Solutions: Problem 2.7

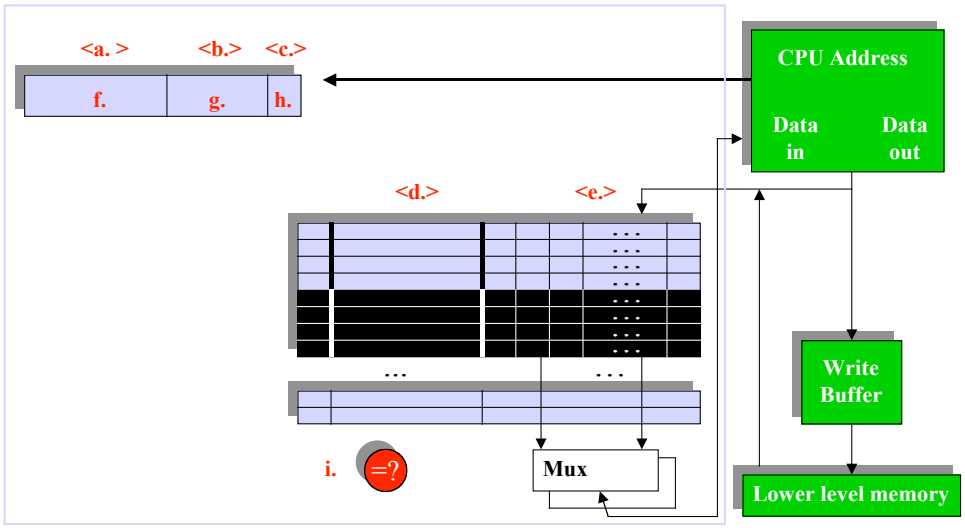
Question 1:

Part A:

You have been asked to design a cache with the following properties:

- Data words are 32 bits each
- A cache block will contain 2048 bits of data
- The cache is direct mapped
- The address supplied from the CPU is 32 bits long
- There are 2048 blocks in the cache
- Addresses are to the word

Pictured below is the general structure of a cache. There are 8 fields (labeled a, b, c, d, e, f, g, and h). In the space below you'll need to indicate the proper name or number of bits for a particular portion of this cache configuration. Whether a name or number should be entered will be specified:



f. (name) _____

- You are being asked to show what part of a physical address form the index, offset, and tag. < f > refers to the most significant bits of the address – so this is the tag.

g. (name) _____

- It follows that the next part of the address is the index.

h. (name) _____

- The least significant bits form the offset.

c. (number) _____

- There are 2^{11} bits / block and there are 2^5 bits / word. Thus there are 2^6 words / block so we need 6 bits of offset.

b. (number) _____

- There are 2^{11} blocks and the cache is direct mapped (or "1-way set associative"). Therefore, we need 11 bits of index.

a. (number) _____

- The remaining bits form the tag. Thus, 32 – 6 – 11 → 15 bits of tag.

d. (number) _____

- Field < d > refers to the fact that a tag must be stored in each block. Thus, 15 bits are kept in each block.

e. (number) _____

- Field < e > asks you to specify the total number of bits / block. This is 2048.

i. What 3 things must be compared at ? to determine if the cache entry is useable or not?

- We need to compare the valid bit associated with the block, the tag stored in the block, and the tag associated with the physical address. The tags should be the same and the valid bit should be 1.

j. What is the total size of the cache?

- There are 2048 blocks in the cache and there are 2048 bits / block.
    - There are 8 bits / byte
    - Thus, there are 256 bytes / block
- 2048 blocks x 256 bytes / block → 2^{19} bytes (or 0.5 MB)

Part B:

Now, let's consider what happens if we make our cache 2-way set-associative instead of direct mapped. However, as before, the following still applies:

- Data words are 32 bits each
- A cache block will contain 2048 bits of data
- The address supplied from the CPU is 32 bits long
- There are 2048 blocks in the cache
- Addresses are to the word

Number of bits in offset?

- There are still 6 bits in the offset; data is still word addressed

Number of bits in index?

- We now need one less bit of index because we address to the set
    - 2^{11} blocks / 2^1 blocks/set = 2^{10} sets
    - (10 bits of index needed)

Number of bits in tag?
- 32 – 6 – 10 = 16 bits.

Part C:
Now, let's consider what happens if we make our cache 4-way set-associative instead of direct mapped. However, as before, the following still applies:
- Data words are 32 bits each
- A cache block will contain 2048 bits of data
- The address supplied from the CPU is 32 bits long
- There are 2048 blocks in the cache
- Addresses are to the word

Number of bits in offset?
- There are still 6 bits in the offset; data is still word addressed

Number of bits in index?
- We now need one less bit of index because we address to the set
    - 2^{11} blocks / 2^2 blocks/set = 2^9 sets
    - (9 bits of index needed)

Number of bits in tag?
- 32 – 6 – 9 = 17 bits.

Part D:
Now, let's consider what happens if data is byte addressable. We'll keep the cache 4-way set associative for this question. However, as before, the following still applies:
- Data words are 32 bits each
- A cache block will contain 2048 bits of data
- The address supplied from the CPU is 32 bits long
- There are 2048 blocks in the cache

Number of bits in offset?
- There *were* 6 bits in the offset
- Now, each of the 4 bytes of a given word can be individually addressed
    - Therefore, we need 2 more bits of address *per word*
- Thus, 2^6 words * 2^2 bytes / word → 2^8
    - 8 bits of offset are needed.

Number of bits in index?
- We need the same number of index bits as in Part D
    - 2^{11} blocks / 2^2 blocks/set = 2^9 sets
    - (9 bits of index needed)

Number of bits in tag?
- 32 – 8 – 9 = 15 bits.

Part E:

Now, let's consider what happens if the size of our physical address changes from 32 bits to 64 bits. We'll keep the cache 4-way set associative and data will still be addressable to the byte for this question. However, as before, the following still applies:

- A cache block will contain 2048 bits of data
- There are 2048 blocks in the cache

Number of bits in offset?

- 8 (as above)

Number of bits in index?

- 9 (as above)

Number of bits in offset?

- 64 – 8 – 9 → 47 bits