HOMEWORK 6 Cache Simulator

Due date:

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, it is called cache miss, and 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.

F740XXXXX (your id) (folder)

src (folder)   * Store your source code

report.docx (project report. The report template is already included. Follow the template to complete the report.)

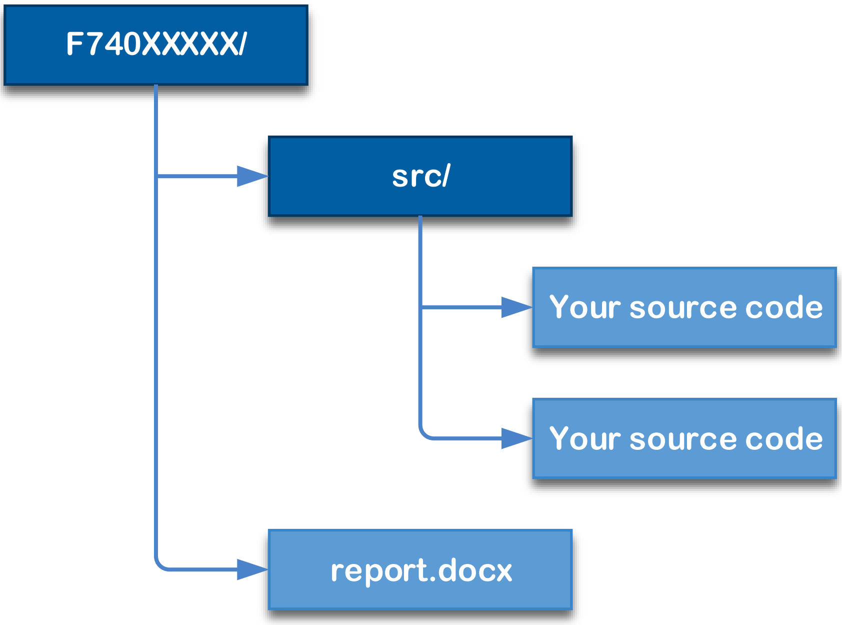

Figure 1. File hierarchy for homework submission

- **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. Cache size : Read from test cases.
b. Block size : Read from test cases.
c. Associativity : Read from test cases. Various associativity from direct-mapped to fully associative .
d. Replaced Algorithm : Read from test cases. Will be FIFO or LRU.
e. 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 (word).

The 2nd line specifies the block size (word).

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 word address of memory accesses executed from some benchmark program.

trace1.txt

Sample Input

```
16      // cache size (word)
1       // block size (word)
0       // associativity
0       // FIFO=0 , LRU=1
3       // word address
180
43
2
191
88
190
14
181
44
186
253
```

Output of your cache simulator

Take trace1 for example, output to file named trace1.out, and output the following information (you need to output which tag is the victim for each request, if there is not any victim then output -1, at last, you need to output the miss rate on the last line).

Sample Output

```
-1
-1
-1
-1
-1
-1
-1
11
-1
-1
-1
-1
Miss rate = 1.000000
```

Score:

Your score is divided into two parts:

a. Functional Simulation (80%): TA will give you score based on the number of correct results. There are 4 test data, one of them is 20 points

b. Report (20%): Follow the report requirement below.

Homework Requirements

a. **Language/Platform**

Please implement this Homework in C or C++ language.

b. **File name**

You need to name your source code as **cache.cpp** or **cache.c**

c. **Command Line Format**

```
./cache <trace.txt> <trace.out>
```

Example: ./cache ./trace1.txt ./trace1.out

d. **Note**

Note that you should write your own make file.

Important

When you upload your file, please make sure you have satisfied all the homework requirements, including the **File hierarchy**, **Requirement file** and **Report format**.

If you have any questions, please contact us.