Lesson 10

Today: direct-mapped cache only

Rules:

1. Always show your work
2. Always work in powers of 2
3. Always show your units
4. Always double check your answer

In a direct-mapped cache, each block of main memory is mapped to EXACTLY ONE line of cache. If that block of memory is in the cache, it must be in that one line. If that block is NOT in that line, then it must not be in the cache.

QUESTION: how many bits make up the TAG, the LINE, and the WORD for:

Main memory: 1 cell/word, 1 MB total, 1 byte/word

Cache: 32 lines, 8 words per line

ANSWER:

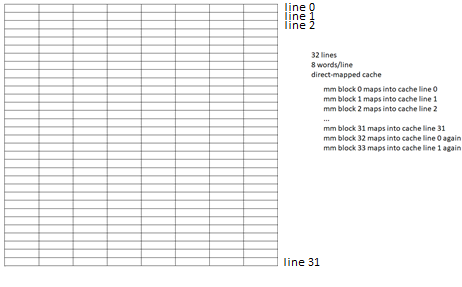
|  |  |  |
| --- | --- | --- |
| TAG | LINE | WORD |
| Total cache:  2^5 lines/cache \* 2^3 words/line  = 2^8 words/cache  Total memory:  2^20 B/mm divided by 2^0 B/word  = 2^20 words/mm  2^20 words/mm divided by  2^8 words/cache = 2^12 cache/mm  12 bits | Log2(32) = 5 because 2^5 = 32  5 bits | Log2(8) = 3 because 2^3 = 8  3 bits |
| Log2(# of caches per memory) | Log2(# of lines per cache) | Log2(# of words per line) |

Double check our work: 12 + 5 + 3 = 20 bits; therefore the main-memory address should be 20 bits long. Therefore there should be 2^20 cells/mm (i.e. 2^20 addresses/mm):  
2^20 words/mm divided by 2^0 cells/word = 2^20 cells per mm. The result looks good.

|  |  |  |
| --- | --- | --- |
| T | L | W |
| 12 | 5 | 3 |

There are 2^12 caches/mm; there are 2^5 lines in cache; there are 2^3 words in each line.

Definition: logarithm: “exponent”



Into what line of cache does mm block 5000 go? 5000%32 = 8 line 8

Which mm cells are in block 5000?

Block 0 contains cells 0 to 7  
 Block 1 contains cells 8 to 15  
 Block 2 contains cells 16 to 23  
 …  
 Block 5000 contains cells 40000-40007

What line of cache does mm block 105 go into? 105%32 = 9 line 9

Which mm cells are in block 105? 840-847

1. Always show your work
2. Always work in powers of 2
3. Always show your units
4. Always double check your answer

QUESTION: how many bits make up the TAG, the LINE, and the WORD for:

Main memory: 1 cell/word, 16-bit address, 4 bytes/word

Cache: 1kB, 32 bytes/line

ANSWER: The number of bits in the tag + line + word must = 16 bits; the same number as in the address

|  |  |  |
| --- | --- | --- |
| TAG | LINE | WORD |
| First technique:  16-bit address  Minus 5-bit line  Minus 3-bit word  = 8-bit tag  Double check: second technique:  Are there really 2^8 caches/mm?  Total cache: 2^10 bytes  Total memory:  2^16 addresses/mm  = 2^16 cells/mm  = 2^16 words/mm  times 2^2 bytes/word  = 2^18 bytes/mm  Total memory/cache:  2^18 bytes/mm divided by  2^10 bytes/cache  = 2^8 caches/mm  Take the logarithm: 8 | 2^10 bytes/cache divided by  2^5 bytes/line =  2^5 lines/cache  Take the logarithm: 5 | 2^5 bytes/line divided by  2^2 bytes/word =  2^3 words/line  Take the logarithm: 3 |
| 8 | 5 | 3 |

1. Always show your work
2. Always work in powers of 2
3. Always show your units
4. Always double check your answer

QUESTION: how many bits make up the TAG, the LINE, and the WORD for:

Main memory: 1 cell/word, 2GB, 30-bit address

Cache: 512 kB, 64 lines

ANSWER: TAG + LINE + WORD bits must equal 30 bits!

|  |  |  |
| --- | --- | --- |
| TAG | LINE | WORD |
| Log2(Total mm/total cache):  2^31 B/mm divided by  2^19 B/cache =  2^12 cache/mm  12 | Log2(64) = 6  6 | 30 – 12 = 6 = 12  Double check:  2^30 addresses/mm  = 2^30 cell/mm  = 2^30 words/mm  2^31 bytes/mm divided by  2^30 words/mm  = 2^1 bytes/word  2^19 bytes/cache divided by  2^6 lines/cache  = 2^13 bytes/line  2^13 bytes/line divided by  2^1 bytes/word  = 2^12 words/line  Log2(2^12) = 12  12 |
| 12 | 6 | 12 |

QUESTION: how many bits make up the TAG, the LINE, and the WORD for:

Main memory: 1 cell/word, 24-bit address, 32 bits per word

Cache: 16 words per line, 256kB total

|  |  |  |
| --- | --- | --- |
| TAG | LINE | WORD |
| 2^24 addresses/mm  = 2^24 cells/mm  = 2^24 words/mm  2^5 bits/word divided by  2^3 bits/byte  = 2^2 bytes/word  2^24 words/mm \* 2^2 B/word  = 2^26 bytes/mm  2^26 B/mm divided by  2^18 B/cache  = 2^8 caches/mm  Log2(2^8) = 8-bit tag | 24-bit address –  8-bit tag –  4-bit word  = 12-bit line  Double check:  Are there really 2^12 lines??  2^4 words/line \*  2^2 bytes/word  = 2^6 bytes per line  2^18 bytes per cache divided by  2^6 bytes per line  = 2^12 lines per cache  Looks good! | Log2(16) = 4-bit word |
| 8 | 12 | 4 |

QUESTION: how many bits make up the TAG, the LINE, and the WORD for:

Main memory: 1 cell/word, 64MB, 2 bytes per word

Cache: 1024 bits in each line; 1MB total cache

ANSWER:

How long should the address be?

2^26 B/mm divided by 2^1 B/word = 2^25 words/mm

Which means the mm address is 25 bits long

|  |  |  |
| --- | --- | --- |
| TAG | LINE | WORD |
| 2^26 B/mm divided by  2^20 B/cache  = 2^6 caches/mm  Log2(2^6) = 6-bit tag | 25 bits – 6 – 6 = 13 bits  Double check!  Are there really 2^13 lines/cache?  2^20 B/cache divided by  2^7 bytes/line  = 2^13 lines/cache | 2^10 bits/line divided by  2^3 bits/byte  = 2^7 bytes/line  2^7 bytes/line divided by  2^1 bytes/word  = 2^6 words/line  Log2(2^6) = 6-bit word address |
| 6 | 13 | 6 |

QUESTION: a) how many bits make up the TAG, the LINE, and the WORD for:

Main memory: 1 cell/word, 20-bit address, 8 bytes per word

Cache: 8 words per line; 128 lines

b) what are the actual TAG, LINE and WORD values for cell 02A51? Answer in base ten.

ANSWER: T-L-W is 10-7-3

02A51

0000 0010 1010 0101 0001

0000001010 1001010 001  
 TAG LINE WORD

Base ten: T-L-W for cell 02A51 is 10-74-1

Therefore cell 2A51 is word 1 in line 74; it is in the 11th block (i.e. block 10) which gets mapped directly into line 74

Recall our first question from today:

QUESTION: how many bits make up the TAG, the LINE, and the WORD for:

Main memory: 1 cell/word, 1 MB total, 1 byte/word

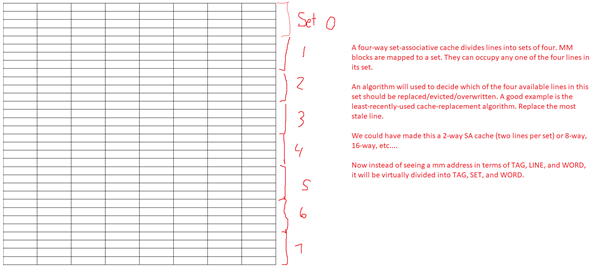
Cache: 32 lines, 8 words per line

In this direct-mapped cache, mm blocks 0, 32, 64, etc all map directly into cache line 0. There is no choice and no flexibility:

Table

Description automatically generated

Let’s change that. We will give each mm block FOUR spots to which they can be mapped. Cache has been divided into SETS of four lines. Each mm block is mapped directly to a SET OF FOUR LINES. Which line does it take? It can take any of those four. An algorithm will determine which one. An example is the least-recently-used cache-replacement algorithm, where a new mm block will evict the least-recently-used line in a set.



Therefore, we need to keep timestamps that are updated whenever a cache line is accessed. If a particular line in a set was just used, then it is the most-recently-used line and it should be kept in the cache and not replaced soon (because of the principle of temporal locality, it is very likely to be used again soon, so let’s keep it in the cache which is faster than mm). QUESTION: Where is the principle of spatial locality being used? ANSWER: cache stores 8 words (a block) instead of just 1 word (therefore takes 3-bits for the cpu to identify which ONE word it wants out of the 8 per line).

Graphical user interface

Description automatically generated with medium confidence

QUESTION: which set does mm block 5000 go into? 5000%8 = 0

Set 0. Block 5000 would replace the least-recently-used line in block 0.

QUESTION: which line does mm block 105 go into? 105%8 = 1; it goes into one of the four lines in set 1; it would replace the most stale (i.e. least-recently-used) line in set 1

QUESTION: how would that picture change if we had an 8-way set-associative cache?

ANSWER: 8 lines per set (in this case, we would have four sets)

QUESTION: how would that picture change if we had a 32-way set-associative cache?

ANSWER: 1 set of 32 lines aka “fully-associative cache”

QUESTION: how would that picture change if we had a 1-way set-associative cache?

ANSWER: this is a direct-mapped cache! 32 sets of 1 line each

Out of all these choices, the 32-way SA cache makes the best usage of the temporal locality principle.

QUESTION: How many bits make up the Tag, Line, and Word for:

Main memory: 1 cell/word, 1MB total, 1 byte/word

Cache: direct mapped, 32 lines, 8 words per line

ANSWER:

|  |  |  |
| --- | --- | --- |
| TAG | LINE | WORD |
| 12 | 5 (2^5 lines) | 3 |

Whole mm address is 12+5+3 = 20 bits long

QUESTION: How many bits make up the Tag, SET, and Word for:

Main memory: 1 cell/word, 1MB total, 1 byte/word

Cache: 4-way set-associative cache, 32 lines, 8 words per line

ANSWER:

|  |  |  |
| --- | --- | --- |
| TAG | SET | WORD |
| 14 | 3 (2^3 sets) | 3 |
| Up by 2 (ie multiplied by 2^2) | Down by 2 (ie divided by 2^2) | NO CHANGE EVER |

Whole mm address is 12+5+3 = 20 bits long

**QUESTION: How many bits make up the Tag, SET, and Word for:**

**Main memory: 1 cell/word, 1MB total, 1 byte/word**

**Cache: 8-way set-associative cache, 32 lines, 8 words per line**

**ANSWER:**

|  |  |  |
| --- | --- | --- |
| **TAG** | **SET** | **WORD** |
| **15** | **2** | **3** |
|  |  |  |

**Whole mm address is 12+5+3 = 20 bits long**