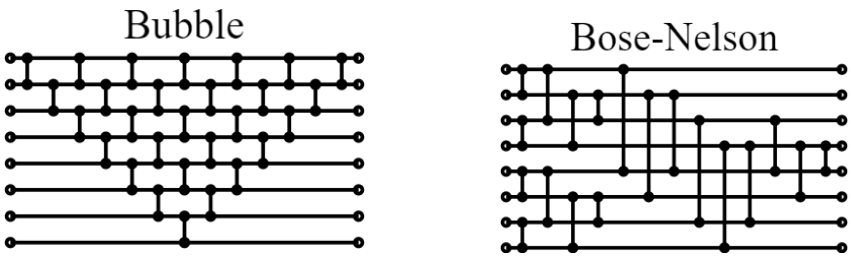
109511207 電機 13 蔡宗儒

(一) 如何設計作業

1.先將 Address 傳入 Register 中，獲得地址後將 Value 丟進 Sorting Block 內做 Sorting，再將 Sorting 後的結果丟進 Mux 內，最後再將第一個 Mux 出來的 Output 丟到第二個 Mux 內做計算。

2.Sorting 的方式應該是影響合成後的面積的最大因素。看到許多同學都使用了兩個 For Loop 並用 Bubble Sort 去做，但是跑兩次迴圈需要 N^2 也就是 36 次，這樣做其實很沒效率也浪費了很多面積。所以我採用的是 Bose-Nelson Sort，簡單來說就是在已知有多少個 Element 要排序的條件下，使用那個 Element 數的最佳方法，這樣可以保證 Sorting 一定是使用了最少次數，而六個 Element 的 Bose-Nelson Sort 只需要比 12 次即可，效率遠勝於 Bubble Sort 的 36 次。下圖為在 8 個 Element 時 Bubble Sort 和 Bose-Nelson Sort 的差異。

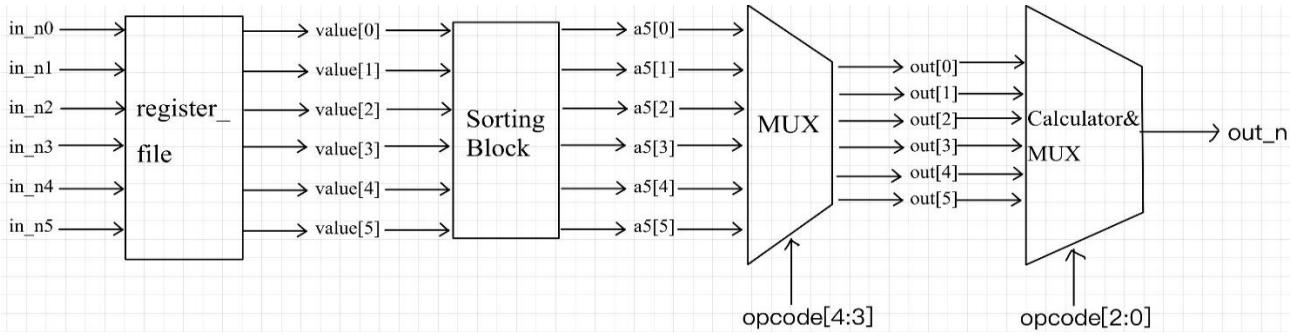


3.另外一個影響效率的因素就是大至小排列以及小至大排列有些人會排列兩次。這在軟體中是很直觀的概念，但在硬體語言卻不是。因為兩種都寫的話，實際的電路就需要把兩種排序都做出來，但這其實是脫褲子放屁。因為我們其實可以只做小至大排列(或只做大至小排序)，再將大至小的排列變成小至大排列的倒序就好，這樣就可以省下一大半的空間。於是我在 Always Block 內先做了小至大的排序，接著再使用 case，根據 opcode[4:3]的值決定排序方式，這樣就不用把同樣的排序方式因為寫了兩次而浪費面積，如下圖，這樣不但少打了很多行 Code，也省下很多面積。

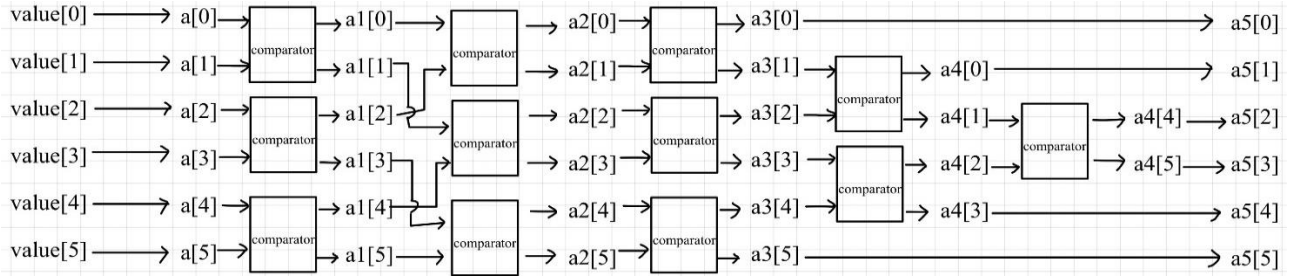
```
case(opcode[4:3])
    2'b11:
    begin
        out[0] = a5[0];
        out[1] = a5[1];
        out[2] = a5[2];
        out[3] = a5[3];
        out[4] = a5[4];
        out[5] = a5[5];
    end
    2'b10:
    begin
        out[0] = a5[5];
        out[1] = a5[4];
        out[2] = a5[3];
        out[3] = a5[2];
        out[4] = a5[1];
        out[5] = a5[0];
    end
```

(二) 架構圖

1.完整架構圖



2.Sorting Block 架構圖



(三) 心得

一開始在做作業的時候一直在找好一點的 Sorting 方法，雖然 Bubble Sort 很簡單，計概就教過，但是心裡很清楚用那種方式會讓做出來的電路面積一大塊，最後在網路上找尋了很久才找到了 Bose-Nelson 這個最佳方法。然後也學到了盡量不要像寫 C 語言一樣用 temp=a; a=b; b=temp; 因為這樣在電路上等同於把電路都接在一起，雖然結果可能不會錯、不會產生 Latch，但這終究不是一個好的寫法。在寫 Verilog 時不應該把他看成是在寫軟體語言，而應該要在心中隨時想著一塊電路，這樣才比較不會出錯。我後來也把這個寫法改掉，改完後面積也從 13029 變成 12633 小幅下降。

(四) Area & Delay

```
Total cell area:          12633.667370
Total area:                undefined
```

```
max_delay                  10.00      10.00
output external delay       0.00      10.00
data required time                    10.00
-------------------------------------------------------
data required time                    10.00
data arrival time                     -9.99
-------------------------------------------------------
slack (MET)                            0.01
```

(五) Reference

[1] Bose-Nelson Sort - https://hoytech.github.io/sorting-networks/

[2] Bose-Nelson Sort - https://reurl.cc/dXj83M