

Computer Organization Lab 5: Cache Performance

Author ID: 0716316 Name: 洪珩均

Teammate ID: 0716085 Name: 賴品樺

(1) Execution result

```
phlai410277@linux1:~/Lab5
[phlai410277@linux1 ~/Lab5]$ make
g++ main.cpp -o simulate_caches
[phlai410277@linux1 ~/Lab5]$ ./simulate_caches a1xb1 out
[phlai410277@linux1 ~/Lab5]$ cat out
70 80 90 90
158 184 210 210
246 288 330 330
334 392 450 450
1553 6016 1648 12144
[phlai410277@linux1 ~/Lab5]$ diff out sample_c1
[phlai410277@linux1 ~/Lab5]$ ./simulate_caches a2xb2 out
[phlai410277@linux1 ~/Lab5]$ cat out
617 630 647 662 677 692 707 722
920 939 968 992 1016 1040 1064 1088
2016 2059 2128 2184 2240 2296 2352 2408
7942 8111 8346 8548 8750 8952 9154 9356
18469 18853 19483 19990 20497 21004 21511 22018
34864 35603 36784 37744 38704 39664 40624 41584
47680 48694 50352 51688 53024 54360 55696 57032
63308 64644 66868 68648 70428 72208 73988 75768
6141 19072 5968 32736
[phlai410277@linux1 ~/Lab5]$ diff out sample_c2
[phlai410277@linux1 ~/Lab5]$ ./simulate_caches a3xb3 out
[phlai410277@linux1 ~/Lab5]$ ./simulate_caches a4xb4 out
[phlai410277@linux1 ~/Lab5]$
```

(2) Analysis and discussions for lab 5

Every time we access the cache, no matter hit or not, we should add up minimum memory stall cycle. We first check if the address hit or not. If miss, go to the next layer searching for the target address, add up miss penalty and also check if there is any invalid cache block. Our program will try to find the invalid slot which is nearest to the starting slot. If all slots are valid, replace the one that is least recently unused. Compare with case a and b, since the bandwidth in case b is 8-words wide, the miss penalty is less than in case a. Compare with case a and c, not only because the miss rate in L1 cache of case 3 is higher but also the block size of L2 cache is 32-word large, case 3 spend much more time on accessing memory and send a word to upper layer one at a time.