Computer Organization 2018

HOMEWORK 4

Due day: 2018/6/14

Overview

In memory hierarchy, the small memory, cache, is used to keep data temporarily for increasing the system performance. If the data is in the cache memory, the processor can get the data with high accessing speed. If the data is not in the cache, called cache miss, the processor will read data from main memory. Accessing data from main memory is slower than accessing it from the cache memory. The goal of this homework is to help you understand the cache. In this homework, you need to write a cache simulator. Please follow the specification in this homework and satisfy all the homework requirements.

General rules for deliverables

- You need to complete this homework INDIVIDUALLY. You can discuss the homework with other students, but you need to do the homework by yourself. You should not copy anything from someone else, and you should not distribute your homework to someone else. If you violate any of these rules, you will get NEGATIVE scores, or even fail this course directly
- When submitting your homework, compress all files into a single zip file, and upload the compressed file to Moodle.
  - Please follow the file hierarchy shown in Figure 1.

F750XXXXX(your id)(folder)

SRC(folder) * Store your source code

F750XXXXX_Report.docx (Project Report. The report template is already included. Follow the template to complete the report.)

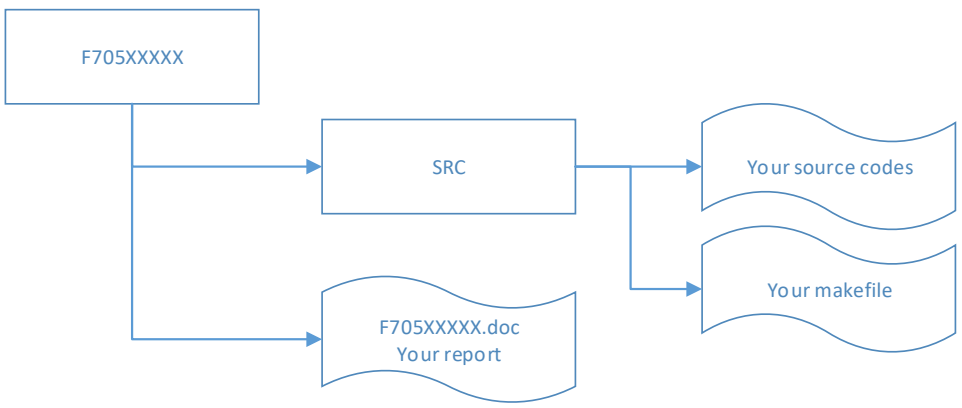

Figure 1. File hierarchy for homework submission

- **Important!** DO NOT submit your homework in the last minute. Late submission is not accepted.
- You should finish all the requirements (shown below) in this homework and Project report.

Homework Specification

a. Width of memory (line size) : 32bits

b. Unit of memory : byte.

c. Cache size : Read from test cases. With variable cache size from 1KB up to 256KB.

d. Block size : Read from test cases. From 16B to 256B.

e. Associativity : Read from test cases. Various associativity from direct-mapped to fully associative .

f. Replaced Algorithm : Read from test cases. Will be FIFO or LRU.

g. All cache is initialized 0.

Input file format

We will give .txt file which contains the test cases.

The file contains a trace of memory accesses executed from some benchmark program.

The 1st line specifies the cache size.

The 2nd line specifies the block size.

The 3rd line specifies the associativity. 0 represents direct-mapped, 1 represents four-way set associative, 2 represents fully associative.

The 4th line specifies the Replace algorithm. 0 represents FIFO, 1 represents LRU.

The rest of the test case is a trace of memory accesses executed from some benchmark program.

trace1.txt

Sample Input
256 // cache size
32 // block size
2 // associativity
0 // FIFO=0 , LRU=1
0xbfa437cc // No. 1
0xbfa437c8 // No. 2 ……
0xbfa437c4
0xbfa437c0
0xbfa437bc
0xbfa437b8
0xbfa437b8
0xbfa43794
0xb8088ea8
0xb8088eac

Output of your cache simulator

Take trace1 for example, Output to file named trace1.out, and output the following information.

Sample Output
Hits instructions: 2,3,4,6,7,10 Misses instructions: 1,5,8,9 Miss rate: 0.4

Homework Requirements

A. Report

Hand in a report answering the following questions

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

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

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

Q4. Briefly describe your data structure of your cache.

Q5. Briefly describe your algorithm of the implementation of LRU.

B. Language/Platform

Please implement this Homework in C or C++ language. The program will be compiled using GCC 5.4.0 on Linux Ubuntu 14.01.

C. Command Line Format

./StudentID –input <trace.txt> -output <trace.out>

Example: ./F750XXXXX –input /benchmark/trace1.txt –output /out/trace1.out

D. Note

Note that you should write your own make file.

Important

When you upload your file, please check you have done and followed all requirements, including **File hierarchy**, **Requirement file** and **Report format**.

If you have any questions, please contact us.