A Primer on Caches

Srinidhi Varadarajan

Types of Caches

- Fully Associative
  - A cache entry can be inserted anywhere in the cache
- Direct Mapped
  - Based on the block address (tag), the cache entry can only be inserted in one location in the cache
- Set associative
  - Based on the block address (tag), the cache entry is allocated a set. The cache entry can be inserted anywhere within the set.
  - Based on the number of sets you can have 2-way, 4-way ... n-way set associative caches.

Cache State

- Block address: The address of the cache block. In the case of the network caches, the block address is the network address.
- Tag: Each entry in the cache has a tag, which indicates the complete address of the cache entry.
- Valid bit: Indicates if the cache entry has valid data in it or not. When the cache is started (cold start), the valid bit is false for all cache entries. ALWAYS CHECK THE VALID BIT EVEN IF TAGS MATCH.
- For a set-associative cache, use the block address to determine the cache set. Then compare the block address with the tags of the blocks within a set to check for a cache hit.

Mapping Block Entries

- Using a block address, you need to determine an appropriate cache set.
  - A hash function is used to map between block addresses and the number of sets.
  - For example, in a 64KB 4-way set associative cache, there are 64KB/4 = 16K sets, each with 4 blocks. Use a hash function to map between the block address to a number between 0 – (16K – 1).
  - Given the cache set, compare the block address with the tags within the individual blocks. If the tag matches the block address, you have a cache hit.
  - For example, in a 4 way set associative cache, there are 4 blocks in a set. Once the set is determined, compare the block address with the tags of the 4 blocks to determine if the cache entry exists in the cache.
  - Always check the valid bit.

Replacement Policies

- What happens if you have a cache miss?
  - Insert the new block address into the cache.
  - What if the cache set is full? All cache entries in the set have their valid flag set.
  - You need to bump out a cache entry to insert the new entry.

  - A cache replacement policy is needed to determine the cache entry that can be replaced when a cache set is full.

- Least Recently Used: Bump out the entry that was used last.
  - How do you measure time for least recently used?
  - Simple mechanism: Use virtual time represented by a number associated with each cache entry. Update the virtual on a cache hit.
  - If the cache set is full, discard the cache entry with the smallest virtual time.

  - Simple to implement. Doesn’t need to store real time.
Replacement Policies

- Least Frequently Used: Bump out the cache entry that was accessed least.
  - Associate a frequency counter with each cache entry.
  - Increment the frequency counter on a cache hit.
  - If the cache set is full, bump out the entry which has the least value for the frequency counter.

Reading

- Recommended
  - Computer Architecture: A Quantitative Approach, Hennessy and Patterson, Morgan Kaufman