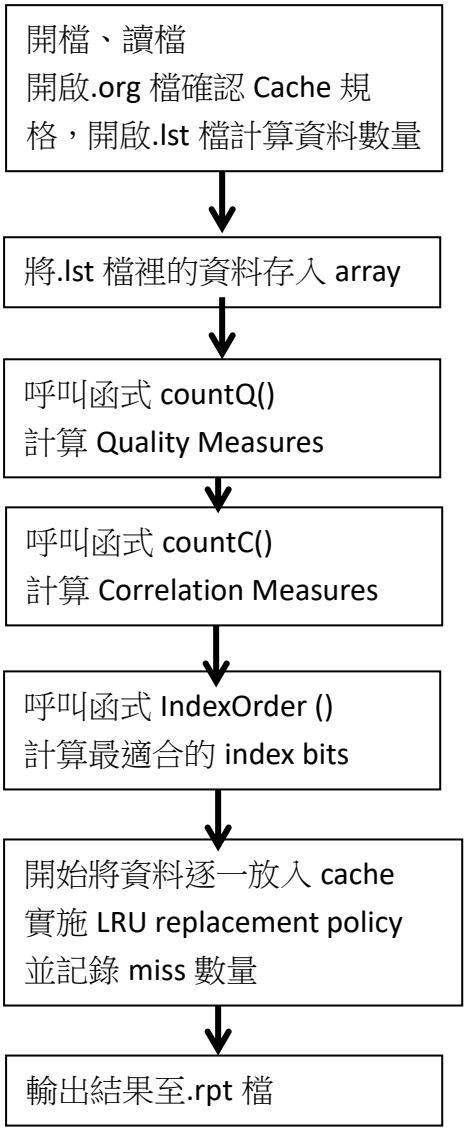
計算機結構 final project report

王立友 103033121

1. 程式流程



2. 函式內容

countQ()

Input: 存放所有資料的 array、Addressing_Bus 大小、資料量、要計算的 bit(一個)

Output: Quality Measures

作法: 計算該 column 的 0 和 1 的數量，分別存入 Z 和 O 這兩的變數，最後輸出結果為: (min(Z,O)/max(Z,O)

countC()

Input: 存放所有資料的 array、Addressing_Bus 大小、資料量、要計算的 bit(兩個)

Output: Correlation Measures

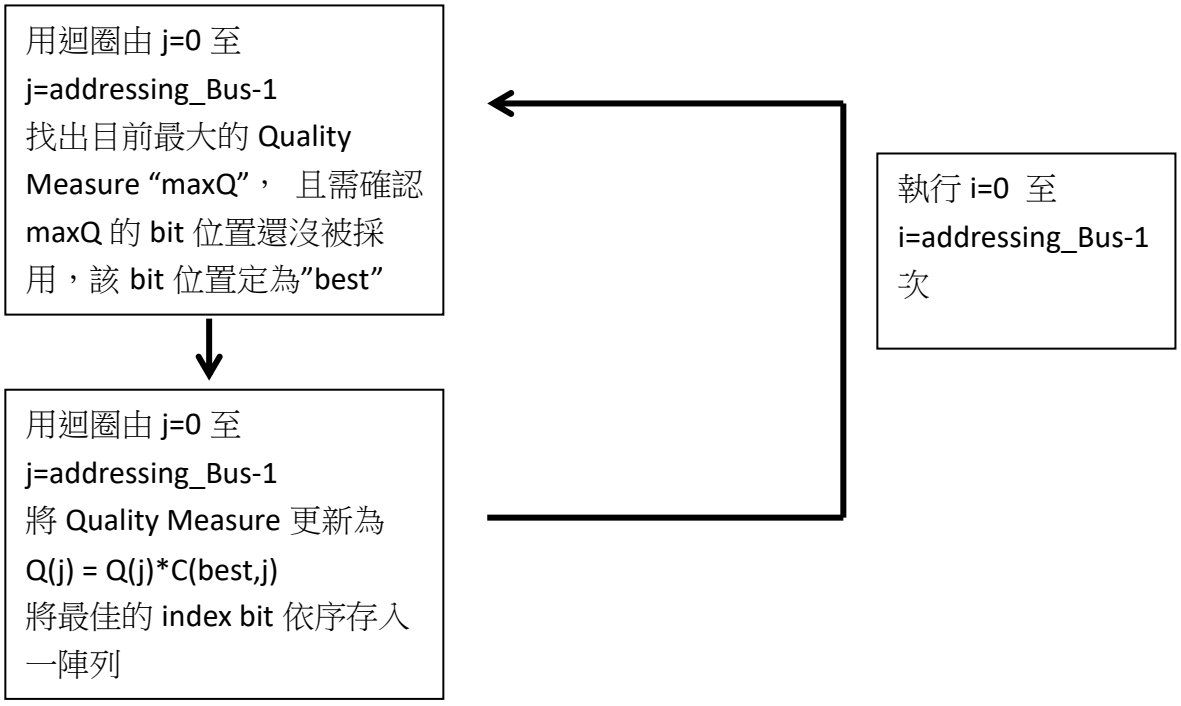
作法: 計算那兩個 column 的相同 bits 和相異 bits 的數量，分別存入 E 和 D 這兩的變數，最後輸出結果為: (min(E,D)/ (max(E,D)

IndexOrder()

Input: addressing_Bus 大小、存 Quality Measures 的陣列、存 Correlation Measures 的陣列、存最佳 index bits 的陣列

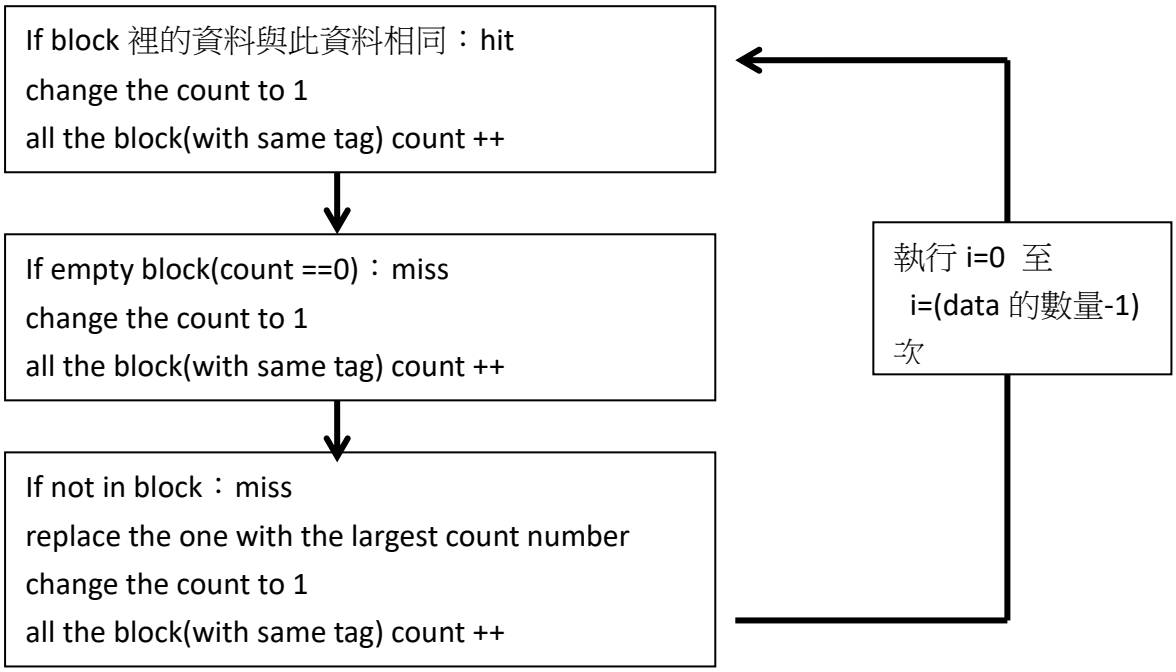
Output: 無回傳值

作法:



3. LRU 作法

定一陣列 count[][]，大小為 entries*associativity，初始時，每個 block 裡的 count 當歸零，每當有新資料存入時，該 block 的 count 設為 1，該 entries 裡的其他 block 的 count+1(除了空的 block)，當有需要 replace 時，尋找 count 最大的 block(表示最久沒更新)，將此 block 裡的資料替換掉，並將 count 設為 1。



執行結果：

執行結果成功通過 verify，所花時間不到一秒。

心得與討論：

我覺得本次 final project 最大的收穫是讓我更了解 cache 的運作原理，包括 index 的對照，LRU 的操作等等。除此之外，也增加了我的程式能力，例如我原本對讀檔寫檔幾乎不瞭解，做完這次的作業，讓我更加熟練了。起初執行結果常常會當掉，後來發現式函示裡有個部份的使用怪怪的，後來加以改善後就很順暢。我對於自己花費無數時間完成的這份 project，很有成就感。