Comp Sci 546 Computer Architecture & Organization
Project 1—Cache Memory

Objective:

To simulate reading and writing to a custom-sized direct-mapped cache, involving a custom-sized main memory.

Inputs:

- total size of accessible main memory (in words)
- total size of the cache (in words)
- block size (words/block)
- main memory address to read from/write to
- contents of the address for writing to the cache

Outputs:

- corresponding cache tag, block, and word for a main memory address
- contents of the address resulting from reading/writing to the cache
- message indicating a miss (if any) to the cache
- Extra Credit: Presented all addresses in both binary & hexadecimal formats

Specification:

The program simulates reading from and writing to a cache based on choosing from a menu of choices, where each choice calls the appropriate procedure, where the choices are:

a. Enter parameters
b. Read cache
c. Write to cache
d. Quit program

Upon entering the parameters, the main memory and cache are to be dynamically allocated based on their respective total sizes. The main memory is constructed as a dynamic 1-dimensional array of integers (words). The cache is constructed as a structure (struct) containing two fields: (i) tag, declared as an integer; (ii) block, declared as a dynamic array of integers (words). Each word i of main memory is initialized with the value (size_of_memory – i). Reading/writing from/to a new block in the cache results in dynamically allocating the block based on the block size. Upon quitting the program, all dynamically allocated memory should be freed.

Submission Product

You must execute the program to the instructor's satisfaction. A softcopy of the source code must be submitted to the instructors flash drive. Be sure to name your source code project1_yourlastname (all lowercase letters). Any deviation from the format for submission will result in an automatic 10% deduction. A printed hardcopy must be provided directly to the instructor.

You can program in any major language, use any editor and/or compiler, but make sure your code compiles and executes in the Pierce Computer Lab otherwise you will receive 0 points for compilation and execution.

Sample Run

Direct Mapping of Main Memory to Cache Memory

```
-------------------------------------
1) Set parameters
2) Read cache
3) Write to cache
4) Exit
Enter selection: 1
Enter main memory size (words): 65536
Enter cache size (words): 1024
Enter block size (words/block): 16
Main memory to Cache memory mapping:
-------------------------------------
1) Set parameters
2) Read cache
3) Write to cache
4) Exit
Enter selection: 3
Enter main memory address to write to: 65535
Enter value to write: 14
Write miss!
Word 15 of block 63 with tag 63 contains value 14
Main memory to Cache memory mapping:
-------------------------------------
1) Set parameters
2) Read cache
3) Write to cache
4) Exit
Enter selection: 2
Enter main memory address to read from: 65535
Word 15 of block 63 with tag 63 contains value 14
Main memory to Cache memory mapping:
-------------------------------------
1) Set parameters
2) Read cache
3) Write to cache
4) Exit
Enter selection: 3
Enter main memory address to write to: 65534
Enter value to write: 512
Word 14 of block 63 with tag 63 contains value 512
Main memory to Cache memory mapping:
-------------------------------------
1) Set parameters
2) Read cache
3) Write to cache
4) Exit
Enter selection: 2
Enter main memory address to read from: 1023
Read miss!
Word 15 of block 63 with tag 0 contains value 64513
Main memory to Cache memory mapping:
-------------------------------------
1) Set parameters
2) Read cache
3) Write to cache
4) Exit
Enter selection: 4
%
```

Parameters for the entire session

The remainder of the values entered in the current session is limited by the parameters set in the initial run above. To reset the parameters, exit the program and reenter, starting a new session.