

Cache Placement Policies

Cache is a memory which holds the recently utilized data by the processor. A block of memory cannot be placed randomly in the cache and is restricted to a single cache line^[1] by the “Placement Policy^{[2][3]}”. In other words, Placement Policy determines where a particular memory block can be placed when it goes into the cache.

There are three different policies available for placement of a memory block in the cache.

Contents

Direct Mapped Cache

- To place a block in the cache
- To search a word in the cache
- Advantages
- Disadvantage
- Example

Fully Associative Cache

- To place a block in the cache
- To search a word in the cache
- Advantages
- Disadvantage
- Example

Set Associative Cache

- To place a block in the cache
- To locate a word in the cache
- Advantages
- Disadvantages
- Example

See also

References

Direct Mapped Cache

In a direct mapped cache structure, the cache is organized into multiple sets^[1] with a single cache line per each set. Based on the address of the memory block, it can only occupy a single cache line. The cache can be framed as a (n*1) column matrix.^[4]

To place a block in the cache

- The set is determined by the index^[1] bits derived from the address of the memory block.
- The memory block is placed in the set identified and the tag^[1] is stored in the tag field associated with the set.
- If the cache line is previously occupied, then the new data replaces the memory block in the cache.

To search a word in the cache

- The set is identified by the index bits of the address.

- The tag bits derived from the memory block address are compared with the tag bits associated with the set. If the tag matches, then there is a cache hit and the cache block is returned to the processor. Else there is a cache miss and the memory block is fetched from the lower memory(main memory, disk).

Advantages

- This placement policy is power efficient as it avoids the search through all the cache lines.
- The placement policy and the replacement policy is simple.
- It requires cheap hardware as only one tag needs to be checked at a time.

Disadvantage

- It has lower cache hit rate, as there is only one cache line available in a set. Every time a new memory is referenced to the same set, the cache line is replaced, which causes conflict miss^[5]

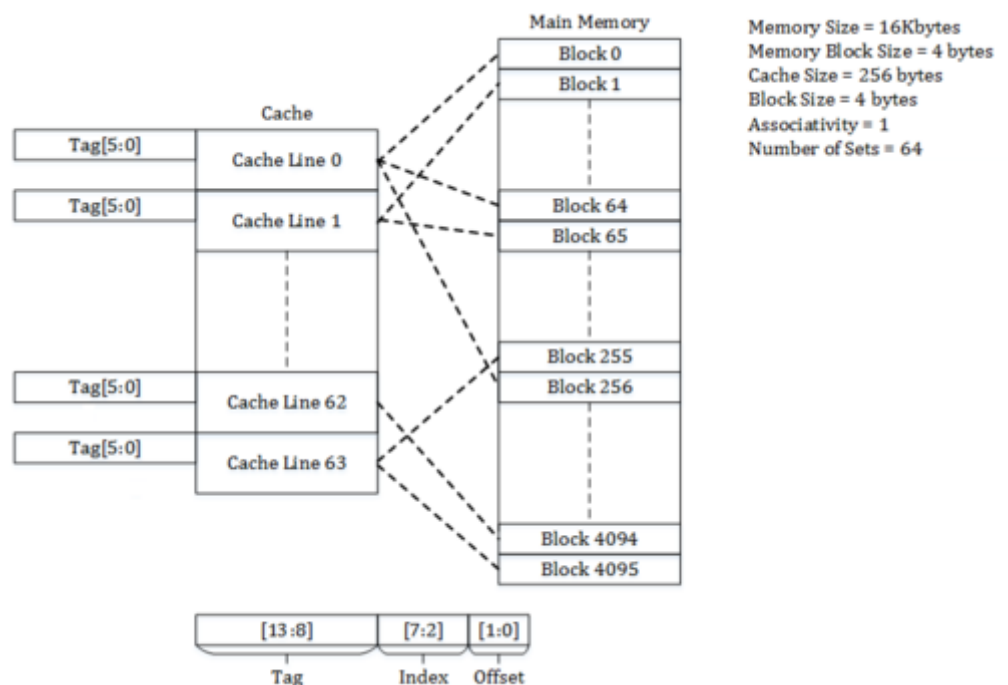
Example

Consider Main memory of 16 Kilobytes, which is organized as 4-byte blocks and Cache of 256 bytes with block size of 4 bytes.

Since each cache block is of size 4 bytes, the total number of sets in the cache is $256/4$, which equals 64 sets.

The incoming address to the cache is divided into bits for Offset, Index and Tag.

Offset corresponds to the bits used to determine the byte to be accessed from the cache line.



Direct-Mapped Cache

In the example, the offset bits are 2 which are used to address the 4 bytes of the cache line.

Index corresponds to bits used to determine the set of the Cache.

In the example, the index bits are 6 which are used to address the 64 sets of the cache.

Tag corresponds to the remaining bits.

In the example, the tag bits are 6 ($14 - (6+2)$), which are stored in tag field to match the address on cache request.

Address 0x0000 (tag - 00_0000, index - 00_0000, offset - 00) maps to block 0 of the memory and occupies the set 0 of the cache.

Address 0x0004 (tag - 00_0000, index - 00_0001, offset - 00) maps to block 1 of the memory and occupies the set 1 of the cache.

Similarly, address 0x00FF(tag – 00_0000, index – 11_1111, offset – 11) maps to block 63 of the memory and occupies the set 63 of the cache.

Address 0x0100(tag – 00_0001, index – 00_0000, offset – 00) maps to block 64 of the memory and occupies the set 0 of the cache.

Fully Associative Cache

In a Fully associative cache, the cache is organized into a single cache set with multiple cache lines. A memory block can occupy any of the cache lines. The cache organization can be framed as (1*m) row matrix.^[4]

To place a block in the cache

- The cache line is selected based on the valid bit^[1] associated with it. If the valid bit is 0, the new memory block can be placed in the cache line, else it has to be placed in an other cache line with valid bit 0.
- If the cache is completely occupied then a block is evicted and the memory block is placed in that cache line.
- The eviction of memory block from the cache is decided by the replacement policy.^[6]

To search a word in the cache

- The Tag field of the memory address is compared with tag bits associated with all the cache lines. If it matches, the block is present in the cache and is a cache hit. If it doesn't match, then it's a cache miss and has to be fetched from the lower memory.
- Based on the Offset, a byte is selected and returned to the processor.

Advantages

- Fully associative cache structure provides us the flexibility of placing memory block in any of the cache lines and hence full utilization of the cache.
- The placement policy provides better cache hit rate.
- It offers the flexibility of utilizing a wide variety of replacements algorithms if a cache miss occurs.

Disadvantage

- The placement policy is slow as it takes time to iterate through all the lines.
- The placement policy is power hungry as it has to iterate over entire cache set to locate a block.
- The most expensive of all methods, due to the high cost of associative-comparison hardware.

Example

Consider Main memory of 16 Kilobytes, which is organized as 4-byte blocks and Cache of 256 bytes and block size of 4 bytes.

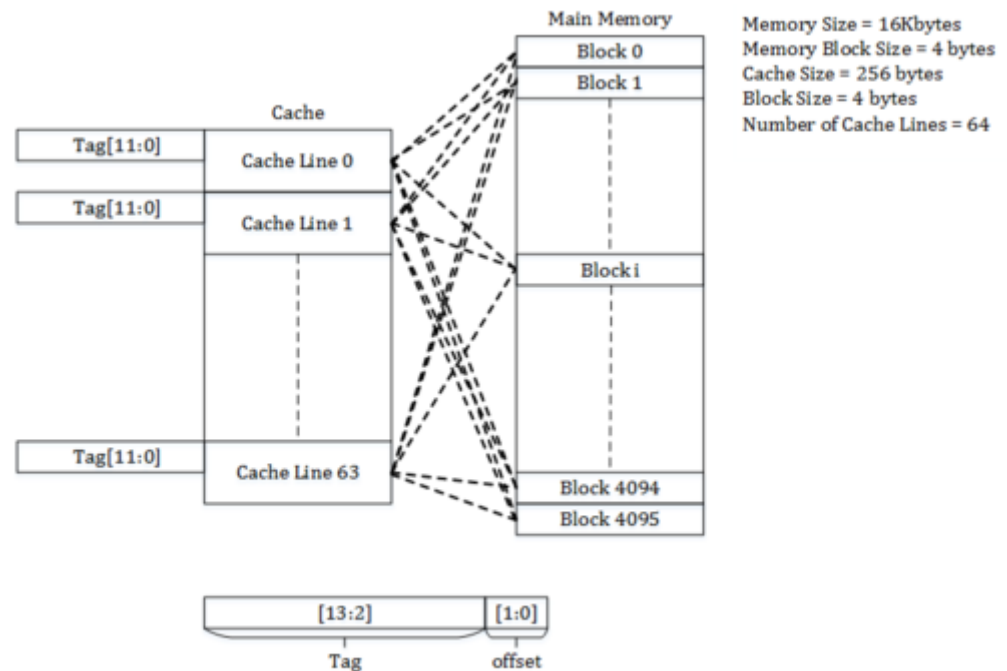
Since each cache block is of size 4 bytes, the total number of sets in the cache is 256/4, which equals 64 sets or cache lines.

The incoming address to the cache is divided into bits for offset and tag.

Offset corresponds to the bits used to determine the byte to be accessed from the cache line.

In the example, the offset bits are 2 which are used to address the 4 bytes of the cache line and the remaining bits form the tag.

In the example, the tag bits are 12 ($14 - 2$), which are stored in the tag field of the cache line to match the address on cache request.



Fully-Associative Cache

Since any block of memory can be mapped to any cache line, the memory block can occupy one of the cache lines based on the replacement policy.

Set Associative Cache

Set associative cache is a trade-off between Direct mapped cache and Fully associative cache.

The Set associative cache can be imagined as a $(n \times m)$ matrix. The cache is divided into 'n' sets and each set contains 'm' cache lines. A memory block is first mapped onto a set and then placed into any cache line of the set.

The range of caches from direct mapped to fully associative is a continuum of levels of set associativity. (Direct mapped is one-way set associative and Fully associative cache with m blocks is m -way set associative.)

Many processor caches in today's design are either direct mapped, two-way set associative, or four-way set associative.^[4]

To place a block in the cache

- The set is determined by the index bits derived from the address of the memory block.
- The memory block is placed in the set identified and the tag is stored in the tag field associated with the set.
- If the cache line is occupied, then the new data replaces the cache block identified with the help of replacement policy.

To locate a word in the cache

- The set is determined by the index bits derived from the address of the memory block.
- The tag bits are compared with the tag of all cache lines present in selected set. If the tag matches, then it's a cache hit and appropriate byte is fetched and delivered to the processor. If the tag doesn't match, then it's a cache miss and is fetched from the lower memory.

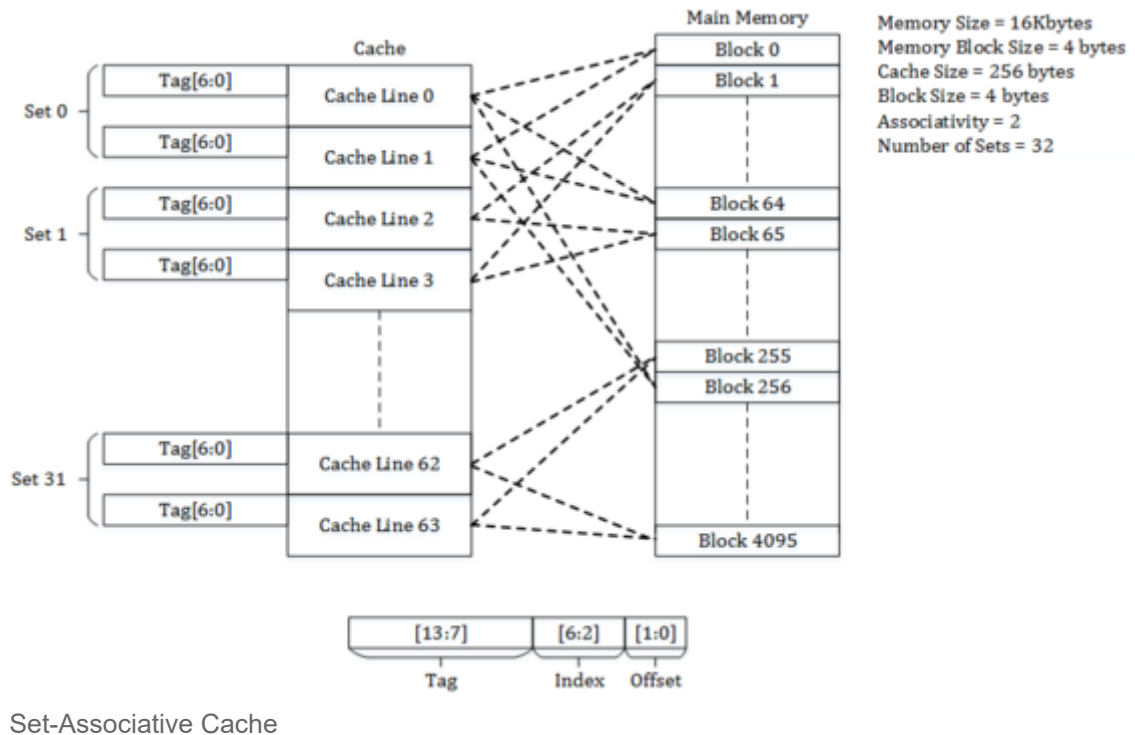
Advantages

- The placement policy is a trade-off between direct mapped and fully associative cache.
- It offers the flexibility of using replacements algorithms if a cache miss occurs.

Disadvantages

- The placement policy will not effectively use all the available cache lines in the cache and suffers from conflict miss.

Example



Consider Main memory of 16 Kilobytes, which is organized as 4-byte blocks and Cache of 256 bytes with block size of 4 bytes and 2-way set associative.

Since each cache block is of size 4 bytes, the total number of sets in the cache is $256/4$, which equals 64 sets or cache lines.

In the example, the offset bits are 2 which are used to address the 4 bytes of the cache line, the index bits are 5 which are used to address the 32 lines of the cache and the tag bits are 7 ($14 - (5+2)$), which are stored in tag to match the address on cache request.

Address 0x0000 (tag – 000_0000, index – 0_0000, offset – 00) maps to block 0 of the memory and occupies the set 0 of the cache. The block occupies one of the cache lines of the set 0 and is determined by the replacement policy for the cache.

Address 0x0004 (tag – 000_0000, index – 0_0001, offset – 00) maps to block 1 of the memory and occupies one of the cache lines of the set 1 of the cache.

Similarly, address 0x00FF (tag – 000_0001, index – 1_1111, offset – 11) maps to block 63 of the memory and occupies one of the cache lines of the set 31 of the cache.

Address 0x0100 (tag – 000_0010, index – 0_0000, offset – 00) maps to block 64 of the memory and occupies one of the cache lines of the set 0 of the cache.

See also

- [Associativity](#)
- [Cache replacement policy](#)
- [Cache hierarchy](#)
- [Writing Policies](#)
- [Cache coloring](#)

References

1. "The Basics of Cache" (<https://cseweb.ucsd.edu/classes/su07/cse141/cache-handout.pdf>) (PDF).
2. "Cache Placement Policies" (http://web.cs.iastate.edu/~prabhu/Tutorial/CACHE/bl_place.html).
3. "Placement Policies" (http://fourier.eng.hmc.edu/e85_old/lectures/memory/node4.html).
4. Solihin, Yan (2015). *Fundamentals of Parallel Multi-core Architecture*. Taylor & Francis. pp. 136–141. [ISBN 978-1482211184](#).
5. "Cache Miss Types" (<http://meseec.ce.rit.edu/eccc551-winter2001/551-1-30-2002.pdf>) (PDF).
6. "Fully Associative Cache" (<https://www.cs.umd.edu/class/sum2003/cmsc311/Notes/Memory/fully.html>).

Retrieved from "https://en.wikipedia.org/w/index.php?title=Cache_Placement_Policies&oldid=751140579"

This page was last edited on 23 November 2016, at 16:55.

Text is available under the [Creative Commons Attribution-ShareAlike License](#); additional terms may apply. By using this site, you agree to the [Terms of Use](#) and [Privacy Policy](#). Wikipedia® is a registered trademark of the [Wikimedia Foundation, Inc.](#), a non-profit organization.