Review Problem 37

- If you can speed up any level’s hit time by a factor of two, which is the best to speed up?

<table>
<thead>
<tr>
<th>Level</th>
<th>Hit Time</th>
<th>Hit Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>1 cycle</td>
<td>95%</td>
</tr>
<tr>
<td>L2</td>
<td>10 cycles</td>
<td>90%</td>
</tr>
<tr>
<td>Main Memory</td>
<td>50 cycles</td>
<td>99%</td>
</tr>
<tr>
<td>Disk</td>
<td>50,000 cycles</td>
<td>100%</td>
</tr>
</tbody>
</table>

\[
= 1 + 0.05 \times 10 + 0.01 \times 50 + 0.05 \times 1 \times 0.1 \times 50,000
\]

\[
= 1 + 0.5 + 0.25 + 2.5 \quad \text{Speed up}
\]

\[
= 4.25
\]
Finding A Block

Each location in the cache can contain a number of different memory locations. Cache 0 could hold 0, 4, 8, 12, ...

We add a tag to each cache entry to identify which address it currently contains. What must we store?

- Portion of address NOT used to indicate the cache line (MS 5 bits)
- Valid bit - 1: is data
  0: no data, all the rest of the tag is garbage
Cache Tag & Index

Assume $2^9$ byte direct mapped cache with 1 byte blocks

Cache Tag = 57
Cache Index = 03
Cache Access Example

Assume 4 byte cache

Access pattern:

```
<table>
<thead>
<tr>
<th>Tag</th>
<th>Valid Bit</th>
<th>Tag</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>00001</td>
<td>0</td>
<td>000</td>
<td>Mem[C</td>
</tr>
<tr>
<td>00110</td>
<td></td>
<td>001</td>
<td>Mem[6</td>
</tr>
<tr>
<td>00001</td>
<td></td>
<td>001</td>
<td>Mem[6</td>
</tr>
<tr>
<td>11010</td>
<td></td>
<td></td>
<td>Mem[C</td>
</tr>
<tr>
<td>00110</td>
<td></td>
<td></td>
<td>Mem[C</td>
</tr>
</tbody>
</table>
```

Tag  C1  miss

Valid Bit  0  0

Tag  000  001

Cache Size Example

How many total bits are required for a direct-mapped cache with 64 KB of data and 1-byte blocks, assuming a 32-bit address?

Index bits: 16 bits

Bits/block:
  Data: 8 bits
  Valid: 16 bits
  Tag: 16 bits

Total size:
\[ 2^{16} \text{ lines} \times 255/\text{line} \times 16 \]
\[ = 25 \times 2^{10} = 200 \text{ KB} \]
Cache Block Overhead

Previous discussion assumed direct mapped cache 1 byte blocks

Uses temporal locality by holding on to previously used values

Does not take advantage of spatial locality

Significant area overhead for tag memory

Take advantage of spatial locality & amortize tag memory via larger block size

<table>
<thead>
<tr>
<th>Valid Bit</th>
<th>Tag</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$2^{n-1}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Cache Blocks

Assume $2^{10}$ line direct mapped cache with $2^9$ byte blocks

Valid Bit | Tag | Data
---|---|---
0 | [Diagram showing valid bit and tag positions] | [Diagram showing data positions]
1
2
3
4
5
6
7
... 
$2^{10}$-1

Cache Tag = 58  Cache Index = 4  Byte Select = 1
Cache Block Example

Given a cache with 64 blocks and a block size of 16 bytes, what block number does byte address $1200_{10}$ map to?

$$BS = Addr \mod \text{block size} = Addr \mod 16$$

$$c = \lceil\frac{Addr}{\text{blocks size}}\rceil$$

$$cT = c \mod \#\text{of blocks}$$

$$cT = \lceil\frac{c}{\#\text{of blocks}}\rceil$$

$$Tag = \lceil\frac{cT}{\#\text{of blocks}}\rceil$$