DCCDL LAB2

matlab

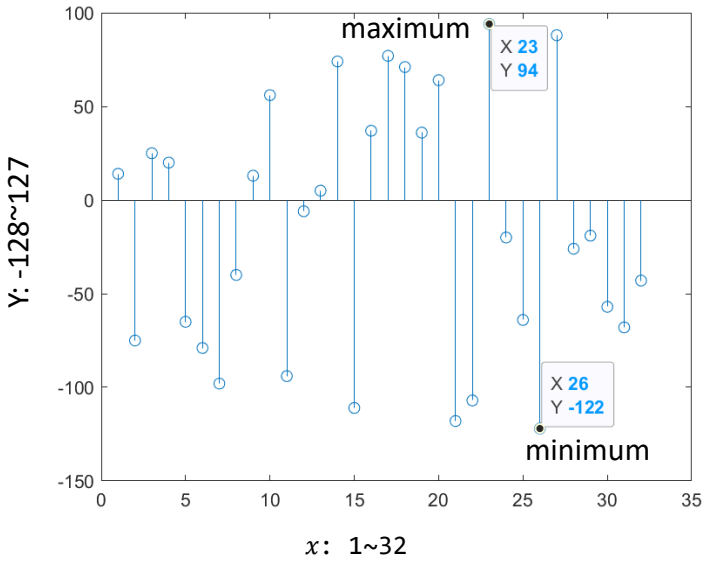
電機碩一　111521035　林豪澤

1.

使用 rng(32,”twister”); 指令設定亂數產生的種子。

The random sequence we generate with 32 elements is:

```
rand_32 = [14 -75 25 20 -65 -79 -98 -40 13 56 -94 -6 5 74 -111 37
           77 71 36 64 -118 -107 94 -20 -64 -122 88 -26 -19 -57 -68 -43]
```



The maximum value with index is　94/23

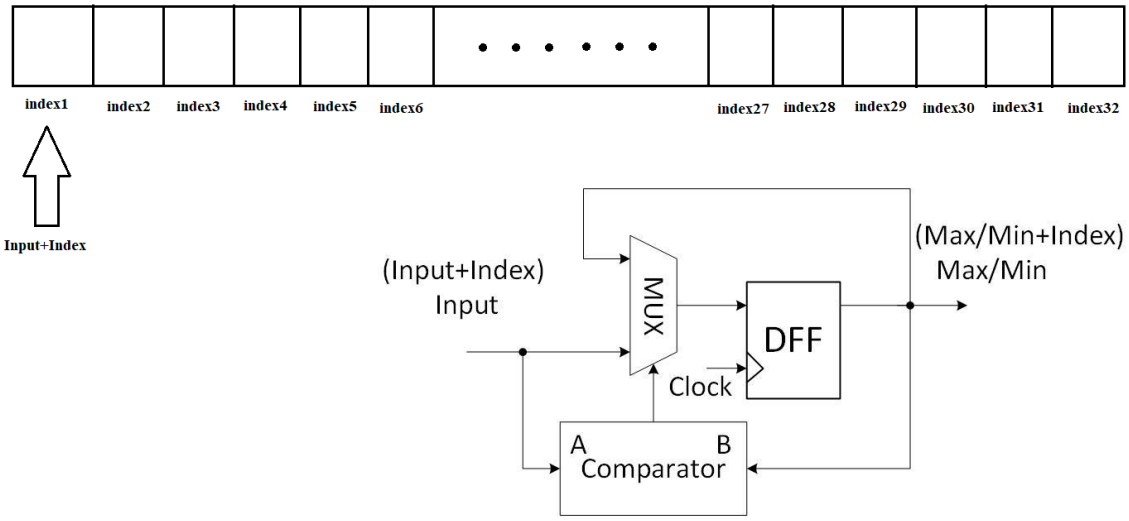
The minimum value with index is　-122/26

將Matrix中的value尾端加上其對應的index，如Matrix中第一個element其值為14，尾數加上其對應的index後即為1401，Matrix第二個element值為-75，經過處理後為-7502，以此類推。

```
mat_32_with_index = [1401 -7502 2503 2004 -6505 -7906 -9807 -4008 1309
                     5610 -9411 -612 513 7414 -11115 3716 7717 7118 3619
                     6420 -11821 -10722 9423 -2024 -6425 -12226 8827 -2628
                     -1929 -5730 -6831 -4332];
```

(execution results 在背面)

Matrix of random seqences contains 32 elements:



如上圖所示，先將陣列中的第一個element塞入DFF。接著將陣列中的元素逐一使用comparator做比較，找出陣列中的最小值。如果 A<B 則comparator輸出給MUX的訊號即為1，DFF儲存(Input+Index)的值；反之若 A>=B 則 comparator 輸出給MUX的訊號為0，DFF儲存上一次DFF儲存的值。

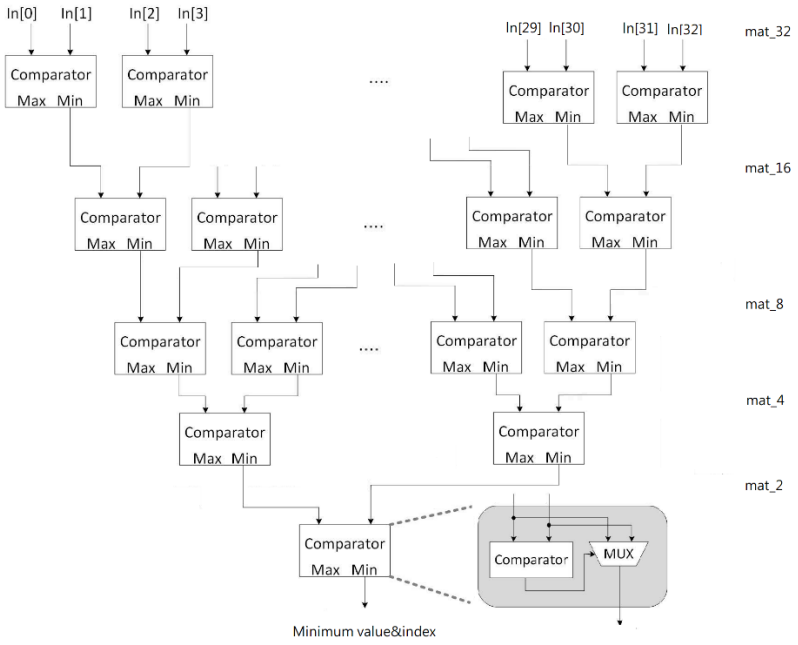
Show your execution results including index and value of your sequential comparator.

使用32個clock依序比較出陣列中Minimum value與其Index。

Command Window

```
Clock  1 : Compare   14 and   14     Value of comparator is 0    Min/Index = 14/1
Clock  2 : Compare   14 and  -75     Value of comparator is 1    Min/Index = -75/2
Clock  3 : Compare  -75 and   25     Value of comparator is 0    Min/Index = -75/2
Clock  4 : Compare  -75 and   20     Value of comparator is 0    Min/Index = -75/2
Clock  5 : Compare  -75 and  -65     Value of comparator is 0    Min/Index = -75/2
Clock  6 : Compare  -75 and  -79     Value of comparator is 1    Min/Index = -79/6
Clock  7 : Compare  -79 and  -98     Value of comparator is 1    Min/Index = -98/7
Clock  8 : Compare  -98 and  -40     Value of comparator is 0    Min/Index = -98/7
Clock  9 : Compare  -98 and   13     Value of comparator is 0    Min/Index = -98/7
Clock 10 : Compare  -98 and   56     Value of comparator is 0    Min/Index = -98/7
Clock 11 : Compare  -98 and  -94     Value of comparator is 0    Min/Index = -98/7
Clock 12 : Compare  -98 and   -6     Value of comparator is 0    Min/Index = -98/7
Clock 13 : Compare  -98 and    5     Value of comparator is 0    Min/Index = -98/7
Clock 14 : Compare  -98 and   74     Value of comparator is 0    Min/Index = -98/7
Clock 15 : Compare  -98 and -111     Value of comparator is 1    Min/Index = -111/15
Clock 16 : Compare -111 and   37     Value of comparator is 0    Min/Index = -111/15
Clock 17 : Compare -111 and   77     Value of comparator is 0    Min/Index = -111/15
Clock 18 : Compare -111 and   71     Value of comparator is 0    Min/Index = -111/15
Clock 19 : Compare -111 and   36     Value of comparator is 0    Min/Index = -111/15
Clock 20 : Compare -111 and   64     Value of comparator is 0    Min/Index = -111/15
Clock 21 : Compare -111 and -118     Value of comparator is 1    Min/Index = -118/21
Clock 22 : Compare -118 and -107     Value of comparator is 0    Min/Index = -118/21
Clock 23 : Compare -118 and   94     Value of comparator is 0    Min/Index = -118/21
Clock 24 : Compare -118 and  -20     Value of comparator is 0    Min/Index = -118/21
Clock 25 : Compare -118 and  -64     Value of comparator is 0    Min/Index = -118/21
Clock 26 : Compare -118 and -122     Value of comparator is 1    Min/Index = -122/26
Clock 27 : Compare -122 and   88     Value of comparator is 0    Min/Index = -122/26
Clock 28 : Compare -122 and  -26     Value of comparator is 0    Min/Index = -122/26
Clock 29 : Compare -122 and  -19     Value of comparator is 0    Min/Index = -122/26
Clock 30 : Compare -122 and  -57     Value of comparator is 0    Min/Index = -122/26
Clock 31 : Compare -122 and  -68     Value of comparator is 0    Min/Index = -122/26
Clock 32 : Compare -122 and  -43     Value of comparator is 0    Min/Index = -122/26
```

2. 由於受到 I/O 的限制，因此使用 32 個 clock 將 32 個隨機產生的數逐一使用 D-flip-flop 讀進 mat_32 中。接著如下圖所示，使用 parallel comparator 將上一層的數值兩兩比較，接者往下傳遞較小值。由此方法花費 5 個 clock 找到 32 個 element 中<u>最小的值</u>。



(Use the same inputs as in Q1)

(以下的 values 皆為加上 index 的數值，在 comparator 比較時會以原始值比較)

```
mat_32 = [1401 -7502 2503 2004 -6505 -7906 -9807 -4008 1309 5610 -9411
          -612 513 7414 -11115 3716 7717 7118 3619 6420 -11821 -10722
          9423 -2024 -6425 -12226 8827 -2628 -1929 -5730 -6831 -4332];
```

```
The inputs values and index is:
 1: Values/Index:   14/ 1       2: Values/Index:  -75/ 2       3: Values/Index:   25/ 3       4: Values/Index:   20/ 4
 5: Values/Index:  -65/ 5       6: Values/Index:  -79/ 6       7: Values/Index:  -98/ 7       8: Values/Index:  -40/ 8
 9: Values/Index:   13/ 9      10: Values/Index:   56/10      11: Values/Index:  -94/11      12: Values/Index:   -6/12
13: Values/Index:    5/13      14: Values/Index:   74/14      15: Values/Index: -111/15      16: Values/Index:   37/16
17: Values/Index:   77/17      18: Values/Index:   71/18      19: Values/Index:   36/19      20: Values/Index:   64/20
21: Values/Index: -118/21      22: Values/Index: -107/22      23: Values/Index:   94/23      24: Values/Index:  -20/24
25: Values/Index:  -64/25      26: Values/Index: -122/26      27: Values/Index:   88/27      28: Values/Index:  -26/28
29: Values/Index:  -19/29      30: Values/Index:  -57/30      31: Values/Index:  -68/31      32: Values/Index:  -43/32
```

```
mat_16 = [-7502 2004 -7906 -9807 1309 -9411 513 -11115 7118 3619 -11821 ...
          -2024 -12226 -2628 -5730 -6831];
```

```
Second layer:
 1: Values/Index:  -75/ 2       2: Values/Index:   20/ 4       3: Values/Index:  -79/ 6       4: Values/Index:  -98/ 7
 5: Values/Index:   13/ 9       6: Values/Index:  -94/11       7: Values/Index:    5/13       8: Values/Index: -111/15
 9: Values/Index:   71/18      10: Values/Index:   36/19      11: Values/Index: -118/21      12: Values/Index:  -20/24
13: Values/Index: -122/26      14: Values/Index:  -26/28      15: Values/Index:  -57/30      16: Values/Index:  -68/31
```

```
mat_8 = [-7502 -9807 -9411 -11115 3619 -11821 -12226 -6831];
```

```
Third layer:
 1: Values/Index:  -75/ 2       2: Values/Index:  -98/ 7       3: Values/Index:  -94/11       4: Values/Index: -111/15
 5: Values/Index:   36/19       6: Values/Index: -118/21       7: Values/Index: -122/26       8: Values/Index:  -68/31
```

```
mat_4 = [-9807 -11115 -11821 -12226];
```

```
Fourth layer:
 1: Values/Index:  -98/ 7       2: Values/Index: -111/15       3: Values/Index: -118/21       4: Values/Index: -122/26
```

```
mat_2 = [-11115 -12226];
```

```
Fifth layer:
 1: Values/Index: -111/15       2: Values/Index: -122/26
```

```
min_value =[-12226]
```

```
Sixth layer:
Values/Index: -122/26
```

3. Please compare the advantage and disadvantage of serial comparator and parallel comparator.

Serial comparator

Advantage:

所需的 I/O 腳位較少，comparator 也只需要一個就足以完成所有運算工作 -> 需要較少的硬體資源

在元件較少的基礎上，理論上消耗的能源也較少 -> 節省能源

Disadvantage: Serial comparator

採取的是一個時間換空間的戰術，在硬體資源較少的情況下理所當然的，完整處理完同樣資料量的工作也會較 parallel 長得多。在此條件下 Serial comparator 比較適合處理沒那麼具急迫性的資料。Non real-time

Parallel comparator

Advantage:

快速處理資料，serial comparator 可能需要 32 個 clock 才能處理完的運算，parallel comparator 只需要 5 個 clock 就能處理完畢。省時快速。

Disadvantage: Serial comparator

採取的是一個空間換時間的戰術，硬體資源相較 serial comparator 多了一倍不止，不僅如此也需要大量 I/O 腳位，成本更是多出許多。雖為相較之下的快速解法但是成本與合理性需要經過多方評估才能確定是否為最佳解。

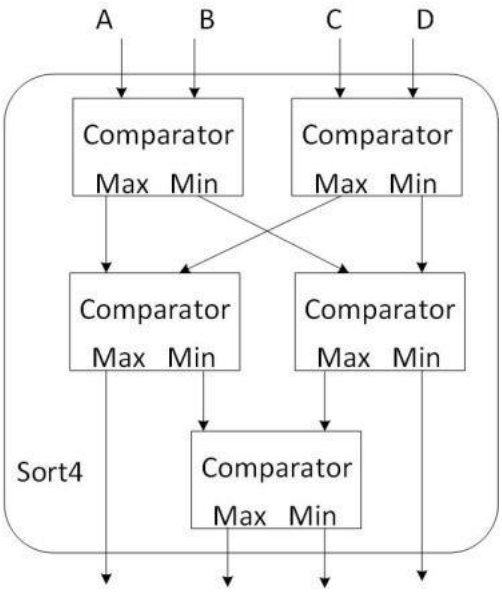
4. Print out your inputs and outputs of your function “Sort4”.

Inputs of function “Sort4” is

```
before_sort_4 = [1401 -7502 2503 2004]
```

Outputs of function “Sort4” is

```
sort_4 = [2503 2004 1401 -7502]
```



Command Window

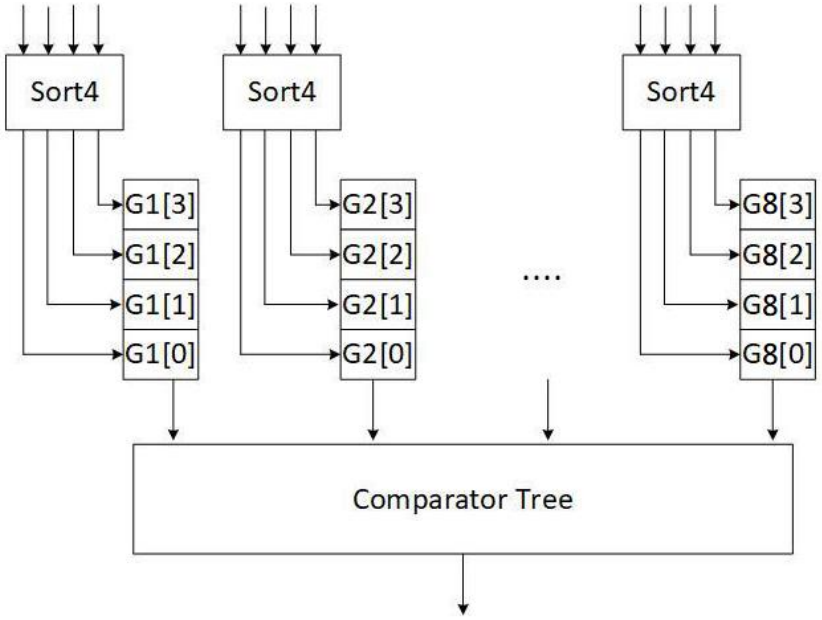
```
before_sort_4 =

        1401       -7502        2503        2004


sort_4 =

        2503        2004        1401       -7502
```

5. Print out the 6 outputs of the block SelectTop6 (value and index) among 32 input elements from your program and the results generated by the Matlab command “sort”.



(1)將 32 個 elements 每 4 個分成一個 group，總共有 8 個 group: G1~G8。
以 4*8 的陣列將所有 32 個 element 儲存起來，每個 column 為一個 group。

(2)先以 sort4 將每個 Group 由大至小排列一遍，由此一來每個 group 中 index 為 3 的位置則為那個 Group 中的最大值。建立一個 index 表，標註 8 個 Group 中即將受比較的值。

(3)使用 serial comparator 依序比較 8 個 group 中每個 Group 中的最大值，藉此找到剩下的 elements 中最大的值。將其儲存在答案的陣列中。

(4)找到此值之後標註其所屬的 Group，將其 index 減掉一，由此一來下次比較時即不會再次比較到該 Group 之最大值，而是次大的值。

(5)重複步驟(3)、(4)直到找到 32 個 elements 中前六大的值。

以下為

```
rand_32 = [14 -75 25 20 -65 -79 -98 -40 13 56 -94 -6 5 74 -111 37
          77 71 36 64 -118 -107 94 -20 -64 -122 88 -26 -19 -57 -68 -43];
```

中使用上述做法所得出的前 6 大 values 及其對應的 index

```
Top6_value = [94 88 77 74 71 64]
Top6_index = [23 27 17 14 18 20]
```

```
Top6_value =

    94    88    77    74    71    64


Top6_index =

    23    27    17    14    18    20
```

以下則是使用”Sort” command 產出的答案

```
using_sort_ans_6 =

    64    71    74    77    88    94
```