International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 2, Issue 1, pp 5-9, March-2018

Dynamic Architecture Way Filtering To Reduce Power Consumption on LLC

[1]R.Karthika, [2]A.Vivekaraj, [3]G.Kanagaraj

[1]PG Scholar, Department of VLSI, AVS Engineering College, Salem, Tamilnadu, India

[2,3]Assistant Professor, Department of ECE, AVS Engineering College, Salem, Tamilnadu, India

Abstract - Last level store (LLC) alludes to the largest amount reserve that is typically shared by all the utilitarian units on the chip (e.g. CPU centers, IGP, and DSP) The term can likewise be utilized as a part of conjunction with a framework whereby the LLC may speak to an alternate progressive level of reserve contingent upon the point of view the segment in setting. Consequently from the CPU center point of view, the LLC is adequately a L3 reserve while from the GPU viewpoint the LLC is a level 4 store. The common last-level reserve (LLC) is a standout amongst the most essential shared assets because of its effect on execution. For store touchy CPU applications, a diminished offer of the LLC could prompt critical execution corruption. To proposed plot is intended to be dynamic in enacting a suitable number of reserve courses with a specific end goal to dispense with the requirement for static profiling to decide a vitality upgraded store arrangement. The exploratory outcomes demonstrate that our proposed dynamic plan lessens the vitality utilization of LLCs by 34% and 40% on single-and double center frameworks, individually, contrasted and the best performing ordinary static reserve design. The proposed design of this paper investigation the rationale size, zone and power utilization utilizing Xilinx 14.2.

Keywords: LLC, Xilinx, Way Filtering, Dynamic Architecture, Chip memory.

I. INTRODUCTION

Last level reserve (LLC) alludes to the most abnormal amount store that is normally shared by all the practical units on the chip (e.g. CPU centers, IGP, and DSP). The term can likewise be utilized as a part of conjunction with a framework whereby the LLC may speak to an alternate progressive level of reserve contingent upon the viewpoint the segment in setting [1]. For instance, in a common standard Sky lake show, there are in an arrangement of individual centers alongside a coordinated illustrations unit. A way-sifting (WF) - based logical–cooperative LLC design to diminish the vitality utilization of LLCs. This engineering legitimately builds the associativity of LLCs when one to three store ways are enacted, and along these lines enhances execution and decreases vitality utilization [2]. Present day processors include a progressive system of reserves. "More elevated amount" stores, which are nearer to the processor center are littler however speedier than bring down level reserves, which are nearer to primary memory [3]-[5].

II.EXISTING SYSTEM

a) System Design

Many circuit-level techniques for reducing the leakage energy of the cache memory have been proposed. The gated-Vdd proposed by Powell et al. is used in many cache designs, but it incurs data loss. The drowsy cache scheme is proposed to reduce leakage energy in unused individual cache lines without data loss. The DRG-cache scheme is proposed to reduce leakage energy of a cache memory without data loss. it is difficult for these two techniques to be applied to real circuits due to process variations, which can make low-voltage SRAM cells faulty [6]. Many researchers tried to reduce the number of ways with little performance degradation. Determining how many cache ways are required is very important. Hwang and Li distinguished repeated and fresh misses to determine the appropriate associativity in a cache set [7]. A dynamic cache resizing technique for multi core systems has recently been proposed to implement the proposed cache size estimation scheme to compare it with our proposed cache architecture.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 2, Issue 1, pp 5-9, March-2018

The selective-sets scheme is proposed to mitigate performance degradation of selective cache ways. It partially turns OFF some cache sets while maintaining cache associatively; it suffers from the following drawbacks. First, its set mapping changes when cache upsizing occurs.

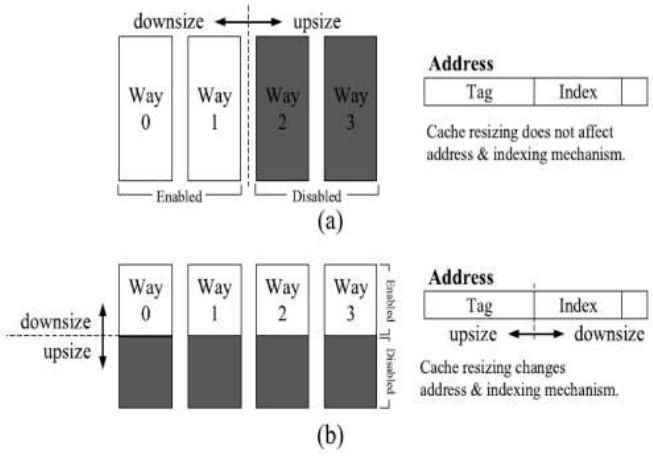

Figure-1: Conceptual diagrams of (a) selective cache ways and (b) selective cache

The selective-sets scheme is proposed to mitigate performance degradation of selective cache ways. It partially turns OFF some cache sets while maintaining cache associativity, it suffers from the following drawbacks. First, its set mapping changes when cache upsizing occurs. Cache indexing is based on modular hash operation. Thus, downsizing (which means decrease of the number of cache sets) does not change cache set mapping, but upsizing (which means increase of the number of cache sets) requires cache set remapping. To deal with this problem, this scheme flushes all cache sets when cache upsizing occurs [8]. This process causes large performance overhead because cold misses occur after flushing. Moreover, this process is not suitable for multi core systems because it flushes all sets, including the cache contents of the other core.

If the running programs of the other core have high temporal locality and/or deadline requirements, this process incurs critical performance degradation and/or deadline misses. In particular, in many core systems, this process causes catastrophic performance degradation because a flushing process caused by just one core affects the performance of all other cores [9]. Second, the implementation overhead of the selective-sets scheme is larger than that of the selective-ways scheme. Changing the number of cache sets also changes the number of tag bits this process causes large performance overhead because cold misses occur after flushing [10]. The selective-sets scheme must use a tag array that is as large as required by the smallest number of supported cache sets, i.e., the selective-sets scheme requires larger tags than the selective-ways scheme, which also means that tag access delay of selective-sets is longer than that of selective-ways.

b) Disadvantages

- Power consumption is high.

III.PROPOSED SYSTEM

a) Proposed Way-Filtering-Based Logical Associative In Cache Architecture

A set-associative scheme is a hybrid between a fully associative cache, and direct mapped cache. It's considered a reasonable compromise between the complex hardware needed for fully associative caches (which requires parallel searches of all slots), and the simplistic direct-mapped scheme, which may cause collisions of addresses to the same slot (similar to collisions in a hash table).

Let's assume, as we did for fully associate caches that we have:

- 128 slots
- 32 bytes per slot

Up to this point, cache has been this magical place that automatically stores data when we need it, perhaps fetching new data as the CPU requires it. However, a good question is what happens when the cache is full and the CPU is requesting additional data not in the cache. In this section, we'll take a look at the internal cache organization and try to answer these questions along with a few others.

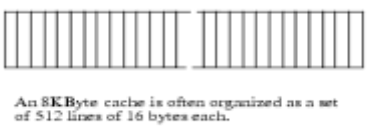

Figure-2: Cache memory

The cache lines are 16 bytes long are an example of so called cache line which holds blocks of 16 bytes whose

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 2, Issue 1, pp 5-9, March-2018

addresses fall in 16-byte boundaries out of main memory (i.e., the L.O. four bits of the address of the first byte in the cache line are always zero).

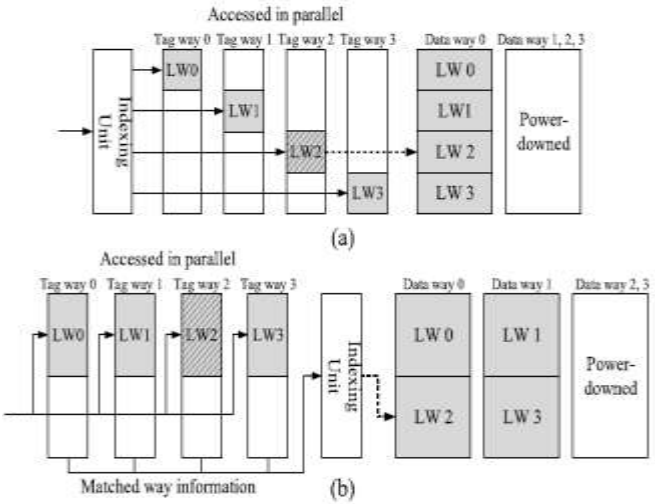

Figure-3: NF parallel logical–associative cache for a four-way cache. (a) And (b) All tag ways are accessed in parallel.

The solid arrows indicate tag way accesses and the dotted arrows indicate data way accesses when a tag hit occurs. LW stands for logical way. Then somehow the controller has to pick one of the other or main memory data that can go into either of both cache lines. The selection is trivial and pick up an unused cache line in either (or both) cache lines are currently left unused. If the cache controller must pick one of the cache lines has both cache lines that are currently in use then it replaces its data with a new data in addition. Ideally, we'd like to keep the cache line that will be referenced first (that is, we want to replace the one whose next reference is later in time). Unfortunately, neither the cache controller nor the CPU is omniscient; they cannot predict which the best one to be replaced.

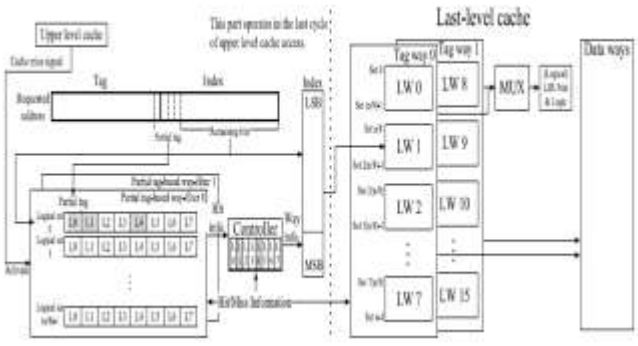

Figure-4: Overall diagram proposed WF-logical–associative cache architecture

To reduce the energy consumption in the tag ways of LLCs, we apply a partial tag matching scheme It extracts a few bits from the tag bits to early identify a cache miss. Fig. 5 shows our proposed partial tag-based way filter. A partial tag consists of a few least significant bits from the original tag bits and a few most significant bits from the index bits. Because the cache lines in a cache way are logically divided, the most significant bits (3 bits for eight logical ways) from the index bits are used as part of a partial tag. It's because the number of logical cache sets becomes smaller than that of cache sets when logical cache ways is applied. A partial tag that is based way filter is where powered down when to their corresponding cache way is turned OFF which can be allocated to each of cache way, and it. Experimentally, to find that a 4-bit partial tag and eight logical ways show optimized results.

b) Cache partitioning

Another one is in sharing of the cache between two different accesses to get studies of every targeted stream. Stone *et al*. studied the optimal allocation of cache memory among multiple access streams. They showed that LRU typically comes close to achieving optimal performance and they focused on partitioning a cache between the instruction and data access streams of a single workload.

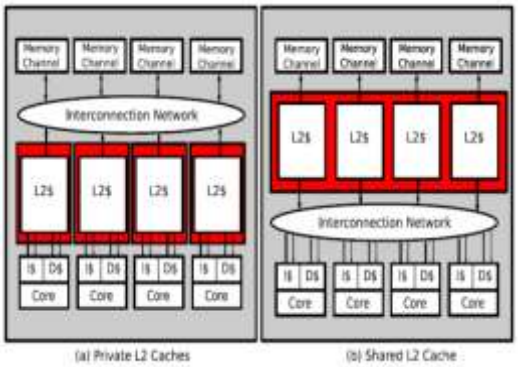

Figure-5: L2 Cache Designs

c) Low Utility

Applications that do not benefit significantly as the available cache space are getting gradually increased. The reason for that could be a very big working set, larger than any available cache, or that the application suffers a large number of compulsory misses.

ISSN (online): 2581-3048
Volume 2, Issue 1, pp 5-9, March-2018

d) Multi Core System

A major objective of this research is to determine an efficient way of using the memory hierarchy of a CMP system. The next step was therefore to study the performance of a dual core system. More specifically, the system was simulated running two of the selected benchmarks in parallel. Each of the processor cores has their own data caches and private L1 instruction of 32 KB and 4way associative each. Both cores share a 4096 KB, 32way associative L2 cache, which employs the LRU replacement policy. The simulation is stopped when one of the benchmarks finishes. At first, saturating utility applications were selected.

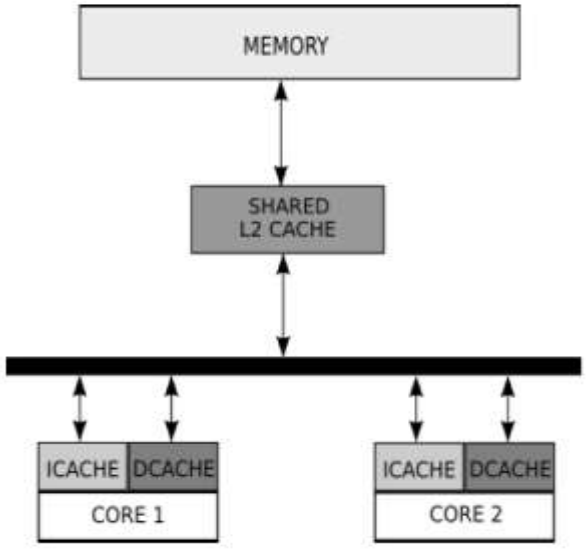

Figure-6: Dual core system

In parallel tagged array with the data cache probe which are similarly accessed to provide every tag stored in the cache block. This stored tag is compared to the tag bits of the original address, and if they match, the Hit/Miss signal is asserted to indicate a cache hit.

IV.CONCLUSION

In Future, to eliminate static profiling, I have to develop the WF-based dynamic logical–associative cache. To evaluated proposed architecture and observed a 3.4% total system energy and 34% LLC energy savings over the selective-cache ways scheme with a relatively small overhead on a single-core system. The proposed scheme shows more energy reduction results on multi core systems with more capacity pressure on the LLC. I have observed that the total energy consumption and EDP are reduced by 9.2% and 11.8%, respectively, over the selective-cache-ways scheme on a quad-core system.

REFERENCES

[1] Abdel-massieh, N. H., Hadhoud, M. M., and Amin, K. M., "Fully automatic liver tumor segmentation from abdominal CT scans," *IEEE International Conference on Computer Engineering and Systems (ICCES)*, pp. 197–202, 2010.

[2] Abdalla, Z., Neveen, I. G., Aboul Ella, H., and Hesham, A. H., "Level set based CT liver image segmentation with watershed and artificial neural networks," HIS, *IEEE*, pp. 96–102, 2012.

[3] Abdalla, M., Hesham, H., Neven, I. G., Aboul Ella, H., and Gerald, S., "Evaluating the effects of image filters in CT liver CAD system," *IEEE-EMBS International Conference on Biomedical and Health Informatics*, The Chinese University of Hong Kong, Hong Kong, 2012.

[4] Hame, Y., and Pollari, M., "Semi-automatic liver tumor segmentation with hidden Markov measure field model and non-parametric distribution estimation," *Med Image Anal.*, vol. 16, no. 1, pp. 140–149, 2012.

[5] Marius George Linguraru, William J. Richbourg, Jianfei Liu, Jeremy M. Watt, Vivek Pamulapati, Shijun Wang, and Ronald M. Summers, "Tumor Burden Analysis on Computed Tomography by Automated Liver and Tumor Segmentation", *IEEE Transactions on Medical Imaging*, vol. 31, no. 10, October 2012.

[6] Shweta, G., and Sumit, K., "Variational level set formulation and filtering techniques on CT images," *International Journal of Engineering Science and Technology*, vol. 4, no.7, July 2012.

[7] Blessingh.T.S, Vincey Jeba Malar.V, Jenish.T, "CAD system for Lung Cancer using Statistical model and Biomarker", *CIIT International Journal Of Digital Image Processing*, Volume 5, No 4, April 2013, ISSN 0974-9691.

[8] Govindaraj.V, Sengottaiyan.G, "Survey of Image Denoising using Different Filters", ISSN: 2278 – 7798, *International Journal of Science,*

ISSN (online): 2581-3048
Volume 2, Issue 1, pp 5-9, March-2018

Engineering and Technology Research (IJSETR), Volume 2, Issue 2,February 2013.

[9] Zhang, X., Tian, J., Xiang, D., Li, X., and Deng, K., "Interactive liver tumor segmentation from ct scans using support vector classification with watershed," *IEEE Conf. Eng Med Biol Soc.,* vol. 2011, pp. 6005–6008, 2011.

[10] Goryawala, M., and Guillen, R., "A 3-D liver segmentation method with parallel computing for selective internal radiation therapy", *IEEE Transaction on Information Technology in Biomedicine,* vol. 16, no. 1, pp. 62-69, Jan. 2012.

How to cite this article:

R.Karthika, A.Vivekaraj, G.Kanagaraj, "Dynamic Architecture Way Filtering To Reduce Power Consumption on LLC", in *International Research Journal of Innovations in Engineering and Technology (IRJIET),* Volume 2, Issue 1, pp 5-9, March 2018.

****

