Lab 4: Cache Simulator Report

0613149 翁羽亮 0613144 葉之晴

1、 Basic Problems

(a) Verilog

用 LAB3 的 v 程式碼實作 cpu

A 改變

1.”Instruction_Memory.v”：改變 instruction_file 的大小，從[0:31]變為[0:64]，讀取 lab4_test_data.txt 的資料

2.”TestBench.v”：用已知的 testbench.v 取代原本的 testbench.v

B 輸出

跑 cpu 之後，會寫入 ICACHE 和 DCACHE 資料進入相對應的.txt 資料裡

(b) cpp codes

1.direct_mapped_cache.cpp

A. function “simulate”:將兩個 double 類別的命中次數和總和相加，計算取用資料時的命中次數

可知 hit rate 和 miss rate：如果 cache hit ，可以在變數 hit 中+1

然後每次讀取輸入資料時，變數 total 可以+1

B. main function

用兩個 loop 來呼叫”simulate”

⇨
```
int i, j;
for(i = 1 ; i < 5 ; i++){
        for(j = 0 ; j < 5 ; j++)
        simulate(pow(2, i*2) * K, pow(2, j+4));
        cout << endl;
  }
```

因為需要 cache size: in 4K, 16K, 64K, 256K

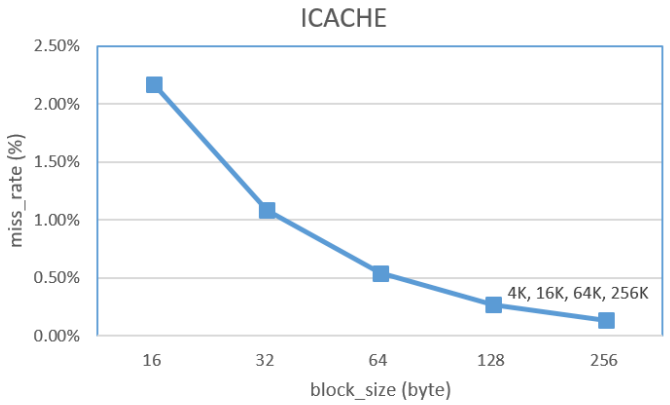
⇨ They are all 2 to the power of i*2

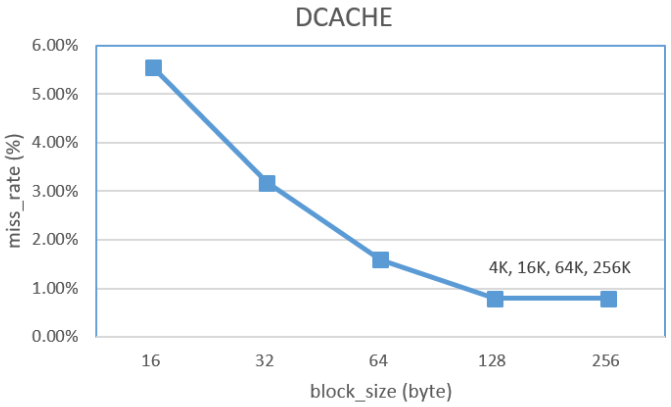
⇨ 2^(1*2) K, 2^(2*2) K, 2^(3*2) K, 2^(4*2) K

And the block_size are 16 B, 32 B, 64 B, 128 B, 256 B

⇨ They are all 2 to the power of j+4

⇨ 2^(0+4) B, 2^(1+4) B, 2^(2+4) B, 2^(3+4) B, 2^(4+4) B





2、 Direct_mapped_cache_LRU.cpp

A. struct “cache_content”: 增加一個 int array “data”(size: 16)來儲存資料

B. function “simulate”: 呼叫 function 時，新增一個參數”ways”，這樣就可以知道現在有多少”ways”

C. main function：用兩個 loop 來呼叫”simulate”

⇨
```
int i, j;
for(i = 0 ; i < 6 ; i++){
        for(j = 0 ; j < 4 ; j++)
        simulate(pow(2, j), pow(2, i) * K, 64);
        cout << endl;
  }
```

Because it has to run 1-way, 2-way, 4-way, 8-way

⇨ They are all 2 to the power of j

⇨ 2^0-way, 2^1-way, 2^2-way, 2^3-way

And cache_size are in 1K, 2K, 4K, 8K, 16K, 32K

⇨ They are all 2 to the power of i

⇨ 2^0 K, 2^1 K, 2^2 K, 2^3 K, 2^4 K, 2^5 K

(※If you run the code, it will automatically print out its cache line, cache_size, block_size, miss_rate, and n-way associative.)