Rethinking Last-Level Cache Management for Multicores Operating at Near-Threshold Voltages

Farrukh Hijaz and Omer Khan
University of Connecticut, Storrs, CT USA
{farrukh.hijaz,khan}@uconn.edu

Abstract

Near-threshold voltage computing promises an order of magnitude improvement in energy efficiency, enabling future processors to integrate 100s of cores running concurrently. However, such low voltage operation accompanies extreme parametric variations, resulting in unreliable operation of the processor. The memory bit-cells in on-chip caches are most vulnerable to failure due to their tight functionality margins. In this paper we focus on the NTV challenges with the last-level cache (LLC) management for future multicores. The state-of-the-art mechanisms enable a hybrid of disabling faulty cache lines and error correction capabilities to maximize the usable cache capacity in the LLC. We postulate that future technologies will suffer from extreme fault rates and some portions of the LLC need to be disabled. This will put unprecedented stress on the LLC capacity impacting the existing management schemes for placement, movement and replication of data. We focus our attention on evaluating the state-of-the-art LLC data management schemes and observe that a scheme optimized for nominal voltage operation may not necessarily perform optimally at NTV. We show that LLC data management schemes must be rethought and redesigned for the emerging energy efficient NTV multicores.

1. Introduction

Multicores have emerged as an alternative to complex sequential processors to overcome the “power wall”. Multicores exploit concurrency to achieve performance, and rely on the simplicity of design and operation of each core to achieve energy efficiency. However, with the integration of many cores on a single die, future multicores will still be constrained by their energy efficiency [30]. Extreme voltage scaling increases the energy efficiency of future processors since energy scales quadratically with voltage [7]. Research has shown that near-threshold voltage (NTV) region is the most energy efficient region to operate in. It can deliver up to 10× reduction in energy [17]. However, the clock frequency of the processor must be reduced, otherwise, the hardware structures may fail due to extremely tight timing guard-bands. Logic elements have been shown to be resilient to timing variations at NTV [17]. However, SRAM memory bit-cells are most vulnerable to failure due to their tight functionality margins [7].

The higher demand for on-chip cache has been steadily increasing to alleviate expensive off-chip accesses. A private last-level cache (LLC) organization (e.g., [11]) has low hit latency due to high data locality. However, its off-chip miss rate is high in workloads with large private working set and/or high degree of data sharing. A popular cache organization is to implement per-core fast private caches backed by a logically shared (physically distributed) last-level cache to minimize the off-chip miss rate [19]. Since the next generation multicores will execute applications with massive data and large working sets, maximizing the capacity of LLC becomes even more important. However, the variable latency to access the shared LLC naturally gives rise to the data locality challenge due to non-uniform cache access (NUCA) [19]. Existing LLC data placement, movement and replication schemes attempt to balance the tradeoffs between data locality and capacity. However, at NTV both latency and capacity of the LLC will worsen and must be reevaluated for optimal data access.

The state-of-the-art NTV proposals either rely on frequency (f_{MIN}) and voltage (V_{ddMIN}) settings such that the processor has zero-errors [16], or allow bit-cell errors to exist at NTV and fix them during design time or at runtime [7, 3, 4]. The first approach delivers reliable operation without increasing the hardware complexity. However, it operates above the near-threshold voltage V_{ddNTC}, and does not fully exploit the energy efficiency. The second approach ensures higher energy efficiency by operating near the threshold voltage (V_{ddNTC}). There are two options for the frequency setting while operating in the NTV region. The first is to run at a low enough frequency (f_{NTC}) to ensure zero-errors [12], however, the overall performance degrades substantially. The second option is to set the frequency close to f_{MIN}, such that the system operates at an acceptable performance level. Because the system now operates at a higher than safe frequency for V_{ddNTC}, timing margins of logic elements and SRAM bit-cells may be violated.

The core pipeline elements and register files can be designed NTV friendly using low overhead circuit solutions [17]. However, a fixed design time approach to harden SRAM bit-cells for NTV operation incurs high area overhead [7, 21, 6], decreasing the available SRAM capacity in a given area, latency and energy budget. The runtime approaches use traditional SRAM design, however, they either disable faulty bit-cells and sacrifice the available memory capacity [28, 25, 2, 4], or use error correcting codes (ECC) to correct bit-cell errors at the cost of additional access latency [9, 20, 31, 10, 24, 3, 14].

We postulate that extreme voltage scaling will result in high bit-cell failure rates in the deep nanoscale technologies. At these high bit-cell failure rates, the previously proposed schemes will reach their error correction limits and resort to disabling portions of the LLC. This will result in reduced LLC capacity and increase the expensive off-chip accesses. Therefore, managing LLC capacity and the associated locality tradeoffs is now ever so critical. In this paper, we explore the effects of different data management schemes at the reduced LLC capacity while operating in the NTV region. An important question that arises is whether the LLC data management schemes used at super-threshold voltages (STV) work efficiently at NTV. We evaluate several LLC data management schemes at different bit-cell failure rates (with varying effective LLC capacity) and highlight some of the locality and capacity tradeoffs in play.

We implement VS-ECC-Disable [3] (cf. Sec. 2.1 for details) on top of a 64-core multicore with a Private-L1 Shared-L2 cache organization. This scheme ensures functionally correct operation of the processor at nominal as well as at NTV. We evaluate state-of-the-art static data management scheme, S-NUCA [19], and dynamic data management schemes, R-NUCA [13] and Victim Replication(VR) [32]. We perform a detailed analysis of the capacity and locality tradeoffs as the effective LLC capacity is decreased at increasing bit-cell failure rates.

The key observations of this paper are:

1. No single LLC management scheme caters for all workloads.
2. A scheme performing optimally at full LLC capacity might not be as effective at NTV and may need NTV specific optimizations.
3. The limited usable capacity and the random nature of faults at NTV pose a serious challenge for optimizing LLC data management schemes.

2. Architecture Framework

2.1. The NTV Multicore Cache Hierarchy

We implement a representative NTV multicore processor. As the focus of this paper is on LLC data management in NTV conditions, we assume that the L1 caches are protected against permanent bit-cell faults using circuit techniques [6, 8]. We assume that LLC tag arrays are also hardened through circuit techniques. The LLC data arrays are modeled based on VS-ECC-Disable architecture [3]. VS-ECC-Disable deploys a hybrid architecture that combines error correcting codes and cache line disabling. Disabling is used when a cache line has more bit-cell faults than the correction capability of ECC.

Our representative NTV multicore is shown in Fig. 1 with shaded regions showing the additional components required on top of an ideal fault-free system. A single “Disable Bit” (DB) is added to the tag of each cache line. For each operating voltage, the disable bits are individually populated and they indicate whether each cache line needs to be disabled. We assume the system is capable of pre-computing all disable bits for a given operating voltage using a traditional memory built-in self-test mechanism (cf. Sec. 3.4).

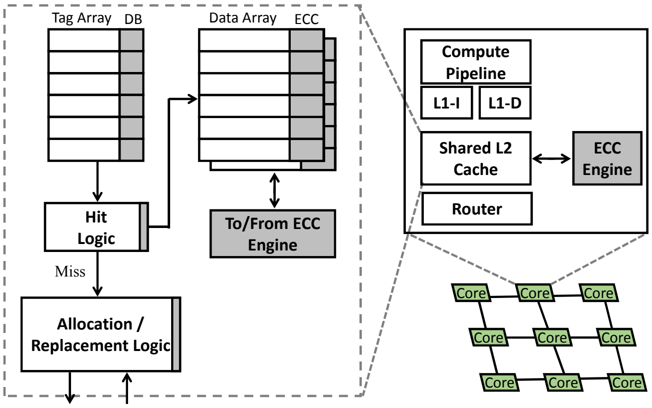

Figure 1: NTV multicore architecture based on VS-ECC-Disable based LLC. VS-ECC-Disable cache operates without any additional latency on fault-free cache lines and incurs two cycle latency for cache lines with either one-bit or two-bit faults. Cache lines with more than two faulty bits are disabled. A “Disable Bit” (DB) is added to the tag array to identify cache lines that cannot be corrected.

The ECC engine and the data structure to store and manage the ECC check bits is added to each LLC data array. The hit logic and the replacement policy is modified to ensure its fault-aware operation. VS-ECC-Disable incurs no extra latency on a cache hit to a fault-free cache line. It incurs two additional cycles on a hit to a cache line with one-bit or two-bit faults. Cache lines with greater than two-bit faults are disabled and not allocated in the LLC. The storage overhead of the VS-ECC-Disable cache architecture is <2% [3].

2.2. LLC Data Management Schemes

We evaluate the impact of increasing fault rates at NTV operation of three state-of-the-art LLC data placement, movement and replication mechanisms, as outlined below.

Static-NUCA [19] is a static LLC data placement scheme. S-NUCA address interleaves the data across all LLC slices. It does not allow replication of data in the LLC and each cache line is only stored in one physical location. The probability that a cache line resides in the local core’s LLC slice is $1/n$ in an n-core processor. Therefore, data is mapped to a remote core with high probability. S-NUCA suffers from high remote LLC slice access rate that results in high on-chip traffic and high average LLC access latency/energy.

Reactive-NUCA [13] classifies data as private or shared at page granularity using the existing operation system virtual memory management mechanism. When a cache line is accessed for the first time, it classifies the associated page as private, and places it at the requester’s LLC slice. If another core then accesses data from that page, it is reclassified as

Table 1: Architectural parameters for evaluation.

Architectural Parameter	Value
Number of Cores	64 @ 1 GHz
Compute Pipeline per Core	In-Order, Single-Issue
Memory Subsystem	
L1-I Cache per core	32 KB, 4-way Assoc., 1 cycle
L1-D Cache per core	32 KB, 4-way Assoc., 1 cycle
L2 Cache per core	256 KB, 8-way Assoc. 2 cycle tag, 6 cycle data Inclusive
Cache Line Size	64 bytes
Directory Protocol	Full Map Invalidation-based MSI
Num. of Memory Controllers	8
DRAM Bandwidth, Latency	5 GBps/Controller, , 75 ns
Electrical 2-D Mesh with XY Routing	
Hop Latency	2 cycles (1-router, 1-link)
Contention Model	Only link contention (Infinite input buffers)

a shared page, and the page is moved to a core based on an address interleaving mechanism. R-NUCA does not allow replication of data. However, instructions are replicated in LLC slice per cluster of 4 cores, using rotational interleaving. This allows R-NUCA to exploit locality for private data. It also avoids cache pollution by not replicating data.

Victim Replication (VR) [32] is a hybrid LLC management scheme that combines the low hit latency of a private LLC and the low off-chip miss rate of a shared LLC. It starts with a private L1, shared LLC configuration and uses the local LLC slice of a core as a victim cache for the cache lines evicted from its L1 cache. Cache line replicas are made at the local LLC slice only if another cache line is found which is either invalid, has no sharers in the L1 cache, or a replica itself, in the stated order. If no such cache line is found, a replica is not created and is sent back to the home core's LLC slice (assigned statically based on address interleaving). On eviction of a replica, it is sent back to the home core's LLC slice. A cache line in the L1 cache has an exclusive relationship with its replica. This means that a replica hit in the local LLC slice results in transferring the cache line to the L1 cache, and its invalidation in the LLC slice. This helps decrease cache pollution, however, cache coherence is further complicated since the cache line is either in the L1 cache or the LLC slice.

3. Evaluation Methodology

3.1. Performance Models

We evaluate a 64-core shared memory multicore. The default architectural evaluation parameters are shown in Table 1. All experiments are performed using the core, cache hierarchy, coherence protocol, memory system and on-chip interconnection network models implemented within the Graphite [23] multicore simulator.

3.2. Energy Models

Table 2: Projected Transistor Parameters for 11 nm Tri-Gate

Parameter	Value
Near Threshold Voltage (V_{NTV})	0.45V
Gate Length	14 nm
Contacted Gate Pitch	44 nm
Gate Cap / Width	2.420 fF/μm
Drain Cap / Width	1.150 fF/μm
Effective On Current / Width (N/P)	739/668 μA/μm
Off Current / Width	1 nA/μm

For energy evaluations of on-chip electrical network routers and links, we use the DSENT [26] tool. Energy estimates for the L1-I, L1-D, L2 (with integrated directory) caches, and DRAM are obtained using McPAT [22]. The energy evaluation is performed at the $11nm$ technology node to account for future scaling trends. We derive models for a tri-gate $11nm$ electrical technology node using the virtual-source transport models of [18] and the parasitic capacitance model of [27]. These models are used to obtain electrical technology parameters (Table 2) used by both McPAT and DSENT.

The static energy (subthreshold and gate leakage) is projected to be the dominant component of the overall energy at NTV [17]. Therefore, we model static energy, in addition to dynamic energy, for the evaluation.

3.3. Benchmarks and Evaluation Metrics

We simulate eleven SPLASH-2 [29] benchmarks (RADIX, FFT, LU_C, LU_NC, CHOLESKY, BARNES, OCEAN_C, OCEAN_NC, WATER-NSQUARED, RAYTRACE, and VOLREND), seven PARSEC [5] benchmarks (BLACKSCHOLES, SWAPTIONS, FLUIDANIMATE, STREAMCLUSTER, DEDUP, FERRET, BODYTRACK, and FACESIM), one Parallel-MI-Bench [15] benchmark (PATRICIA), and one graph benchmark (CONNECTED-COMPONENTS) [1]. Each application is run to completion using the medium or large input sets. For each simulation run, we measure the *Completion Time*, i.e., the time in parallel region of the benchmark; this includes the instruction processing latency, memory access latency[1] , and the synchronization latency.

3.4. NTV Model

Memory built-in self-test (MBIST) is a popular mechanism used to detect memory faults at runtime. We deploy MBIST to test the integrity of on-chip caches and identify faulty cache lines. MBIST is run during the boot-up process of the system for the target operating voltage and frequency. It identifies all faulty bits and constructs a bit-mask of the disable bits for each cache line accordingly. This disable bit-mask is then loaded into the LLC before executing the user applications.

[1] The memory access latency is broken down into five components. **L1 Cache to LLC replica latency (L1C-LLCReplica)** is the time spent by the L1 cache miss request to the LLC replica and the corresponding reply from the LLC replica including time spent accessing the LLC. **L1 Cache to LLC home latency (L1C-LLCHome)** is the time spent by the L1 cache miss request to the LLC home location and the corresponding reply from the LLC home including time spent in the network and first access to the LLC. **LLC home waiting time (LLCHome-Waiting)** is the queueing delay at the LLC home incurred because requests to the same cache line must be serialized to ensure memory consistency. **LLC to sharers latency (LLCHome-Sharers)** is the round-trip time needed to invalidate sharers and receive their acknowledgments. This also includes time spent requesting and receiving synchronous write-backs. **LLC home to off-chip memory latency (LLCHome-OffChip)** is the time spent accessing memory including the time spent communicating with the memory controller and the queueing delay incurred due to finite off-chip bandwidth.

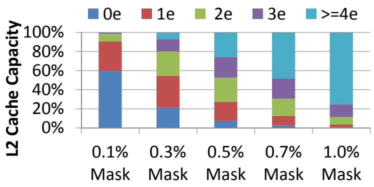

Figure 2: The average LLC capacity with 0, 1, 2, 3, and >=4 random bit-cell faults at NTV.

3.5. Bit-cell Fault Masks for Caches at NTV

Near-threshold voltage depends on the process technology, and can vary within and across generations. At a given NTV operating point, each bit-cell can be modeled as operational or not. Furthermore, these probabilities exhibit normal distribution and random occurrence [21].

We evaluate our architecture at several bit-cell fault rates (0.1%, 0.3%, and 0.5%) with varying usable LLC capacity. Figure 2 shows the average cache capacity that is available with zero-bit, one-bit, two-bit, three-bit, and more than three-bit faults. The average usable LLC capacity at 0.7% bit-cell fault rate is only 30.5% with the double correction capability of VS-ECC-Disable. This capacity is too low to even expect any reasonable performance and hence operating at such an extreme region might need stronger ECC correction or a more efficient NTV architecture to recover more LLC capacity. The fault rates used in this paper represent varying capacity levels and with a stronger correction capability these capacity levels might occur at the higher fault rates.

4. Results

In this section, we discuss the impact of reduced available capacity and the random nature of bit-cell faults on the state-of-the-art LLC management schemes. Figures 3 and 4 plot the average completion time and energy breakdown for various random fault rates (0%, 0.1%, 0.3%, and 0.5%). The results are normalized to a fault-free (ideal) S-NUCA baseline with 100% capacity (S-NUCA at 0% fault rate).

R-NUCA and VR perform consistently better than S-NUCA at all fault rates evaluated. VR performs better than R-NUCA at low fault rates, however, it performs on-par with R-NUCA at 0.5% fault rate. This is because as the fault rate increases, the available capacity decreases and VR has fewer opportunities to make local replicas. Due to this effect, VR accesses remote LLC slices more often (even for private data), whereas R-NUCA accesses remote LLC slices for shared data only.

The energy results show similar trends as completion time. R-NUCA and VR reduce the energy consumption over S-NUCA. VR shows some benefits over R-NUCA at low fault rates, however, at 0.5% fault rate the energy consumption is more than that of R-NUCA.

We observe three different trends in completion time across the fault rates, namely (1) workloads where benefits of VR over R-NUCA diminish with increasing fault rates, (2) workloads where benefits of VR are consistently higher than R-NUCA at all fault rates, and (3) workloads where benefits of R-NUCA increase over VR with increasing fault rates. Figures 5 and 6 plot one representative benchmark highlighting each trend. The benchmarks not shown either show similar trends or are not interesting to study in the paper's context.

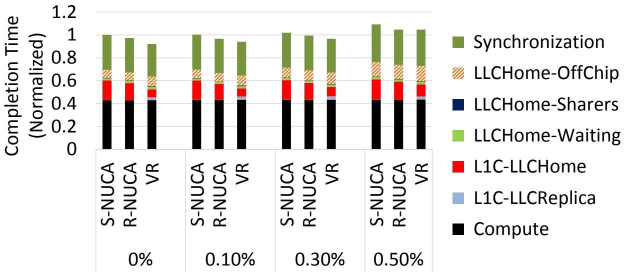

Figure 3: Completion time breakdown for S-NUCA, R-NUCA, and VR at 0%, 0.1%, 0.3%, and 0.5% fault rates. The results are normalized to an ideal fault-free S-NUCA baseline.

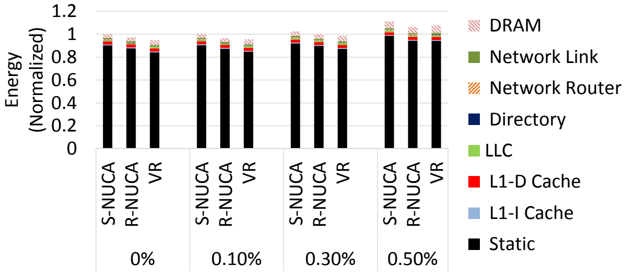

Figure 4: Energy breakdown for S-NUCA, R-NUCA, and VR at 0%, 0.1%, 0.3%, and 0.5% fault rates. The results are normalized to an ideal fault-free S-NUCA baseline.

4.1. BARNES

BARNES (cf. fig. 5) has a working set that fits in the LLC at low fault rates. We observe high number of LLC accesses to shared read-write data. As S-NUCA and R-NUCA do not allow replication of shared read-write data, they suffer from excessive accesses to remote LLC slices. VR replicates such data in the local LLC slice and exploits the lower access latency. This trend holds for 0%, 0.1% and 0.3% fault rates. At 0.5% fault rate the performance advantage of VR over R-NUCA diminishes because of higher stress on the LLC capacity. We note that as the fault rate increases due to NTV operation, the off-chip miss rate also increases. This results in performance degradation for all evaluated schemes. However, the rate of degradation varies making it hard to pick one of the schemes as an optimal candidate. WATER-NSQUARED, and SWAPTIONS show similar behavior.

The energy results (cf. fig. 6) follow the completion time results closely and are dominated by the static energy component. The overall energy increases with increase in fault rate because of the higher number of off-chip accesses.

4.2. OCEAN_NC

OCEAN_NC exhibits a large number of LLC accesses to private data. Since R-NUCA places private data in its local LLC slice, one expects it to show some performance and energy benefits over S-NUCA. However, false sharing is exhibited

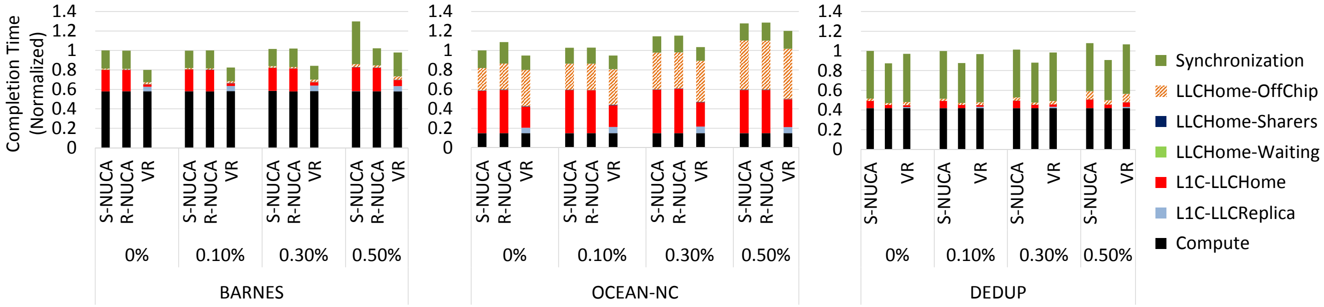

Figure 5: Normalized completion time breakdown at 0%, 0.1%, 0.3%, and 0.5% fault rates.

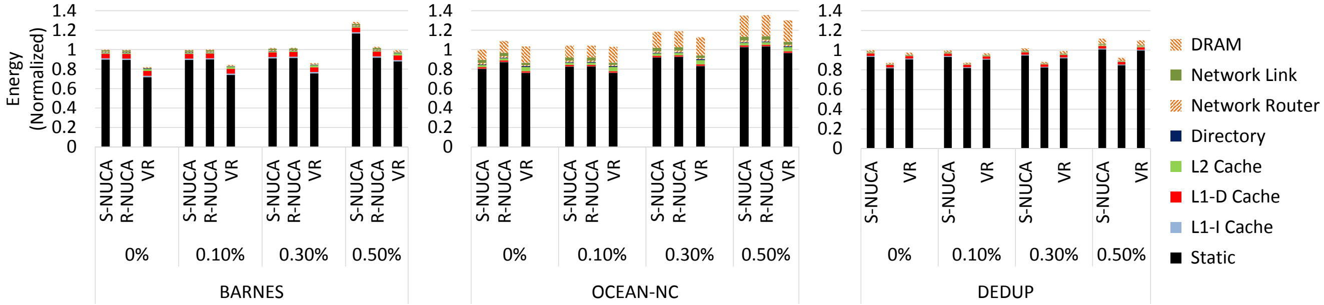

Figure 6: Normalized energy breakdown at 0%, 0.1%, 0.3%, and 0.5% fault rates.

at the page-level, i.e., multiple cores privately access non-overlapping cache lines in a page. As R-NUCA classifies data at the page level, it is unable to place all private data at the local LLC slice. On the other hand, VR replicates cache lines in the local LLC slice upon eviction from the L1 cache. Therefore, it is able to capture private data at cache line granularity and replicate it in the local LLC slice. This helps reduce the average LLC access latency for subsequent accesses to such cache lines.

As the fault rate increases, the overall completion time and energy increases for VR due to limited opportunities for replication. However, VR reduces completion time and energy compared to R-NUCA. Similar trends are observed in BLACKSCHOLES, LU_NC, PATRICIA, and RAYTRACE.

4.3. DEDUP

DEDUP is a benchmark that shows significant number of accesses to thread private data. As S-NUCA stripes the data across the chip based on address interleaving, the probability of the private data being mapped to a remote core is high. This hurts the average LLC access latency and is clearly visible in this benchmark. On the other hand, R-NUCA places private data in the core's local LLC slice and hence exploits lower data access latency. The improvement is significant even at fault rates as high as 0.5%. The reason is that working set of DEDUP suffers (almost) equally from increase in off-chip accesses in all LLC management schemes. VR is built on top of S-NUCA, so it also suffers from the same issue of private data being mapped to a remote LLC slice. It tries to combat the issue by replicating data in the local LLC slice, however, R-NUCA's placement of private data in local LLC slice is more effective and results in higher benefits. Moreover, at high fault rates the usable capacity comes under greater stress and the opportunity to make replicas in the local LLC slice diminishes significantly. Similar behavior is observed in RADIX, FFT, LU_C, CHOLESKY, and OCEAN_C.

The energy results for DEDUP closely follow its completion time results. R-NUCA spends significantly lower amount of static and dynamic energy compared to S-NUCA and VR.

4.4. Summary

Our evaluation highlights that there is no one-size-fits-all data management scheme at the lower usable LLC capacity when operating at NTV. Moreover, a scheme that works optimally at higher LLC capacity might not be effective at the lower usable capacity. We observe this behavior in BARNES and LU_NC where the significant advantage over R-NUCA diminishes at high fault rates. We also observe that there is a tradeoff between locality optimization and the LLC capacity. Optimizing locality ends up putting extra stress on the LLC, increasing the off-chip miss rate. These off-chip accesses cost more than what the locality optimizations improve. Hence, choosing an optimal LLC data management scheme becomes even more challenging. This change in trends motivate further investigation into LLC data management schemes that are specifically designed for NTV operation. Such a scheme needs to not only utilize LLC capacity more intelligently but also possess the ability to handle the random distribution of faults.

5. Conclusion

At near-threshold voltage operation, the usable LLC capacity reduces significantly due to high bit-cell fault rates. In such operation conditions, intelligent management of the LLC capacity becomes critical. In this paper we evaluate three LLC data management schemes. We observe that no scheme caters for optimal data access latency/energy for all benchmarks. We also observe that a scheme that performs optimally at full capacity might not be optimal at NTV. Based on these observations, we conclude that there is a need for variation-aware data management schemes specifically designed for NTV operation of the multicore last-level cache.

References

[1] DARPA UHPC Program BAA. https://www.fbo.gov/spg/ODA/DARPA/CMO/DARPA-BAA-10-37/listing.html, March 2010.

[2] J. Abella, J. Carretero, P. Chaparro, X. Vera, and A. González. Low vccmin fault-tolerant cache with highly predictable performance. In *Int'l Symposium on Microarchitecture*, 2009.

[3] A. R. Alameldeen, I. Wagner, Z. Chishti, W. Wu, C. Wilkerson, and S.-L. Lu. Energy-efficient cache design using variable-strength error-correcting codes. In *Int'l Symposium on Computer Architecture*, 2011.

[4] A. Ansari, S. Feng, S. Gupta, and S. Mahlke. Archipelago: A polymorphic cache design for enabling robust near-threshold operation. In *Int'l Symposium on High Performance Computer Architecture*, 2011.

[5] C. Bienia, S. Kumar, J. P. Singh, and K. Li. The PARSEC Benchmark Suite: Characterization and Architectural Implications. In *International Conference on Parallel Architectures and Compilation Techniques*, 2008.

[6] B. Calhoun and A. Chandrakasan. A 256kb sub-threshold sram in 65nm cmos. In *Solid-State Circuits Conference, 2006. ISSCC 2006. Digest of Technical Papers. IEEE International*, pages 2592–2601, Feb 2006.

[7] A. Chandrakasan, D. Daly, D. Finchelstein, J. Kwong, Y. Ramadass, M. Sinangil, V. Sze, and N. Verma. Technologies for ultradynamic voltage scaling. *Proceedings of the IEEE*, 98(2):191–214, Feb 2010.

[8] I.-J. Chang, J.-J. Kim, S. P. Park, and K. Roy. A 32 kb 10t sub-threshold sram array with bit-interleaving and differential read scheme in 90 nm cmos. *Solid-State Circuits, IEEE Journal of*, 44(2):650–658, Feb 2009.

[9] C. Chen and M. Hsiao. Error-correcting codes for semiconductor memory applications: A state-of-the-art review. *IBM Journal of Research and Development*, 28(2):124–134, 1984.

[10] Z. Chishti, A. Alameldeen, C. Wilkerson, W. Wu, and S.-L. Lu. Improving cache lifetime reliability at ultra-low voltages. In *Int'l Symposium on Microarchitecture*, 2009.

[11] P. Conway, N. Kalyanasundharam, G. Donley, K. Lepak, and B. Hughes. Cache hierarchy and memory subsystem of the amd opteron processor. *Micro, IEEE*, 30(2):16–29, 2010.

[12] R. Dreslinski, D. Fick, B. Giridhar, G. Kim, S. Seo, M. Fojtik, S. Satpathy, Y. Lee, D. Kim, N. Liu, M. Wieckowski, G. Chen, D. Sylvester, D. Blaauw, and T. Mudge. Centip3de: A 64-core, 3d stacked near-threshold system. *IEEE Micro*, 2013.

[13] N. Hardavellas, M. Ferdman, B. Falsafi, and A. Ailamaki. Reactive NUCA: Near-Optimal Block Placement and Replication in Distributed Caches. In *International Symposium on Computer Architecture*, 2009.

[14] F. Hijaz, Q. Shi, and O. Khan. A private level-1 cache architecture to exploit the latency and capacity tradeoffs in multicores operating at near-threshold voltages. In *Computer Design (ICCD), 2013 IEEE 31st International Conference on*, pages 85–92, Oct 2013.

[15] S. Iqbal, Y. Liang, and H. Grahn. ParMiBench - an open-source benchmark for embedded multiprocessor systems. *Computer Architecture Letters*, 2010.

[16] U. R. Karpuzcu, A. Sinkar, N. S. Kim, and J. Torrellas. Energysmart: Toward energy-efficient manycores for near-threshold computing. In *High Performance Computer Architecture (HPCA2013), 2013 IEEE 19th International Symposium on*, pages 542–553, Feb 2013.

[17] H. Kaul, M. Anders, S. Hsu, A. Agarwal, R. Krishnamurthy, and S. Borkar. Near-threshold voltage (ntv) design: opportunities and challenges. In *Design Automation Conference*, 2012.

[18] Khakifirooz, A. and Nayfeh, O.M. and Antoniadis, D. A Simple Semiempirical Short-Channel MOSFET Current-Voltage Model Continuous Across All Regions of Operation and Employing Only Physical Parameters. *Electron Devices, IEEE Transactions on*, 56(8):1674 –1680, aug. 2009.

[19] C. Kim, D. Burger, and S. W. Keckler. An Adaptive, Non-Uniform Cache Structure for Wire-Delay Dominated On-Chip Caches. In *ASPLOS*, 2002.

[20] J. Kim, N. Hardavellas, K. Mai, B. Falsafi, and J. Hoe. Multi-bit error tolerant caches using two-dimensional error coding. In *Int'l Symposium on Microarchitecture*, pages 197–209, 2007.

[21] J. Kulkarni, K. Kim, and K. Roy. A 160 mv robust schmitt trigger based subthreshold sram. *IEEE Journal of Solid-State Circuits*, 2007.

[22] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi. Mcpat: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In *MICRO*, 2009.

[23] J. E. Miller, H. Kasture, G. Kurian, C. Gruenwald, N. Beckmann, C. Celio, J. Eastep, and A. Agarwal. Graphite: A distributed parallel simulator for multicores. In *HPCA*, pages 1–12, 2010.

[24] T. Miller, R. Thomas, J. Dinan, B. Adcock, and R. Teodorescu. Parichute: Generalized turbocode-based error correction for near-threshold caches. In *Int'l Conf on Microarchitecture*, 2010.

[25] D. Roberts, N. S. Kim, and T. Mudge. On-chip cache device scaling limits and effective fault repair techniques in future nanoscale technology. In *Euromicro Conference*, 2007.

[26] C. Sun, C.-H. O. Chen, G. Kurian, L. Wei, J. Miller, A. Agarwal, L.-S. Peh, and V. Stojanovic. DSENT - a tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling. In *Int'l Symposium on Networks-on-Chip*, 2012.

[27] L. Wei, F. Boeuf, T. Skotnicki, and H.-S. Wong. Parasitic Capacitances: Analytical Models and Impact on Circuit-Level Performance. *Electron Devices, IEEE Transactions on*, 58(5):1361 –1370, may 2011.

[28] C. Wilkerson, H. Gao, A. R. Alameldeen, Z. Chishti, M. Khellah, and S.-L. Lu. Trading off cache capacity for reliability to enable low voltage operation. In *Int'l Symposium on Computer Architecture*, 2008.

[29] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta. The SPLASH-2 Programs: Characterization and Methodological Considerations. In *International Conference on Computer Architecture*, 1995.

[30] K. Yelick. Ten ways to waste a parallel computer. In *Keynote at International Symposium on Computer Architecture*, 2009.

[31] D. H. Yoon and M. Erez. Memory mapped ecc: low-cost error protection for last level caches. In *Int'l Symp. on Computer Arch.*, 2009.

[32] M. Zhang and K. Asanovic. Victim replication: Maximizing capacity while hiding wire delay in tiled chip multiprocessors. In *Int'l Symposium on Computer Architecture*, 2005.