1. 我這次作業的程式大致上和講義的架構一樣，由 CN.sv 接收 input，同時利用 register_file 來將收到的 register 位址轉成數值，再去做相對應的計算。計算的過程分為兩個部分，我將 sort 的部分寫成一個 module，他可以將 6 個數值，依照傳入的 opcode 來做排序。這個 module 會回傳排好的陣列，再將結果連到 Calc 的 module 中去做運算。
2. 排序：
   我是用 case 來判斷 opcode 的最後兩 bit 的順序，然後再用 bubble sort 將陣列排好。原本我是將大到小和小到大的迴圈部分分開，但是面積一直都是 22765。一開始我以為是迴圈的 interger 會開 32bit 導致佔了些空間，所以將它改成 3bit 試看看，但發現面積竟然一樣，我猜或許是被優化改掉了吧所以才沒差。但後來突然想到，在不同 case 中的 block 好像會被分開來合成，導致多了一大塊都是 for 部分的 gate，而 bubble sort 的 gate 感覺面積蠻大的，所以我就改成在得到數值陣列之後，就直接由小到大排序，再去看說要不要反過來擺，透過這樣的方法我的面積少了 6 千多!反過來擺的方式也是用 for 迴圈，但就是只要三個拿來交換的 register 就好了(2 個 2 個一組)，比起把兩個 for 分開寫小蠻多的。
3. 其他：
   我原本有查到利用 insertion sort 的方法，但是這次作業似乎比較不適合，因為資料是一次全部送進來的，若是要多用其他的判斷，將值一個一個 insert 到另一個 Array 的話似乎不用 clock 會有點麻煩。
   一開始的時候我是不想多開 5 個 register 抓資料，想直接把結果塞到 Array 中，但這樣會變成 combinational，沒辦法再開一個 always block 去做排序的動作，所以也是放棄這方法。
4. 心得：
   這次作業還算蠻簡單的，感覺都能夠蠻直覺地想到解法。
   另外我這學期有修計算機組織，感覺這次作業有點像是給我們一些指令集，當初點開來看到 OPCODE 和 register address 還以為點錯檔案了，想說計組明明還沒出作業阿 XD。不過也剛好因為這樣，在看敘述的時候能夠蠻快抓住重點的，算是運氣不錯吧。
   這門課剛開學給我的感覺有點像是 verilog tutorial 然後由助教發一些 lab 跟 HW 給我們練習上課的語法，雖然到目前為止的內容都還是自己之前就知道的，修起來目前也蠻輕鬆的，不過會希望能學到更多有關設計電路時的概念，譬如說像是在做加法器有哪些寫法，以及不同寫法會合成的不同電路模型，該如何在寫程式的時候同時心中想著電路等等，畢竟 verilog 語法就那些，如何從架構上去思考才是比較難學習到的部分~