Computer Organization, Spring 2019

Lab 4: Cache Simulator

Due: 2019/06/10

1. Goal

Cache performance is important for system performance. In this lab, you are asked to simulate cache behaviors by C/C++ style cache simulators. By this training, you would understand the performance difference between different cache architectures.

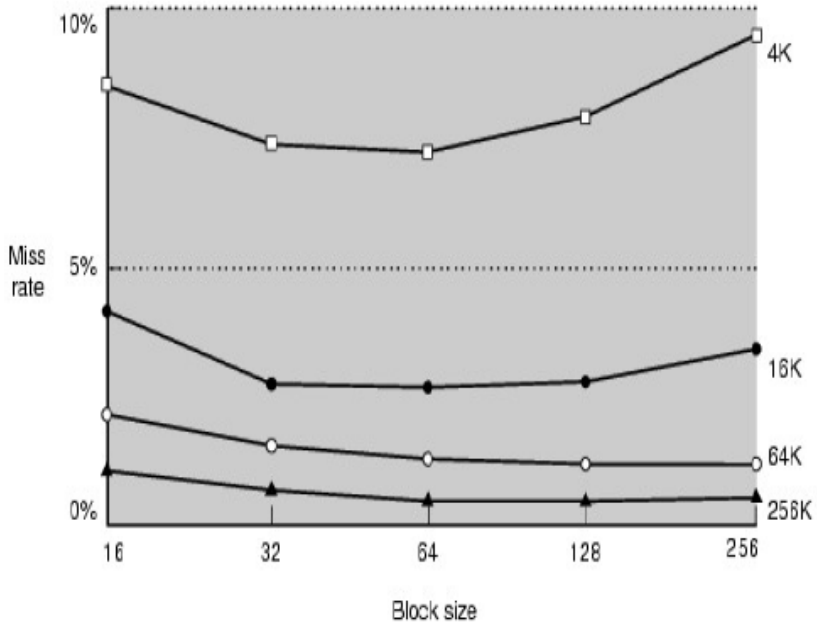
2. Basic Problem (60%)

You should trace the memory addresses in this lab. The supplied files are described as follows:

a. **"lab4_test_data.txt" -**
The file is a 3-dimensional matrix multiplication program. You can find the explanation of the code in "lab4_test_data_assembly.pdf".

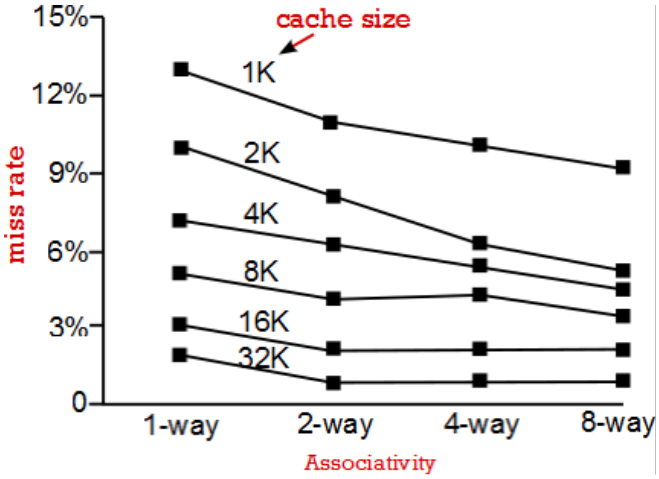
b. **"ICACHE.txt" & "DCACHE.txt" -**
The memory trace from "lab4_test_data.txt". (Do not need to write any verilog code in this lab, just use these two files as inputs of your simulators)

c. **"direct_mapped_cache.cpp" -**
A simple cache simulator. You should modify this simulator so that it can output the miss rate of the cache. Please take the given file "ICACHE.txt" and "DCACHE.txt" as inputs of the simulator and then run it. First, change the parameters (cache size or block size) when you do the simulation. Then draw a graph as the following example and describe the reason of rise and fall of the lines in the report. (Please separate ICACHE from DCACHE)



3. Advanced Problem (40%)

LRU stands for Least-Recently Used, which is a replacement policy of choosing the one unused for the longest time. In this problem, you have to implement an n-way set-associative cache simulator using LRU (by C/C++, refer to the supplied file "direct_mapped_cache.cpp"). Please name this new simulator as "direct_mapped_cache_lru.cpp". Set block size = 64 bytes. Then input the file "LU.txt" and "RADIX.txt" that are the memory trace from two benchmarks to the simulator. Please draw a graph as the following example and describe the reason of rise and fall of the lines in the report. (Please separate the discussion of LU and RADIX). Also, please compute the total bits (including tags and one valid bit) required for each cache, and show them by table (like following table) or plot the figure.



Associativit / Cache Size	1-way	2-way	4-way	8-way
1K				
2K				
4K				
8K				
16K				
32K				

4. Grade

a. Total score: 100%, **Copy will get 0 point!**

b. Basic score: 50% + 10% (Report)

c. Advanced score: 30% + 10% (Report)

d. **Delay: 10% off per day**

5. Hand in your Assignment

a. Please upload your assignment to new E3.

b. Please zip your folder and submit only one *.zip file per team. Either you or your teammate (but not both) may make the submission. Name the *.zip file with your student IDs (e.g., 0516001_0516002.zip). Other filenames and formats such as *.rar and *.7z are **NOT** accepted!

c. Please include **ONLY** *.cpp, makefile (for compilation) and your report (only pdf is accepted) ißn the zipped folder - NO OTHER FILE!

d. Platform: linux

e. Please write your own makefile to compile your own C/C++ code. Make sure your makefile could work !!

6. Q&A

If you have any question regarding Lab 4, please send email to 陳均騰 (k.james0629@gmail.com).