VOL. 10, NO. 7, APRIL 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences


www.arpnjournals.com

# SURVEY ON CACHE MEMORY DESIGN TECHNIQUES FOR LOW POWER HIGH PERFORMANCE PROCESSOR

R. Ramya, T. Ravi and M. Manoranjani
VLSI Design, Department of Electronics and Communication Engineering, Sathyabama University, Chennai, India
E-Mail: ramesh.ramya3@gmail.com

**ABSTRACT**

Cache memory is an additional and fast memory unit that has to be placed between the processing unit and the physical memory. The most recently used instructions and data, where this information are needed to be accessed again are stored in cache .The use of cache memory makes the processing to be faster. As the cache memory moves away from the CPU, the access time and the size of the cache memory storage unit increase as well. The physical memory and external disk storage devices can be accessed faster by the internal registers and cache which are located near to CPU. Cache which are accessed faster and also with less miss rate can be considered to be more power efficient. On-chip caches which are of large size are used in order to overcome cache miss are increasingly used by the modern processors. Also, with each CMOS technology generation, the leakage power consumption is said to be increasing for the past few decade. For this reason, cache power management has become a very important scenario in modern processor design. To address this challenge and also sustainable computing goals are met by several energy efficient techniques that are proposed for the cache architecture. The static and dynamic power consumption will lead to the total power consumption. The power consumption of set associative cache due to access of every cache is more when compared to the direct mapping method; this includes both the tag and the data parameter of the cache. This paper presents a review on different cache technique and the cache optimization parameters and there are different low power techniques for low power cache.

**Keywords:** cache miss, cache hit, conflict miss, cache energy saving techniques, dynamic energy, leakage energy, power management, low power cache.

## 1. INTRODUCTION

Today computer each have a small amount of very high speed memory so called Cache memory, where data from memory locations that are used more frequently may not be permanently stored. This small sized cache is placed near to a large sized main memory, this memory is memory system. This memory is connected by a very large memory system. This memory is connected by a very large disk and various removable media. Cache memory is intended to give memory speed approaching that of the fastest memories that are used and at the same period provide less expensive and types of semiconductor memories which are of large memory size. There is correspondingly main memory which is large but slow together with a smaller as well faster cache memory. The cache memory contains a copy of instruction from main memory. The processor when it needs to read from or write to in the main memory locations, it first checks whether the cache memory contains the required data .If so, the processor will reads from or writes immediately to the cache, which are much smaller and faster than reading from or writing to main memory. Three independent caches are used in the modern desktop and server CPUs, such as an instruction cache to speed up executable fetch from instruction, a data cache is used to speed up data fetch and store, transition look aside table used to speed up virtual-to-physical address translation for both executable instructions and data. The use of two levels of memory to reduce average access time works in principle, during the course of execution of a program. A logical cache is also known as a virtual cache in which virtual address are used to store the data required. The processor accesses the cache directly, without going through the MMU. A physical cache stores data using main memory physical addresses. One obvious advantage of the logical cache is that cache access speed is faster than a physical cache, because this cache memory will respond before the MMU performs an address translation. The disadvantage is that the same virtual address in two different applications refers to two different physical addresses.

Cache systems are on-chip memory element used to store data. A cache controller is used for tracking induced miss rate that occurred in cache memory. The data requested by microprocessor is found in cache memory then the term is called "cache hit". The most important advantage of storing data on cache, as compared to RAM, is that it has faster retrieval times, but energy consumption is one of the drawback of on-chip. In fact, Rogers et al., [1] have shown that due to bandwidth-limitations, SRAM-based caches may occupy 90% of the chip-area in upcoming fourth CMOS generation. Thus, With each CMOS (complementary metal oxide semiconductor) technology generation will lead to considerable amount of leakage power [2], [3], due to increasing size of LLCs, along with large leakage energy consumption of SRAM devices, the energy consumption of LLCs is becoming a significant fraction of processor energy consumption [4]. SMART technique uses cache reconfiguration to dynamically tune cache active fraction to reduce leakage energy consumption, while choosing reasonably small write latency [5]. Although the techniques designed to improve performance are also likely to conserve energy, in this survey we focused on only those techniques which aim to optimize energy efficiency and have been shown to


improve energy efficiency. In this paper we have 3 sections, Section 1 briefly describes the different optimization parameters that cache consist of during cache design. Section 2 describes about the power efficient cache which leads to low power cache and Section 3 describes different cache mapping techniques.

## 2. DESIGN PARAMETERS OF CACHE MEMORY

### 1) Cache hits

When the particular information that is wanted by the processor and it is found to be in cache. It was storing the item and it is able to satisfy the query , the transaction is said to be cache hit.

### 2) Cache miss

When the information requested by the processor is not found in the cache memory, the transaction is said to be a cache miss.

### Types of Cache miss

### 2(a) Compulsory miss

A Compulsory caches miss refers to, that the very primary data access to the needed block results in a miss. Under this type of cache miss category, the block that is needed must be bought into the cache from main memory. This type of miss in cache is otherwise known as cold start miss or first reference miss.

### 2(b) Capacity miss

During the execution of a large program or process, it may be impossible for the cache to contain all the blocks needed during the execution. Therefore, a capacity miss occurs when the needed block has been discarded before and now must be placed back in the cache since it is again needed. Capacity miss can be reduced by Compressing the data stored in the cache can increase the effective size of the cache [7], but this increases latency as cache blocks have to be decompressed when needed. Compression and decompression also requires additional logic and increases complexity.

### 2(c) Conflict miss

Conflict misses only occur in set-associative or direct-mapped caches, since these are the only cache architectures in which a block is discarded in order to make room for another set of block .The most important strategy such as set-associative or direct-mapped placement is used, conflict misses, in addition to that compulsory and capacity misses it will also occur when the set is mapped by too many blocks. Conflict misses are otherwise known as collision misses. These misses can be reduced by Associatively can be increased in order to reduce the number of conflict misses and however, the associativity is already quite high in modern last-level caches and increasing it further will have diminishing results on reducing the number of misses while increasing complexity, latency, chip area and power consumption. Using victim caches is one inexpensive way to reduce the number of conflict misses without increasing the associativity [8]. The victim cache holds recently evicted lines, typically less than the last six evicted and is fully associative. This improves performance of a direct mapped first-level cache, which are less suitable for direct mapped last-level caches.

### 2(d) Cache consistency

Since cache is a copy of a small piece of main memory, it is important that the cache always reflects what is in main memory. Some common terms used to describe the process of maintaining a cache consistency are: 1) Snoop, When a cache is watching the address lines for transaction, this is called a snoop. This function allows the cache to see if any transactions are accessing memory it contains within itself. 2) Snarf, when a cache takes the information from the data lines, the cache is said to have snarfed the data. This function allows the cache to be updated and consistency can be maintained. Snoop and snarf are the important mechanisms that the cache can use to maintain consistency. The other terms that are most commonly used to describe the inconsistencies in the cache data, these terms are:

- Dirty Data

When data is modified within cache but not modified or updated in main memory, the data in the cache is called "dirty data."

- Stale Data

When data is updated or modified within main memory but not updated in cache, the data in the cache is called stale data.

### 2(e) Coherence miss

In a system with multiple cores, one processor can invalidate cache blocks in private caches for other processors through the coherence protocol and hence later cause cache misses. Coherence misses are caused by parallel programs that share and use a write invalidate protocol and modify the same data structures. If any one processor modifies a cache block that is also present in some other private cache, where these data are invalidated in some other processor(s) cache. There are different ways to reduce the number of misses caused by this invalidation such as changing the hardware scheme to update the data rather than invalidate it. The programmer or compiler can also reduce the number of misses by improving how the processors share the data [9].

The number of cold misses can be reduced by prefetching data. Prefetching can be done by software or hardware and reduces the number of cold misses The simplest prefetching method is sequential, when a cache block is requested, the following cache block is also needed to be fetched. More advanced prefetching methods store information about memory access pattern in dedicated hardware tables [10] in order to prefetch more complex access patterns. Prefetching can be done by


software controlled. In this type the program provides a special instruction which causes prefetching [11].

**Methods to overcome cache miss**

- Miss cache
- Victim cache
- Stream buffer

**Miss cache** aims to reduce conflict misses where Fully associative cache is inserted between level1 and level2 cache.

**Victim cache** is used to overcome the wasted space in the pure miss caching. We replace that with victim cache, it is alternate to 2-way set associative cache, it provides better space utilization where it can be used for the same size cache.

**Stream buffers** can be used when a miss occurs in L1 at address A, the stream buffer immediately start to prefetch elements at A+1.The subsequent access check the head of stream buffer before going to L2.stream buffer is a FIFO queue placed in between L1 and L2 cache. The performance is good in stream buffer where 72% of instruction misses are removed and 25% of data misses are removed. The drawback is, non sequential misses will cause the line to restart prefetching.

**3) Misspenality:** Miss penality refers to when the data is not found in cache memory, the processor loads data from main memory and copies into cache. This results in extra delay in the processing unit.

## 3. LOW POWER CACHE

The power is a function of both dynamic and static power. The power consumption of a set-associative cache tends to be higher than that of a direct mapped cache, due to the activation of available cache ways upon every cache access. This includes the simultaneous activation of both the tag and the data parts of the cache. Power dissipation in caches has been heavily researched and several proposals made to reduce both dynamic and static power in caches.

Total Power = Power Dynamic + Power Static

### 3(a) Static power

Static power, also commonly known as leakage power, is due to the leakage current that flows even when the transistor is turned-off. Static power is proportional to the transistor count. It is calculated every cycle. The following formula gives the static power,

Power Static=Voltage × N × k× Current Static

N is the number of transistors, k is a design dependent parameter. Hence, static power is proportional to the number of transistors and it decreases the cache efficiency as there is an increase in size of the transistor.

- **Drowsy cache,** [12] keeps the cache line in a low power state, known as the drowsy state. When a cache line is accessed, its drain voltage V*dd*should is high or at value1 for the contents to be preserved. By lowering the V*dd*to a predetermined threshold value say for example 0.7V*dd*, the contents are still preserved. When V*dd*is further lowered beyond the threshold value, the cache contents are lost. The key challenge in designing the drowsy cache is to determine the threshold value. This is the value at which all the cache lines, irrespective of their irregularities in fabrication, should able to preserve the cache contents, while being in the lowest V*dd*for the maximum static energy reduction. When a cache line is in the drowsy state, its contents cannot be read/write, it requires one cycle to activate or to bring the cache lines to V*dd*high, before the read/written action. Thus it incurs one cycle extra hit latency for activating and then accessing the cache line. First level caches have short hit time constraints. Hence drowsy caches can be implemented in the lower level caches, whose hit time is not in the critical path and also these lower level caches are larger than the first level caches, providing more room for energy saving whilst maintaining high performance. As the CMOS technology advances, static energy will be more dominant than dynamic energy, making drowsy caches increasingly relevant.
- **Gated-Vddtechnique**, drains out the V*dd* completely, making the cache lines lose their contents. This scheme operates at 0V*dd* compared to the drowsy scheme, offering more static energy reductions. The hardware overhead increases, with a gating transistor added for every cache line. Thus connecting several cache lines to the gating transistor, simplifies the power gating scheme. Powering down the cache lines can be carried out at a bank level, rather than at line granularity, making the scheme more coarse grain. The power down signal can be connected to all the gated V*dd*transistors, which when enabled drains out the power completely in a bank, resulting in turning-off the entire bank. Gated V*dd*can be combined with the selective cache ways to turn-off the disabled cache ways, offering both dynamic and static energy savings. The cache decay which is a state-destroying low power scheme. The upper limits of reducing leakage power by combining both drowsy and gated-Vdd techniques, this work is only a theoretical upper bound since it assumes the existence of an ideal pre-fetcher which is impossible to provide in practice. A gated-ground scheme for turning off cache lines whilst preserving their contents, these kind of circuits require a single supply voltage [13].

### 3(b) Dynamic power

Dynamic power is due to the switching activity that takes place in transistors. It is proportional to the product of the number of switching transistors and the switching rate. It is calculated for every cache access.

Pd= 0.5×CapacitiveLoad×Voltage2×Frequency switched

Dynamic power is proportional to the product of load capacitance of the transistor, the square of voltage and the frequency of switching. Thus, reducing the switching activity reduces the dynamic power consumption.

- **Filter cache,** is a small direct mapped cache, typically a level-0 cache, present before the level-1 cache. As the name implies, it filters out the frequently accessed data present in level-1 cache. If most of the level-1 requests are serviced by the filter cache, then most of the level-1 cache switching activity can be greatly reduced. Thus, it provides a fast hit time and also reduces significant dynamic energy consumption of the level-1 caches. The Data part of the cache accounts for more size in bytes compared to the tag part of the cache. Thus, it consumes more dynamic energy compared to the tag access [14, 16].
- **phased cache**, divides the cache access into two phases. First, all the tags in the set are examined in parallel and no data accesses occur during this phase. Second, if there is a hit, then a data access is performed for the hit way. Hence a phased lookup will reduce the dynamic power on every cache access. For the first level cache, whose hit time is in the critical path, this will incur an additional delay of one cycle hit latency. Therefore the scheme is beneficial in lower level caches but not in first level cache, where a cache hit time is in the critical path. Hence this scheme level cache, where a cache hit time is in the critical path. Hence this scheme is used to reduce the dynamic energy in lower level caches. Way-Prediction ,on the other hand, reduces dynamic energy by predicting exactly one way on every cache access. On a prediction hit, only one way is accessed, resulting in a fast tag and data look-up along with the reduction in dynamic energy. On a prediction miss, the cache access time is increased by one cycle latency, asin the next cycle it needs to check all the tags and data. Suppose if there is a cache hit occurs then request is serviced immediately, if not the next lower level cache is accessed. Thus, during a prediction miss, it behaves like a normal set associative cache. Way prediction is beneficial when the cache access is predictable, such as in instruction cache. For the data cache and the next lower level cache, whose cache access is irregular and un predict able, way prediction incurs more latency for miss prediction. Since the instruction cache is accessed more frequently, more dynamic energy saving is achieved. Way guarding, uses counting bloom filters to determine whether an incoming address is present in the cache or not. If there is a hit in the entries of the counting bloom filters, then several cache ways that might contain the requested address are enabled simultaneously. If there is a miss in the entries, then it is guaranteed that the requested address will not be present in the cache. This has an advantage over way prediction, as it does not incur extra look-ups on a miss. On a hit, way guarding enables more cache ways than way prediction [15, 17].Recently 3D die stacking technology promises lower power consumption of the interconnect, chip with smaller size and increasing bandwidth of on-chip and for these consideration, this recent technology is already has its footprint on commercial adoption .However, 3D die stacking technology also presents significant power and thermal challenges.

**3(c) Leakage power**

Leakage energy saving approaches work by turning off a part of the cache to reduce the cache leakage energy consumption. Based on the data retention capacity of the blocks that are turned off, the techniques used for the leakage energy saving are classified into two categories, which are called as state-preserving and state-destroying techniques. circuit-level mechanisms have been used by the architectural techniques for leakage control in both state-preserving and state-destroying method [4].

- **gated Vdd:** which facilitates leakage control by state-destroying method. In this technique an additional transistor is used in the supply voltage path or ground path of the SRAM (static random access memory) cell. The leakage energy of the SRAM cell is reduced by turning off the additional transistor and also by stacking effect of the additional transistor, the leakage current can be reduced to orders of magnitude.
- **drowsy-cache:** which facilitates leakage control by state preserving method. In this technique the cache can make use of two different voltage supplies, one among them is of low voltage and the other supply voltage is high. The leakage energy of the SRAM cell can be reduced by the cache controller, It just provide low voltage as operating voltage to the cell, thus low-leakage mode is maintained in the cell. During next time when the same line is being accessed, the low supply voltage is changed to high supply voltage, by which normal power can be used by the cache block.
- **super-drowsy, gated-ground:** both of which behave similar to the method called drowsy cache, but the difference is that these two methods need single voltage supply. The other methods of state-preserving circuit design, named multi threshold CMOS dynamically changes the threshold voltage of the SRAM cell by modulating the back gate bias voltage to transition the cell to low-leakage mode**.**

## 4. CACHE MAPPING TECHNIQUES

**4(a) Direct mapping**

In a direct mapped cache, each block has only one place that it can go. Thus when the CPU needs a certain block from the main memory, the cache memory is the only one place that it could possibly reside. The needed block can also be fetched from the lower level of memory if it is not there in the cache memory. In order to find where the block resides in the cache, the block frame address is divided by the number of blocks in the cache. Thus direct mapping is used to access the data at faster rate .

**Advantages**

- trivial mapping function.
- cache line immediately available.
- faster access.
- optimum case: sequential processing.


**Disadvantages**

- If cache not yet full line may be replaced and replacing scheme may be inadequate.
- worst case: referencing always contains same line number.

**4(b) Fully associative mapping**

In a fully associative cache, the block can be placed anywhere in the cache since there are no restrictions as to where it has to be placed. When the CPU needs a certain memory block, the cache mustbe checked for each block that resides in it, to determine if the required information is present in the cache. A block is only evicted from a fully associative cache if the cache is full.

**Advantages**

- trivial mapping function.
- Only if the cache is full then the line can be replaced
- optimum case: repeated wide-spread access.

**Disadvantages**

- cache line has to be searched.
- extended tag field (all bits).
- additional counter for LRU.
- replacing scheme may be inadequate.
- worst case: The area that are slightly larger than cache are done sequentially.

**4(c) Set associative mapping**

In a set associative cache, certain set of places are allocated for each memory block. A set consists of a group of two or more blocks in the cache. When a block is placed into cache memory from the main memory, first the block is mapped to a set, and within that set the block is free to go anywhere. This method of set associative mapping combines the likes of both direct mapped and fully associative mapping in that the block is directly mapped to a set, and then is fully associative within that set. In order to determine the set that the block should be placed in, the block frame address is divided (modulo) by the number of sets that are in the cache. A cache is said to be n-way set associative if there are n blocks in each set.

**Advantage**

- Cheaper than fully associative cache.
- Lower miss ratio is achieved when compared to direct mapped cache.

**Other Cache technique**

- Blocking Cache
- Non-blocking Cache
- Victim Cache
- Sub block placement
- Pseudo-set Associative cache
- Pipelined cache Access
- Trace cache

## CONCLUSIONS

Caches can be implemented in different ways, but the basic concepts behind the cache technique remain same. The challenge in cache design is to ensure that the desired data and instructions are in the cache. The cache should achieve a high hit ratio. The cache system must be quickly searchable as it is checked every memory reference. Total power consumption of any processors is increasing tremendously and it reaches the “power wall” imposed by thermal limitations of cooling solutions and power delivery. Thus, using technological scaling higher performance can be achieved and managing the power consumption of processors has become a vital necessity. In this paper, we have reviewed different techniques proposed for managing low power cache, different parameters of cache that has to be considered during design and different mapping techniques.

## REFERENCES

[1] B.M Rogers and A. Krishna. *et al.* 2009. Scaling the bandwidth wall: challenges in and avenues for CMP scaling”, ACM SIGARCH computer Architecture, Journal article. Vol. 37, pp. 371-382.

[2] S. Borkar. *et al.* 1999. Design challenges of technology scaling”, journal article, in sustainable computing: informatics and systems.

[3] G. Gammie and A. Wang *et al.* 2010. ”Smartreflex power and performance management technologies for 90nm, 65nm, 45nm mobile Application processors”, IEEE.

[4] Sparsh Mittal. 2013. A Survey of Architectural techniques for improving cache power efficiency”, Elsevier sustainable computing: Informatics and systems. Vol. 4 no. 1, pp. 33-43.

[5] Sparsh Mittal,” A cache energy optimization technique for STT-RAM last level cache”, 2207v1,CS.AR, 2013.

[6] ching-Long su, and Alvin M. Despain. Cache design Trade-offs for power and performance optimization” in ISLPED’95 Proceedings of international symposium on low power, pp. 63-68, ISBN: 0-89791-7448.

[7] H. Dybdahl *et al*. 2005. Destructive read in embedded DRAM impact on power consumption “, Journal of embedded computing.

[8] Norman P. Jouppi. 1990. Improving direct mapped cache performance by the addition of a small fully associative cache and prefetch buffers.

[9] Bradford M. Beckmann and David A. 2004. wood," Managing wire delay in large chip multiprocessor caches", International symposium Microarchitecture.

[10] C. L. Hwang and T. Kirihata *et al.* 2002. "A 2.9ns random Access cycle embedded DRAM with a destructive read Architecture", IEEE symposium, pp 174-175 ISBN: 0-7803-7310-3.

[11] G. Kirsch. 2003. Parallel and distributed processing", on International symposium, ISBN: 0-7695-0990-8

.

[12] Krisztian flaunter, nam sung Kim *et al.* 2002. Drowsycaches: simple techniques for reducing leakage power", International symposium on computer Architecture.

[13] Michael powell *et al.* 2000. Gatedvdd:a circuit technique to reduce leakage in deep-submicron cache memories", International Symposium on low power electronics and design, pp. 90-95,ISBN: 1-58113-190-9.

[14] Ravi T. and Kannan V. 2012. Design and Analysis of Low power CNTFET TSPC D Flip-Flop based shift Registers", Applied Mechanics and Materials journal, pp. 1651-1655, ISBN: 229-231.

[15] Ravi Thiyagarajan and Kannan Veerappan. 2013. "Ultra Low Power Single Edge Triggered Delay Flip Flop Based Shift Registers using 10-Nanometer Carbon Nano Tube Field Effect Transistor", American Journal of Applied Sciences, October, Vol.10, Issue 12, pp.1509-1520.

[16] Johnson Kin and Munisha Gupta *et al.* 1997. The filter cache: an energy efficient memory structure", International symposium on Microarchitecture.

[17] Atsushi Hasegawa and Ikuya Kawasaki *et al.* 1995. "High code density low power", IEEE Micro.