CSE 325 **Spring 2020**

Computer Project #8

Assignment Overview

This assignment focuses on the interactions between primary storage and the data cache. You will design and implement additional functionality to extend the previous project, as described below.

It is worth 40 points (4% of course grade) and must be completed no later than 11:59 PM on Thursday, 3/19.

Assignment Deliverables

The deliverables for this assignment are the following files:

> `proj08.makefile` – the makefile which produces `proj08`
> `proj08.student.c` – the source code file for your solution

Be sure to use the specified file names and to submit your files for grading via the CSE Handin system before the project deadline.

Assignment Specifications

The program will simulate the steps required to manage primary storage and the data cache. Primary storage consists of 1,048,576 bytes of RAM; physical addresses are 20 bits in length. The data cache is a direct-mapped, write-back cache which contains 8 lines. Each cache line contains 16 bytes of data, as well as the control information (valid bit, modified bit, and tag bits).

1. Your program will accept command-line arguments to specify the file which contains the memory references (the "-refs" option) and to control the display of debugging information (the "-debug" option).

2. The "-refs" option will be followed by the name of the file which contains zero or more memory references. Each line of the file will contain the following information:

a) physical address being referenced (five hexadecimal digits, with leading zeroes)
b) operation being performed (one character; 'R' for read and 'W' for write)
c) if the operation is a write, the value being written (four bytes, where each byte is given as two hexadecimal digits, with leading zeroes)

Items in the line will be separated by exactly one space, and the line will terminate with a newline. For example:

```
00bd8 R
20ac4 W 02 04 56 78
```

3. For each memory reference in the file, your program will display one line with the following information:

a) physical address being referenced (five hexadecimal digits, with leading zeroes)
b) operation being performed (one character; 'R' for read and 'W' for write)
c) tag bits (four hexadecimal digits, with leading zeroes)
d) cache line accessed (one hexadecimal digit)
e) byte offset (one hexadecimal digit)
f) result of access (one character; 'H' for hit and 'M' for miss)
g) data value (four bytes, where each byte is given as two hexadecimal digits, with leading zeroes

Items in the line will be separated by exactly one space, and the line will terminate with a newline. For example:

```
00bd8 R 0017 5 8 H 44 55 66 77
20ac4 W 0415 4 4 M 02 04 56 78
```

4. After the simulation is completed, your program will display the contents of the data cache. The display will contain one line for each data cache entry:

a) index of the data cache entry (one hexadecimal digit)
b) V bit (one character; ‘0’ for not valid, ‘1’ for valid)
c) M bit (one character; ‘0’ for not modified, ‘1’ for modified)
d) tag bits (four hexadecimal digits, with leading zeroes)
e) data block (16 bytes of data, where each byte is given as two hexadecimal digits, with leading zeroes)

Items within a line will be separated by exactly one space, and each line will terminate with a newline. The data cache display will begin and end with a blank line (for readability), and will include appropriate column headers.

5. After the simulation is completed, your program will display the current contents of the subset of RAM which begins at address 20000 and contains 128 bytes. Each line of that display will include an address (five hexadecimal digits) and the 16 bytes of data starting at that address (two hexadecimal digits, with leading zeroes).

Items in the line will be separated by exactly one space, and the line will terminate with a newline. The display will begin and end with a blank line (for readability). For example, the first two lines of that display might appear as:

```
20000 11 22 33 44 12 34 56 78 b4 82 59 7a b3 74 01 ef
20010 9d e3 bf a0 31 00 00 86 b0 16 21 d4 55 66 77 88
```

6. If the "-debug" option has been selected, your program will also display the following:

a) the contents of the data cache at the start of the simulation
b) the contents of the data cache after each memory reference is processed

7. The program will include appropriate error-handling.

Assignment Notes

1. As stated above, your source code file will be named "proj08.student.c"; that source code file may contain C or C++ statements.

2. You must use "g++" to translate your source code file in the CSE Linux environment.

3. Valid executions of the program might appear as follows:

```
proj08 –debug –refs test1
proj08 –refs test2 -debug
proj08 –refs test3
```

4. Your program must create a data structure representing the data cache and set all of the entries to zero at the start of the simulation. Your program will update the data cache in response to the memory references.

5. Your program must create a data structure representing the RAM and set all of the entries to zero at the start of the simulation. Your program will update the RAM in response to the memory references.