**Computer Organization**

**0716214 江岳勳**

**0716222 黃偉傑**

**Detailed description of the implementation:**

* **direct\_mapped\_cache**
* **set\_assoiciative\_cache**

程式在啟動的時候會先將LRU.txt中的資料先讀取進來，然後按照各種不同的cache size/associativity組合來執行測試。

測試的一開始會先根據cache size和associativity算出block和set的數量，使用一個vector<vector<unsigned int>>來模擬set associative cache，內層的vector代表一個一個的set，新touch過的位址會被push到尾端，這會讓LRU永遠在vector的頭部。接著便從預先讀好的data中拿記憶體位址來做處理。

讀出來的記憶體位址要先除以block size，轉換成block address，然後再用set\_count對block address分別取餘和商，得到該block address的set編號及tag。接下來分為三種情況(從上至下if、else if、else)：

1. cache[set\_idx]包含tag，這樣的話代表該block已經在cache中了，將tag從vector erase掉，重新push\_back到尾端。**(HIT + 1)**
2. cache[set\_idx]大小未滿associativity，代表該block不在cache中，且該set仍有invalid的空白區域，直接將tag push\_back進去。**(MISS + 1)**
3. 以上皆非的話意味著要將LRU給替換成要寫入的tag，將vector的頭部元素erase掉，再將tag push\_back進去。**(MISS + 1)**

模擬完後只要將hit rate和miss rate算出輸出便結束了。

**Implementation results:**

* **direct\_mapped\_cache**
* **set\_assoiciative\_cache**

以下是miss rate在不同cache size和associativity組合下的數值：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1K | 2K | 4K | 8K | 16K | 32K | 64K | 128K |
| 1-way | 11.07% | 8.28% | 5.47% | 4.03% | 3.16% | 2.54% | 2.34% | 2.33% |
| 2-way | 8.36% | 5.18% | 3.63% | 2.98% | 2.37% | 2.33% | 2.29% | 2.28% |
| 4-way | 7.78% | 4.19% | 3.07% | 2.67% | 2.34% | 2.28% | 2.28% | 2.28% |
| 8-way | 7.83% | 3.98% | 2.81% | 2.45% | 2.29% | 2.28% | 2.28% | 2.28% |

從圖表中可以發現，miss rate隨著cache size和associativity的變大而降低，而且有出現趨近一個定值的情況。其中比較有趣的是1K-8way的miss rate反而比1K-4way的miss rate高了一些，我推測應該是由於測資的量不夠大，剛好有一些巧合讓8way反而比4way的miss rate還大。

**Problems encountered and solutions:**

* **direct\_mapped\_cache**
* **set\_assoiciative\_cache**

在一開始讀檔的時候，我使用ifstream搭配std::hex來處理十六進位的資料，卻發現一直失敗，打開LRU.txt手動模擬了一下後才發現裡面有會讓signed int overflow的數字，將code內的int換成unsigned int後便解決了。

模擬set associative cache的部分則沒有什麼問題。

**Comment:**

**0716214 江岳勳：**

本次我負責set associative的部分，實作過程中剛好在複習了一次整個第五章有關cache的部分，期末考前在複習時就有發現我常常會把byte address、block address、tag、index之類的名詞搞混在一起，經過這次作業有確實感覺到自己整個搞懂這個部份了。

**0716222 黃偉傑：**

aa