# CACHE MEMORY

Yashika Arora, Tanushree, Shilpa Yadav
*Computer Science, Maharishi Dayanand University*

***Abstract*— CPU cache is a cache which used by the central processing unit of computer to decrease the average time which is taken to access the data from main memory. The cache is smaller and faster memory which stores the copies of data from commonly used main memory locations. Most of the CPUs have dissimilar independent caches, which includes the instructions and data caches where data cache is generally organized as hierarchy of more cache levels.**

***Index Terms*—Cache,Heuristic,Multiprocessors,Write-through**

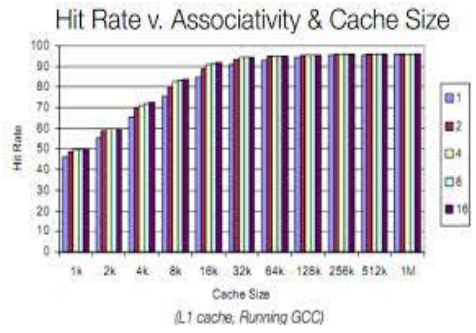
## I. INTRODUCTION

CPU cache is a cache which used by the central processing unit of computer to decrease the average time which is taken to access the data from main memory. The cache is smaller and faster memory which stores the copies of data from commonly used main memory locations. Most of the CPUs have dissimilar independent caches, which includes the instructions and data caches where data cache is generally organized as hierarchy of more cache levels.When processor requires reading from or writing to a site in the main memory. It first examines in case a copy of that particular data is in cache. If so, processor then instantly reads from or writes to cache, and this is much faster than writing to or reading from the main memoryMany modern desktops and server CPUs have somewhat three independent caches: one is instruction cache which is used to speed up executable instruction fetch, data cache is used to speed up data fetching and storing of data, and translation look aside buffer i.e. TLB is used for speeding up virtual-to-physical address translation for data and executable instructions.

## II. CACHE ENTRIES

The data is shifted between the memory and cache in the blocks which are of fixed size, and are known as cache lines. When cache line is copied from the memory into cache, then a cache entry is made. The cache entry includes the data which is copied as well as the memory location which is requested and it is now called a tag.When processor wants to read or write to a location in the main memory then it first inquires for an equivalent entry in cache. The cache inquires for contents of requested memory location in any of the cache lines which might contain that particular address. If processor discovers that memory location is in the cache then a cache hit occurs. On the other hand, if the processor does not find memory location in cache then a cache miss occurs. In case of:

- a cache hit, processor instantly reads or writes data in cache line
- a cache miss, cache assigns a new entry, and copies the data from the main memory; and then the appeal is fulfilled from the contents of cache.




## III. REPLACEMENT POLICIES

In order to make room for new entries on cache miss, cache may have to expel one of those current entries. The heuristic that is used to choose the entry to expel and this is called replacement policy. The basic problem with any of the replacement policy is that: it must forsee which of the existing cache entry will be used least in future. Predicting future is hard to do, so there is no excellent way to choose between the replacement policies which are available.One of the popular replacement policies is the least-recently used or LRU which replaces the entry which is least recently accessed.

Marking of some of the memory ranges as non-cacheable memory can improve the performance, by preventing caching of the memory regions which are hardly re-accessed. This prevents the overhead of loading into cache, without any reuse.

- The entries of cache could also be locked or disabled depending on the circumstances.

## IV. WRITE POLICIES

If the data is written to cache then at some point it should also be written to the main memory. The timing which is used in this write is known as write policy.

- In write-through cache, every write which is given to cache causes write to the main memory.

Rather, in write-back or copy-back cache, write is not instantly mirrored to main memory. Alternatively, cache tracks the locations which have been written over and then these locations are marked as dirty. Data in these dirty locations are written back to main memory only when the data which was present is evicted from cache. For this particular

reason, read miss in write-back cache sometimes might require two memory accesses to provide the service:

- one is to the first write the dirty location to the memory and then the another one to read a new location from the memory.

There are some intermediate policies also. The cache could be write-through, but the write might be held in store data queue for some limited time, usually so that numerous stores can be processed together and this reduces bus turnarounds and also improve utilization of bus.

Data in the main memory which is being cached can be changed by the other entities like peripherals using multi-core processor or direct memory access, in which case copy in cache might become out-of-date or we can say, stale. Rather, when a CPU in the multiprocessor system updates the data in cache, copies of the data in the cache is correlated with other CPUs that will become stale. The communication protocols between cache managers which keeps the data consistent are called as cache coherence protocols.

## V. Conclusion

This paper forms part of the guideline for future work for researchers interested in optimization of memory hierarchy for scalable multi core processors, as it presents a survey of all such techniques proposed in recent publications. The techniques are also presented along with the comments about their effectiveness.. The effect of the mechanisms and policies of operating system on the memory hierarchy, especially the on-chip cache hierarchy is another direction of research that can be explored. High coherence traffic gives rise to congestion at the first level cache. Directory-based coherence protocols may reduce the overall coherence traffic but this comes with the cost of maintaining the directory and keeping it updated. These and other research directions shall be explored in future research.

## References

[1] Rolán, D., B. Fraguela, R. Doallo, (2009), Adaptive Line Placement
[2] with the Set balancing Cache, 42nd Annual IEEE/ACM International
[3] Symposium on Micro-architecture (MICRO-42), p. 529-540
[4] Salapura, Blumrich, Gara, (2008), Design and Implementation of Blue
[5] Gene/P Snoop Filter, Proceedings of the 14th International Symposium
[6] on High Performance Computer Architecture (HPCA), p.5-14
[7] Sun, G., X. Dong, Y. Xie, J. Li, (2009), A Novel Architecture of the
[8] 3D Stacked MRAM L2 Cache for CMPs, IEEE 15th International
[9] Symposium on High Performance Computer Architecture (HPCA), p.

**First Author** personal profile which contains their education details, their publications, research work, membership, achievements, with photo that will be maximum 200-400 words.

**Tanushree**, Pursuing Btech in computer science , published Five research paper in different journalsand having membership of various societies like CSI,ISOC,UNICEF,ISTE, etc,

**Third Author** personal profile which contains their education details, their publications, research work, membership, achievements, with photo that will be maximum 200-400 words.