

# Jack Sampson

## Research Statement

### Research Overview

For several decades, improvements to CMOS process technologies fueled rapid growth in processor performance and throughput. Recently, however, power limitations have radically altered performance scaling for single-threaded applications, and the number of cores per processor has lagged behind the historical rate of transistor increase. This transition leaves architects with far more transistors than power constraints will allow them to simultaneously utilize. This in turn has led to portions of chip area that must lay fallow at any given time in order for the design to stay within the chip's power budget, commonly referred to as *dark silicon*. Further process shrinks and increasing 3D integration only serve to increase the ratio of dark silicon to active area. Thus, managing and exploiting the growth of dark silicon now looms among the largest challenges facing processor designers.

My research was among the first efforts to succinctly describe how we now contend with a **Utilization Wall**: *With each generation of Moore's Law, the percentage of a chip that can actively switch drops exponentially due to power constraints.* [8, 3] So long as transistor density continues to increase exponentially, the Utilization Wall will be an exponentially-growing problem. Moreover, the effects of the Utilization Wall are already apparent in modern process nodes: Processor frequency growth has slowed and new chips offer features such as "Turbo boost" that accelerate one core only when the others are inactive. The Utilization Wall demands urgent and aggressive solutions because it presents a substantial challenge to the long-term trend of increasing on-chip integration: For traditional chip-multiprocessors (CMPs), being able to pack more cores into a given area provides little benefit if the power budget does not support using them.

My current research focuses on surmounting the utilization wall and finding ways to manage and exploit dark silicon. Whereas dark silicon limits the potential for homogeneous chip-multiprocessors (the currently dominant processor paradigm), the relative shift in the cost of power and area will make new design paradigms profitable as area becomes cheap and power becomes expensive. In the dark silicon regime, the opportunity cost in area for adding specialized hardware is small: Even if the specialized hardware is used infrequently, the transistors it requires could not have been otherwise fully utilized. Domains that have long been power- and energy-constrained, such as mobile computing, already rely heavily on energy-efficient coprocessors. My work aims to harness the potential of customized and specialized architectures to continue historical scaling trends by converting increasing areas of dark silicon into increasing degrees of customization or increasing breadth and depth of specialization.

To provide an architectural means of trading area for energy-efficiency, I designed a class of automatically-generated energy-efficient coprocessors called *Conservation Cores* [8, 7, 1, 5, 9] and led my colleagues in developing the infrastructure necessary to generate many such coprocessors and integrate them into a larger CMP. Each Conservation Core, or *C-Core*, is a drop-in replacement for a region of code in the source application. C-Cores, unlike many other approaches, are parallelism-agnostic and can target nearly arbitrary code regions. C-Cores improve the energy efficiency of the code they target by converting dark silicon into a collection of energy-saving, application-specialized cores. These cores are produced by an automated toolchain which transforms regions of C source into C-Cores and transparently modifies the original applications to use the C-Cores.

C-Cores have a different goal than conventional accelerators that focus on improving performance without regard to energy efficiency. Designers rarely deploy accelerators for code that lacks abundant, highly structured parallelism they can exploit for performance gains. In contrast, C-Cores, focus on energy reduction and serial or irregular codes are valid targets for energy reduction via C-Cores, even when performance benefits are limited. C-Cores aim to always reduce energy consumption and to accelerate where practical. C-Cores are thus parallelism-agnostic: Even a code region with a tight critical path, little parallelism, and/or very poor memory behavior remains an excellent candidate for a C-Core if it executes frequently because C-Cores can greatly reduce the number of transistor toggles required to execute that code, saving energy. Even if, for a given C-Core, the hardware specialization that provides energy efficiency does not translate into better performance, energy-efficiency improvements can translate directly to better throughput at the chip level by freeing up more of the power budget for other computation. C-Cores that are also accelerators are possible. In [7], I introduced two new techniques for improving the performance of C-Cores and other energy-saving specialized circuits without reducing energy-efficiency.

Shifting the focus from performance-at-all-costs to efficiency allows C-Cores to target a fundamentally broader range of applications than traditional accelerators. The hardware generation toolchain my colleagues and I created generates coprocessors from unmodified program source and produces circuits whose structure corresponds very closely to the original program. This close correspondence eases automation, increases the breadth of code segments to which the approach applies, and makes it possible to reason about re-use of existing coprocessors by future software. This applicability to arbitrary code makes C-Cores an effective response to dark silicon: As the quantity of dark silicon continues to grow exponentially, architectures can incorporate increasing numbers of C-Cores, thus reducing energy per operation across ever broader portions of the target workload.

To better characterize and evaluate the potential of these coprocessors and the systems that contain them, our toolchain produced fully-synthesizable designs that we ran through industry-standard CAD flows to generate fully placed and routed designs targeting modern process nodes. We have emulated our entire system in FPGA prototyping platforms, and will tape-out a test chip for our GreenDroid [3, 4] mobile applications processor within the year.

While simulation and modeling remain invaluable research and prototyping tools, I believe that evaluating realizable hardware is important for advancing the field of computer architecture. The dark silicon era fundamentally changes traditional trade-offs among area, power, and performance. We cannot afford to be complacent in our assumptions and rules-of-thumb. Instead, we must embrace investigations of radically new designs. This will often require developing these designs to the point where we can adequately determine not only the benefits they could provide, but the unexpected challenges they may pose.

Prior to working on the challenges of dark silicon, my research efforts focused on enhancements for CMP memory systems. This is an area of inquiry in which I remain interested. To enable software to profit from increasing on-chip resources, hardware should provide richer interfaces to memory that let applications exploit the advantages of an on-chip communication environment. To that end, I have explored architectural enhancements to multi-core memory systems to accelerate and extend support for parallel software constructs [6] and richer memory semantics [2]. The barrier filters I developed in [6] enable vector computations to be efficiently distributed across the cores of a CMP by ensuring that all threads arriving at a barrier require an unavailable cache line to proceed. Additional hardware in the shared portions of the memory subsystem starves the pending requests until all participating threads have arrived. In [2], we introduced mechanisms to efficiently support transactional memory with unbounded transactions. Our approach integrates transaction metadata into the virtual memory paging system to support fast transactional operations and to manage speculative data.

### **Future Directions**

The coming years will continue to be a time of increasingly plentiful transistors and increasingly constrained power budgets. From battery-limited smart phones and sensor motes to the warehouse-scale installations of data centers and supercomputers, continued scaling in the performance of processors will rely on techniques that improve the energy-efficiency of computation. At the same time, abundant transistors and impediments to the growth of single-threaded performance will make it practical to spend area on hardware to implement new features and capabilities, such as security guarantees or support for common language-level constructs.

Hardware specialization will continue to be a valuable tool for improving energy efficiency. However, there are many research questions left to answer before we can readily build large-scale, concurrent, general-purpose systems composed primarily of specialized circuits. To scale the breadth and depth of specialization in pace with the growth of dark silicon, architects will need to design and verify specialized processors in increasing numbers. Area budgets will soon provide sufficient transistor resources to build systems that contain hundreds or even thousands of coprocessors on a single die, but assembling so many coprocessors into a single architecture can erode the performance and/or efficiency savings that each coprocessor provided in isolation. Integrating all of these processors into a single design will also yield a massively heterogeneous platform; Heterogeneous platforms are widely considered to be more difficult to program, and there is little benefit to energy-efficient hardware that software leaves unused. Traditionally, many of the performance gains from specialized hardware stem from customized memory designs, and it is not yet clear how best to integrate multiple such memory designs together into a single architecture. These and other challenges provide fertile ground for further research into managing dark silicon through hardware specialization.

Changing our design and usage of on-chip memory systems will be critical for building energy-efficient systems. The data movement required to fetch operands from distant caches, let alone off-chip memory, already requires far more energy than the computation itself. Increasing levels of datapath specialization due to the utilization wall will only exacerbate this divide. As the number of processing elements connected to memory will grow faster than the number of simultaneously active processing elements, there are opportunities for reconsidering the appropriate topologies and protocols for systems where, for reasons of data locality as well as hardware specialization, execution may frequently migrate. Likewise, computational specialization will make the memory access patterns of each each processing element more distinctive, enabling improvements in both performance and power through non-uniform interfaces to memory.

Finally, to make significant strides in performance and efficiency, we may need to revise some of our fundamental abstractions about memory. A single, flat, shared, typeless memory has been a historically convenient abstraction and programmers use it for purposes ranging from synchronization to storage, but it hides or discards information that is increasingly critical to efficiency. The growing non-uniformity in accessing different on-chip memories and the imminent integration of disruptive new technologies, such as non-volatile memories, leave a simple shared memory abstraction lacking. Preserving and exposing language-level information will allow us to better manage locality, improve the capacity of hardware to reason about the validity of accesses for correctness and security, and provide resources that are appropriate for the type of access, such as narrow, low-latency channels for synchronization and throughput-oriented block transfers for moving objects in and out of memory.

The phenomenon of dark silicon has greatly changed the computing landscape, and has fundamentally altered the balance among power, area, and performance concerns. Dealing with the rising tide of dark silicon will require significant and diverse research investments at many levels of the hardware/software stack ranging from energy-efficient circuit design to coprocessor-aware compilers and operating systems designed for massively heterogeneous platforms. I look forward to building my own research group to address these issues, and to collaborating with architects and other systems researchers working on customizable processors, hardware specialization, aggressive power management, energy-efficient circuit design, and other means of controlling and exploiting the growth of dark silicon.

## References

- [1] M. Arora, J. Sampson, N. Goulding-Hotta, J. Babb, G. Venkatesh, M. B. Taylor, and S. Swanson. Reducing the energy cost of irregular code bases in soft processor systems. In *Field-Programmable Custom Computing Machines (FCCM), 2011 IEEE 19th Annual International Symposium on*, pages 210–213. IEEE, 2011.
- [2] W. Chuang, S. Narayanasamy, G. Venkatesh, J. Sampson, M. Van Biesbrouck, G. Pokam, B. Calder, and O. Colavin. Unbounded page-based transactional memory. In *ASPLOS-XII: Proceedings of the 12th international conference on Architectural support for programming languages and operating systems*, pages 347–358, New York, NY, USA, 2006. ACM.
- [3] N. Goulding, J. Sampson, G. Venkatesh, S. Garcia, J. Auricchio, J. Babb, M. B. Taylor, and S. Swanson. GreenDroid: A mobile application processor for a future of dark silicon. *Hot Chips*, 2010.
- [4] N. Goulding-Hotta, J. Sampson, G. Venkatesh, S. Garcia, J. Auricchio, P. Huang, M. Arora, S. Nath, V. Bhatt, J. Babb, M. B. Taylor, and S. Swanson. The GreenDroid mobile application processor: An architecture for silicon’s dark future. *Micro, IEEE*, 31(2):86–95, 2011.
- [5] J. Sampson, M. Arora, N. Goulding-Hotta, G. Venkatesh, J. Babb, V. Bhatt, S. Swanson, and M. B. Taylor. An evaluation of selective depipelining for FPGA-based energy-reducing irregular code coprocessors. In *Field Programmable Logic and Applications (FPL), 2011 International Conference on*, pages 24–29. IEEE, 2011.
- [6] J. Sampson, R. Gonzalez, J.-F. Collard, N. P. Jouppi, M. Schlansker, and B. Calder. Exploiting fine-grained data parallelism with chip multiprocessors and fast barriers. In *MICRO 39: Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 235–246, Washington, DC, USA, 2006. IEEE Computer Society.
- [7] J. Sampson, G. Venkatesh, N. Goulding, S. Garcia, S. Swanson, and M. B. Taylor. Efficient complex operators for irregular codes. In *International Symposium on High-Performance Computer Architecture*, Feb. 2011.
- [8] G. Venkatesh, J. Sampson, N. Goulding, S. Garcia, V. Bryksin, J. Lugo-Martinez, S. Swanson, and M. B. Taylor. Conservation cores: Reducing the energy of mature computations. In *International Conference on Architectural Support for Programming Languages and Operating Systems*, Mar. 2010.
- [9] G. Venkatesh, J. Sampson, N. Goulding-Hotta, S. Venkata, M. B. Taylor, and S. Swanson. Qscores: Trading dark silicon for scalable energy efficiency with quasi-specific cores. In *Proceedings of the 44th Annual IEEE/ACM International Symposium on Microarchitecture*. IEEE Computer Society, 2011.