Sungkyunkwan Univ.
College of Software

Assignment 2: Cache Simulator

Due date: 12/25 24:00

Submission: iCampus (Report), Homework Server (Source Code)

In this project, you will implement a cache simulator. The simulator must be able to handle the following characteristics:

- Split Cache: I-cache and D-cache
- Cache Sizes: 1024, 2048, 4096, 8192, 16384 bytes
- Block Sizes: 16 bytes, 64 bytes
- Associativities: Direct Mapped, 2-way, 4-way, 8-way
- Replacement Policies: LRU(Least Recently Used)
- Write and Allocate Policy: Write Back and Write Allocate for D-cache
- Multi-level Cache: L1 (I-cache, D-cache), L2 (Unified cache)
- L2 cache configuration: 8-way, same block size as L1 cache, 16384bytes size
- L2 cache also uses the Write Back and Write Allocate policy.
- Multi-level Cache Policy: inclusive, exclusive

The cache simulator should keep track of cache misses for each combination of cache size, block size, associativity, and replacement policy.

The number of sets of the target cache system is determined based on the associativity, the block size and the total cache size. For example, if the associativity, the block size, and the total cache size are 2-way, 16 bytes, and 8KB, respectively, the number of sets is 256. (256 sets * 2 way * 16 bytes = 8KB)

The input to your program will be a sequence of addresses. For each address, you should simulate a read or write from the cache. Therefore, given an address you would first check to see if it is contained in the cache. If it isn't, you would increment the number of misses and update the cache using the replacement policies.

If the replacement evicts a dirty block, you would increment the number of memory writes. Assume the size of an address is 32 bits.

You are provided with two files of address traces (trace1.din and trace2.din). I also provide a short version of each trace for your test (trace1-short.din and trace2-short.din). You can find them at your home directory in Homework server. Please be sure to unzip it before you use them.

The addresses are represented in hexadecimal format. Each address is a byte address and follows its label which gives the access type of a reference. (0: read data, 1: write data, 2: instruction fetch)

A sample address trace looks like the following:

```
2 403664

1 7ffd8ce0

2 404bc8
```

Two-level cache architecture adopts an additional CPU cache, which is slower but larger than the first-level cache, to minimize the number of external memory accesses. On each memory request of CPU, the level 1 (L1) cache is first examined; For L1 cache miss, the next level cache (level 2, L2) is checked before external memory is accessed.

On inclusive policy, the L2 cache must include all the data in the L1 cache. ($\mathrm{L1} \subset \mathrm{L2}$)

- L1 hit: update the LRU order of L1 cache blocks.
- L1 miss & L2 hit: add the requested block in L1 cache, update the LRU order of L2 cache blocks.
- L1 miss & L2 miss: add the requested block in both L1 cache and L2 cache.
- Block eviction from L1 cache: move the victim block to L2 cache (do not update the LRU order of L2 cache).
- When a block is evicted from L2 cache, the same block in L1 cache must also be evicted.

On exclusive policy, a cached block should be in either L1 cache or L2 cache, never in both. ($\mathrm{L1} \cap \mathrm{L2} = \emptyset$)

- L1 hit: update the LRU order of L1 cache blocks.
- L1 miss & L2 hit: add the requested block in L1 cache, remove the block in L2 cache.
- L1 miss & L2 miss: add the requested block only in L1 cache.
- Block eviction from L1 cache: move the victim block to L2 cache, and update the LRU order of L2 cache

The simulator should output the following two tables for L1 I-cache and D-cache, respectively. Each entry in the table contains the miss ratio for each combination of size and associativity. The tables should contain the results for different block sizes. Output the tables in the following format:

Cache Miss Ratio (block size = 16B)					
LRU/16	1024	2048	4096	8192	16384
Direct					
2-way					
4-way					
8-way					

Cache Miss Ratio (block size = 64B)					
LRU/64	1024	2048	4096	8192	16384
Direct					
2-way					
4-way					
8-way					

The simulator also should output the following two tables for inclusive and exclusive L2 caches, respectively.

L2 Miss Ratio (block size = 16B)					
LRU/16	1024	2048	4096	8192	16384
Direct					
2-way					
4-way					
8-way					

L2 Miss Ratio (block size = 64B)					
LRU/64	1024	2048	4096	8192	16384
Direct					
2-way					
4-way					
8-way					

For D-cache, the number of memory block writes should be reported with the following tables. Each entry in the table contains the number of memory block writes generated by block replacements.

Number of Memory Block Writes (Inclusive)					
LRU/16	1024	2048	4096	8192	16384
Direct					
2-way					
4-way					
8-way					

Number of Memory Block Writes (Inclusive)					
LRU/64	1024	2048	4096	8192	16384
Direct					
2-way					
4-way					
8-way					

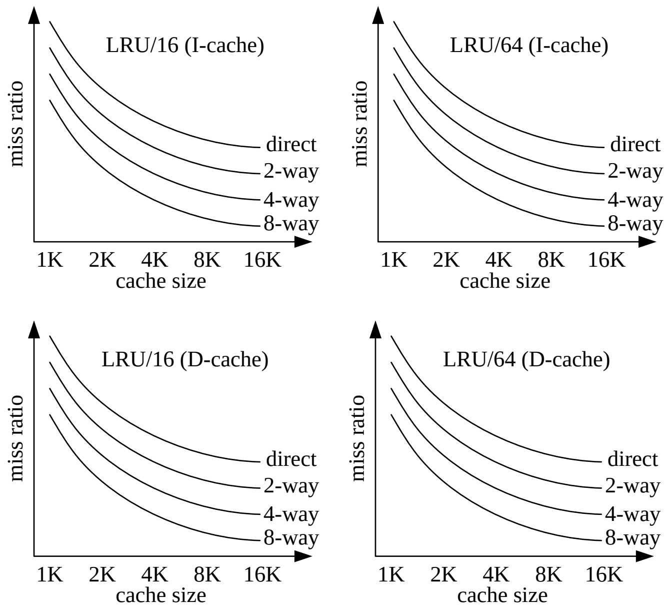
Number of Memory Block Writes (exclusive)					
LRU/16	1024	2048	4096	8192	16384
Direct					
2-way					
4-way					
8-way					

Number of Memory Block Writes (exclusive)					
LRU/64	1024	2048	4096	8192	16384
Direct					
2-way					
4-way					
8-way					

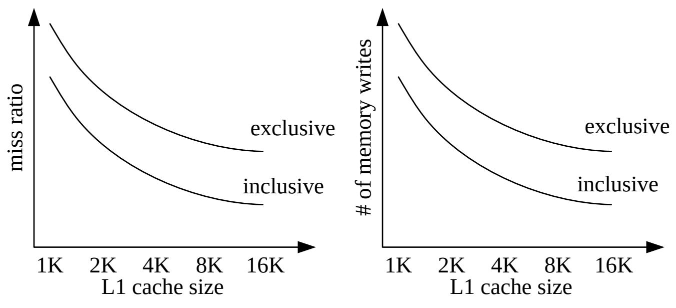
Your simulator should be one executable file which can generate all the output results for various combinations of cache size, block size, associativity, and inclusive policy. (Don't make multiple programs each of which can run only one configuration)

What to Submit

1. The report summarizing the results for the provided two test files. (The report should be a pdf file.)
   - Rename the report into (***yourstudentid.pdf)*** and upload it at iCampus.
   - Explain on how you implemented the simulator. (Descriptions on main data structures and algorithms)
   - Include the tables that were output from the simulator.
   - Include four graphs which contain miss rate on the y-axis and cache size on the x-axis for each trace. Plot four different results on each graph, one for each associativity.



   - Include two graphs which compare the L2 miss rates and the number of memory writes of the inclusive and exclusive policies at D-cache. Argue on the different performances of inclusive and exclusive policies.
     - The graphs should show the L2 miss rates or the number of memory writes on the y-axis and the L1 cache size on the x-axis for each trace.
     - For L1 cache configuration, the block size is 64 bytes and the associativity is 2-way.



2. The source code of cache simulator. The source code must include all project files for building your program (makefile, source files, header files).
   - You must use our homework server. To connect to Homework server, refer to the Assignment 1.
   - All the commands on the homework server will be written in a log file, which will be used to hunt out cheating.
   - Make an archive file(yourstudentid.zip) of your source code and leave it on your home folder of homework server. You don't need to submit the code to iCampus.
   - I will regard the modified time(ctime) of yourstudentid.zip as the submit time.
   - The simulator should be written in C or C++.
   - For all the program blocks (functions, loops, if-then-else, etc) and the variables that you implemented, you should add proper comments. If sufficient comments are not provided, I will not give any points.
   - Don't upload the final source code after you edit it at your PC. You have to use the Linux editor (vi) in Homework Server. We will check the log file.

◈ **Notice**

- **If I would find two or more source codes which are much alike, all the corresponding students will fail this course. At the Homework server, we can monitor all your operations such shell commands and logging IP. So, we can produce an evidence of your cheating.**
- **If your program satisfies only a subset of the specified requirements, you can get a partial point. To get a partial point, specify the list of complete functions of your program.**
- **One day delay has -20 points penalty.**
- **If you have any questions, send email to or visit TA (**pdaejun@gmail.com**, room# 85465).**