# The Rise and Fall of Dark Silicon

Nikos Hardavellas is the June and Donald Brewer Assistant Professor of Electrical Engineering and Computer Science at Northwestern University. His research interests include parallel computer architecture, dark silicon, memory systems, optical interconnects, and data-oriented software architectures. Prior to joining Northwestern University, he contributed to the design of several generations of Alpha processors and high-end multiprocessor servers at Digital Equipment Corp. (DEC), Compaq Computer Corp., and Hewlett-Packard. Nikos holds a PhD in Computer Science from Carnegie Mellon University.
nikos@northwestern.edu

Industry experts predict that transistor counts will continue to grow exponentially for at least another decade. Historically, we were able to harness all of these transistors to deliver exponential increases in computational power by capitalizing on both technological improvements and micro-architectural innovation. However, after decades of reaping Moore's bounty, processors eventually hit a power wall. Technological limitations will soon prevent us from powering all transistors simultaneously, leaving a large fraction of the chip powered off (or dark). Short of a technological miracle, we head towards an era of "dark silicon," able to build dense devices we cannot afford to power. Without the ability to use more transistors, or run them faster, performance improvements are likely to stagnate, unless we change course.

In this article, we (see Acknowledgments) discuss the technological trends that give rise to dark silicon, and outline our current efforts to curb them. One of the most important realizations is that instead of wasting the dark silicon, we can harness it to implement specialized cores, where each core is able to perform a narrow set of tasks at significantly lower energy. We envision an architecture that provides a sea of specialized cores, with the executing workload powering up only the most application-specific hardware, while the rest of the chip is switched off to conserve energy. The new architecture shows promise in solving the problem of dark silicon for the foreseeable future, but requires us to overcome several challenges across the entire computing stack to realize it.

## The Energy Cost of Computing

Computer systems have already become indispensible and ubiquitous in every aspect of our life. Our continuous reliance on computers generates data at exponential rates. For example, during the month of March 2011, over 1.6 PB of data were generated and transferred for processing among the Tier-1 sites of CERN's Large Hadron Collider computing grid [1]. Typical business data sets have grown by 29% annually over the last decade, surpassing Moore's Law [2], and personal data have been shown to grow at similar rates. Processing all these data requires unprecedented amounts of computation, with commensurate energy demands. To put the energy demands in perspective, it suffices to note that a 1,000m$^2$ datacenter is a 1.5 MW facility. Gartner estimates that personal computers and servers consumed 408 TWh of energy globally in 2010 [3]. Computer energy consumption in the US alone is estimated at 150 TWh annually, accounting for 3.8% of domestic energy generation for a total of $15B. This appetite for energy has created an IT

industry with approximately the same carbon footprint as the airline industry, accounting for 2% of the greenhouse gas emissions [3]. With a forecasted 10% annual growth on installed computing systems [3], these figures rise rapidly, negatively impacting both the environment and the economics of computing.

It is not surprising, then, that energy consumption is quickly becoming a limiter for big science. Building an exascale machine with today's technology is impractical due to the inordinate power draw it would require, hampering large-scale scientific efforts. The average energy-per-instruction needs to decrease 200-fold (from 2 nJ to 10 pJ) for exascale machines to be within a reasonable (20 MW) power target [4]. Even this target would require 3% of the output of an average nuclear plant to feed a single machine.

Unfortunately, a large fraction of this energy is wasted. Processor chips account for 38% of the power consumption of a typical computer system [5], but the general-purpose computing substrate they provide is highly power inefficient. For example, conventional multicore processors consume 157–707 times more energy [6] than customized hardware designs (ASICs—application-specific integrated circuits). To enter an environmentally sustainable path and lower the operational costs of large computing installations, we must find ways to eliminate these energy overheads.

While the energy demands of computing at the macro-scale have received widespread attention, there is another quiet revolution taking place at the chip level, which threatens to topple today's hardware landscape.

## The Rise of Dark Silicon

In the past several decades, technological progress and micro-architectural innovation allowed processor performance to ride Moore's Law. However, a few years ago the semiconductor industry hit a power wall. When processors approached the limits of air-cooling technologies, the on-chip frequency growth halted, and micro-architectural techniques alone proved inadequate to continue the upward performance trend. With the traditional performance driver gone, processor manufacturers turned to multicores to satisfy the users' need for performance.

Since the number of transistors on chip is growing exponentially fueling the multicore revolution, operating all transistors simultaneously requires exponentially more power per chip. However, whereas the power requirements grow, chip power delivery and cooling limitations remain largely unchanged across technologies [7]. We are already at a point where we cannot deliver and cool efficiently more than 130 W per chip [7], and as a result we will soon be incapable of operating all transistors simultaneously, pushing multicore scaling to an end [8, 9].

The reason behind this profound change is that while transistor counts grow exponentially, the voltage required to power them does not decrease fast enough: a 10-fold increase in transistor counts over the last decade was followed by only a 30% drop in supply voltage [7]. To the first order, power ($P$) is related to supply voltage ($V_{dd}$) by the equation

$$P = P_{static} + P_{dynamic} = \alpha \times N \times I_{leak} \times V_{dd} \times K + (1-\alpha) \times N \times C \times f \times V_{dd}^2$$

where $N$ is the transistor count, $\alpha$ is the average fraction of transistors in the off state (not switching), $I_{leak}$ is the leakage current, $K$ is a circuit-design-style constant, $C$ is the transistor capacitance, and $f$ is the operational frequency. The

supply voltage $V_{dd}$ cannot be reduced much, as it is limited by the threshold voltage ($V_{th}$) required to switch transistors reliably. To reduce $V_{dd}$ one needs to reduce $V_{th}$, which itself cannot be reduced much as it will increase $I_{leak}$ exponentially, raising the power consumption. Thus, while $V_{dd}$ is stuck at virtually the same voltage level, the transistor count $N$ grows exponentially, raising the power consumption of the chip.

Unfortunately, the power consumption of the additional transistors can no longer be mitigated through circuit-level techniques. Voltage-frequency scaling may decrease power consumption by lowering the operating frequency ($f$) and voltage, but its operational voltage range has shrunk by 70% over the last decade, rendering it incapable of solving computing's energy woes. At the same time, the frequency $f$ cannot be reduced sufficiently to keep the power consumption at bay, and simultaneously deliver reasonable performance [9]. With the power envelope remaining constant, and voltage scaling much slower than the exponential growth in transistor density, we'll soon be unable to power all transistors simultaneously. Short of a technological miracle, we head towards an era of "dark silicon," able to build dense devices that we cannot afford to power [2, 8, 9].

## Modeling Methodology

To assess the extent of dark silicon, we developed first-order analytical models of the dominant components of a processor's power consumption, bandwidth utilization, area occupancy, and performance, relating the effects of technology-driven physical constraints to the performance of workloads running on future multicores. We construct detailed parameterized models that conform to the projections of the International Technology Roadmap for Semiconductors (ITRS) [7] for future manufacturing technologies. Detailed descriptions of the models appear elsewhere [2, 9]. Our models have been independently vetted and used in a recent study of heterogeneous computing [10]. Similar models were independently developed and validated against PARSEC benchmarks, demonstrating that multicore performance will not scale with technology [8].

The goal of the analytical models is to describe the processor's physical constraints, and derive peak-performance designs by jointly optimizing the models' parameters (including core type and performance, core count, silicon area, memory hierarchy characteristics, manufacturing process, transistor type, cache miss models, application data-set growth, application parallelism, 3D-stacking, supply and threshold voltage, and clock frequency). The models facilitate the design space exploration of multicores by relating a design's performance to its power consumption, bandwidth requirements, and area occupancy. The modeling methodology allows us to select the design that attains peak performance, out of all the potential designs that conform to technology projections and physical constraints. It is important to note that the goal of this modeling effort is not to offer absolute numbers of the cache size or number of cores that produce the best design. Rather, the purpose of the models is to capture the first-order effects of technology scaling and unveil the trends that ultimately lead to dark silicon.

### *Example*

To illustrate the use of the models, we explain the progression of the design-space exploration algorithm based on the results plotted in Figure 1. To develop Figure 1, we first constrain the models to a single technology node (20 nm), transistor

technology (high-performance double-gate FinFETs), core design (conventional general-purpose cores modeled after Sun UltraSPARC), memory technology (off-chip DRAM), die area (310 mm2), and application characteristics (cache miss rates modeled after TPC-H on DB2, with 99% parallel code). The plot shows the aggregate chip performance as a function of the L2 cache size on the chip. Thus, for each point in the figure, a fraction of the die area is dedicated to an L2 cache of size denoted in the X-axis, 25% of the die area is used for supporting structures (e.g., memory controllers, interconnect, component spacing), and the remaining area can be populated with cores.

The *Area* curve shows the performance of designs that are constrained only in the on-chip die area (they have unlimited power and off-chip bandwidth). These designs can achieve high performance by populating the entire die with cores. As the L2 cache size grows to the right, even though fewer cores fit in the remaining area, each core's performance is higher due to the higher hit rate of the bigger cache, leading to an increase in aggregate chip performance. Eventually, the performance benefit of a large cache is outweighed by the cost of reducing the core count, leading to an aggregate performance drop at larger cache sizes.

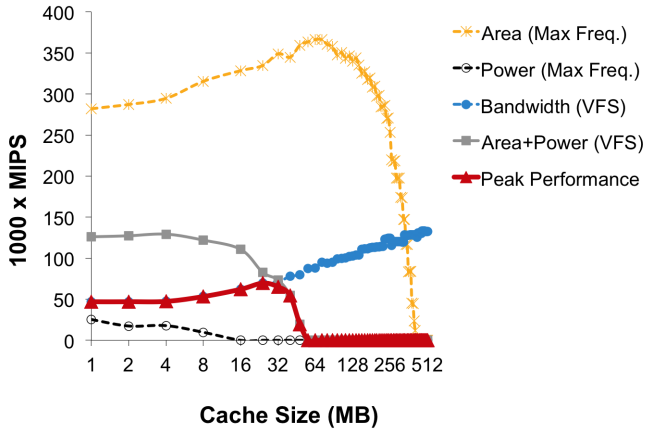


**Figure 1:** Performance of a multicore with general-purpose cores running TPC-H queries against a DB2 database at 20 nm

The *Power* curve shows designs populated with cores running at the maximum frequency allowed for the corresponding technology node, with power constrained by conventional forced-air cooling, but having unlimited area and off-chip bandwidth. Running the cores at maximum frequency requires so much power that it restricts these designs to a handful of cores, severely limiting the aggregate chip performance.

The *Bandwidth* curve shows designs that are limited only in off-chip bandwidth, permitting unlimited area and power use. The core count and core frequency are jointly optimized to find the peak-performing configuration, subject to bandwidth constraints. Larger caches reduce the off-chip bandwidth pressure, leading to improved performance. Conversely, the *Area+Power* curve shows designs limited in area and power but permitted to consume unlimited off-chip bandwidth. The Area+Power curve jointly optimizes the core voltage and frequency, selecting the peak-performing design for each L2 cache size.

Finally, the *Peak Performance* curve shows only the feasible multicore designs that conform to all physical constraints. At small cache sizes, the off-chip bandwidth is the performance-limiting factor. Beyond 24 MB, however, power becomes the main limiter, and the peak-performance design lies at the intersection of the power and bandwidth constraints. The gap between the Peak Performance and Area curves at 24 MB cache indicates that the majority of the silicon area of the best possible design for this technology node cannot be used for more cores because they cannot be powered up.

## Forecasting Dark Silicon

Using an identical analysis, but varying all the other parameters as well (e.g., core type, transistor technology, memory technology, workload, etc.), we find the multicore design that attains the highest performance on average across all workloads for a given process technology. We determine the technology trends by repeating this process for each technology node. Our workload suite includes online transactional processing (TPC-C on Oracle and DB2), Web servers (SPECweb on Apache), and decision support systems (TPC-H on DB2).

Figure 2 presents the results of this analysis, where the X-axis plots the year that each corresponding technology becomes mainstream according to ITRS, and the Y-axis plots the number of cores. The dashed lines indicate the maximum number of cores that fit on chip, and the solid lines indicate the number of cores in the peak-performance design estimated by our models. There are two pairs of lines, one for multicore processors with conventional general-purpose (GPP) cores, and one for embedded (EMB) cores modeled after ARM11.

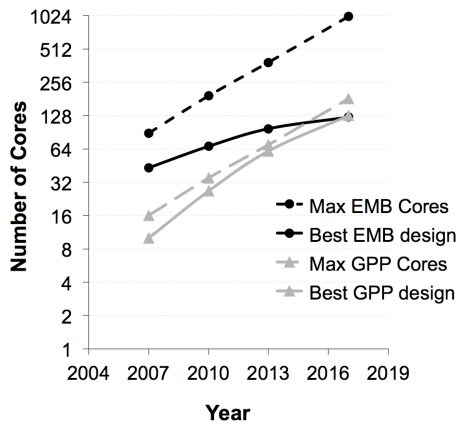


**Figure 2:** Number of cores for peak-performance designs across technologies

The gap between each pair of lines is an indicator of the impeding dark silicon. Even though up to 1000 cores can fit in a single die at 20 nm, populating the chip with an order of magnitude fewer cores is close to ideal. Placing more cores would require more power and bandwidth, which would force the supply voltage down and the entire system to run slower and lose performance.

The advent of dark silicon is even more pronounced once the bandwidth wall is alleviated: for example, through the use of 3D-stacked memory. In this technology, DRAM chips are stacked within the same package on top of the logic chip

that implements the processing cores. Communication between the stacked dies is very fast; the signals propagate vertically for only a few microns, as the dies are exceedingly thin. Retrieving data from a stacked DRAM is 60x more energy-efficient than using the conventional pin interface to off-chip DRAM. Also, the tight integration and small area footprint allows vertical buses to deliver three orders of magnitude higher bandwidth.

As a result, processor packages with 3D-stacked memory realize superior performance without the need for a large L2 on-chip cache, freeing both area and power to be used by the cores. Even these systems, however, can support only a fraction of the cores than can fit in a die because of power limitations. As a result, a significant portion of the die real estate remains unutilized, or dark. Our models predict that, under 3D-die stacking, as much as 54% of the die area may be left unutilized at the 20 nm technology node (Figure 3).

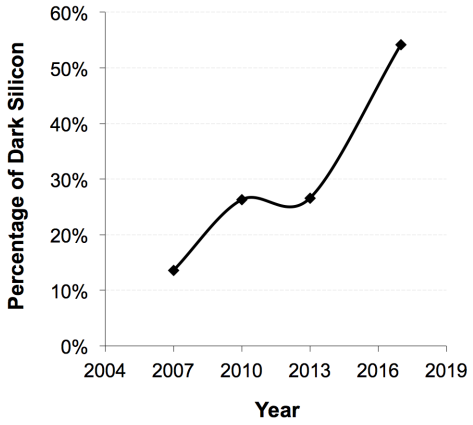


**Figure 3:** Dark silicon for peak-performance multicores with 3D-stacked DRAM

It becomes apparent that the power constraint is the main cause of dark silicon. While techniques like 3D-die stacking and photonic interconnects push the bandwidth wall far enough to practically free processor chips from the bandwidth limitation, there is little we can do to free future chips from the perils of the power wall. Even if exotic cooling technologies were employed, such as liquid cooling coupled with microfluidics, power delivery to the chip would likely impose a new constraint. Based on ITRS, power delivery poses significant challenges due to poor signal integrity when large currents are delivered at low voltages, for which no mainstream solutions have been proposed to date.

Without much hope of fixing the problem by raising the power budget, the only solution to dark silicon seems to be frugality: we need to identify the sources of energy overheads and remove them to increase the energy efficiency of computing. Today's processors are wasting energy at the circuit layer to provide reliable execution, and at the architectural layer to provide general-purpose computation and transfer data to the cores. An ideal solution would attack all these overheads.

## The Energy Overhead of the Circuit Layer

The power consumption of conventional processors is high in part because the supply voltage is determined by conservative guardbands based on worst-case scenarios. To ensure the correct operation of circuits, designers supply the transistors

with a voltage high enough to guarantee that signals will be produced and communicated successfully through the chip under all operating conditions, no matter how rare worst-case conditions may be. However, this design approach results in significant power and area overheads. Even a 40% reduction in guardbands results in 13–19% reduction in power consumption, and 13% reduction in area occupancy and wire lengths [11].

To mitigate the overheads of wide guardbands, recent research advocates the near-threshold-voltage operation of circuits. Keeping all else constant, decreasing the operating voltage ($V_{dd}$) reduces power consumption at a quadratic rate. However, as $V_{dd}$ approaches $V_{th}$, some transistors become slower than others, and some signals are not propagated correctly through the circuit and violate timing constraints, causing errors. The surge of errors at the circuit layer is for the most part the result of process variations: material impurities and the uneven distribution of dopants result in a variation in the electrical properties and switching speed of transistors, making them unreliable. The smaller the transistors become, the more susceptible they are to variations.

Building reliable circuits from unreliable components requires a host of techniques to mitigate the influx of errors, including wide guardbands and specialized error detection and recovery circuits. Unfortunately, these techniques induce significant power overheads [11]. But, what if we let go of these guardbands and allow the components of the processor to fail sometimes?

### *Elastic Fidelity*

While traditional designs correct all errors and provide accurate computation, not all computations and all data in a workload need to maintain 100% accuracy. Rather, different portions of the execution and data exhibit different sensitivity to errors, and perfect computation is not always required to present acceptable results. For example, computations that involve human perception (e.g., image, audio, motion) provide a lot of leeway in occasional errors as visual and auditory after-effects compensate for them. Some applications (e.g., networking) already have strong built-in error correction facilities as they assume unreliable components [12]. Computations performed on noisy data (e.g., from sensors) are typically equipped with techniques to correct inaccuracies within reasonable error bounds. Computations that iteratively converge to a certain value will continue to converge successfully after the introduction of small errors, albeit a few iterations later. However, the freedom to introduce errors allows the processor to operate components at significantly lower voltage, quadratically reducing the dynamic energy consumption.

Elastic Fidelity capitalizes on this observation by occasionally relaxing the reliability guarantees of the hardware based on the software requirements, and judiciously letting errors manifest in the error-resilient portion of an application's data set. Programming language subtyping designates error-tolerant sections of the data set, which are communicated through the compiler to the hardware subsystem. The hardware then operates components (e.g., functional units, cache banks) occasionally at low voltage to reduce the power and energy consumption. Portions of the application that are error-sensitive execute at full reliability, while the error-tolerant computations are scheduled to run on low-voltage units to produce an acceptable result. Similarly, error-tolerant sections of the data are stored in

low-power storage elements (e.g., low-voltage cache banks, low-refresh-rate DRAM banks) that allow for the occasional error.

By not treating all code and all data the same with respect to reliability, Elastic Fidelity exploits sections of the computation that are error-tolerant to lower power and energy consumption, without negatively impacting executions that require full reliability. Overall, 13–50% energy savings can be obtained without noticeably degrading the output [13, 14], while reaching the reliability targets required by each computational segment [14].

## The Energy Overhead of Data Transfers

While relaxing the conservative guardbands offers some respite from dark silicon, there remain significant overheads in conventional multicore computing. One of the most important ones is the high cost of data movement: fetching operands requires orders of magnitude more energy than computing on them. While a typical integer arithmetic operation consumes 0.5 pJ at 28 nm, and a 64-bit double-precision floating-point operation consumes about 20 pJ, reading two operands from a nearby cache memory requires around 100 pJ, which is 5x–200x more than the floating-point and integer operations, respectively. Retrieving the data from a cache 10 mm away requires 356 pJ, a 712-fold energy increase over an integer add, while retrieving them from across a 400 $mm^2$ chip requires 1100 pJ, a 55x–2200x increase over the floating-point and integer operations, respectively [21]. Even worse, retrieving the data from off-chip DRAM requires 16,000 pJ, three to five orders of magnitude more energy than the energy required to compute on the data!

### *Elastic Caches*

As a result of the high cost of data movement, there has been a significant research effort to minimize data transfers. Aggressive scheduling techniques aim to schedule data-sharing threads together [14], advanced caching aims to minimize trips to main memory [18], and elastic data placement schemes [16] and memory hierarchies [17] aim to bring data and computation within physical proximity. Similarly, recent efforts have shown the advantages of moving computation (i.e., code, which is typically small) close to the data (which are typically large), rather than the reverse [19].

Even simple optimizations to minimize data transfers with negligible implementation cost have been shown to reduce the energy demands by a large margin. Elastic Caches, for example, adaptively co-locate data with computation [16] and can reduce the energy demands of a processor by 17%; by placing the on-chip directory entries close to the computation, they provide an additional 13% energy savings [15].

## The Energy Overhead of General-Purpose Computing

While a simple arithmetic operation requires around 0.5–20 pJ, modern cores spend about 2000 pJ to schedule it. This tremendous source of overhead is the price we pay for general-purpose computing. Fetching instructions described in a generic ISA, decoding, tracking instructions in flight, discovering dependencies and forwarding the right values, renaming registers, reordering instructions, and predicting branch targets and target addresses all contribute to this overhead. As a result, compared to ASICs, conventional multicore processors consume two to three orders of magnitude more energy [6].

This profound energy waste contributes not only to the excessive energy consumption of modern computing, but also to the onset of dark silicon. We propose to harness the second problem to fix the first: instead of wasting dark silicon, we should embrace it and use it to realize energy-efficient computation.

### *Repurposing Dark Silicon for Specialized Computing*

Our models indicate that chips will not scale efficiently beyond a few tens to low hundreds of cores, while upwards of 1000 cores can fit in a single chip [2, 9]. Thus, an increasing fraction of the chip in future technologies will be dark silicon. We envision SeaFire, an architecture that implements a sea of specialized cores on dark silicon, where each core is able to perform a narrow set of tasks at significantly lower energy. The executing workload would fire up only the cores most useful to the computation at hand, while the rest of the chip remains switched off to conserve energy. Examples of specialized cores include cores to execute Java bytecode and JVM natively (like Azul's Vega and aJile's JEMcore), or cores that implement server functionality common in the target workloads (e.g., compression, encryption, video decoding). Similarly, some cores could be ideally suited to data-parallel computation (e.g., SIMD), while others could be optimized for long-latency operations (e.g., memory accesses) or ILP (e.g., out-of-order cores). Finally, some cores could simply provide commonly used principles (e.g., string manipulation, convolution, file scanning and filtering).

To estimate the potential of SeaFire, we evaluate an extreme application of this approach. Rather than representing a specific core design, we consider a heterogeneous multicore populated with specialized cores that exhibit ASIC-like properties on the code they execute. We model such cores after ASICs running a CABAC (Context-Adaptive Binary Arithmetic Coding) segment of the H.264 video encoder. CABAC is a form of entropy encoding used in H.264/MPEG-4 AVC video encoding. While it provides higher compression than most alternative algorithms, it requires a large amount of processing, it is difficult to parallelize and vectorize, and its highly control-intensive nature does not lend to an efficient hardware implementation. Thus, we choose to model the specialized cores after CABAC in order to obtain conservative bounds, as opposed to modeling them after data-parallel computations that may provide significantly lower energy in a specialized design.

While conventional multicores require extreme parallelism by the software to be fully utilized, a processor with specialized cores on dark silicon attains peak performance with only a handful of cores powered up at a time [2, 9]. This observation holds across technologies, and for applications with up to 99.9% parallelism. The low core count across technologies hints that peak-performance designs with specialized cores can be realized in an increasingly smaller silicon area, leaving an exponentially larger portion of the chip powered-off (dark). SeaFire repurposes the otherwise-dark silicon to implement a large collection of specialized cores to increase the likelihood of finding a core suitable for the computation. Overall, we estimate SeaFire reduces the energy consumption 12-fold over homogeneous architectures implemented within the same physical constraints [2, 9]. Thus, SeaFire promises to not only reduce the energy consumption by a large margin, but in doing so, utilize the otherwise wasted dark silicon.

## Concluding Remarks

It is apparent that, unless we change course, we'll soon find ourselves unable to utilize fully the chips we build, and the inordinate energy consumption of computing will make its expansion prohibitive. The culprits seem to be the overheads associated with traditional implementation choices: we waste energy to guarantee correct circuit execution under all circumstances, no matter how rare they may be or how much correctness really matters; we waste energy to move data from far away locations because until now maybe we didn't have a strong enough incentive to solve the problem; we waste energy in computations, a price we have been willing to pay until recently to gain generality in computing. However, our current path is unsustainable and needs to change.

But change doesn't come easy. Significant challenges need to be addressed to realize the solutions identified above. Every aspect of computing will need to be revisited to make this vision a reality. How to modify programming languages and applications to identify and specify error tolerance? Which computations are ideal candidates for off-loading to specialized hardware, and common enough to be utilized by several workloads? What are the appropriate language and runtime techniques to drive the execution migration across specialized cores? How to restructure software and algorithms for heterogeneity? The list quickly grows long, and the fight seems tough. However, the stakes are high enough to make it a fight worth fighting.

### *Acknowledgments*

The author acknowledges the contributions of his collaborators in various aspects of this research: M. Ferdman, S. Roy, A. Das, K. Liu, T. Clemons, S.M. Faisal, B. Falsafi, A. Ailamaki, S. Parthasarathy, S. Memik, M. Schuchhardt, G. Tziantzioulis, G. Memik, A. Choudhary, I. Pandis, R. Johnson, and the members of the PARAG@N Lab.

#### References

[1] CERN, Worldwide LHC computing grid, realtime VO-wise data transfer: http://lcg.Web.cern.ch/lcg/, 2011.

[2] N. Hardavellas, "Chip Multiprocessors for Server Workloads," PhD thesis, Carnegie Mellon University, 2009.

[3] *Financial Times*, "Computing: Powerful Argument for Cutting IT Energy Consumption": http://www.ft.com/cms/s/0/4e926678-bf90-11df-b9de-00144feab49a.html#axzz18EH%YGujk, September 2010.

[4] B. Dally, "Power and Programmability: The Challenges of Exascale Computing," DoE ASCR Exascale Research PI Meeting, 2011.

[5] X. Fan, W-D. Weber, and L.A. Barroso, "Power Provisioning for a Warehouse-Sized Computer," Proceedings of the 34th Annual International Symposium on Computer Architecture, 2007.

[6] R. Hameed, W. Qadeer, M. Wachs, O. Azizi, A. Solomatnikov, B.C. Lee, S. Richardson, C. Kozyrakis, and M. Horowitz, "Understanding Sources of Inefficiency in General-Purpose Chips," Proceedings of the 37th Annual International Symposium on Computer Architecture, 2010.

[7] Semiconductor Industry Association, The International Technology Roadmap for Semiconductors (ITRS): http://www.itrs.net/, 2008.

[8] H. Esmaeilzadeh, E. Blem, R. St Amant, K. Sankaralingam, and D. Burger, "Dark Silicon and the End of Multicore Scaling," Proceedings of the 38th Annual International Symposium on Computer Architecture, 2011.

[9] N. Hardavellas, M. Ferdman, B. Falsafi, and A. Ailamaki, "Toward Dark Silicon in Servers," *IEEE Micro*, vol. 31, no. 4, 2011.

[10] E.S. Chung, P.A. Milder, J.C. Hoe, and K. Mai, "Single-Chip Heterogeneous Computing: Does the Future Include Custom Logic, FPGAs, and GPUs?" Proceedings of the 43rd International Symposium on Microarchitecture, 2010.

[11] K. Jeong, A.B. Kahng, and K. Samadi, "Impact of Guardband Reduction on Design Outcomes: A Quantitative Approach," *IEEE Transactions on Semiconductor Manufacturing*, vol. 22, no. 4, 2009.

[12] A. Mallik and G. Memik, "A Case for Clumsy Packet Processors," Proceedings of the 37th International Symposium on Microarchitecture, 2004.

[13] A. Sampson, W. Dietl, E. Fortuna, D. Gnanapragasam, L. Ceze, and D. Grossman, "EnerJ: Approximate Data Types for Safe and General Low-Power Computation," Proceedings of the 32nd ACM SIGPLAN Conference on Programming Language Design And Implementation, 2011.

[14] S. Roy, T. Clemons, S.M. Faisal, K. Liu, N. Hardavellas, and S. Parthasarathy, "Elastic Fidelity: Trading-Off Computational Accuracy for Energy Reduction, CoRR, abs/1111.4279, 2011.

[15] S. Chen, P.B. Gibbons, M. Kozuch, V. Liaskovitis, A. Ailamaki, G.E. Blelloch, B. Falsafi, L. Fix, N. Hardavellas, T.C. Mowry, and C. Wilkerson, "Scheduling Threads for Constructive Cache Sharing on CMPs," Proceedings of the 19th Annual ACM Symposium on Parallelism in Algorithms and Architectures, 2007.

[16] N. Hardavellas, M. Ferdman, B. Falsafi, and A. Ailamaki, "Near-Optimal Cache Block Placement with Reactive Nonuniform Cache Architectures," *IEEE Micro*, vol. 30, no. 1, 2010.

[17] A. Das, M. Schuchhardt, N. Hardavellas, G. Memik, and A. Choudhary, "Dynamic Directories: Reducing On-Chip Interconnect Power in Multicores," Proceedings of the Design, Automation, and Test in Europe, 2012.

[18] A. Jaleel, K. Theobald, S.C. Steely Jr., and J. Emer, "High Performance Cache Replacement Using Re-Reference Interval Prediction," Proceedings of the 37th Annual International Symposium on Computer Architecture, 2010.

[19] I. Pandis, R. Johnson, N. Hardavellas, and A. Ailamaki, "Data-Oriented Transaction Execution," Proceedings of the VLDB Endowment, vol. 3, no. 1, 2010.

[20] J.H. Patel, "CMOS Process Variations: A Critical Operation Point Hypothesis," Computer Systems Colloquium, Stanford University, 2008.

[21] Stephen W. Keckler, "Echelon: NVIDIA & Team's UHPC Project," DARPA Ubiquitous High Performance Computing Program Meeting, Houston, TX, May 2011.