= 4.25 cycles

\[
1 + 0.5 + 0.25 + 2.5
\]

\[
0.05 \times 10 + 0.05 \times 1 \times 0.1 \times 0.5 \times 0.0001 \times 1000
\]

\[
AMAT = 1 + 0.05 \times 10 + 0.05 \times 1 \times 0.5 \times 1 \times 0.1 \times 0.5 \times 0.0001 \times 1000
\]

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

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

Review Problem 37
1. Send the original PC to the memory
2. Instruct memory to perform a read and wait
3. Write the result to the appropriate cache line
4. Restart the instruction

Data Miss

1. Stall the pipeline (freeze following instructions)
2. Instruct memory to perform a read and wait (no write enables)
3. Return the result from memory and allow the pipeline to continue

Instruction Miss

Processor expects a cache hit (1 cycle), so no effect on hit.

Handling a Cache Miss
more recently accessed items

When we must evict items to make room for new ones, attempt to keep

keep more recently accessed items closer to the processor

Temporal locality

Move blocks consisting of multiple contiguous words to upper level

Spatial locality

Exploiting Locality
Cache Arrangement

The cache depends upon the memory address. Data can be placed in a limited number of places in

N-way Set Associative – Data can be placed anywhere in the cache.

Fully Associative – Data can be placed anywhere in the cache.

Direct Mapped – Memory addresses map to particular location in that cache.

How should the data in the cache be organized?

Caches are smaller than the full memory, so multiple addresses must map to the same cache “line.”
Memory Address

Optimize for spatial locality (close blocks likely to be accessed soon)

4 byte direct mapped cache with 1 byte blocks

Rem Address % cache size

Bother = 6, 15

Mod
Finding a Block

What must we store?

We add a tag to each cache entry to identify which address it currently contains.

Cache 0 could hold 0, 4, 8, 12, ...

Each location in the cache can contain a number of different memory locations...
Assume 2^9 byte direct mapped cache with 1 byte blocks

Cache Tag & Index

Cache Tag = 57

Cache Index = 03
Cache Access Example

Assume 4 byte cache

Access Pattern:

00110
11010
10000
00110
00001

Data
Tag
Valid Bit

W
W
W
W

0 0 0 0 0 0 0 0
1 0 1 0 1 0 1 0
0 0 0 0 0 0 0 0