# COMP 228 Winter 2020

## Memory & Cache Practice Problems

A memory address as seen by the cache looks like this

| tag | Index | offset |
|---|---|---|

Index: Contains frame index in direct-mapped cache or set index in m-way associative cache.

Cache-line-number The tag + Index fields. This is the block number in main memory.

(Direct-mapped cache) Frame Index = cache-line-number mod # cache frames

(Associative cache) Set Index = cache-line-number mod # cache Sets

Offset: Relative address of a particular byte inside a cache frame or memory block.

E.g. if frame size is 4 bytes, then offset field would have require 2-bits and has 4 possible values (00,01,10,11) for each of the bytes inside the frame.

Cache-line: The block of data transferred from main memory to cache and vice-versa.

Size of cache-line = size of a cache frame = size a block in main memory

Exercises

1- Suppose a computer using direct mapped cache has $2^{32}$ bytes of byte-addressable main memory, and a cache size of 512 bytes, and each cache block contains 64 bytes.

a- How many blocks of main memory are there?

b- What is the format of a memory address as seen by the cache? i.e. what are the sizes of the tag, block and offset fields?

c- To which cache block, will the memory address $13A4498A_{16}$ map?

Solution:

a- Block size (memory or cache) = 64 = $2^6$ → offset 6 bits
# main memory blocks = $2^{32}$ / $2^6$ = $2^{26}$

b-
# cache frames = 512/64 = 8 = $2^3$ → Index 3 bits

| tag | Index | offset |
|---|---|---|
| ---------23---------- | ----------3---------- | --------6-------- |

c- We only need to look at the lower 3 digits of address: 98A = 100110001010

Cache block 6

2- Suppose a byte-addressable computer using set associative cache has $2^{21}$ of bytes of Main memory and a cache of 64 frames, where each cache frame contains 4 bytes.

a- If this cache is 2-way associative, what is the format of a memory address?

b- If this cache is 4-way associative, what is the format of a memory address?

Solution:

a- Cache frame size = 4 = $2^2$ → offset 2 bits

# sets = 64 / 2 = 32 = $2^5$ → set index 5 bits

Tag 14 bits

| tag | Set Index | offset |
|---|---|---|
| ---------14---------- | ----------5---------- | --------2-------- |

b- # sets = 64 / 4 = 16 = $2^4$ → set index 4 bits

Tag 15 bits

| tag | Set Index | offset |
|---|---|---|
| ---------15---------- | ----------4---------- | --------2-------- |

3- Suppose that we have a computer that uses a memory address of 8-bits. This computer has a 16 byte cache and 4 bytes per block. The computer accesses a number of memory locations throughout the course of running a program. Suppose the computer uses direct-mapped cache.
The system accesses memory addresses (in hex) in this exact order: 6E, B9, A9, E0, 4E, A8, A9, AA, 93, and 94. What memory blocks will be in the cache after the last memory access?
Solution:
Memory size = $2^8$ bytes
# memory blocks = $2^8$ / 4 = $2^6$= 64 blocks
There will be 4 cache frames
Cache will look like this

| Frame | Content (memory block) | Tag |
|---|---|---|
| 0 | | |
| 1 | | |
| 2 | | |
| 3 | | |

Initially Cache is empty. Each time a memory block is requested; its address would be mapped and placed in the cache along with the tag. Here is the order of requests in this example. These are 8-bit memory byte addresses (each address is for a particular byte in the frame) that we map to get frame index.
Remember that each cache frame contains 4 bytes so it is possible that multiple addresses map to the same frame with no conflict.

| Memory Req. | Cache line number Tag+fr Index Mem Block # | Frame Index | Tag | Cache Content | Hit/Miss |
|---|---|---|---|---|---|
| 6E 01101110 | 011011 = 27 | 27 mod 4 =3 | 6 | Cache[3] = block 27 Tag[3] = 6 | Frame 3 empty. Compulsory miss |
| B9 10111001 | 101110 = 46 | 46 mod 4 =2 | B | Cache[2] = block 46 Tag[2] = B | Frame 2 empty. Compulsory miss |
| A9 10101001 | 101010 = 50 | 50 mod 4 =2 | A | Cache[2] = block 50 Tag[2] = A | Frame 2 is not empty tags compared. A9 tag is A and Tag[2] is B. Frame 2 is evicted. This is a conflict miss. |
| E0 11100000 | 111000 = 56 | 56 mod 4 =0 | E | Cache[0] = block 56 Tag[0] = E | Frame 0 empty. Compulsory miss |
| 4E | 010011 = 19 | 19 mod 4 =3 | 4 | Cache[3] = block 19 | Frame 3 is not empty |

| | | | | | |
|---|---|---|---|---|---|
| 01001110 | | | | Tag[3] = 4 | tags compared. 4E tag is 4 and Tag[3] is 6. Frame 3 is evicted. This is a conflict miss. |
| A8<br>10101000 | 101010=50 | 50 mod 4 =2 | A | Cache[2] = block 50<br>Tag[2] = A | Frame 2 is not empty tags compared. A8 tag is A and Tag[2] = A. So, it is a hit. |
| A9<br>10101001 | 101010=50 | 50 mod 4 =2 | A | Cache[2] = block 50<br>Tag[2] = A | Again a hit. |
| AA<br>10101010 | 101010=50 | 50 mod 4 =2 | A | Cache[2] = block 50<br>Tag[2] = A | Again a hit. |
| 93<br>10010011 | 100100=36 | 36 mod 4 =0 | 9 | Cache[0] = block 36<br>Tag[0] = 9 | Frame 0 is not empty. Tags compared. 93 tag is 9 and Tag[0] = E. Frame 0 has to be evicted. This is a conflict miss. |
| 94<br>10010100 | 100101=37 | 37 mod 4 =1 | 9 | Cache[1] = block 37<br>Tag[1] = 9 | Frame 1 empty. Compulsory miss |