttu.ee)](https://ati.ttu.ee/IAY0340/labs/Tutorials/VHDL/CPU_3.html)