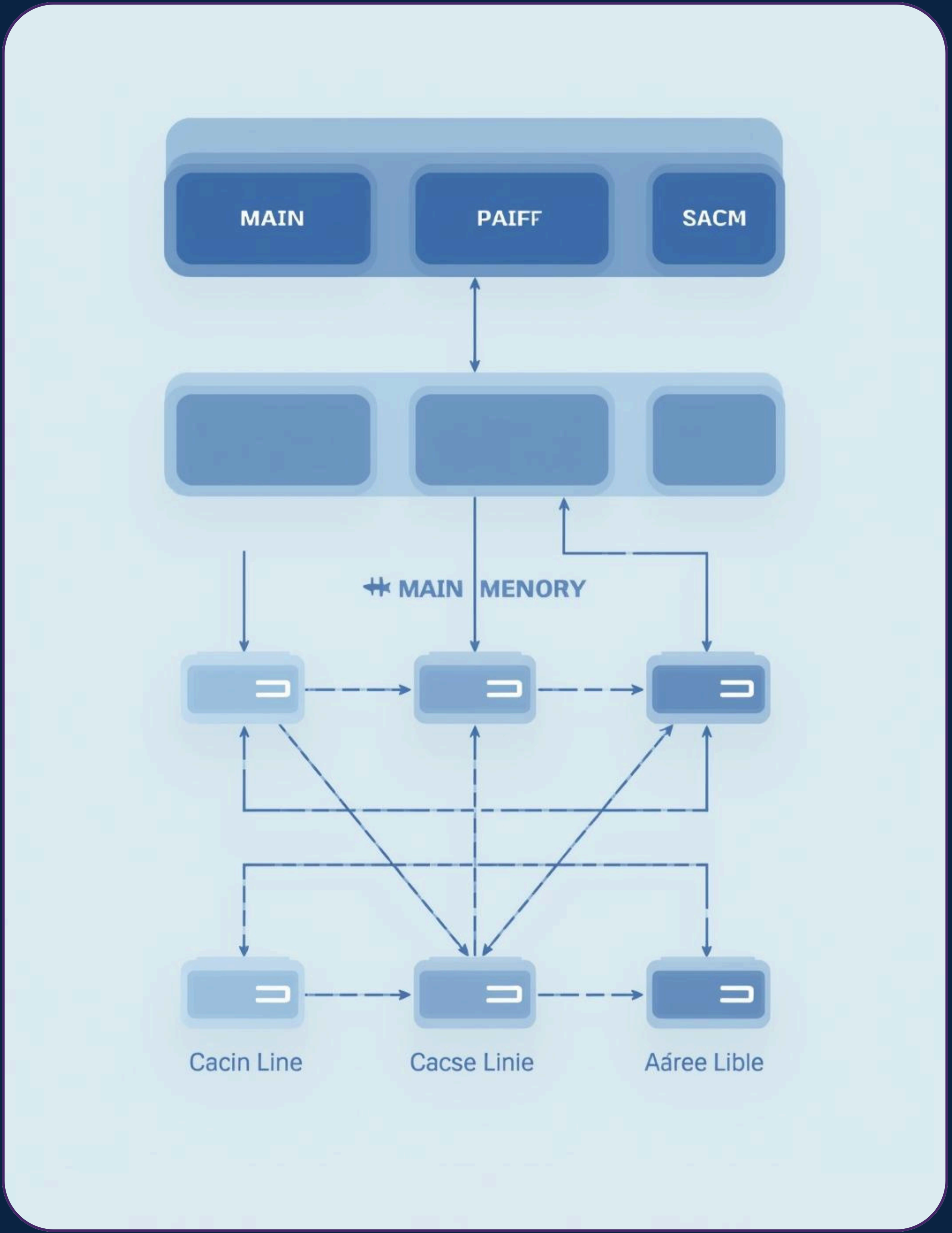
# Direct Mapping in Cache Memory

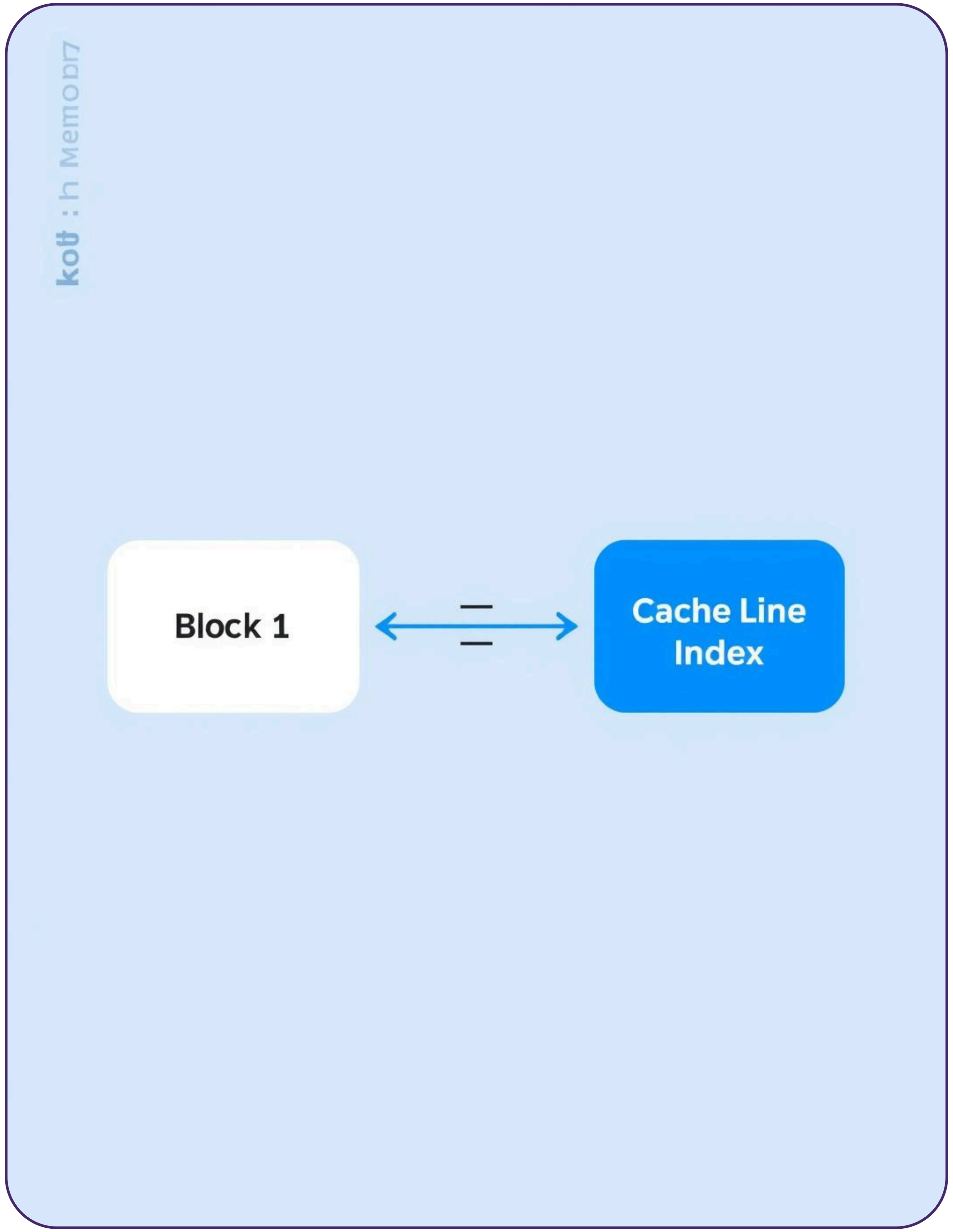
# Definition of Direct Mapping

In direct mapping, **each main memory block uniquely maps** to a single cache line, ensuring efficient data retrieval.



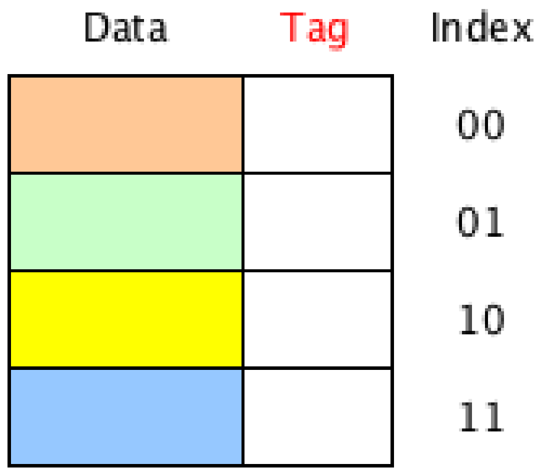
# How Mapping Works

Each block number is processed through a modulo operation to determine its corresponding cache line index.



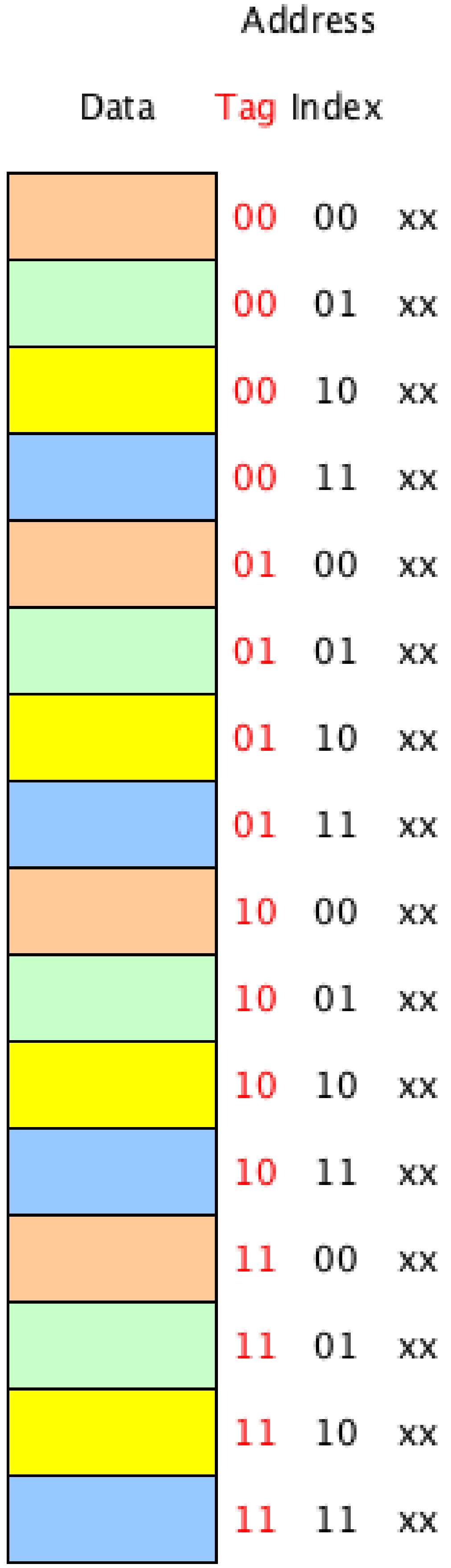
# A SMALL DIRECT-MAPPED CACHE

## Cache



| Data | Tag | Index |
|---|---|---|
| 🟧 | | 00 |
| 🟩 | | 01 |
| 🟨 | | 10 |
| 🟦 | | 11 |

## Main Memory



| | Address | | |
|---|---|---|---|
| Data | Tag | Index | |
| 🟧 | 00 | 00 | xx |
| 🟩 | 00 | 01 | xx |
| 🟨 | 00 | 10 | xx |
| 🟦 | 00 | 11 | xx |
| 🟧 | 01 | 00 | xx |
| 🟩 | 01 | 01 | xx |
| 🟨 | 01 | 10 | xx |
| 🟦 | 01 | 11 | xx |
| 🟧 | 10 | 00 | xx |
| 🟩 | 10 | 01 | xx |
| 🟨 | 10 | 10 | xx |
| 🟦 | 10 | 11 | xx |
| 🟧 | 11 | 00 | xx |
| 🟩 | 11 | 01 | xx |
| 🟨 | 11 | 10 | xx |
| 🟦 | 11 | 11 | xx |

Note the following characteristics of this diagram:

1. a Memory word can be copied only into the Cache location that has the same color
2. only one Memory word of each color can be in the Cache at any given time
3. we need a mechanism to tell us which word of each color is in the cache
4. the address of each word in Main Memory is divided into three fields
   - a. the Tag field is in red ink
   - b. the Index field is in black ink
   - c. the byte offset is always "xx" because it is  not interesting
5. the fat red lines divide Main Memory into 4 blocks
6. each block has a color pattern that matches that of the Cache
7. the Index values of each block also match those of the Cache
8. in every block, the Tag value is the same for all of the words
9. every block has a different Tag value
10. items 8 and 9 are discussed in more detail on the page "Fun with Address Spaces"
11. when we place a word into the Cache, it goes into the place whose Index value is the same as the Index field of the word's address
12. this just happens to be the place that is the same color as the word
13. at the same time, we copy the Tag field of the word's address into the Tag field of the Cache location into which the word is going
14. when we need to find a word in the Cache, the only place to look is the Cache entry whose Index matches the Index field of the word we are looking for
15. if the Tag field of that Cache entry also matches the Tag field of the word's address, then we have found the word in the Cache; i.e. we have a hit
16. if the Tag fields do not match, then the word we are looking for is not in the Cache; i.e. we have a miss

# Cache Structure

In direct mapping, each cache line contains an **index**, a **tag**, and **data**. The tag is crucial for cache hits.

Tag: The remaining, higher-order bits of the address. This tag is stored in the cache line and is compared to the tag portion of the address during a memory request to verify that the correct memory block is present.

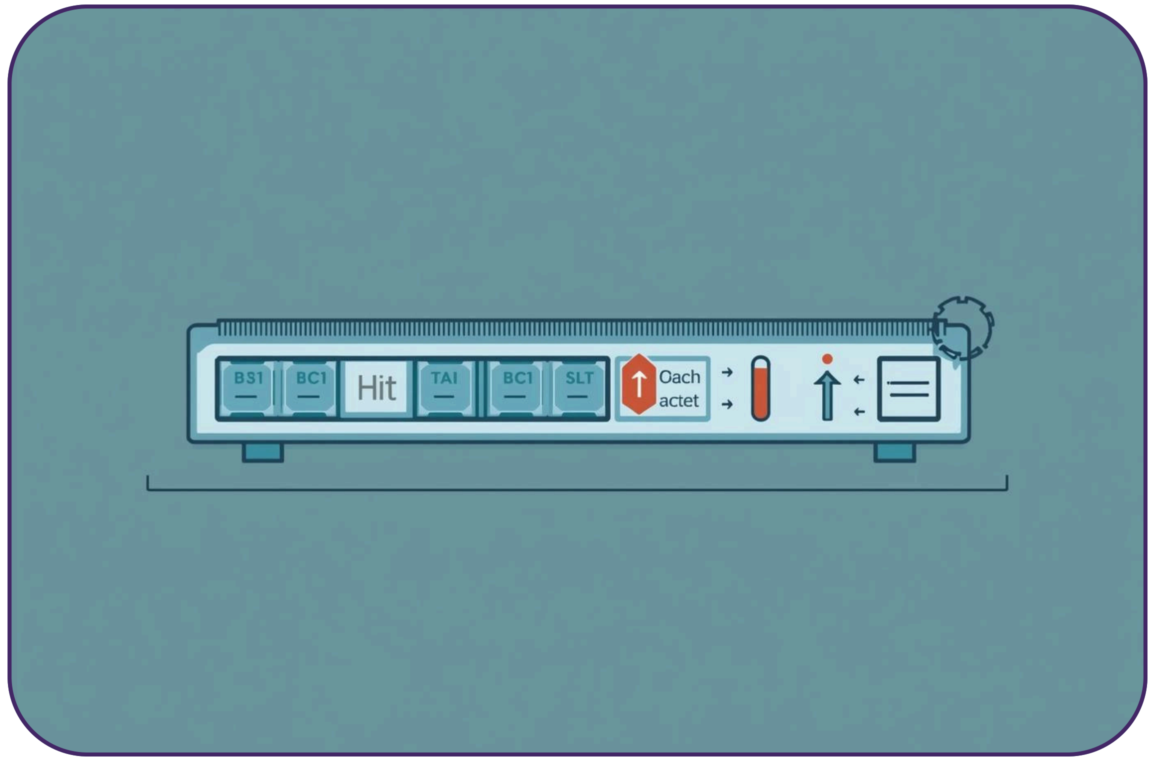
Data: The actual data from main memory that is stored in the cache line. This is the data that the CPU will use if a cache hit occurs.

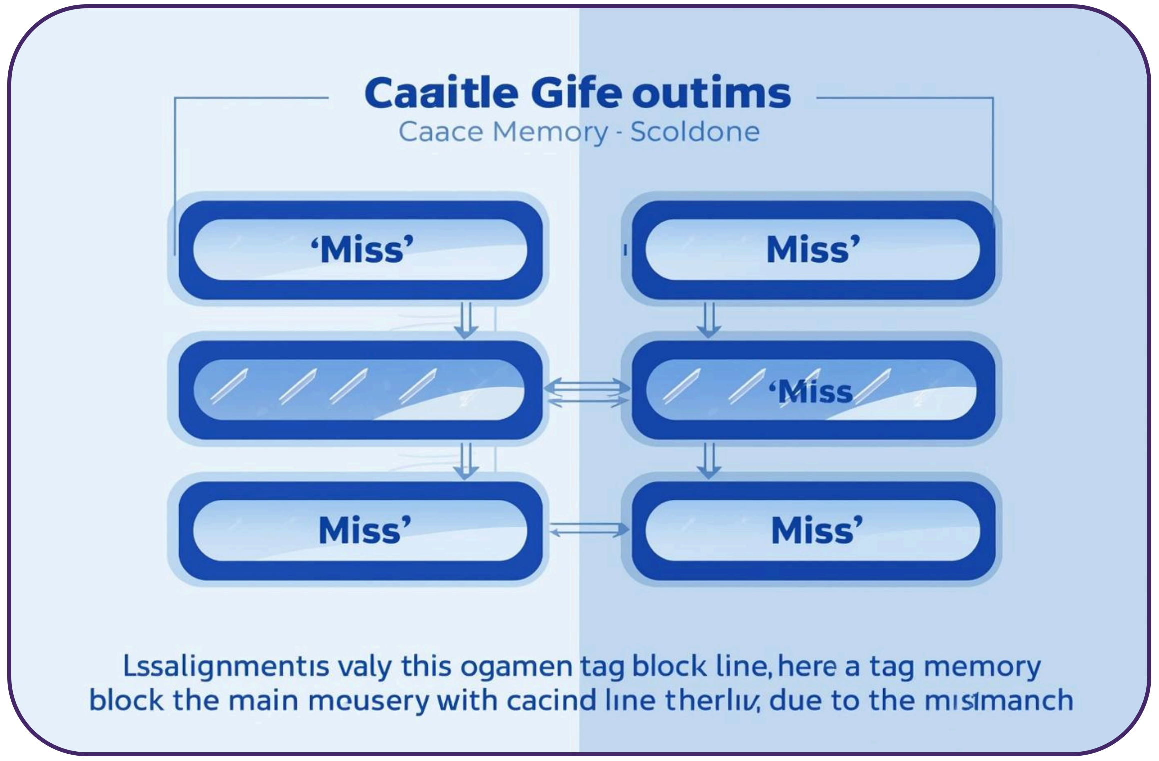
Valid bit: An additional bit in each cache line that indicates whether the line contains valid data or is empty.

# Hit and Miss Scenarios

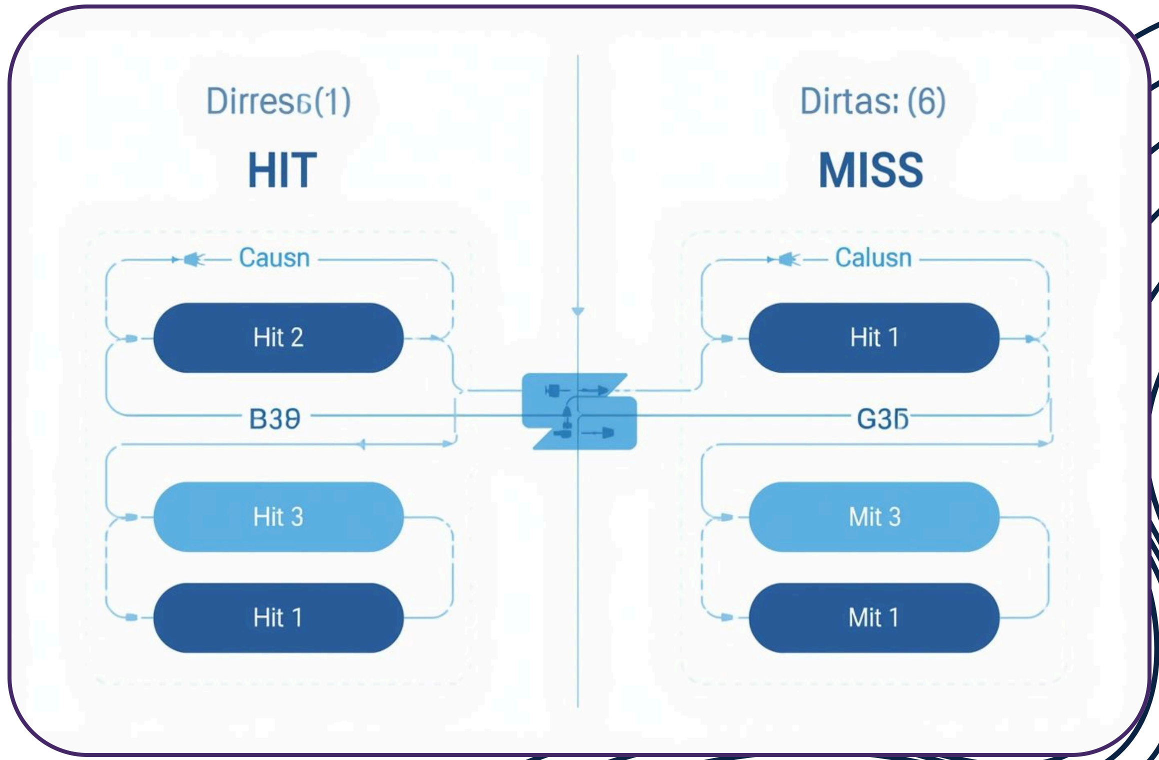
## Understanding cache access results clearly

In direct mapping, cache accesses can result in a **tag match (hit)** or a **tag mismatch (miss)**, significantly impacting performance and efficiency.





Lssalignmentıs valy this ogamen tag block line, here a tag memory block the main mcusery with cacind lıne therlıv, due to the mısmanch



# Advantages & Limitations

**Understanding the trade-offs of direct mapping**

### Speed

Direct mapping offers **high-speed access** to data since each memory block maps to one cache line, minimizing delays and ensuring quick retrieval of frequently used information.

### Simplicity

The architecture of direct mapping is **simple and straightforward**, making it easy to implement in hardware designs, which can reduce complexity and development time for cache systems.

### Conflict Misses

One significant drawback is the occurrence of **conflict misses**, where multiple blocks compete for the same cache line, leading to performance inefficiencies and increased access times.