IMPROVING MEMORY RELIABILITY, POWER AND PERFORMANCE USING MIXED-CELL DESIGNS

Contributors

Alaa R. Alameldeen
Intel Labs

Nam Sung Kim
University of Wisconsin-Madison

Samira M. Khan
Intel Labs and Carnegie Mellon University

Hamid Reza Ghasemi
University of Wisconsin-Madison

Chris Wilkerson
Intel Labs

Jaydeep Kulkarni
Intel Labs

Daniel A. Jiménez
Texas A&M University

Many enterprise and mobile systems attempt to maximize energy-efficient performance, dynamically trading off performance and power to have the best performance while keeping power within specified limits. Cache and memory system behavior plays a large role in this tradeoff, since power optimizations may jeopardize memory cell reliability.

In this article, we show that mixed-cell memory designs could play a key role in achieving the right balance between performance, power, and reliability for single-core and multi-core systems. In such designs, part of the memory structure is built with cells that are more robust and failure-resistant, while the rest is designed using traditional cells. Robust cells ensure resiliency under low-voltage conditions to protect the most vulnerable data, while the rest of the memory structure could be used to store redundant data to improve performance. We demonstrate this concept using two specific examples: (1) A cache system that only turns on the robust portion at low-voltage, achieving good reliability and power savings while providing high-voltage performance improvements; (2) A cache system that uses the whole cache (including the non-robust portion) at low voltage, achieving good performance and reliability while not exceeding power limits. While the specific examples we explore in this article are cache-related, the same concept could be used throughout the entire memory hierarchy to improve memory resiliency without sacrificing performance or energy efficiency.

Introduction

Power is a key design constraint for modern multiprocessors used across market segments, from mobile systems to servers. In mobile systems, thermal design power (TDP) plays a key role in determining the form factor of the mobile device, and therefore optimizing processor power is critical. Likewise, data centers are built with fixed power and cooling capabilities, and improving processor performance within a given power budget yields direct economic benefits by increasing the compute capability supported by a fixed investment in data center infrastructure.

"Increasing core counts, in the context of fixed power budgets, is a key challenge for future systems."
"In today's TDP-limited systems, the voltage of active cores has to decrease as the number of active cores increases."

To address these power constraints, new processor generations have provided improvements in core performance and efficiency and have also increased the number of cores on a die. Today, state-of-the-art server processors may contain tens of cores, and even mobile products, including tablets and smart phones, have more than one core. Increasing core counts, in the context of fixed power budgets, is a key challenge for future systems.

In today's TDP-limited systems, the voltage of active cores has to decrease as the number of active cores increases.[6] Conversely, as cores become inactive,

the voltage of the remaining cores is raised to maximize performance. Changing the voltage in response to changes in core activity allows the power budget of these systems to remain constant regardless of the number of active cores.

Voltage reduction, however, comes at the cost of dramatically reducing reliability for memory cells that operate at a low voltage. To circumvent this problem, prior work has explored using separate voltages for the core logic and caches. This captures most of the power benefits by reducing the core voltage, while ensuring reliable cache operation at a higher voltage. However, separate voltage domains greatly increase design complexity.[13] This added complexity can be avoided by building memories with robust cells better suited for low voltage operation (using larger cells with upsized transistors or more transistors). Unfortunately, robust cells significantly increase power and area for a memory structure.

"Voltage reduction, however, comes at the cost of dramatically reducing reliability for memory cells that operate at a low voltage."

The high overhead of cell upsizing has led architects to propose mixed (heterogeneous) cell cache architectures, consisting of traditional cells and robust cells[4][5][8], with the goal of minimizing the use of expensive, robust memory cells, while continuing to harvest their low voltage benefits. Mixed-cell cache architectures achieve this by implementing a small portion of the cache with robust cells that can operate reliably at low voltage, and the remainder with non-robust cells. When operating at a high voltage, both portions would be used to maximize cache capacity and performance. When operating at low voltage, the failure-prone non-robust cells would be turned off, reducing cache capacity by up to 75 percent.[4][5] Conversely, the non-robust cells can be turned on but are only used to store noncritical data.[8]

We advocate using mixed (heterogeneous) cell cache architectures to build reliable and scalable memory structures. Memory structures do not need to be uniformly reliable. With careful design mechanisms, a robust (reliable) portion can be used to store critical data, while the non-robust portion can be power-gated or used for noncritical data.

"We advocate using mixed (heterogeneous) cell cache architectures to build reliable and scalable memory structures."

In the remainder of this article, we demonstrate how mixed-cell architectures help achieve memory resiliency at low voltage. We highlight two examples for cache hierarchies designed with mixed cells:

- In the first design[5], a last-level cache is designed with a fraction of all cells built with robust cells, while the rest are built using standard cells that are power-gated at low voltage. Such a system helps maintain low-voltage cache reliability while allowing the whole cache to be active at high voltage/frequency to maximize performance.
- In the second design[8], both robust and non-robust cells are enabled at low voltage, but special logic needs to be implemented to ensure critical data (that is, the only copy in the system) is stored in robust cells. Such a design helps maximize performance for a multi-core system where all cores could be active only at low voltage.

Background

Achieving the highest possible density is a main design goal for different memory technologies. SRAM bit cells, for example, generally employ minimum-geometry transistors, which are susceptible to systematic as well as random process variations such as random dopant fluctuations (RDF) and line edge roughness (LER). Process variations produce V_T (threshold voltage) mismatch between neighboring transistors, resulting in asymmetric bit cell characteristics, and making bit cells susceptible to failure at low voltage. DRAM cells are also designed with minimum-sized transistors in a given process technology, making some cells less reliable when power-saving optimizations are used (such as lower refresh frequency). We'll use SRAM caches as the main topic of discussion in this article, but similar tradeoffs could also apply to other memory technologies.

"…large memory structures in the core, such as caches, become unreliable at low voltage."

With bit cells susceptible to failure, large memory structures in the core, such as caches, become unreliable at low voltage. This limits voltage and frequency scaling for the cores, which must operate at a minimum voltage (Vmin) to ensure reliable operation. Reducing cache Vmin has become an area of active research. Prior work in this area fits into two broad categories: circuit solutions and architectural solutions.

Circuit Solutions

Circuit techniques generally aim to reduce Vmin by improving the bit cell. One approach is to reduce the voltage for the core logic and use a separate (higher) voltage for caches. Unfortunately, a partitioned power supply increases power grid routing complexity, reduces on-die decoupling capacitance, increases susceptibility to voltage droops, and may require level shifters that add latency to signals that cross voltage domains. [13] Most commercial processors use multiple voltages generated off-chip by high-efficiency off-chip voltage regulators (~95-percent efficiency). As the number of cores increases, providing multiple voltages for each core becomes increasingly impractical. A four-core system with separate voltages for the core and its private L1/L2 caches would require three voltage domains per core (a total of 12 power supplies), in addition to those needed for other system components.

"…upsizing devices can dramatically reduce variations and improve Vmin."

Another way to improve bit cell Vmin involves upsizing its constituent devices. Threshold voltage (V_T) variation depends inversely on the transistor gate area.[9] Consequently, upsizing devices can dramatically reduce variations and improve Vmin. Zhou et al.[22] designed and optimized six different 6T SRAM cells (C1-C6 cells), and analyzed the failure probabilities of the cells due to process variations in a 32 nm technology. These analyses demonstrated that increasing a cell's size can reduce its failure probability by orders of magnitude.

Unfortunately, the Vmin benefits of upsizing a typical 6T bit cell diminish as device size increases. Figure 1 compares the Vmin for four different caches implemented in a 65 nm technology. Each cache is implemented using one

of four different 6T cells. [11] We set Vmin at the point when the cache failure probability is 1/1000.[20] The figure depicts the probability (y-axis) that the cache will contain a single failing bit as a function of voltage (x-axis). A 4-MB cache constructed with a minimum-sized 6T cell, 4M-min, exhibits very high failure rates (~30 percent) even at high voltages (>900 mV). The 4M-2X implementation of a 4-MB cache doubles the device sizes in each memory cell, increasing cell area by 33 percent. 4M-4X quadruples the size of the devices, doubling the size of the cell. The 4M-8X implementation uses the most robust cell with devices that are eight times as large and a 233 percent larger cell size than 4M-min. Increasing cell sizes initially yields dramatic improvements over minimum-sized cells (note the 275 mV improvement moving from 4M-min to 4M-2X). But further size increases yield smaller benefits, 60 mV and 55 mV for the 4M-4X and 4M-8X, respectively.

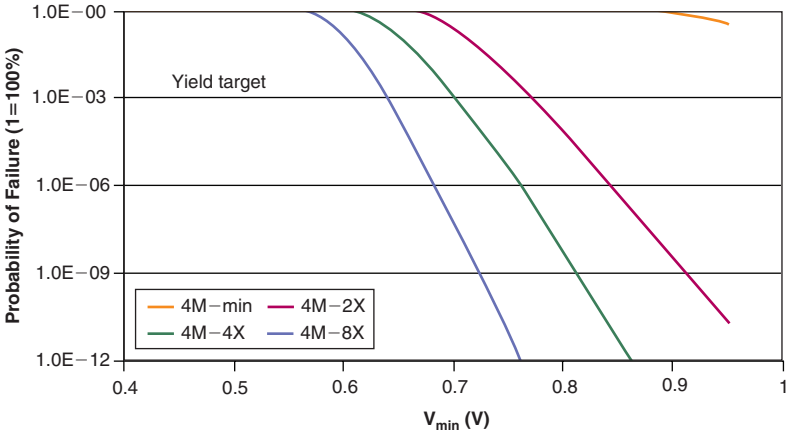

Figure 1: Vmin improvements with bit cell upsizing (Source: Khan et al., 2013[8])

Cell upsizing also causes increases in static and dynamic power. Static power (leakage) varies linearly as a function of transistor dimensions, therefore increasing with larger cells. Larger cells also add switching capacitance on the word lines (WL) and bit lines (BL) increasing dynamic power. Upsizing from the minimum cell to the 2X cell yields a substantial benefit since the reduction in Vmin (275 mV) more than compensates for the additional power introduced by larger devices. Further upsizing, however, increases power since the costs of larger devices outweigh the savings from voltage reductions (–60 mV, –55 mV).[8]

"Cell upsizing also causes increases in static and dynamic power."

Architectural Solutions

Another approach to reducing Vmin uses failure-prone cells with smaller devices, but augments the memory array with the capability to repair bit failures. Prior work introduced many repair mechanisms that depend on memory tests to identify bad bits.[17][18][20] Relying on memory tests limits the applicability of these approaches when memory tests are expensive or failures are erratic.[1] Other repair mechanisms rely on coding techniques, such as

error-correcting codes (ECC), to autonomously identify and repair defective bits.[3][10]

Fundamentally, each of these approaches trades off the repair mechanism overhead for the ability to compensate for defective bits. For memory designs with very high failure rates, this tradeoff may be unattractive.

To address the high overhead of operating at low voltage, Wilkerson et al.[20] improve Vmin by storing error-correction patterns in cache resources, trading off cache capacity for low voltage operation. Chishti et al.[3] identify the limitations of testing-based implementations and propose to provide error correction capability using orthogonal Latin square codes. Chakraborty et al.[2] also trade off cache capacity for lower voltage. A multi-copy cache stores two copies of each clean datum and three copies of each dirty datum to allow detection and correction of corrupted bits, respectively.

"More recently, designs with mixed (heterogeneous) cell designs have been proposed to achieve low voltage with modest area cost."

More recently, designs with mixed (heterogeneous) cell designs have been proposed to achieve low voltage with modest area cost. Dreslinski et al.[4] propose to combine the low voltage benefits of robust upsized cells and the cost benefits of smaller cells by building caches with a mixture of cell types. Cache lines consisting of robust cells operate at low voltage, while a separate power supply provides a higher voltage to less robust cells. By moving recently accessed data to the low voltage cache lines, Dreslinksi et al. service the majority of requests using low voltage cache lines, and reduce active power in the L1 cache.

While using mixed cell architectures could help achieve reliable low voltage operation, it is important to ensure that such design has a minimal impact on high-voltage performance (for a single-core system) or low-voltage performance (for a multi-core system). In the next two sections, we highlight two mixed-cell cache architectures we explored in our prior work. The first architecture[5] is tailored towards high-performance systems, where high-voltage performance is critical but we need to maintain reliability at low voltage using robust cells. The second architecture[8] targets multi-core TDP-limited systems, where the highest performing point is when all cores are active at low voltage, so low-voltage performance is critical.

A Mixed-Cell Architecture for High-Performance Systems

A typical last-level cache (LLC) consists of hundreds or thousands of SRAM sub-arrays. We proposed an architecture for a single-core system that uses multiple cell sizes in a single LLC.[5] When high performance is needed, the processor runs at high voltage/frequency states where even small (non-robust) cells can operate reliably. As supply voltage is lowered, the failure rate of small cells increases exponentially, so we disable ways or sets one after another beginning with those consisting of the smallest SRAM cells. Ways or sets implemented with large cells remain active

(and reliable) at lower supply voltage, providing the needed LLC capacity. To avoid failures, a uniform implementation of the cache using only robust cells would significantly decrease cache capacity for the same area, therefore hurting high-voltage performance. Alternatively, we propose a heterogeneous-cell architecture to avoid low-voltage failures without hurting high-voltage performance.

"…we propose a heterogeneous-cell architecture to avoid low-voltage failures without hurting high-voltage performance."

LLC Implementation Using Heterogeneous Cell Sizes to Support Low Vmin

Consider a four-way set-associative cache. Figure 2 illustrates an example of building a four-way set-associative LLC, where each group of sub-arrays is associated with a cache way and has a different cell size. In this illustration, the total number of sub-arrays is divided into four groups where each group represents a particular way with a particular cell size; the sub-arrays with larger cells become taller since the cell size increases in the horizontal direction.[22] In this example, the processor and LLC are operating at 0.7 V. Thus, the LLC sections corresponding to ways three and four are disabled at 0.7 V since they are comprised of small cells, many of which will fail at such a voltage.

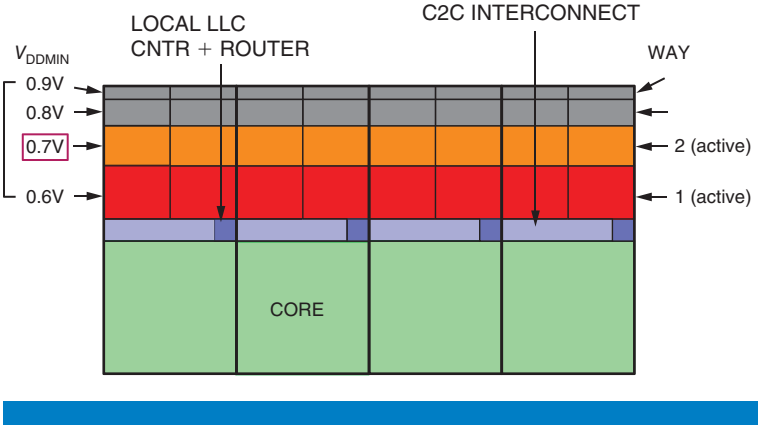

Figure 2: An example four-way LLC, where each way uses a different cell size and is active at a different voltage. The processor runs at 0.7 V, so only ways 1 and 2 are active
(Source: Ghasemi et al., 2011[5])

Consider an 8-MB LLC comprised of C5 and C3 cells[22] where each cell size provides 4-MB capacity. The cell failure probability of these cells is presented in Table 1. In this particular architecture, we can reduce the total cell area by 15 percent, that is, the total LLC area by 13 percent considering SRAM array efficiency equal to 85 percent.[26] When the voltage (frequency) is higher than 0.8 V (1.6 GHz), the processor is able to use the full 8-MB LLC capacity. If the voltage (frequency) is below 0.8 V (1.6 GHz), the 4-MB section of the LLC consisting of the smaller C3 cells will be disabled. However, the 4-MB section consisting of the larger C5 cells will operate reliably in the whole voltage range from 0.7 V to 0.9 V.

	C1	C2	C3	C4	C5	C6
Relative Cell Size	1.00	1.12	1.23	1.35	1.46	1.58
Pfail at 0.90 V	3.2×10^{-7}	2.5×10^{-9}	7.0×10^{-11}	4.5×10^{-12}	5.1×10^{-13}	1.2×10^{-13}
Pfail at 0.85 V	5.4×10^{-7}	1.0×10^{-8}	3.1×10^{-10}	1.6×10^{-11}	3.8×10^{-12}	1.0×10^{-12}
Pfail at 0.80 V	1.0×10^{-6}	3.0×10^{-8}	1.5×10^{-9}	7.6×10^{-11}	2.9×10^{-11}	9.0×10^{-12}
Pfail at 0.75V	2.0×10^{-6}	8.1×10^{-8}	7.4×10^{-9}	4.1×10^{-10}	2.2×10^{-10}	7.9×10^{-11}
Pfail at 0.70V	4.1×10^{-6}	2.1×10^{-7}	3.7×10^{-8}	2.2×10^{-9}	1.6×10^{-9}	7.0×10^{-10}

Table 1: Cell Size and Voltage vs. Probability of Cell Failure at Voltages from 0.9 V to 0.7 V (Source: Ghasemi et al., 2011[5])

"Using this architecture, we can reduce the total area dedicated to SRAM cells by 18 percent…"

To minimize the LLC area further, we can design a more heterogeneous LLC composed of C5, C4, C3, and C2 cells (Table 1) where each cell size gives 2 MB of capacity (for compactness of notation we refer to this as a 2-MB/2-MB /2-MB/2-MB C5/C4/C3/C2 LLC). As the voltage is decreased from 0.9 V to 0.8 V, to 0.75 V, and to 0.7 V, the LLC capacity is reduced from 8 MB to 6 MB, to 4 MB, and to 2 MB. In this architecture, the full 8-MB capacity operates reliably at 0.9 V. As the voltage decreases to 0.8 V, 0.75 V, and 0.7 V, each 2-MB section consisting of C2, C3, and C4 cells will, respectively, be disabled in turn. Within their range of valid operating voltages the resulting cache failure probability of each of the 6-MB, 4-MB, and 2-MB sections of the LLC is acceptable. Using this architecture, we can reduce the total area dedicated to SRAM cells by 18 percent, and therefore, the total LLC area by 16 percent if we assume 85-percent array efficiency. We also explored two other LLC architectures: (1) a 4-MB/2-MB/2-MB LLC consisting of C2/C3/C4 cells, and (2) a 2-MB/2-MB/4-MB LLC consisting of C1, C2, and C4 cells. These two additional LLC architectures satisfy the yield target for the given voltage range, 0.7–0.9 V as long as the proper section of the LLC is shut down for each voltage down-transition. Figure 3 shows the total LLC cell area and the operating voltage range of each section for four different LLC architectures relative to the baseline 8-MB one.

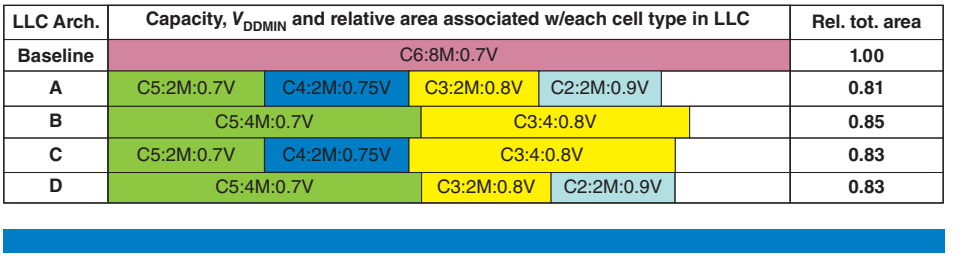

Figure 3: Total LLC cell area for different LLC configurations relative to the baseline. In each colored box whose area is proportional to the total cell area for a given cell size X:Y: Z represents cell size capacity and minimum operating voltage (Source: Ghasemi et al., 2011[5])

Microarchitecture Techniques for LLC Way Shutdown

In our example in Figure 2, as supply voltage decreases, one LLC way after another will be disabled in ascending order of cell size; a cell size is associated with an LLC way. When a voltage/frequency down-transition is triggered by dynamic voltage and frequency scaling (DVFS), an LLC way that cannot operate reliably at the new voltage is shut down. In such a case, the dirty LLC lines in the LLC way must be written back to main memory. The mechanism for shutting down a subset of LLC is already available in commercial multi-core processors to reduce leakage power consumption.[15]

Once the DVFS controller decides to decrease the operating voltage/frequency of the processor, each local LLC controller shown in Figure 2 examines each line in the way that is being shut down. If the line is dirty, it is either (a) written back to the memory controller queue (cache to memory or C2M) or (b) moved to another way after evicting a least recently used clean line in the same set (cache to cache or C2C). The next line is then examined after the status bit of the dirty line is set to the "invalid" state. This process is repeated until all lines are examined in the way that needs to be shut down. Note that a way shutdown process using option (a) may increase the traffic between on-chip cores and off-chip memory (and thus power consumption). On the other hand, the LLC can still service read/write requests to minimize the performance impact associated with the shutdown operations.

Performance and Power Impact

Mixed-cell LLC architectures may impact performance and power both positively and negatively. First, the leakage power remains significant due to the use of larger cells. However, our heterogeneous LLC architectures can reduce a substantial amount of the LLC leakage power since some LLC ways are automatically disabled at low voltage/frequency operating states. Second, the heterogeneous LLC architectures require significantly less die area for the same capacity (Figure 3) compared to a cache with all-robust cells. This freed-up die area can, in turn, be used to increase the LLC capacity, providing higher peak performance at the highest voltage/frequency state.

"…heterogeneous LLC architectures can reduce a substantial amount of the LLC leakage power…"

On the downside, two factors contribute to increasing memory traffic and higher power consumption. First, the flushing operations required before reducing voltage/frequency and disabling LLC ways increases memory traffic. Second, the reduced LLC capacity at low voltage causes more misses and therefore more memory traffic. These effects reduce overall performance and increase memory system power consumption. However, one should note first that workloads that need high performance would spend a substantial fraction of their runtime at the high voltage/frequency states. Furthermore, the interval of voltage/frequency changes is often longer than 10 milliseconds in a commercial operating system, mainly due to the performance penalty associated with PLL re-locking time (tens of microseconds) for changing

frequency.[16] This makes the overall performance impact of the flushing operations quite small; on average, the performance degradation is less than 0.5 percent even with the 1ms voltage/frequency change interval when combined with the C2C scheme.

Evaluation Summary

We evaluated our architecture using Simics[12] augmented with GEMS[14] running four commercial workloads and two memory-intensive SPEC[19] workloads. A more detailed analysis of our results is presented in.[5] Our proposed LLC architecture reduces the LLC total cell area by 15–20 percent without impacting high-voltage performance, compared to an all-robust LLC. Comparing the same cache area as an all-robust LLC, our architecture provides a higher cache capacity, leading to an average 15 percent higher peak performance. The performance impact of the proposed architecture is negligible when various voltage/frequency states are explored by DVFS as a function of changing performance and power demands. The proposed LLC architecture reduces their leakage power due to way disabling at low voltage. Overall energy consumption is reduced by 5–10 percent even though extra energy consumption is required to support the slightly longer runtimes and more frequent accesses to the LLC and off-chip memory.

"Our proposed LLC architecture reduces the LLC total cell area by 15–20 percent without impacting high-voltage performance…"

A Mixed-Cell Architecture for Multi-Core Systems

The motivation for our second mixed-cell cache architecture[8] is to enable the whole cache at low voltage, and therefore avoid a higher cache miss rate and improve low-voltage performance. This is needed for TDP-limited multi-core architectures where all cores can only be active at low voltage. To achieve this goal, we need to protect modified lines by storing them in robust cells, while using the remainder of the cache for clean lines. We use simple error detection and correction mechanisms to detect errors in clean lines, allocate write misses to robust lines, and read misses to clean lines. On a subsequent write to a clean line, we examined three alternatives to ensure modified data is not lost.

"…we need to protect modified lines by storing them in robust cells…"

Cache Hierarchy with Mixed-Cell Support

Figure 4 shows all three levels of our cache hierarchy with support for robust cells. Our baseline cache hierarchy uses a 32-KB 8-way L1 cache, 256-KB 8-way L2 cache, and a 4-MB 16-way LLC (L3). For each level in the cache hierarchy, we implement two ways with robust cells, while the remaining ways use standard (non-robust) cells. This adds an area overhead of 25 percent (L1 and L2) and 12.5 percent (L3) for the cache data array. We add a status bit associated with each tag indicating whether the associated line is a robust way or a non-robust way. We don't necessarily need this extra bit if the robust ways are fixed to two specific ways (Way 0 and Way 1 in Figure 4). We also add an extra LRU bit since we implement a different replacement algorithm in the low-voltage mode.

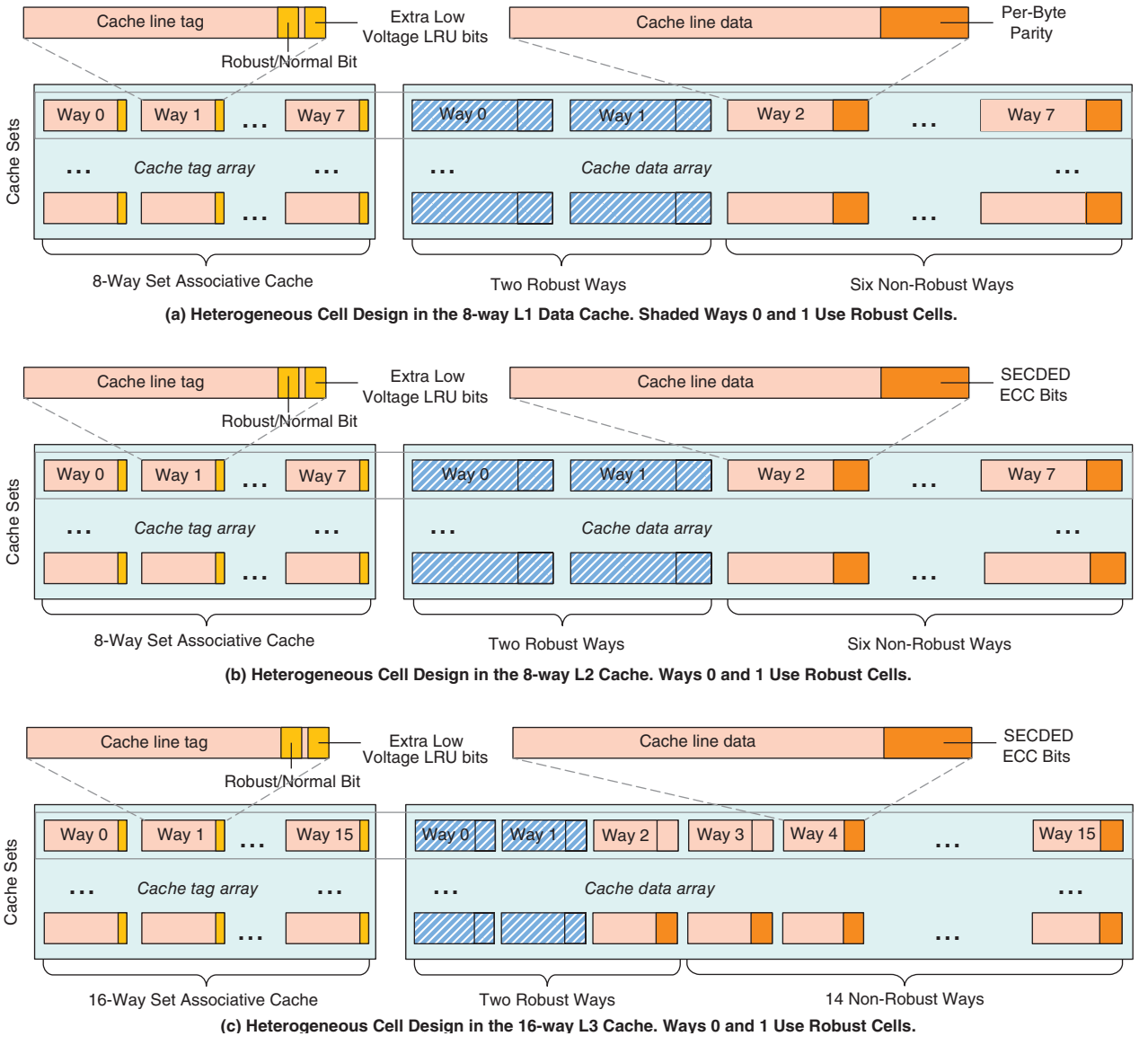

Figure 4: A Mixed-cell cache hierarchy: L1 cache uses parity, while the L2 and L3 use SECDED ECC (Source: Khan et al., 2013[8])

Each cache level has a different requirement for error detection and correction. Since the L1 cache is byte-accessible and extremely latency sensitive, we use a parity bit for each byte in the L1, similar to many Intel® Atom™ and Intel Core™ processors. We use simple SECDED ECC for each line in the L2 and L3 caches. We provide this protection for both robust and non-robust lines to account for soft errors as well as voltage-dependent failures. In general, detectable errors in clean data are recoverable from the next cache level or from memory. However, detectable errors in dirty lines may not be recoverable. To minimize detectable unrecoverable errors (DUEs), we handle modified data differently from unmodified data.

"To minimize detectable unrecoverable errors (DUEs), we handle modified data differently from unmodified data."

If an error is detected in a clean line, it is treated like a cache miss and is obtained from the next cache/memory level. For modified data, however, we must ensure a very low probability of failure, which we achieve through the use of robust cells. This is particularly true in the L1, where parity is unable to correct bit errors and the increased robustness of the cell allows us to minimize the likelihood of bit errors.

To simplify our L1 cache implementation, we handle all accesses to failing lines as cache misses. Since the number of such lines is small, this has little impact on performance. For the L2 and L3 caches, SECDED ECC corrects most errors. Errors that are detected but not corrected (for example, lines with two errors) are handled as cache misses and obtained from the next cache level or from memory. L2 and L3 lines that incur double-bit errors can be disabled to avoid undetectable errors, that is, silent data corruption (SDC), when soft errors hit the same line. Our analysis shows that the probability of failures in robust cells is extremely low at the voltages we consider. For example, we find that 99.9 percent of the L3 caches will suffer failures in less than 1 percent of all lines at low voltage.[8]

Our mixed-cell cache handles writes differently from reads. We need to satisfy the condition of storing modified data only in robust ways. To achieve this objective, we modify the cache replacement policy to handle write misses differently from read misses, and also need to handle subsequent writes to non-robust lines, as we explain in the next two subsections.

Changes to Cache Replacement Policy

We assume the baseline caches implement a least recently used (LRU) replacement policy to simplify our explanation, though the proposed mechanism could be applied to other replacement policies. In our mixed-cell cache architecture, we allocate write misses only to robust ways and read misses to non-robust ways.

"…we allocate write misses only to robust ways and read misses to non-robust ways."

The flowchart in Figure 5 demonstrates changes we made to the cache replacement policy. On a read miss, we choose a replacement victim, NR_LRU, only from non-robust ways based on LRU bits. On a write miss, we choose a victim, GLOBAL_LRU, which is the LRU line among all ways of the set (both robust and non-robust). If the victim line is robust, we trigger a writeback for modified data and allocate the new line in its place. If the chosen victim is in a non-robust way, we choose the LRU line from the two robust ways (RB_LRU), trigger a writeback for modified data to convert the RB_LRU line to a clean line, move the RB_LRU line to use the GLOBAL_LRU line's storage, and allocate the new line to the RB_LRU line.

An alternative implementation we investigated was to implement LRU for two disjoint groups of lines for each cache set: robust lines and non-robust lines. However, some benchmarks, where writes represent a significant fraction of all misses, suffered significant performance losses when write-allocates were limited to choose a victim only from robust ways. We still observed significant

performance losses even when the number of L3 robust ways increased from two to four or eight. This motivated our modified algorithm that chooses a GLOBAL_LRU victim on a write miss. While some benchmarks are affected due to limiting victim selection for read misses to only non-robust ways, the performance losses are small since each cache set has many more non-robust lines than robust lines (6 out of 8 for the L1 and L2 caches, and 14 out of 16 for the L3 cache).

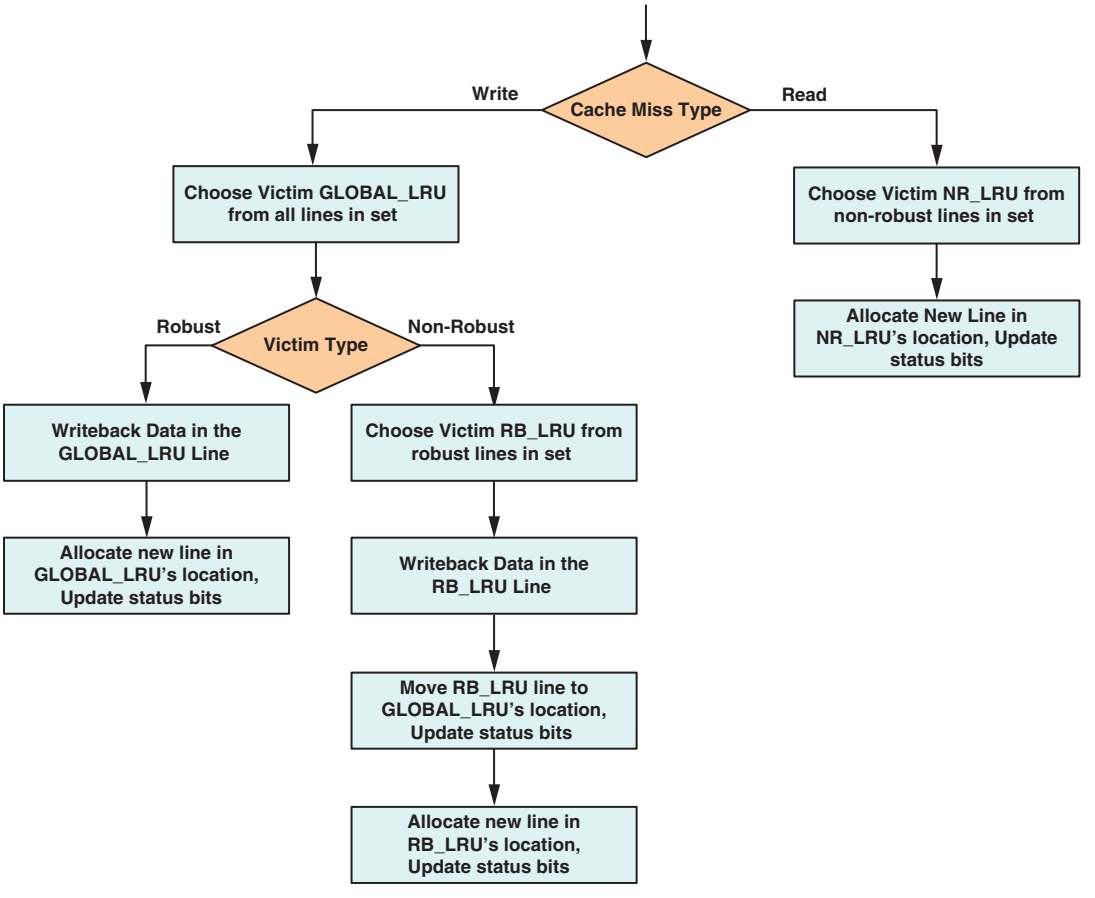

Figure 5: Changes to cache replacement policy (Source: Alameldeen et al., 2013)

Handling Writes to Non-Robust Lines

Our mixed-cell cache architecture needs to prevent DUE and SDC for modified data. It is straightforward to implement this for lines allocated on a write miss, since the cache replacement algorithm would allocate them to robust cells. However, for lines that were allocated to non-robust ways on a read miss, we explore different alternatives to prevent failures.

"Our mixed-cell cache architecture needs to prevent DUE and SDC for modified data."

Writeback

We handle the write to a non-robust line like we would for a write-through cache. We store modified data in the same non-robust line, but convert it to a clean line by writing back the data immediately to the next cache level.

This writeback traffic causes significant network congestion, as well as power and latency overhead. A write to the L1 cache can trigger cascading writes all the way to memory if the L2 and L3 caches allocated the same line to non-robust ways.

Swap

We observe that a write to a cache line is often followed by more writes to the same cache line. To reduce writeback traffic, we handle a write to a non-robust line by swapping with the LRU way of robust lines in the set, RB_LRU. The RB_LRU line triggers a writeback to convert to a clean line. The RB_LRU line is then swapped with the written line. The status and LRU bits are also swapped between the two cache tags. This approach reduces traffic as it is more likely to write to the most recently written line than it is to write to the LRU robust line. We model this mechanism's overhead by blocking access to the cache for three cycles (L1) or six cycles (L2 and L3 that have 32-byte accesses) to account for using the cache read and write ports to perform the swap.

Duplication

To avoid writeback traffic and the additional swap latency, we explore trading off capacity to save this overhead. In this mechanism, we assign each two consecutive non-robust lines as "partner lines" similar to. [21] For example, in Figure 5's L1 cache, the line in way 2 is a partner line to that in way 3, the line in way 4 is a partner line to that in way 5, and the line in way 6 is a partner line to that of way 7. When a write occurs to a non-robust line, we evict its partner line and write the data to both lines, using two extra cycles. We modify the replacement algorithm so that the partner line is always invalid and not a candidate for replacement. This duplication causes losing some cache capacity, but avoids writeback traffic and swap overhead. When writing to a duplicate line, we perform the write to both the original line and its partner. When reading from a duplicate line, we check parity (L1) or ECC (L2/L3), and trigger a read from the partner line if an error is detected.

Evaluation Summary

We evaluated a power-constrained system with the ability to operate one, two, and four cores within the same power budget using CMP$im[7] and SPEC benchmarks.[19] A more detailed analysis of our results is presented in.[8] To support four active cores, our hypothetical system used a mixed-cell cache architecture to operate at 590 mV. In this mode, the 75 percent capacity loss experienced by our baseline mixed-cell cache architecture resulted in a 12 percent performance loss. Our proposal delivers a 9.5 percent performance benefit relative to a non-mixed cell baseline using only robust memories, which is similar to the performance improvement for our mechanism in the section "LLC Implementation Using Heterogeneous Cell Sizes to Support Low Vmin." However, our design avoids significant reductions in cache size at low voltage, improving multi-core performance by up to 17 percent on average and saving 50 percent of the L1 dynamic power compared to using only robust cells. While the writeback mechanism incurs significant overheads, both swap and duplication achieve significant performance improvements and power reductions.

"…our design avoids significant reductions in cache size at low voltage…"

"…improving multi-core performance by up to 17 percent on average and saving 50 percent of the L1 dynamic power…"

Conclusions

In this article, we showed that mixed-cell memory designs could play a key role in achieving the right balance between performance, power, and reliability for single-core and multi-core systems. For mixed-cell designs, part of the memory structure is built with cells that are more robust and failure-resistant, while the rest is designed using traditional cells. Robust cells ensure resiliency under low-voltage conditions to protect the most vulnerable data, while the rest of the memory structure could be used to store redundant data to improve performance.

We showed how this concept works using two specific examples. First, our heterogeneous LLC system only turns on the robust portion at low voltage, while using the whole cache at high voltage. This mechanism achieves significant power savings at low voltage and significant performance improvements at high voltage compared to a uniformly robust cache design. Second, our multi-core mixed cell architecture uses the whole cache (including the non-robust portion) at low voltage while ensuring modified data is not lost. This mechanism enables a multi-core system where all cores are active in a TDP-limited design and achieves significant performance improvements and power savings at low voltage. The same concept could be used throughout the entire memory hierarchy to improve memory resiliency without sacrificing performance or energy efficiency.

"This mechanism enables a multi-core system where all cores are active in a TDP-limited design…"

Acknowledgements

We are very grateful to Aamer Jaleel who helped us with his simulator, CMP$im. Work at the University of Wisconsin was supported by an NSF grant (CCF-1016262), a Fall Competition Multidisciplinary Research Award from the University of Wisconsin-Madison Graduate School, and NSF CAREER Award (CCF-0953603).

References

[1] M. Agostinelli, et al., "Erratic fluctuations of SRAM cache Vmin at the 90 nm process technology node," IEDM Technical Digest, pp. 655–658, Dec. 2005.

[2] Arup Chakraborty, et al., "E < MC^2: Less Energy through Multi-Copy Cache," Conference on Compilers, Architecture, and Synthesis for Embedded Systems (CASES), pp. 237–246, Oct. 2010.

[3] Zeshan Chishti, et al., "Improving Cache Lifetime Reliability at Ultra-low Voltages," International Symposium on Microarchitecture, pp. 89–99, Dec. 2009.

[4] Ronald G. Dreslinski, et al., "Reconfigurable Energy Efficient Near Threshold Cache Architectures," International Symposium on Microarchitecture, pp. 459–470, Dec. 2008.

[5] Hamid Reza Ghasemi, Stark Draper, and Nam Sung Kim, "Low-Voltage On-Chip Cache Architecture using Heterogeneous Cell Sizes for Multi-Core Processors," International Symposium on High-Performance Computer Architecture, pp. 38–49, Feb. 2011.

[6] Intel Corporation, "Intel® Turbo Boost Technology in Intel® Core™ Microarchitecture (Nehalem) Based Processors," White Paper, 2008.

[7] Aamer Jaleel, et al., "CMP$im: A Pin-Based On-The-Fly Multi-Core Cache Simulator," 4th Workshop on Modeling, Benchmarking and Simulation, Beijing, China, June 2008.

[8] Samira M. Khan, et al., "Improving Multi-Core Performance Using Mixed-Cell Cache Architecture," International Symposium on High-Performance Computer Architecture (HPCA), February 2013.

[9] Muhammad M. Khellah, et al., "Read and Write Circuit Assist Techniques for Improving Vccmin of Dense 6T SRAM Cell," Conf. on Int. Circuit Design and Technology, pp. 185–189, June 2008.

[10] Jangwoo Kim, et al., "Multi-bit Error Tolerant Caches Using Two-Dimensional Error Coding," International Symposium on Microarchitecture, pp. 197–209, Dec. 2007.

[11] Jaydeep Kulkarni and Kaushik Roy, "Ultra-low Voltage Process Variation Tolerant Schmitt Trigger based SRAM Design," IEEE Transactions on VLSI Systems, 2011.

[12] P. Magnusson et al., "Simics: A full system simulation platform," IEEE Computer, 35(2): 50–58, Feb. 2002.

[13] Wai-Kei Mak and Jr-Wei Chen, "Voltage Island Generation under Performance Requirement for SoC Designs," Asia and South Pacific Design Automation Conference, Jan. 2007.

[14] M. Martin et al., "Multifacet's General Execution-Driven Multiprocessor Simulator (GEMS) Toolset," Computer Architecture News, 33(4): 92–99, Sep. 2005.

[15] A. Naveh et al., "Power and thermal management in the Intel® Core Duo Processor," Intel Technology Journal, 10(2): 109–122, May 2006.

[16] Jaehyun Park, et al., "Accurate Modeling and Calculation of Delay and Energy Overheads of Dynamic Voltage Scaling in Modern High-Performance Microprocessors," International Symposium on Low-Power Electronics and Design (ISLPED), pp. 419–424, Aug. 2010.

[17] David Roberts, Nam Sung Kim and Trevor Mudge, "On-Chip Cache Device Scaling Limits and Effective Fault Repair Techniques in Future Nanoscale Technology," Digital System Design Architectures, Methods and Tools, pp. 570–578, Aug. 2007.

[18] Stanley Schuster, "Multiple Word/Bit Line Redundancy for Semiconductor Memories," IEEE Journal of Solid-State Circuits, vol. 13, no. 5, pp. 698–703, Oct. 1978.

[19] SPEC CPU2006 Benchmarks, http://www.spec.org/cpu2006/

[20] Chris Wilkerson, et al., "Trading off Cache Capacity for Reliability to Enable Low Voltage Operation," International Symposium on Computer Architecture, pp. 203–214, June 2008.

[21] Wei Zhang, et al., "ICR: In-Cache Replication for Enhancing Data Cache Reliability," International Conference on Dependable Systems and Networks, pp. 291–300, June 2003.

[22] S.-T. Zhou et al., "Minimizing total area of low-voltage SRAM arrays through joint optimization of cell size, redundancy, and ECC," Proc. IEEE International Symposium on Computer Design, pp. 112–117, Oct. 2010.

Author Biographies

Alaa R. Alameldeen received BSc and MSc degrees from Alexandria University, Egypt, in 1996 and 1999, respectively, and MSc and PhD degrees from the University of Wisconsin–Madison, in 2000 and 2006, respectively, all in computer science. He is a research scientist at Intel Labs, Hillsboro, Oregon. His current research focuses on energy-efficient memory and cache design.

Nam Sung Kim is an assistant professor at the University of Wisconsin–Madison. He has been conducting interdisciplinary research that cuts across device, circuit, and architecture for power-efficient computing. His research has been supported by National Science Foundation (NSF), Semiconductor Research Corporation (SRC), Defense Advanced Research Project Agency (DARPA), AMD, IBM, Samsung, Microsoft, and Korean Ministry of Education, Science, and Technology. Prior to joining the University of Wisconsin–Madison, he was a senior research scientist at Intel from 2004 to 2008, where he conducted research in power-efficient digital circuits and processor architecture. He also has served several prominent international conferences as a technical program committee member. Nam Sung Kim has been the recipient of IEEE Design Automation Conference (DAC) Student Design Contest Award in 2001, Intel Fellowship in 2002, and IEEE International Conference on Microarchitecture (MICRO) Best Paper Award in 2003, NSF CAREER Award in 2010, and IBM Faculty Award in 2011 and 2012. His current research interest is designing robust, low-power computing systems in nanoscale technology. He is an IEEE senior member

and holds a PhD in Computer Science and Engineering from the University of Michigan–Ann Arbor, and both an MS and a BS in Electrical Engineering from Korea Advanced Institute of Science and Technology.

Samira M. Khan is a post-doctoral researcher associated with Intel Labs and Carnegie Mellon University. She received her PhD from the University of Texas at San Antonio. Her research focuses on new microarchitectural designs that can make future microprocessors fast and energy efficient. She received her BSc degree from Bangladesh University of Engineering and Technology.

Hamid Reza Ghasemi is a PhD candidate at the Computer Sciences Department of University of Wisconsin–Madison. He received a BS and MS degree from the Electrical and Computer Engineering Department of University of Tehran. His research interests include computer architecture, power efficient high-performance processing, and low power architecture.

Chris Wilkerson received a master's degree from Carnegie Mellon University in 1996. He is currently a research scientist at Intel Labs, Hillsboro, Oregon. He has authored or coauthored a number of papers published on microarchitectural topics including value prediction, branch prediction, cache organization, and runahead and advanced speculative execution. His current research interests include microarchitectural mechanisms to enable low-power operation for microprocessors.

Jaydeep Kulkarni received a BE degree from the University of Pune, India, in 2002, an M. Tech. degree from the Indian Institute of Science (IISc), Bangalore, India, in 2004, and a PhD from Purdue University, in 2009, all in electrical engineering. During 2004–2005, he was with Cypress Semiconductors, Bangalore, India, where he was involved with low-power SRAM design. He is currently with the Circuit Research Lab, Intel Corporation, Hillsboro, Oregon, working on embedded memories, low-voltage circuits, and power management circuits. Dr. Kulkarni was the recipient of the Best M. Tech Student Award from IISc Bangalore in 2004, two SRC Inventor Recognition Awards, the 2008 ISLPED Design Contest Award, the 2008 Intel Foundation PhD Fellowship Award, and the Best Paper in Session Award at 2008 SRC TECHCON.

Daniel A. Jiménez is an Associate Professor in the Department of Computer Science and Engineering at Texas A&M University. Previously he was a full Professor and Chair of the Department of Computer Science at The University of Texas at San Antonio and Associate Professor in the Department of Computer Science at Rutgers University. His research focuses on microarchitecture and low-level compiler optimizations. He introduced and developed the perceptron branch predictor which has inspired the design of two implemented microarchitectures: the AMD "Bobcat" core and the Oracle SPARC T4. Daniel earned his BS (1992) and MS (1994) in Computer Science at The University of Texas at San Antonio and his PhD (2002) in Computer Sciences at The University of Texas at Austin. From

2002 through 2007, Daniel was an Assistant Professor in the Department of Computer Science at Rutgers. In 2005 Daniel took sabbatical leave at the Technical University of Catalonia (UPC) in Barcelona, Catalonia, Spain. In 2008 he was promoted to Associate Professor with tenure at Rutgers. He returned to his native Texas to take a position at UT San Antonio. He recently returned from a second sabbatical leave in Spain at the Barcelona Supercomputing Center and was General Chair of the 2011 IEEE HPCA conference. He is an NSF CAREER award recipient and ACM Distinguished Scientist.