Computer Organization 2021
HOMEWORK 4

系級: 不分系112　　學號: F64081020　姓名: 劉翊安

Q1. How do you know the number of block from input file?

題目有給cache size以及block size，利用這兩個條件可以算出block的數量。

Number of block = cache size * 1024 / block size。

以下附上我的程式碼：

```
63 |    cache_block = cache_size * 1024 / block_size;
```

Q2. How do you know how many set in this cache?

Set的數量取決於associativity。

當associativity是direct-mapped，set = block的數量。

當associativity是4_way，set = block的數量除以4。

當associativity是fully，set = 1。

以下附上我的程式碼：

```
65 |    if(associativity == 0){//direct-mapped
66 |    |    cache_set = cache_block;
67 |    |    cache_set_length = 2;//#1 valid + #1 tag
68 |    }
69 |    else if(associativity == 1){//4_way
70 |    |    cache_set = cache_block / 4;
71 |    |    cache_set_length = 5;//#1 valid + #4 tag
72 |    }
73 |    else{//fully
74 |    |    cache_set = 1;
75 |    |    cache_set_length = 1 + cache_block;//#1 valid + #cache_block tag
76 |    }|   //printf("%d", cache_block);
```

Q3. How do you know the bits of the width of the Tag ?

我是將所有數字轉成十進位來運行程式的，因為tag在binary的位數是扣掉offest+index位數的address最高位數。因此tag = address的十進位數字 / 2^(offest位數 + index位數)

以下附上我的程式碼：

```
15 int get_tag(long int dec_string_addr, int index_length, int offset_length){
16 |    int tag;
17 |    tag = dec_string_addr / pow(2, index_length + offset_length);|  //printf("%d %d ", index_length, offset_length); |  //printf("%ld, %f\n
   ", dec_string_addr, pow(2, index_length + offset_length));
18 |    return tag;
19 }
```

Q4. Briefly describe your data structure of your cache.

1. 算出所有的變數
2. 用二維矩陣建cache
3. While(讀取address)

//逐行讀取address，根據不同的動作做相應的指令，印出結果

//判斷valid, hit or miss

If(not valid)

Vaild記為1

矩陣第一格紀錄tag

印-1

Else if(valid)

If(hit) //根據replace policy做相應的動作

If(associvity = LRU)

找到該tag移到array最後一個位置

印-1

Else if(miss)

If(array沒有滿)

在array中第一個空的位置放入tag

印-1

Else if(array滿了)

If(replace == your policy)

隨機印出一個tag

新tag取代被印出的那一格

else

印出array第一格的tag

全部往前移一格，把新的tag放到最後一格

Q5. Briefly describe your algorithm of LRU.

Array的越前面放的是越早使用到的tag

If(hit) 把該tag一到最後一個

Else if(miss)

If(NOT full)在最後一格加入新的tag

Else(full)

印出第一個tag

全部往前移一格

最後一格放入新的tag

Q6. Briefly describe your algorithm of your policy.

當miss且array滿的時候，利用random函數隨機print一個tag，並且用新的tag取代。

Q7. Run trace2.txt, trace3.txt and then makefile to get the miss rate and put it in your report.

Trace2.txt:

```
peggy@ubuntu:/mnt/hgfs/109-2(2-2)/computer_organization/hw4_cache_ian/F740XXXXX/
src$  make verify REQ_FILE=trace2.txt VIC_FILE=trace2.out
./cache  trace2.txt trace2.out
1, 129# @python3   trace2.txt trace2.out
./Verify  trace2.txt trace2.out
 ======   Configuration   ======

    cache size :            2 KB
    block size :           16  B
 associativity :         128-way
        policy :            LRU

 ======     Simulation    ======

 =========== SUMMARY ===========
 |     Miss rate: 0.000597     |
 ===============================
```

Trace3.txt:

```
peggy@ubuntu:/mnt/hgfs/109-2(2-2)/computer_organization/hw4_cache_ian/F740XXXXX/
src$ make verify REQ_FILE=trace3.txt VIC_FILE=trace3.out
./cache  trace3.txt trace3.out
65536, 5# @python3   trace3.txt trace3.out
./Verify  trace3.txt trace3.out
 ======   Configuration   ======

    cache size :          256 KB
    block size :            1  B
 associativity :           4-way
        policy :  Design by self

 ======     Simulation    ======

 =========== SUMMARY ===========
 |     Miss rate: 0.000076     |
 ===============================
```

心得(Report)

這次的作業在一開始下載ubuntu虛擬環境時碰到了一些問題，兩週的作業我用了一週的時間才把環境建好。

開始打程式以後我認為最困難的地方是剛開始要動筆的時候，如果沒有把上課內容讀到透徹，很難在腦海中有一個明確的架構要怎麼寫。因此我又花了幾天好好的研究cache的運作方式。在心裡有明確架構以後，其實城市打起來就很順利了，只是偶爾會有一些bug要仔細找才找得到。

這次我有稍微紀錄了一下當中碰到的困難。1.讀、寫command line上的檔案。2. 傳送二維array到函式。3.找程式中的bug。基本上這些問題上網找就能夠解決，程式中的bug　　我最後是靠直接把所有的array印出來解決的。
整體來說，我覺得這次作業並不困難，只是花時間。