1. **Awasthi, M.; Shevgoor, M.; Sudan, K.; Rajendran, B.; Balasubramonian, R.; Srinivasan, V.; , "Efficient scrub mechanisms for error-prone emerging memories,"** ***High Performance Computer Architecture (HPCA), 2012 IEEE 18th International Symposium on*** **, vol., no., pp.1-12, 25-29 Feb. 2012**
   doi: 10.1109/HPCA.2012.6168941
   Abstract: Many memory cell technologies are being considered as possible replacements for DRAM and Flash technologies, both of which are nearing their scaling limits. While these new cells (PCM, STT-RAM, FeRAM, etc.) promise high density, better scaling, and non-volatility, they introduce new challenges. Solutions at the architecture level can help address some of these problems; e.g., prior research has proposed wear-leveling and hard error tolerance mechanisms to overcome the limited write endurance of PCM cells. In this paper, we focus on the soft error problem in PCM, a topic that has received little attention in the architecture community. Soft errors in DRAM memories are typically addressed by having SECDED support and a scrub mechanism. The scrub mechanism scans the memory looking for a single-bit error and corrects it before the line experiences a second uncorrectable error. However, PCM (and other emerging memories) are prone to new sources of soft errors. In particular, multi-level cell (MLC) PCM devices will suffer from resistance drift, that increases the soft error rate and incurs high overheads for the scrub mechanism. This paper is the first to study the design of architectural scrub mechanisms, especially when tailored to the drift phenomenon in MLC PCM. Many of our solutions will also apply to other soft-error prone emerging memories. We first show that scrub overheads can be reduced with support for strong ECC codes and a lightweight error detection operation. We then design different scrub algorithms that can adaptively trade-off soft and hard errors. Using an approach that combines all proposed solutions, our scrub mechanism yields a 96.5% reduction in uncorrectable errors, a 24.4 × decrease in scrub-related writes, and a 37.8% reduction in scrub energy, relative to a basic scrub algorithm used in modern DRAM systems.
   URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6168941&isnumber=6168936
2. **Negi, A.; Titos-Gil, R.; Acacio, M.E.; Garcia, J.M.; Stenstrom, P.; , "π-TM: Pessimistic invalidation for scalable lazy hardware transactional memory," High Performance Computer Architecture (HPCA), 2012 IEEE 18th International Symposium on , vol., no., pp.1-12, 25-29 Feb. 2012**
   doi: 10.1109/HPCA.2012.6168951
   Abstract: Lazy hardware transactional memory has been shown to be more efficient at extracting available concurrency than its eager counterpart. However, it poses scalability challenges at commit time as existence of conflicts among concurrent transactions is not known prior to commit. Non-conflicting transactions may have to wait before committing, severely affecting performance in certain workloads. Early conflict detection can be employed to allow such transactions to commit simultaneously. In this paper we show that the potential of this technique has not yet been fully utilized, with design choices in prior work severely burdening common-case transactional execution to avoid some relatively uncommon correctness concerns. The paper quantifies the severity of the problem and develops μ-TM, an early conflict detection - lazy conflict resolution design. This design highlights how, with modest extensions to existing directory-based coherence protocols, information regarding possible conflicts can be effectively used to achieve true parallelism at commit without burdening the common-case. We leverage the observation that contention is typically seen on only a small fraction of shared data accessed by coarse-grained transactions. Pessimistic invalidation of such lines when committing or aborting, therefore, enables fast common-case execution. Our results show that μ-TM performs consistently well and, in particular, far better than previous work on early conflict detection in lazy HTM. We also identify a pathological scenario that lazy designs with early conflict detection suffer from and propose a simple hardware workaround to sidestep it.
   URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6168951&isnumber=6168936
3. **Xuehai Qian; Sahelices, B.; Torrellas, J.; , "BulkSMT: Designing SMT processors for atomic-block execution," High Performance Computer Architecture (HPCA), 2012 IEEE 18th International Symposium on , vol., no., pp.1-12, 25-29 Feb. 2012**
   doi: 10.1109/HPCA.2012.6168952
   Abstract: Multiprocessor architectures that continuously execute atomic blocks (or chunks) of instructions can improve performance and software productivity. However, all of the prior proposals for such architectures assume single-context cores as building blocks - rather than the widely-used Simultaneous Multithreading (SMT) cores. As a result, they are underutilizing hardware resources. This paper presents the first SMT design that supports continuous chunked (or transactional) execution of its contexts. Our design, called BulkSMT, can be used either in a single-core processor or in a multicore of SMTs. We present a set of BulkSMT configurations with different cost and performance. We also describe the architectural primitives that enable chunked execution in an SMT core and in a multicore of SMTs. Our results, based on simulations of SPLASH-2 and PARSEC codes, show that BulkSMT supports chunked execution cost-effectively. In a 4-core multicore with eager chunked execution, BulkSMT reduces the execution time of the applications by an average of 26% compared to running on single-context cores. In a single core, the average reduction is 32%.
   URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6168952&isnumber=6168936
4. **Sheng Ma; Jerger, N.E.; Zhiying Wang; , "Supporting efficient collective communication in NoCs," High Performance Computer Architecture (HPCA), 2012 IEEE 18th International Symposium on , vol., no., pp.1-12, 25-29 Feb. 2012**
   doi: 10.1109/HPCA.2012.6168953
   Abstract: Across many architectures and parallel programming paradigms, collective communication plays a key role in performance and correctness. Hardware support is necessary to prevent important collective communication from becoming a system bottleneck. Support for multicast communication in Networks-on-Chip (NoCs) has achieved substantial throughput improvements and power savings. In this paper, we explore support for reduction or many-to-one communication operations. As a case study, we focus on acknowledgement messages (ACK) that must be collected in a directory protocol before a cache line may be upgraded to or installed in the modified state. This paper makes two primary contributions: an efficient framework to support the reduction of ACK packets and a novel Balanced, Adaptive Multicast (BAM) routing algorithm. The proposed message combination framework complements several multicast algorithms. By combining ACK packets during transmission, this framework not only reduces packet latency by 14.1% for low-to-medium network loads, but also improves the network saturation throughput by 9.6% with little overhead. The balanced buffer resource configuration of BAM improves the saturation throughput by an additional 13.8%. For the PARSEC benchmarks, our design offers an average speedup of 12.7% and a maximal speedup of 16.8%.
   URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6168953&isnumber=6168936

5. **Lei Jiang; Bo Zhao; Youtao Zhang; Jun Yang; Childers, B.R.; , "Improving write operations in MLC phase change memory," High Performance Computer Architecture (HPCA), 2012 IEEE 18th International Symposium on , vol., no., pp.1-10, 25-29 Feb. 2012**
   doi: 10.1109/HPCA.2012.6169027
   Abstract: Phase change memory (PCM) recently has emerged as a promising technology to meet the fast growing demand for large capacity memory in modern computer systems. In particular, multi-level cell (MLC) PCM that stores multiple bits in a single cell, offers high density with low per-byte fabrication cost. However, despite many advantages, such as good scalability and low leakage, PCM suffers from exceptionally slow write operations, which makes it challenging to be integrated in the memory hierarchy. In this paper, we propose architectural innovations to improve the access time of MLC PCM. Due to cell process variation, composition fluctuation and the relatively small differences among resistance levels, MLC PCM typically employs an iterative write scheme to achieve precise control, which suffers from large write access latency. To address this issue, we propose write truncation (WT) to reduce the number of write iterations with the assistance of an extra error correction code (ECC). We also propose form switch (FS) to reduce the storage overhead of the ECC. By storing highly compressible lines in SLC form, FS improves read latency as well. Our experimental results show that WT and FS improve the effective write/read latency by 57%/28% respectively, and achieve 26% performance improvement over the state of the art.
   URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6169027&isnumber=6168936
6. **Raghavan, A.; Yixin Luo; Chandawalla, A.; Papaefthymiou, M.; Pipe, K.P.; Wenisch, T.F.; Martin, M.M.K.; , "Computational sprinting," High Performance Computer Architecture (HPCA), 2012 IEEE 18th International Symposium on , vol., no., pp.1-12, 25-29 Feb. 2012**
   doi: 10.1109/HPCA.2012.6169031
   Abstract: Although transistor density continues to increase, voltage scaling has stalled and thus power density is increasing each technology generation. Particularly in mobile devices, which have limited cooling options, these trends lead to a utilization wall in which sustained chip performance is limited primarily by power rather than area. However, many mobile applications do not demand sustained performance; rather they comprise short bursts of computation in response to sporadic user activity. To improve responsiveness for such applications, this paper explores activating otherwise powered-down cores for sub-second bursts of intense parallel computation. The approach exploits the concept of computational sprinting, in which a chip temporarily exceeds its sustainable thermal power budget to provide instantaneous throughput, after which the chip must return to nominal operation to cool down. To demonstrate the feasibility of this approach, we analyze the thermal and electrical characteristics of a smart-phone-like system that nominally operates a single core (~1W peak), but can sprint with up to 16 cores for hundreds of milliseconds. We describe a thermal design that incorporates phase-change materials to provide thermal capacitance to enable such sprints. We analyze image recognition kernels to show that parallel sprinting has the potential to achieve the task response time of a 16W chip within the thermal constraints of a 1W mobile platform.
   URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6169031&isnumber=6168936
7. **Martin, R.; Demme, J.; Sethumadhavan, S.; , "TimeWarp: Rethinking timekeeping and performance monitoring mechanisms to mitigate side-channel attacks," Computer Architecture (ISCA), 2012 39th Annual International Symposium on , vol., no., pp.118-129, 9-13 June 2012**
   doi: 10.1109/ISCA.2012.6237011
   Abstract: Over the past two decades, several microarchitectural side channels have been exploited to create sophisticated security attacks. Solutions to this problem have mainly focused on fixing the source of leaks either by limiting the flow of information through the side channel by modifying hardware, or by refactoring vulnerable software to protect sensitive data from leaking. These solutions are reactive and not preventative: while the modifications may protect against a single attack, they do nothing to prevent future side channel attacks that exploit other microarchitectural side channels or exploit the same side channel in a novel way. In this paper we present a general mitigation strategy that focuses on the infrastructure used to measure side channel leaks rather than the source of leaks, and thus applies to all known and unknown microarchitectural side channel leaks. Our approach is to limit the fidelity of fine grain timekeeping and performance counters, making it difficult for an attacker to distinguish between different microarchitectural events, thus thwarting attacks. We demonstrate the strength of our proposed security modifications, and validate that our changes do not break existing software. Our proposed changes require minor - or in some cases, no - hardware modifications and do not result in any substantial performance degradation, yet offer the most comprehensive protection against microarchitectural side channels to date.
   URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6237011&isnumber=6236993
8. **Ting Cao; Blackburn, S.M.; Tiejun Gao; McKinley, K.S.; , "The Yin and Yang of power and performance for asymmetric hardware and managed software," Computer Architecture (ISCA), 2012 39th Annual International Symposium on , vol., no., pp.225-236, 9-13 June 2012**
   doi: 10.1109/ISCA.2012.6237020
   Abstract: On the hardware side, asymmetric multicore processors present software with the challenge and opportunity of optimizing in two dimensions: performance and power. Asymmetric multicore processors (AMP) combine general-purpose big (fast, high power) cores and small (slow, low power) cores to meet power constraints. Realizing their energy efficiency opportunity requires workloads with differentiated performance and power characteristics.
   URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6237020&isnumber=6236993
9. **Doudalis, I.; Prvulovic, M.; , "Euripus: A flexible unified hardware memory checkpointing accelerator for bidirectional-debugging and reliability," Computer Architecture (ISCA), 2012 39th Annual International Symposium on , vol., no., pp.261-272, 9-13 June 2012**
   doi: 10.1109/ISCA.2012.6237023
   Abstract: Bidirectional debugging and error recovery have different goals (programmer productivity and system reliability, respectively), yet they both require the ability to roll-back the program or the system to a past state. This rollback functionality is typically implemented using checkpoints that can restore the system/application to a specific point in time. There are several types of checkpoints, and bidirectional debugging and error-recovery use them in different ways. This paper presents Euripus1, a flexible hardware accelerator for memory checkpointing which can create different combinations of checkpoints needed for bidirectional debugging, error recovery, or both. In particular, Euripus is the first hardware technique to provide consolidation-friendly undo-logs (for bidirectional debugging), to allow simultaneous construction of both undo and redo logs, and to support multi-level checkpointing for the needs of error-recovery. Euripus incurs low performance overheads (<;5% on average), improves roll-back latency for bidirectional debugging by >;30%, and supports rapid multi-level error recovery that allows >;95% system efficiency even with very high error rates.

URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6237023&isnumber=6236993

10. **Pellegrini, A.; Greathouse, J.L.; Bertacco, V.; , "Viper: Virtual pipelines for enhanced reliability," Computer Architecture (ISCA), 2012 39th Annual International Symposium on , vol., no., pp.344-355, 9-13 June 2012**
doi: 10.1109/ISCA.2012.6237030
Abstract: The reliability of future processors is threatened by decreasing transistor robustness. Current architectures focus on delivering high performance at low cost; lifetime device reliability is a secondary concern. As the rate of permanent hardware faults increases, robustness will become a first class constraint for even low-cost systems. Current research into reliable architectures has focused on ad-hoc solutions to improve designs without altering their centralized control logic. Unfortunately, this centralized control presents a single point of failure, which limits long-term robustness.To address this issue, we introduce Viper, an architecture built from a redundant collection offine-grained hardware components. Instructions are perceived as customers that require a sequence of services in order to properly execute. The hardware components vie to perform what services they can, dynamically forming virtual pipelines that avoid defective hardware. This is done using distributed control logic, which avoids a single point offailure by construction. Viper can tolerate a high number of permanent faults due to its inherent redundancy. As fault counts increase, its performance degrades more gracefully than traditional centralized-logic architectures. We estimate that fault rates higher than one permanentfaults per 12 million transistors, on average, cause the throughput of a classic CMP design to fall below that of a Viper design of similar size.
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6237030&isnumber=6236993

11. **Qureshi, M.K.; Franceschini, M.M.; Jagmohan, A.; Lastras, L.A.; , "PreSET: Improving performance of phase change memories by exploiting asymmetry in write times," Computer Architecture (ISCA), 2012 39th Annual International Symposium on , vol., no., pp.380-391, 9-13 June 2012**
doi: 10.1109/ISCA.2012.6237033
Abstract: Phase Change Memory (PCM) is a promising technology for building future main memory systems. A prominent characteristic of PCM is that it has write latency much higher than read latency. Servicing such slow writes causes significant contention for read requests. For our baseline PCM system, the slow writes increase the effective read latency by almost 2X, causing significant performance degradation. This paper alleviates the problem of slow writes by exploiting the fundamental property of PCM devices that writes are slow only in one direction (SET operation) and are almost as fast as reads in the other direction (RESET operation). Therefore, a write operation to a line in which all memory cells have been SET prior to the write, will incur much lower latency. We propose PreSET, an architectural technique that leverages this property to pro-actively SET all the bits in a given memory line well in advance of the anticipated write to that memory line. Our proposed design initiates a PreSET request for a memory line as soon as that line becomes dirty in the cache, thereby allowing a large window of time for the PreSET operation to complete. Our evaluations show that PreSET is more effective and incurs lower storage overhead than previously proposed write cancellation techniques. We also describe static and dynamic throttling schemes to limit the rate of PreSET operations. Our proposal reduces effective read latency from 982 cycles to 594 cycles and increases system performance by 34%, while improving the energy-delay-product by 25%.
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6237033&isnumber=6236993

12. **Satish, N.; Kim, C.; Chhugani, J.; Saito, H.; Krishnaiyer, R.; Smelyanskiy, M.; Girkar, M.; Dubey, P.; , "Can traditional programming bridge the Ninja performance gap for parallel computing applications?,"Computer Architecture (ISCA), 2012 39th Annual International Symposium on , vol., no., pp.440-451, 9-13 June 2012**
doi: 10.1109/ISCA.2012.6237038
Abstract: Current processor trends of integrating more cores with wider SIMD units, along with a deeper and complex memory hierarchy, have made it increasingly more challenging to extract performance from applications. It is believed by some that traditional approaches to programming do not apply to these modern processors and hence radical new languages must be discovered. In this paper, we question this thinking and offer evidence in support of traditional programming methods and the performance-vs-programming effort effectiveness of common multi-core processors and upcoming manycore architectures in delivering significant speedup, and close-to-optimal performance for commonly used parallel computing workloads. We first quantify the extent of the “Ninja gap”, which is the performance gap between naively written C/C++ code that is parallelism unaware (often serial) and best-optimized code on modern multi-/many-core processors. Using a set of representative throughput computing benchmarks, we show that there is an average Ninja gap of 24X (up to 53X) for a recent 6-core Intel® Core™ i7 X980 Westmere CPU, and that this gap if left unaddressed will inevitably increase. We show how a set of well-known algorithmic changes coupled with advancements in modern compiler technology can bring down the Ninja gap to an average of just 1.3X. These changes typically require low programming effort, as compared to the very high effort in producing Ninja code. We also discuss hardware support for programmability that can reduce the impact of these changes and even further increase programmer productivity. We show equally encouraging results for the upcoming Intel® Many Integrated Core architecture (Intel® MIC) which has more cores and wider SIMD. We thus demonstrate that we can contain the otherwise uncontrolled growth of the Ninja gap and offer a more stable and predictable performance growth over future archit- ctures, offering strong evidence that radical language changes are not required.
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6237038&isnumber=6236993

13. **Lotfi-Kamran, P.; Grot, B.; Ferdman, M.; Volos, S.; Kocberber, O.; Picorel, J.; Adileh, A.; Jevdjic, D.; Idgunji, S.; Ozer, E.; Falsafi, B.; , "Scale-out processors," Computer Architecture (ISCA), 2012 39th Annual International Symposium on , vol., no., pp.500-511, 9-13 June 2012**
doi: 10.1109/ISCA.2012.6237043
Abstract: Scale-out datacenters mandate high per-server throughput to get the maximum benefit from the large TCO investment. Emerging applications (e.g., data serving and web search) that run in these datacenters operate on vast datasets that are not accommodated by on-die caches of existing server chips. Large caches reduce the die area available for cores and lower performance through long access latency when instructions are fetched. Performance on scale-out workloads is maximized through a modestly-sized last-level cache that captures the instruction footprint at the lowest possible access latency. In this work, we introduce a methodology for designing scalable and efficient scale-out server processors. Based on a metric of performance-density, we facilitate the design of optimal multi-core configurations, called pods. Each pod is a complete server that tightly couples a number of cores to a small last-level cache using a fast interconnect. Replicating the pod to fill the die area yields processors which have optimal performance density, leading to maximum per-chip throughput. Moreover, as each pod is a stand-alone server, scale-out processors avoid the expense of global (i.e., interpod) interconnect and coherence. These features synergistically maximize throughput, lower design complexity, and improve technology scalability. In 20nm technology, scaleout chips improve throughput by 5x-6.5x over conventional and by 1.6x-1.9x over emerging tiled organizations.

URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6237043&isnumber=6236993

14. **Singh, A.; Narayanasamy, S.; Marino, D.; Millstein, T.; Musuvathi, M.; , "End-to-end sequential consistency," Computer Architecture (ISCA), 2012 39th Annual International Symposium on , vol., no., pp.524-535, 9-13 June 2012**
doi: 10.1109/ISCA.2012.6237045
Abstract: Sequential consistency (SC) is arguably the most intuitive behavior for a shared-memory multithreaded program. It is widely accepted that language-level SC could significantly improve programmability of a multiprocessor system. However, efficiently supporting end-to-end SC remains a challenge as it requires that both compiler and hardware optimizations preserve SC semantics. While a recent study has shown that a compiler can preserve SC semantics for a small performance cost, an efficient and complexity-effective SC hardware remains elusive. Past hardware solutions relied on aggressive speculation techniques, which has not yet been realized in a practical implementation. This paper exploits the observation that hardware need not enforce any memory model constraints on accesses to thread-local and shared read-only locations. A processor can easily determine a large fraction of these safe accesses with assistance from static compiler analysis and the hardware memory management unit. We discuss a low-complexity hardware design that exploits this information to reduce the overhead in ensuring SC. Our design employs an additional unordered store buffer for fast-tracking thread-local stores and allowing later memory accesses to proceed without a memory ordering related stall. Our experimental study shows that the cost of guaranteeing end-to-end SC is only 6.2% on average when compared to a system with TSO hardware executing a stock compiler's output.
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6237045&isnumber=6236993

15. **Chao Li; Qouneh, A.; Tao Li; , "iSwitch: Coordinating and optimizing renewable energy powered server clusters," Computer Architecture (ISCA), 2012 39th Annual International Symposium on , vol., no., pp.512-523, 9-13 June 2012**
doi: 10.1109/ISCA.2012.6237044
Abstract: Large-scale computing systems such as data centers are facing increasing pressure to cap their carbon footprint. Integrating emerging clean energy solutions into computer system design therefore gains great significance in the green computing era. While some pioneering work on tracking variable power budget show promising energy efficiency, they are not suitable for data centers due to lack of performance guarantee when renewable generation is low and fluctuant. In addition, our characterization of wind power behavior reveals that data centers designed to track the intermittent renewable power incur up to 4X performance loss due to inefficient and redundant load matching activities. As a result, mitigating operational overhead while still maintaining desired energy utilization becomes the most significant challenge in managing server clusters on intermittent renewable energy generation. In this paper we take a first step in digging into the operational overhead of renewable energy powered data center. We propose iSwitch, a lightweight server power management that follows renewable power variation characteristics, leverages existing system infrastructures, and applies supply/load cooperative scheme to mitigate the performance overhead. Comparing with state-of-the-art renewable energy driven system design, iSwitch could mitigate average network traffic by 75%, peak network traffic by 95%, and reduce 80% job waiting time while still maintaining 96% renewable energy utilization. We expect that our work can help computer architects make informed decisions on sustainable and high-performance system design.
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6237044&isnumber=6236993

16. **Dushyanth Narayanan and Orion Hodson. 2012. Whole-system persistence. In Proceedings of the seventeenth international conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '12). ACM, New York, NY, USA, 401-410.**
DOI=10.1145/2150976.2151018 http://doi.acm.org/10.1145/2150976.2151018
Today's databases and key-value stores commonly keep all their data in main memory. A single server can have over 100 GB of memory, and a cluster of such servers can have 10s to 100s of TB. However, a storage back end is still required for recovery from failures. Recovery can last for minutes for a single server or hours for a whole cluster, causing heavy load on the back end. Non-volatile main memory (NVRAM) technologies can help by allowing near-instantaneous recovery of in-memory state. However, today's software does not support this well. Block-based approaches such as persistent buffer caches suffer from data duplication and block transfer overheads. Recently, user-level persistent heaps have been shown to have much better performance than these. However they require substantial application modification and still have significant runtime overheads. This paper proposes whole-system persistence (WSP) as an alternative. WSP is aimed at systems where all memory is non-volatile. It transparently recovers an application's entire state, making a failure appear as a suspend/resume event. Runtime overheads are eliminated by using "flush on fail": transient state in processor registers and caches is flushed to NVRAM only on failure, using the residual energy from the system power supply. Our evaluation shows that this approach has 1.6--13 times better runtime performance than a persistent heap, and that flush-on-fail can complete safely within 2--35\% of the residual energy window provided by standard power supplies.

17. **Jichuan Chang, Justin Meza, Parthasarathy Ranganathan, Amip Shah, Rocky Shih, and Cullen Bash. 2012. Totally green: evaluating and designing servers for lifecycle environmental impact. SIGARCH Comput. Archit. News 40, 1 (March 2012), 25-36. DOI=10.1145/2189750.2150980** http://doi.acm.org/10.1145/2189750.2150980
The environmental impact of servers and datacenters is an important future challenge. System architects have traditionally focused on operational energy as a proxy for designing green servers, but this ignores important environmental implications from server production (materials, manufacturing, etc.). In contrast, this paper argues for a lifecycle focus on the environmental impact of future server designs, to include both operation and production. We present a new methodology to quantify the total environmental impact of system design decisions. Our approach uses the thermodynamic metric of exergy consumption, adapted and validated for use by system architects. Using this methodology, we evaluate the lifecycle impact of several example system designs with environment-friendly optimizations. Our results show that environmental impact from production can be important (around 20% on current servers and growing) and system design choices can reduce this component (by 30--40%). Our results also highlight several, sometimes unexpected, cross-interactions between the environmental impact of production and operation that further motivate a total lifecycle emphasis for future green server designs.

18. **Hadi Esmaeilzadeh, Adrian Sampson, Luis Ceze, and Doug Burger. 2012. Architecture support for disciplined approximate programming. In Proceedings of the seventeenth international conference on Architectural Support for Programming Languages and Operating Systems(ASPLOS '12). ACM, New York, NY, USA, 301-312. DOI=10.1145/2150976.2151008 http://doi.acm.org/10.1145/2150976.2151008**

Disciplined approximate programming lets programmers declare which parts of a program can be computed approximately and consequently at a lower energy cost. The compiler proves statically that all approximate computation is properly isolated from precise computation. The hardware is then free to selectively apply approximate storage and approximate computation with no need to perform dynamic correctness checks. In this paper, we propose an efficient mapping of disciplined approximate programming onto hardware. We describe an ISA extension that provides approximate operations and storage, which give the hardware freedom to save energy at the cost of accuracy. We then propose Truffle, a microarchitecture design that efficiently supports the ISA extensions. The basis of our design is dual-voltage operation, with a high voltage for precise operations and a low voltage for approximate operations. The key aspect of the microarchitecture is its dependence on the instruction stream to determine when to use the low voltage. We evaluate the power savings potential of in-order and out-of-order Truffle configurations and explore the resulting quality of service degradation. We evaluate several applications and demonstrate energy savings up to 43%.

19. **Asim Kadav and Michael M. Swift. 2012. Understanding modern device drivers. In Proceedings of the seventeenth international conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '12). ACM, New York, NY, USA, 87-98.**
DOI=10.1145/2150976.2150987 http://doi.acm.org/10.1145/2150976.2150987
Device drivers are the single largest contributor to operating-system kernel code with over 5 million lines of code in the Linux kernel, and cause significant complexity, bugs and development costs. Recent years have seen a flurry of research aimed at improving the reliability and simplifying the development of drivers. However, little is known about what constitutes this huge body of code beyond the small set of drivers used for research. In this paper, we study the source code of Linux drivers to understand what drivers actually do, how current research applies to them and what opportunities exist for future research. We determine whether assumptions made by most driver research, such as that all drivers belong to a class, are indeed true. We also analyze driver code and abstractions to determine whether drivers can benefit from code re-organization or hardware trends. We develop a set of static-analysis tools to analyze driver code across various axes. Broadly, our study looks at three aspects of driver code (i) what are the characteristics of driver code functionality and how applicable is driver research to all drivers, (ii) how do drivers interact with the kernel, devices, and buses, and (iii) are there similarities that can be abstracted into libraries to reduce driver size and complexity? We find that many assumptions made by driver research do not apply to all drivers. At least 44% of drivers have code that is not captured by a class definition, 28% of drivers support more than one device per driver, and 15% of drivers do significant computation over data. From the driver interactions study, we find USB bus offers an efficient bus interface with significant standardized code and coarse-grained access, ideal for executing drivers in isolation. We also find that drivers for different buses and classes have widely varying levels of device interaction, which indicates that the cost of isolation will vary by class. Finally, from our driver similarity study, we find 8% of all driver code is substantially similar to code elsewhere and may be removed with new abstractions or libraries.

20. **Andy A. Hwang, Ioan A. Stefanovici, and Bianca Schroeder. 2012. Cosmic rays don't strike twice: understanding the nature of DRAM errors and the implications for system design. InProceedings of the seventeenth international conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '12). ACM, New York, NY, USA, 111-122.**
DOI=10.1145/2150976.2150989 http://doi.acm.org/10.1145/2150976.2150989
Main memory is one of the leading hardware causes for machine crashes in today's datacenters. Designing, evaluating and modeling systems that are resilient against memory errors requires a good understanding of the underlying characteristics of errors in DRAM in the field. While there have recently been a few first studies on DRAM errors in production systems, these have been too limited in either the size of the data set or the granularity of the data to conclusively answer many of the open questions on DRAM errors. Such questions include, for example, the prevalence of soft errors compared to hard errors, or the analysis of typical patterns of hard errors. In this paper, we study data on DRAM errors collected on a diverse range of production systems in total covering nearly 300 terabyte-years of main memory. As a first contribution, we provide a detailed analytical study of DRAM error characteristics, including both hard and soft errors. We find that a large fraction of DRAM errors in the field can be attributed to hard errors and we provide a detailed analytical study of their characteristics. As a second contribution, the paper uses the results from the measurement study to identify a number of promising directions for designing more resilient systems and evaluates the potential of different protection mechanisms in the light of realistic error patterns. One of our findings is that simple page retirement policies might be able to mask a large number of DRAM errors in production systems, while sacrificing only a negligible fraction of the total DRAM in the system.

21. **Michael B. Taylor. 2012. Is dark silicon useful?: harnessing the four horsemen of the coming dark silicon apocalypse. In Proceedings of the 49th Annual Design Automation Conference(DAC '12). ACM, New York, NY, USA, 1131-1136. DOI=10.1145/2228360.2228567 http://doi.acm.org/10.1145/2228360.2228567**
Due to the breakdown of Dennardian scaling, the percentage of a silicon chip that can switch at full frequency is dropping exponentially with each process generation. This utilization wall forces designers to ensure that, at any point in time, large fractions of their chips are effectively dark or dim silicon, i.e., either idle or significantly underclocked. As exponentially larger fractions of a chip's transistors become dark, silicon area becomes an exponentially cheaper resource relative to power and energy consumption. This shift is driving a new class of architectural techniques that "spend" area to "buy" energy efficiency. All of these techniques seek to introduce new forms of heterogeneity into the computational stack. We envision that ultimately we will see widespread use of specialized architectures that leverage these techniques in order to attain orders-of-magnitude improvements in energy efficiency. However, many of these approaches also suffer from massive increases in complexity. As a result, we will need to look towards developing pervasively specialized architectures that insulate the hardware designer and the programmer from the underlying complexity of such systems. In this paper, I discuss four key approaches--the four horsemen--that have emerged as top contenders for thriving in the dark silicon age. Each class carries with its virtues deep-seated restrictions that requires a careful understanding of the underlying tradeoffs and benefits.

Zhenman Fang, Qinghao Min, Keyong Zhou, Yi Lu, Yibin Hu, Weihua Zhang, Haibo Chen, Jian Li, and Binyu Zang. 2012. Transformer: a functional-driven cycle-accurate multicore simulator. InProceedings of the 49th Annual Design Automation Conference (DAC '12). ACM, New York, NY, USA, 106-114.
DOI=10.1145/2228360.2228381 http://doi.acm.org/10.1145/2228360.2228381
Full-system simulators are extremely useful in evaluating design alternatives for multicore. However, state-of-the-art multicore simulators either lack good extensibility due to their tightly-coupled design between functional model (FM) and timing model (TM), or cannot guarantee cycle-accuracy. This paper conducts a comprehensive study on factors affecting cycle-accuracy and uncovers several contributing factors ignored before. Based on the study, we propose a loosely-coupled functional-driven full-

system simulator for multicore, namely Transformer. To ensure extensibility and cycle-accuracy, Transformer leverages an architecture-independent interface between FM and TM and uses a lightweight scheme to detect and recover from execution divergence between FM and TM. Based on Transformer, a graduate student only needs to write about 180 lines of code and takes about two months to extend an X86 functional model (QEMU) in Transformer. Moreover, the loosely-coupled design also removes the complex interaction between FM and TM and opens the opportunity to parallelize FM and TM to improve performance. Experimental results show that Transformer achieves an average of 8.4% speedup over GEMS while guaranteeing the cycle-accuracy. A further parallelization between FM and TM leads to 35.3% speedup.

22. **Sara Vinco, Debapriya Chatterjee, Valeria Bertacco, and Franco Fummi. 2012. SAGA: SystemC acceleration on GPU architectures. In Proceedings of the 49th Annual Design Automation Conference (DAC '12). ACM, New York, NY, USA, 115-120.**
    DOI=10.1145/2228360.2228382 http://doi.acm.org/10.1145/2228360.2228382
    SystemC is a widespread language for HW/SW system simulation and design exploration, and thus a key development platform in embedded system design. However, the growing complexity of SoC designs is having an impact on simulation performance, leading to limited SoC exploration potential, which in turns affects development and verification schedules and time-to-market for new designs. Previous efforts have attempted to parallelize SystemC simulation, targeting both multiprocessors and GPUs. However, for practical designs, those approaches fall far short of satisfactory performance. This paper proposes SAGA, a novel simulation approach that fully exploits the intrinsic parallelism of RTL SystemC descriptions, targeting GPU platforms. By limiting synchronization events with ad-hoc static scheduling and separate independent dataflows, we shows that we can simulate complex SystemC descriptions up to 16 times faster than traditional simulators.
23. **Azalia Mirhoseini, Miodrag Potkonjak, and Farinaz Koushanfar. 2012. Coding-based energy minimization for phase change memory. In Proceedings of the 49th Annual Design Automation Conference (DAC '12). ACM, New York, NY, USA, 68-76.**
    DOI=10.1145/2228360.2228374 http://doi.acm.org/10.1145/2228360.2228374
    We devise new coding methods to minimize Phase Change Memory write energy. Our method minimizes the energy required for memory rewrites by utilizing the differences between PCM read, set, and reset energies. We develop an integer linear programming method and employ dynamic programming to produce codes for uniformly distributed data. We also introduce data-aware coding schemes to efficiently address the energy minimization problem for stochastic data. Our evaluations show that the proposed methods result in up to 32% and 44% reduction in memory energy consumption for uniform and stochastic data respectively.
24. **Luis Gabriel Murillo, Juan Eusse, Jovana Jovic, Sergey Yakoushkin, Rainer Leupers, and Gerd Ascheid. 2012. Synchronization for hybrid MPSoC full-system simulation. In Proceedings of the 49th Annual Design Automation Conference (DAC '12). ACM, New York, NY, USA, 121-126.**
    DOI=10.1145/2228360.2228383 http://doi.acm.org/10.1145/2228360.2228383
    Full-system simulators are essential to enable early software development and increase the MPSoC programming productivity, however, their speed is limited by the speed of processor models. Although hybrid processor simulators provide native execution speed and target architecture visibility, their use for modern multi-core OSs and parallel software is restricted due to dynamic temporal and state decoupling side effects. This work analyzes the decoupling effects caused by hybridization and presents a novel synchronization technique which enables full-system hybrid simulation for modern MPSoC software. Experimental results show speed-ups from 2x to 45x over instruction-accurate simulation while still attaining functional correctness.
25. **Chi-Hao Chen, Pi-Cheng Hsiu, Tei-Wei Kuo, Chia-Lin Yang, and Cheng-Yuan Michael Wang. 2012. Age-based PCM wear leveling with nearly zero search cost. In Proceedings of the 49th Annual Design Automation Conference (DAC '12). ACM, New York, NY, USA, 453-458.**
    DOI=10.1145/2228360.2228439 http://doi.acm.org/10.1145/2228360.2228439
    Improving the endurance of PCM is a fundamental issue when the technology is considered as an alternative to main memory usage. In the design of memory-based wear leveling approaches, a major challenge is how to efficiently determine the appropriate memory pages for allocation or swapping. In this paper, we present an efficient wear-leveling design that is compatible with existing virtual memory management. Two implementations, namely, bucket-based and array-based wear leveling, with nearly zero search cost are proposed to tradeoff time and space complexity. The results of experiments conducted based on popular benchmarks to evaluate the efficacy of the proposed design are very encouraging.
26. **Farinaz Koushanfar, Saverio Fazzari, Carl McCants, William Bryson, Matthew Sale, Peilin Song, and Miodrag Potkonjak. 2012. Can EDA combat the rise of electronic counterfeiting?. InProceedings of the 49th Annual Design Automation Conference (DAC '12). ACM, New York, NY, USA, 133-138.**
    DOI=10.1145/2228360.2228386 http://doi.acm.org/10.1145/2228360.2228386
    The Semiconductor Industry Associates (SIA) estimates that counterfeiting costs the US semiconductor companies $7.5B in lost revenue, and this is indeed a growing global problem. Repackaging the old ICs, selling the failed test parts, as well as gray marketing, are the most dominant counterfeiting practices. Can technology do a better job than lawyers? What are the technical challenges to be addressed? What EDA technologies will work: embedding IP protection measures in the design phase, developing rapid post- silicon certification, or counterfeit detection tools and methods?