Programming Assignment #4: Cache Simulator

Prof. Jae W. Lee (jaewlee@snu.ac.kr)
Department of Computer Science and Engineering
Seoul National University

By your friendly TAs (snu-arc-uarch-ta@googlegroups.com)

Contents

Goal of this project

- **You need to implement some features in a 2-level cache simulator**
  - Cache entry structures and function prototypes in C are given
  - Implement a **2-level cache** and observe memory access patterns in a multi-level cache
  - Examine memory access patterns in multiple **eviction policies**

Environment setup

- This assignment should be written in C

- You can get skeleton code and test bench from git repo
  - `git clone https://github.com/SNU-ARC/2023_fall_comarch_PA4`

Instructions

- **There are 2 skeleton code files (.c), 1 header file (.h), 2 executables (csim-ref, test-csim), 8 traces (.trace) for evaluation (+ 1 hidden trace)**

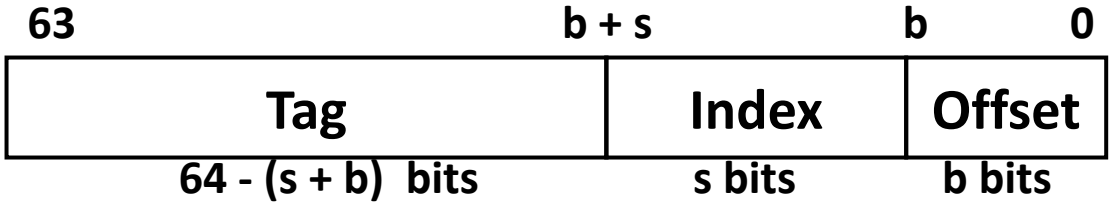
- **You can only modify the csim.c file, and you will have to submit it.**
  - You should not modify or submit other files

Instructions: Skeleton Code

- **cachelab.h**
  - header file containing data structures and function declarations used in this assignment
- **cachelab.c**
  - contains implementations for helper functions
  - Please read carefully - **stats counted with Verbose() function is used for evaluation**
- **csim.c (submission required)**
  - Implementation of 2-level inclusive cache simulator
  - Initialization of data structures and argument parser is already there - do not modify them
  - Should support LRU and FIFO (First-In First-Out) eviction policies
  - Prints # of cache hit/misses/evictions for both levels and **writes to L2 cache/memory**

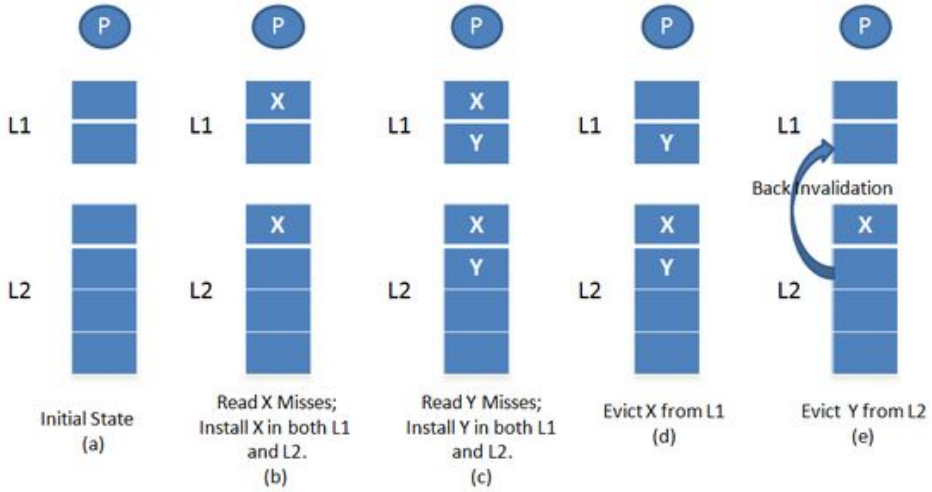
Instructions: Skeleton Code

- **Cache spec**
  - Adopts inclusive cache policy with write-back write policy
  - (# of set index bits of L2) = 2 * (# of set index bits of L1)
  - (# of blocks for each set of L2) = 4 * (# of blocks for each set of L1)



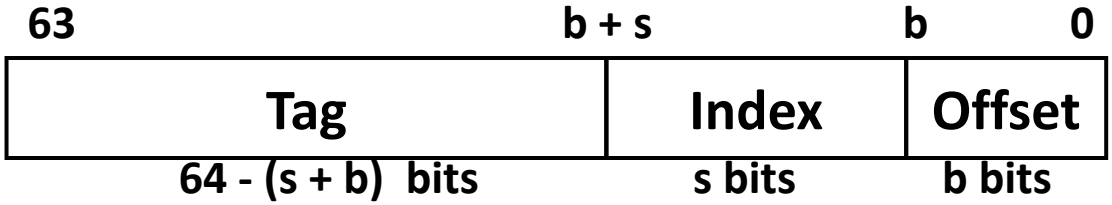
Instructions: Skeleton Code

- **Inclusive cache**
  - Cached blocks in L1 cache should be a subset of those in L2 cache
  - If a block is evicted from L2 cache, the same block in L1, if any, should also be invalidated (L1 eviction occurs)



Instructions: Arguments

- **Usage: ./csim [-hv] -s <num> -E <num> -b <num> -p <num> -t <file>**
  - -h : Print help message
  - -v : Verbose mode. Prints actions taken in cache for each line in trace.
  - -s <num> : Number of set index bits in L1
  - -E <num> : Number of blocks per set in L1
  - -b <num> : Number of block offset bits
  - -p <num> : Cache policy. 0 for LRU, 1 for FIFO
  - -t <file> : Path of input trace file



Helper Program

- **Use provided csim-ref to compare your code with answer**
  - It is a binary compiled with golden (answer) code
  - Command is same as **csim** - just change **./csim** to **./csim-ref**
  - Result of your code should be identical to that of **csim-ref** for full marks

```
(base) lks@kslim:~/2023_2_comparch/2023_fall_comarch_PA4$ ./csim-ref -v -s 1 -E 1 -b 1 -p 1 -t traces/yi2.trace
L 0x0,1 L1 Miss L2 Miss
L 0x1,1 L1 Hit
L 0x2,1 L1 Miss L2 Miss
L 0x3,1 L1 Hit
S 0x4,1 L1 Miss L1 Eviction L2 Miss
L 0x5,1 L1 Hit
S 0x6,1 L1 Miss L1 Eviction L2 Miss
L 0x7,1 L1 Hit
S 0x8,1 L1 Miss L1 Eviction L2 Write L2 Miss L2 Eviction
L 0x9,1 L1 Hit
S 0xa,1 L1 Miss L1 Eviction L2 Write L2 Miss L2 Eviction
L 0xb,1 L1 Hit
S 0xc,1 L1 Miss L1 Eviction L2 Write L2 Miss L2 Eviction
L 0xd,1 L1 Hit
S 0xe,1 L1 Miss L1 Eviction L2 Write L2 Miss L2 Eviction
M 0xf,1 L1 Hit L1 Hit
level          hits           misses         evictions
L1             9              8              6
L2             0              8              4
writes to L2: 4, writes to memory: 0
```

Helper Program

- **Use provided test-csim to check your score for open test cases**
  - Usage: ./test-csim
  - It compares results of your code with golden code for test cases used for evaluation
  - Please note that there are **hidden test cases** - getting full marks with **test-csim** does not guarantee full marks in this assignment

```
(base) lks@kslim:~/2023_2_comparch/2023_fall_comarch_PA4$ ./test-csim
                                                                  Reference simulator                                                             Your
simulator
Points (s,E,b)     L1 Hits   L1 Misses   L1 Evicts     L2 Hits   L2 Misses   L2 Evicts   L2 Writes  Mem Writes     L1 Hits   L1 Misses   L1 Evicts     L2 Hits   L2 Misses   L2 Evicts   L2 Writes  M
em Writes
    18 (1,4,1)           9           3           5           5           4           0           0           0           0           0           0           0           0           0           0
        0  traces/belady.trace
    30 (3,2,1)           3           3           0           3           0           0           0           0           0           0           0           0           0           0           0
        0  traces/l1.trace
     6 (2,1,1)          15           0          14          13           2          11           7           6           0           0           0           0           0           0           0
        0  traces/l2.trace
    18 (2,2,4)           3           2           1           3           0           0           1           0           0           0           0           0           0           0           0
        0  traces/dave.trace
    12 (4,2,4)           5           4           2           4           1           0           1           0           0           0           0           0           0           0           0
        0  traces/yi.trace
    12 (1,1,1)           8           9           6           8           0           4           4           0           0           0           0           0           0           0           0
        0  traces/yi2.trace
     0 (5,1,5)       21775      265189       21743       18508        3267       18288       17393       16223           0           0           0           0           0           0           0
        0  traces/long.trace
     0 (5,1,5)       21775      265189       21743        8213       13562        7957       17393        6097           0           0           0           0           0           0           0
        0  traces/long.trace
     0 (2,1,3)          71         167          67          30          41          10          33           4           0           0           0           0           0           0           0
        0  traces/trans.trace
     0 (2,2,3)          52         186          44          24          28           2          28           1           0           0           0           0           0           0           0
        0  traces/trans.trace
    96

TEST_CSIM_RESULTS=96
```

Grading Policy

- **Test bench Score : 90 %**
  - 10 test cases provided along with skeleton code (60%)
  - 6 hidden test cases to test accuracy of your code (30%)
- **Write-up** : **10%**
- **For late submission:**
  - A deduction of 10% p per 24 hours
  - After next 120 hours, submission will not be accepted

Submission

- **Write-up**
  - Briefly describe your implementation.
  - Filename: [student_id].txt (example: 2023-12345.txt)
  - Please use ‘UTF-8’ encoding if possible
  - **Please** submit it in **txt** format. Other formats are not accepted.
- **Compress your source code and write-up into a single zip file**
  - Compress **csim.c** and your report
  - Filename should be [student_id].zip (example: 2023-12345.zip).
  - **Please** submit it in **ZIP** format. Other formats are not accepted.
  - Don’t mind numbers automatically attached to your file name for multiple submission in eTL - we will grade the most recent one
- **Submission deadline: by 23:59 on November 22, 2023 (Wed)**