Memory & Caches III
CSE 351 Winter 2024

Instructor:
Justin Hsia

Teaching Assistants:
Adithi Raghavan
Aman Mohammed
Connie Chen
Eyoel Gebre
Jiawei Huang
Malak Zaki
Naama Amiel
Nathan Khuat
Nikolas McNamee
Pedro Amarante
Will Robertson

http://xkcd.com/908/
Relevant Course Information

- HW15 due tonight, HW16 due Friday, HW17 due Wednesday (2/21)
- Lab 3 due Friday (2/16)
- Lab 4 released Friday, due two weeks later (3/1)
  - Can do Part 1 after today; will need Lesson 18 to do Part 2
- No lecture on Monday for President’s Day!
Caches III
Lesson Summary (1/2)

❖ **Associativity** gives us flexibility in where to place blocks in the cache
  - Group $E$ slots into **sets**, means there are $E$ ways to place block within each set
    - Direct-mapped is $E = 1$, **fully associative** is $E = \# \text{ of slots in cache} \ (i.e., S = 1)$
    - Helps avoid conflicts in each set at the expense of slightly longer and more complex searching & placing in the cache
    - By default, we will replace the **least-recently used** block in a set
  - Index → Set, $S = (C/K)/E$, still $s = \log_2(S)$
Lesson Summary (2/2)

❖ Management bits

- Information needed for proper management of the cache & its data, but not counted in the cache size
- **Valid bit** for validity of data
- **Tag bits** for identifying which block

\[
E = \text{blocks (or lines) per set} \\
S = 2^s \quad \text{(sets)} \\
K = \text{bytes per block} \\
C = K \times E \times S \quad \text{data bytes (doesn't include V or Tag)}
\]
Lesson Q&A

❖ Learning Objectives:
  ▪ Determine how memory addresses and data interact with the cache (i.e., cache lookups, data movement).
  ▪ Analyze how changes to cache parameters and policies affect performance metrics such as AMAT.

❖ What lingering questions do you have from the lesson?
  ▪ Chat with your neighbors about the lesson for a few minutes to come up with questions
Caches III – Practice
Polling Questions

❖ We have a cache of size 2 KiB with block size of 128 B. If our cache has 2 sets, what is its associativity?
   A. 2
   B. 4
   C. 8
   D. 16

❖ If addresses are 16 bits wide, how wide is the Tag field?
Homework Setup

Addresses are 13 bits wide, and the cache is two-way set-associative ($E=2$) with 4-byte block size ($K=4$) and eight sets ($S=8$).

<table>
<thead>
<tr>
<th>Set Index</th>
<th>Tag</th>
<th>Valid</th>
<th>Byte 0</th>
<th>Byte 1</th>
<th>Byte 2</th>
<th>Byte 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>21</td>
<td>1</td>
<td>68</td>
<td>65</td>
<td>79</td>
<td>20</td>
</tr>
<tr>
<td>1</td>
<td>43</td>
<td>1</td>
<td>74</td>
<td>88</td>
<td>65</td>
<td>72</td>
</tr>
<tr>
<td>2</td>
<td>7F</td>
<td>1</td>
<td>65</td>
<td>20</td>
<td>69</td>
<td>74</td>
</tr>
<tr>
<td>3</td>
<td>29</td>
<td>0</td>
<td>27</td>
<td>73</td>
<td>20</td>
<td>73</td>
</tr>
<tr>
<td>4</td>
<td>3D</td>
<td>1</td>
<td>61</td>
<td>6D</td>
<td>6D</td>
<td>79</td>
</tr>
<tr>
<td>5</td>
<td>2C</td>
<td>0</td>
<td>20</td>
<td>77</td>
<td>6F</td>
<td>6C</td>
</tr>
<tr>
<td>6</td>
<td>1A</td>
<td>1</td>
<td>66</td>
<td>64</td>
<td>61</td>
<td>77</td>
</tr>
<tr>
<td>7</td>
<td>2B</td>
<td>1</td>
<td>67</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set Index</th>
<th>Tag</th>
<th>Valid</th>
<th>Byte 0</th>
<th>Byte 1</th>
<th>Byte 2</th>
<th>Byte 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00</td>
<td>0</td>
<td>F3</td>
<td>29</td>
<td>84</td>
<td>55</td>
</tr>
<tr>
<td>1</td>
<td>21</td>
<td>1</td>
<td>63</td>
<td>F7</td>
<td>E3</td>
<td>D1</td>
</tr>
<tr>
<td>2</td>
<td>0B</td>
<td>1</td>
<td>22</td>
<td>AA</td>
<td>BB</td>
<td>CC</td>
</tr>
<tr>
<td>3</td>
<td>3B</td>
<td>0</td>
<td>2F</td>
<td>5C</td>
<td>93</td>
<td>4B</td>
</tr>
<tr>
<td>4</td>
<td>00</td>
<td>0</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>5</td>
<td>00</td>
<td>0</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>6</td>
<td>2F</td>
<td>1</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>F3</td>
</tr>
<tr>
<td>7</td>
<td>C1</td>
<td>0</td>
<td>49</td>
<td>AB</td>
<td>DF</td>
<td>CC</td>
</tr>
</tbody>
</table>

- What are the widths of the tag, index, and offset fields?
Homework Setup

- Addresses are 13 bits wide, and the cache is two-way set-associative \((E=2)\) with 4-byte block size \((K=4)\) and eight sets \((S=8)\).

<table>
<thead>
<tr>
<th>Set Index</th>
<th>Tag</th>
<th>Valid</th>
<th>Byte 0</th>
<th>Byte 1</th>
<th>Byte 2</th>
<th>Byte 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>21</td>
<td>1</td>
<td>68</td>
<td>65</td>
<td>79</td>
<td>20</td>
</tr>
<tr>
<td>1</td>
<td>43</td>
<td>1</td>
<td>74</td>
<td>68</td>
<td>65</td>
<td>72</td>
</tr>
<tr>
<td>2</td>
<td>7F</td>
<td>1</td>
<td>65</td>
<td>20</td>
<td>69</td>
<td>74</td>
</tr>
<tr>
<td>3</td>
<td>29</td>
<td>0</td>
<td>27</td>
<td>73</td>
<td>20</td>
<td>73</td>
</tr>
<tr>
<td>4</td>
<td>3D</td>
<td>1</td>
<td>61</td>
<td>6D</td>
<td>6D</td>
<td>79</td>
</tr>
<tr>
<td>5</td>
<td>2C</td>
<td>0</td>
<td>20</td>
<td>77</td>
<td>6F</td>
<td>6C</td>
</tr>
<tr>
<td>6</td>
<td>1A</td>
<td>1</td>
<td>66</td>
<td>64</td>
<td>61</td>
<td>77</td>
</tr>
<tr>
<td>7</td>
<td>2B</td>
<td>1</td>
<td>67</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Line 0</th>
<th>Line 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag</td>
<td>Valid</td>
</tr>
<tr>
<td>-----</td>
<td>-------</td>
</tr>
<tr>
<td>00</td>
<td>0</td>
</tr>
<tr>
<td>21</td>
<td>1</td>
</tr>
<tr>
<td>0B</td>
<td>1</td>
</tr>
<tr>
<td>3B</td>
<td>0</td>
</tr>
<tr>
<td>00</td>
<td>0</td>
</tr>
<tr>
<td>00</td>
<td>0</td>
</tr>
<tr>
<td>2F</td>
<td>1</td>
</tr>
<tr>
<td>C1</td>
<td>0</td>
</tr>
</tbody>
</table>

- What addresses will hit in Set 0?
Caches III – Context
Example Code Analysis Problem

Assuming the cache starts **cold** (all blocks invalid) and sum, i, and j are stored in registers, calculate the **miss rate**:

- \( m = 10 \) bits, \( C = 64 \) B, \( K = 8 \) B, \( E = 2 \)

```
#define SIZE 8
short ar[SIZE][SIZE], sum = 0; // &ar=0x200
for (int i = 0; i < SIZE; i++)
    for (int j = 0; j < SIZE; j++)
        sum += ar[j][i];
```
Cache Simulator

- [link](https://courses.cs.washington.edu/courses/cse351/cachesim/)
  - From course website: Simulators → Cache Simulator
  - Allows you to play around with the effects of cache parameters and policies
  - Lots of neat features like highlighting, hover text, ability to rewind and replay accesses, and copy-and-paste access patterns

- Ways to use:
  - Take advantage of “explain mode” and navigable history to test your own hypotheses and answer your own questions
  - Self-guided Cache Sim Demo posted along with Section 7
  - Will be used in HW18 – Lab 4 Preparation
Cache Simulator Demo

- [https://courses.cs.washington.edu/courses/cse351/cachesim/](https://courses.cs.washington.edu/courses/cse351/cachesim/)
  - From course website: Simulators → Cache Simulator
  - Allows you to play around with the effects of cache parameters and policies
  - Lots of neat features like highlighting, hover text, ability to rewind and replay accesses, and copy-and-paste access patterns

- Let’s simulate the example problem from the lesson:

```c
#define SIZE 8
short ar[SIZE][SIZE], sum = 0;  // &ar=0x200
for (int i = 0; i < SIZE; i++)
    for (int j = 0; j < SIZE; j++)
        sum += ar[j][i];
```
Group Work Time

- During this time, you are encouraged to work on the following:
  1. If desired, continue your discussion
  2. Work on the homework problems
  3. Work on the lab (if applicable)

- Resources:
  - You can revisit the lesson material
  - Work together in groups and help each other out
  - Course staff will circle around to provide support