

Spring 2000 CDA 4101 Programming Assignment 2

Due date: April 19, 2000.

Implementing an Instruction/Data Cache Simulator

Your assignment is to write a program that simulates the behavior of an instruction and a data cache. The program will read a trace of references from standard input and produce statistics about the trace to standard output. The trace of references has the following format:

```
<streamtype>:<accesstype>:<size>:<hexaddress>
```

where <streamtype> can be the characters:

- I - indicates a reference to an instruction
- D - indicates a reference to data

<accesstype> can be the characters:

- R - indicates a read access
- W - indicates a write access

<size> is the size of the reference in bytes, and <hexaddress> is the starting byte address of the reference expressed as a hexadecimal number. You may assume that the format of the trace of references is valid. You should check that the size of each instruction reference is 4 bytes and the size of each data reference is 1, 2, 4, or 8 bytes. You should also check if each reference's address is properly aligned. The example trace below contains the addresses that correspond to the example references described on page 547 and 548 of your text. **Note the addresses here are hexadecimal byte addresses rather than decimal word addresses as used in the text.** The file can be downloaded from the class webpage (trace.dat).

```
I:R:4:58
I:R:4:68
I:R:4:58
I:R:4:68
I:R:4:40
I:R:4:c
I:R:4:40
I:R:4:48
```

Before processing the trace file, the cache simulator should first determine the characteristics of the instruction and data caches. This is accomplished by reading a configuration file that specifies the number of cache sets, the number of lines in each set (associativity level), and the size of a cache line in bytes for both of the caches. For the data cache you will also have to specify a write policy. This information should be read from a file named "trace.config" in the current directory. The format of the configuration file is shown below and this file is available on the class webpage (trace.config). (Note that the characters after the colons are the portion that can change). You may assume that the format of the file is valid.

```
Number of instruction sets: 8
Instruction set size: 1
Instruction line size: 4
Number of data sets: 4
Data set size: 1
Data line size: 8
Write through/no write allocate: y
```

Your simulator should support a maximum number of sets of 8K and a maximum associativity level of 8. The minimum line size is 4 bytes for the instruction cache, the minimum line size for the data cache is 8 bytes, and the number of sets and line size are powers of 2. Your cache simulator should use an LRU replacement algorithm. For data writes, it should employ a write-through and no write-allocate policy if the answer in the configuration file to that question is 'y'. Otherwise, it should use a write-back and write-allocate policy, i.e., write-back and write-allocate is indicated by putting 'n' in the write-through/no write allocate line of the configuration file.

The cache simulator output should first print information about the cache configuration used. Next, it should print information for each reference. This information is the stream type (inst or data), the access type (read or write), the address, tag, index, offset, result (hit or miss), and the number of lines sent to/from memory by the reference (0, 1, or 2). **Note that the address and the tag should be printed in hex. The index and offset should be printed in decimal.** This information will help you debug problems with your simulator. Though it is not required, you may also wish to write a function to dump the state of the caches to help debug problems.

After the last reference, the cache simulator should output the following summary information:

- a. the configuration of the two caches
- b. the number of hits for each cache
- c. the number of misses for each cache
- d. total accesses for each cache
- e. hit ratio for each cache

- f. miss ratio for each cache
- g. the number of cache accesses that were reads
- h. the number of cache accesses that were writes
- i. the fraction of cache accesses that were reads
- j. the number of cache accesses for instructions
- k. the number of cache accesses for data
- l. the fraction of cache accesses that were instructions
- m. the total amount of data moved to/from memory in bytes (i.e. the number of lines moved times the size of the line in bytes)
- n. the total number of bytes requested by the processor (i.e. sum of the bytes requested by the references in trace file)
- o. the ratio of m to n

The output below is the result of the simulator after processing only the references in the trace file and configuration given above.

You should try to make your output match this as closely as possible to facilitate grading of the assignment.

Cache Configurations

Instruction cache contains:

8 1-way instruction set associative entries
of line size 4 bytes

Data cache contains:

4 1-way data set associative entries
of line size 8 bytes
with a no write-allocate and write-through policy.

Results for Each Reference

Type	Access	Address	Tag	Index	Offset	Result	lines to/from mem.
inst	read	58	2	6	0	miss	1
inst	read	68	3	2	0	miss	1
inst	read	58	2	6	0	hit	0
inst	read	68	3	2	0	hit	0
inst	read	40	2	0	0	miss	1
inst	read	c	0	3	0	miss	1

```
inst  read    40    2    0    0  hit    0
inst  read    48    2    2    0  miss   1
```

Simulation Statistics

Instruction Cache Results

```
-----
Total inst hits      : 3
Total inst misses    : 5
Total inst accesses  : 8
Inst hit ratio       : 0.375000
Inst miss ratio      : 0.625000
```

Data Cache Results

```
-----
Total data hits      : 0
Total data misses    : 0
Total data accesses  : 0
```

Read vs. Writes

```
-----
Total reads          : 8
Total writes         : 0
Ratio of reads       : 1.000000
```

Inst vs. Data Refs

```
-----
Total inst refs      : 8
Total data refs      : 0
Ratio of insts       : 1.000000
```

Main Memory Accesses

```
-----
Total main memory refs: 20 bytes
Total CPU memory refs : 32 bytes
Total main mem ratio  : 0.625000
```

You should again comment your program so that others (e.g. the TA) can understand it. E-mail your files to karwande@cs.fsu.edu before 5 pm on the due date. Make sure that your program properly compiles and executes on xi or quake depending on graduate or undergraduate status.