Set-associative caches

Recall that in a set-associative cache, the index field identifies a set of blocks (unlike a direct-mapped cache, where the index identifies a unique block) that may contain the data. Every block within this set is examined (in parallel) to see if the tag field matches and the valid bit is set. If both these conditions are met, we have a cache hit. Otherwise, on a cache miss, we must load a new block from memory into this set, replacing the least recently used (LRU) block in that set.

Problem 1: Suppose a cache is 2-way set associative. What additional information should the cache store so that it can identify the LRU block in a set? How would your answer change if the cache was 4-way set associative?

Solution: For a 2-way set associative cache, each set contains two blocks (block 0 and block 1). We need one bit per set to identify which of these blocks is the LRU block. Note that if block 0 is the LRU block, then block 1 is the MRU (most recently used) block, and vice versa. If the cache was 4-way set associative, each set needs to record the LRU block, the second-LRU block, ..., the MRU block. There are 4! = 24 permutations of 4 blocks, and hence the cache needs 5 LRU bits per set.

Problem 2: Suppose memory addresses are n bits long and the cache can hold M bytes of data. If the cache block-size is $2^b$ bytes and the cache is k-way set associative, write a formula for the total number of bits to implement the cache with a write-through policy. How would your answer change if the cache had a write-back policy?

Solution: The block-offset is clearly b-bits long. The cache contains $M/2^b$ blocks, and since it is k-way set associative, it contains $s = \frac{M/2^b}{k}$ sets. Thus, the index field is $\log_2 s$ bits long, and so the tag field is $t = n - \log_2 s - b$ bits long. Each block contains $8 \times 2^b$ data bits, a t-bit tag field and a valid bit. If the cache had a write-back policy, each block would also contain a dirty bit. Each set contains $L = \lceil \log_2(k!) \rceil$ LRU bits. Thus, the total number of bits to implement the cache is:

$$s \times L + (M/2^b) \times (8 \times 2^b + t + 1 + 1 \text{ (if write back)})$$

Problem 3: A byte addressable memory system has 32-bit addresses.

(a) The memory system has a single-level write back cache with 16 byte blocks, a 32-bit bus with a 1 cycle latency between cache and memory, interleaved memory with four banks, each with a 10 cycle latency. Draw a pipeline diagram to show how a dirty cache block will be written to memory.

Solution: Since the width of the cache-to-memory bus is 4 bytes but cache blocks are 16 bytes long, the bus will be used four times as shown below (A = address, D = data):

\[
\begin{array}{cccccccccccccccc}
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
\end{array}
\]

Notice that the cache finishes its job by the end of cycle 8, although memory is not completely done until the end of cycle 18.
(b) When a new cache block needs to be loaded from memory, what is the miss penalty for the cache? (Note: Does your answer depend on whether the block being replaced is dirty or clean?) Draw a pipeline diagram to support your answer.

Solution: The answer certainly depends on whether the block being replaced is dirty or clean: the miss penalty in the former case is more. First, however, consider loading a cache block when the block being replaced is clean:

1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8
A - - - - - - - - - - D
A - - - - - - - - - - D
A - - - - - - - - - - D
A - - - - - - - - - - D

The miss penalty in this case is clearly 15 cycles. Now suppose the cache block being replaced was dirty. This block would first have to be written to memory, and then the new block would have to be read. In order to avoid a structural hazard where a bank would need to both write and read, the pipeline would have to be stalled for six cycles as follows:

1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
A D - - - - - - - - - -
A D - - - - - - - - - -
A D - - - - - - - - - -
A D - - - - - - - - - -
** * * * * * A - - - - - - - - D
A - - - - - - - - - - D
A - - - - - - - - - - D
A - - - - - - - - - - D

Thus the miss penalty in this case is 29 cycles.

(c) The cache has a 1 cycle hit time and a 90% hit rate. How long does it take on average to read a byte of data from a virtual address, if 10% of blocks replaced in the cache are dirty?

Solution: We need to compute the AMAT of the cache. The miss penalty is $p = 0.9 \times 15 + 0.1 \times 29$. Hence the AMAT of the cache is $a = 1 + (1 - 0.9) \times p$ cycles.