Chapter 4b: Direct-Mapped Cache

Gene Cooperman

THIS IS STILL A WORK IN PROGRESS.

0.0.1 2. User Request (Direct-Mapped Cache Description)

A **direct-mapped CPU cache** is the simplest cache organization where a **memory block can be placed in only one specific cache line** (or "slot"), determined by a portion of the memory address . This mapping is not flexible, meaning a unique **index** (derived from the address) points directly to the single cache line where the block must reside.

0.1 Address Decoding and Mapping

When the CPU requests an address, the address is logically divided into three parts:

1. **Block Offset:** Identifies the specific data unit (e.g., a byte) within a data **block** (or "cache line").
2. **Index (or Set Index):** Used to select the specific cache line (or set) where the memory block might be stored. For a direct-mapped cache, the index directly selects the *one and only one* line a block can occupy.
3. **Tag:** The remaining, most significant bits of the address. This uniquely identifies the memory block *within* the main memory, distinguishing it from other blocks that map to the same **Index**.

0.2 Hit Check and Comparators

When the CPU requests data, a **cache hit** occurs only if the following conditions are met for the single cache line pointed to by the **Index**:

1. The **Valid Bit** (V) of the cache line is set to 1 $(V = 1)$.
2. The **Tag** extracted from the CPU's requested address **matches** the Tag stored in that specific cache line.

The check for the Tag match is performed by a **comparator circuit**.

- **Direct-Mapped Cache:** Because each memory block can map to only **one** specific cache line, a direct-mapped cache requires only a **single comparator** circuit. This single comparator compares the Tag from the CPU address with the Tag read from the one indexed cache line.
- **Fully Associative Cache Contrast:** In contrast, a fully associative cache must search *every* cache line simultaneously for a match, requiring **one comparator for every single cache line** in the entire

cache. This is why a direct-mapped design is simpler and faster in terms of lookup logic, though it is more prone to conflict misses.

0.3 Diagram (but with mistakes, as created by Google Gemini)

Note the accompanying diagram created by Google Gemini in this directory at: 4b-cache-direct-mapped-diagram.png

Howeer, the diagram has several mistakes, and Google Gemini semms incapable of fixing the mistakes.