SCE212 Project 3: Cache Design

Due 11:59pm, December 11th

1. Overview

This project is intended to help you understand the principle of caching by implementing a data cache. The main part of this project is to simulate a data cache from an input trace file. The cache should be configurable to adjust capacity, associativity, and block size with command-line options. You must support extra options to print out the content of the cache.

2. Simulation Details

In this project, you should design a trace-based cache simulator. An input trace file contains a sequence of read or write operations with addresses. For each step, an entry of the trace is fed into the cache simulator, to simulate the internal operation of the cache. The write policy of the cache must be *write-allocate* and *write-back*. The replacement policy must be the perfect LRU.

2.1 Cache Parameters

The capacity, associativity, and block size of the cache must be configurable. The parameters are specified with '`-c`' option: `-c <capacity>:<assoc>:<blocksize>`
Configurable parameters:

- Capacity: 4B (one word) – 8KB
- Associativity: 1 – 16 way
- Block size: 4B – 32B

When you specify both capacity and block size, you should specify the number with byte granularity and power of two.
e.g.,) Capacity 4KB, Associativity 4-way, Block size 32B → **4096:4:32**

3. Simulator Options and Output

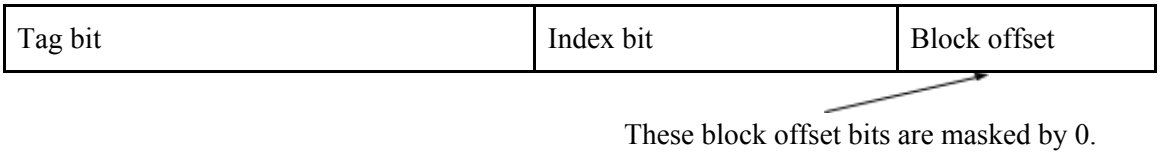
3.1 Options

```
$ ./sce212cache -c cap:assoc:block_size [-x] input_trace
```

- `-c` : cache configuration
- `-x` : dump the cache content at the end of simulation

Cache content is not data content but the address which is aligned with block size.
e.g.,) Block size 16B

When the address 0x10001234 is stored in the cache, the content of the cache entry is 0x10001230.
The last 4bits (block size bit) are masked by 0.



We provide skeleton code for displaying output format within `main.c`. You are free to change the parameters and corresponding parts of the code as long as it can print out the same format.

The example of output format with options `-c` and `-x` is attached as below.

```
Cache Configuration:
-------------------------------------
Capacity: 256B
Associativity: 4way
Block Size: 8B

Cache Stat:
-------------------------------------
Total reads: 0
Total writes: 0
Write-backs: 0
Read hits: 0
Write hits: 0
Read misses: 0
Write misses: 0

Cache Content:
-------------------------------------
          WAY[0]      WAY[1]      WAY[2]      WAY[3]
SET[0]:   0x00000000  0x00000000  0x00000000  0x00000000
SET[1]:   0x00000000  0x00000000  0x00000000  0x00000000
SET[2]:   0x00000000  0x00000000  0x00000000  0x00000000
SET[3]:   0x00000000  0x00000000  0x00000000  0x00000000
SET[4]:   0x00000000  0x00000000  0x00000000  0x00000000
SET[5]:   0x00000000  0x00000000  0x00000000  0x00000000
SET[6]:   0x00000000  0x00000000  0x00000000  0x00000000
SET[7]:   0x00000000  0x00000000  0x00000000  0x00000000
```

3.2 Input

The input trace contains a sequence of read or write operations with addresses as we mentioned.

e.g.,)

```
R 0x10000000
W 0x10003231
R 0x12341245
R 0x10003231
W 0x10023414
…
```

We provide several input traces which are real-world traces.

3.3 Output

Your output must contain the following statistics:

- the number of total reads
- the number of total writes
- the number of write-backs
- the number of read hits
- the number of write hits
- the number of read misses
- the number of write misses

The order of output results is Cache Configuration (if `-c` option is on) → Cache Statistics → Cache Content (if `-x` option is on).

4. Download Project3 Repository

Like earlier projects, you can clone/download the sce212-project3 repo into your local machine to work on the project. Then you are ready to start the project. Following instruction is identical to the earlier projects.

(1) Go to the following page: http://github.com/csl-ajou/sce212-project3. The page is Project3 repository.
(2) Change directory to the location you want to clone your project and clone!
```
$ git clone http://github.com/csl-ajou/sce212-project3.git
```
(3) You will get the cloned repo in your machine.

Be sure to read the `README.md` file for some useful information. It includes the explanation of each file and which files you are allowed to modify for this project.

5. Grading Policy

Grades will be given based on the 4 examples provided for this project provided in the `sample_input` directory. Your simulator should print the exact same output as the files in the `sample_output` directory.

We will be automating the grading procedure by seeing if there are any differences between the files in the `sample_output` directory and the result of your simulator executions. Please make sure that your outputs are identical to the files in the `sample_output` directory.

You are encouraged to use the `diff` command to compare your outputs to the provided outputs.

```
$ ./sce212cache -c 1024:8:8 sample_input/simple > my_output
$ diff -Naur my_output sample_output/simple
```

If there are any differences (including whitespaces) the diff program will print the different lines. If there are no differences, nothing will be printed. Furthermore, we have provided a simple checking mechanism in the `Makefile`. Executing the following command will automate the checking procedure.

```
$ make test
```

There are 4 code segments to be graded and you will be granted 20 points for each correct binary code and **being "Correct" means that every digit and location is the same** to the given output of the example. If a digit is not the same, you will receive **0 score** for the example.

6. Submission

Before submitting your code, it is highly recommended to complete the work on the Linux server we provided, the Linux virtual machine, or Windows Subsystem for Linux as previously. In this project, you should only submit the `main.c` file, thus you just need to upload the `main.c` file to https://sslab.ajou.ac.kr/pasubmit.

After then, you should check whether your code works well or not. Of course, you can test your code on `PAsubmit` as many as you want. Please make sure that you can see the same result on the submission site as well.

7. Updates/Announcements

If there are any updates to the project, including additional tools/inputs/outputs, or changes, we will post a notice on the Ajou BB, and will send you an e-mail using the Ajou BB system. **Frequently check your AjouBB linked e-mail account or the AjouBB notice board for updates.**

8. Misc

We will accept your late submissions up to 5 days without delay tokens. For each late day, your score will lose 10% so that up to 50% for 5 days. Please do not give up!

Be aware of plagiarism! Although it is encouraged to discuss with others and refer to extra materials, copying other students or opening your code is strictly banned. The TAs will compare your source code with other's code. If you are caught, you will receive a penalty for plagiarism.

If you have any requests or questions regarding administrative issues (such as late submission due to an unfortunate accident, `PAsubmit` is not working, how many remain tokens) please send an e-mail to the TAs (tome01@ajou.ac.kr / jjw8967@ajou.ac.kr).