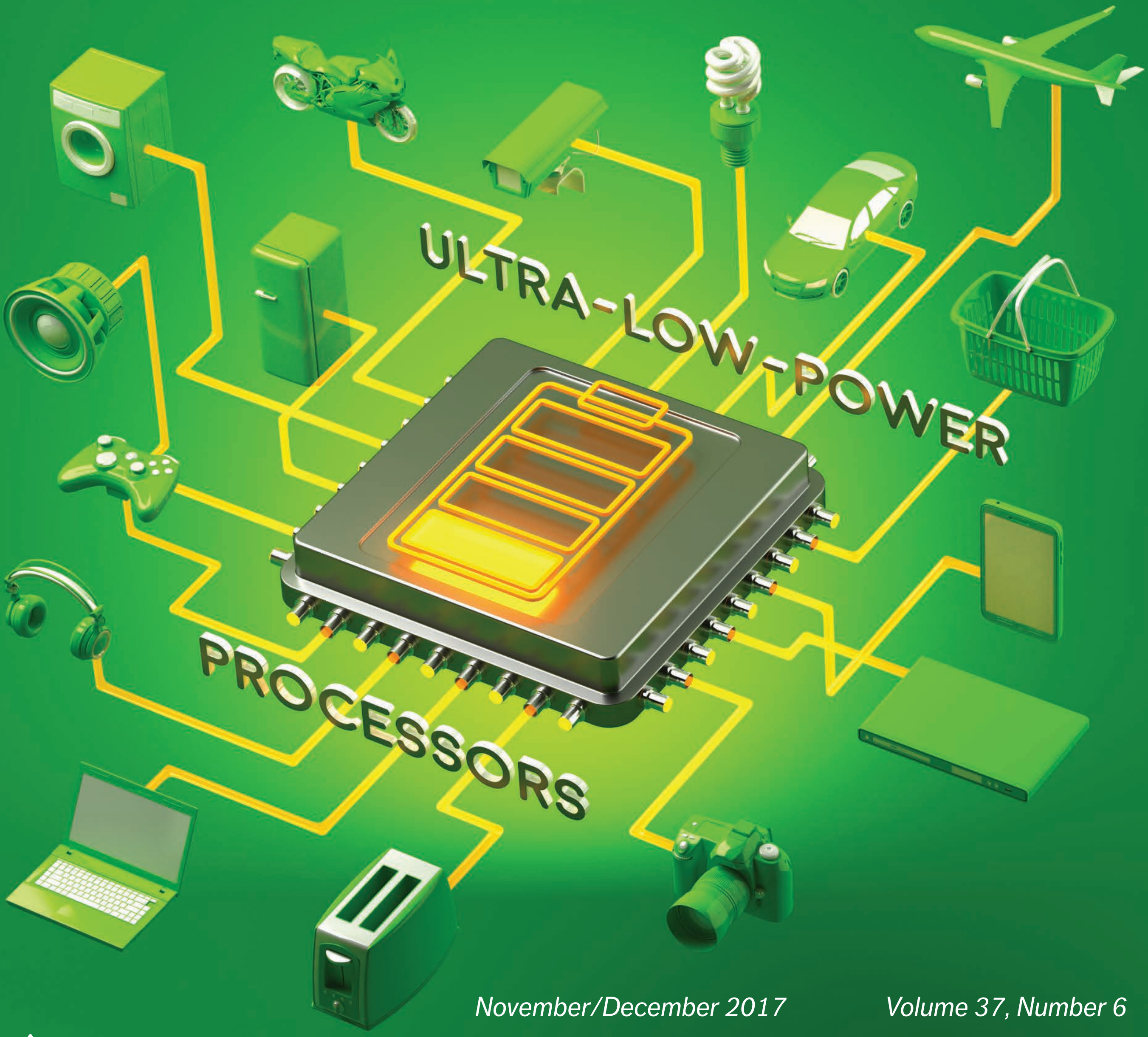
Architectural Agility
p. 6

# IEEE micro

*The magazine for chip and silicon systems designers*



*November/December 2017* *Volume 37, Number 6*

IEEE

IEEE computer society

*www.computer.org/micro*

# Achieve your career goals with the fit that's right for you.

## Explore your options below.

| Select your membership | Preferred Plus | | Training & Development | | Research | | Basic | | Student |
|---|---|---|---|---|---|---|---|---|---|
| | **$60**<br>*IEEE Member* | **$126**<br>*Affiliate Member* | **$55**<br>*IEEE Member* | **$115**<br>*Affiliate Member* | **$55**<br>*IEEE Member* | **$115**<br>*Affiliate Member* | **$40**<br>*IEEE Member* | **$99**<br>*Affiliate Member* | **$8**<br>*Does not include IEEE membership* |
| *Computer* magazine (12 digital issues)* | ✓ | | ✓ | | ✓ | | ✓ | | ✓ |
| *ComputingEdge* magazine (12 issues) | ✓ | | ✓ | | ✓ | | ✓ | | ✓ |
| Members-only discounts on conferences and events | ✓ | | ✓ | | ✓ | | ✓ | | ✓ |
| Members-only webinars | ✓ | | ✓ | | ✓ | | ✓ | | ✓ |
| Unlimited access to *Computing Now*, computer.org, and the new mobile-ready myCS | ✓ | | ✓ | | ✓ | | ✓ | | ✓ |
| Local chapter membership | ✓ | | ✓ | | ✓ | | ✓ | | ✓ |
| Skillsoft's Skillchoice™ Complete with 67,000+ books, videos, courses, practice exams and mentorship resources | ✓ | | ✓ | | | | | | ✓ |
| Books24x7 on-demand access to 15,000 technical and business resources | ✓ | | ✓ | | | | | | ✓ |
| Two complimentary Computer Society magazine subscriptions | ✓ | | | | ✓ | | | | |
| myComputer mobile app | *30 tokens* | | | | *30 tokens* | | | | *30 tokens* |
| Computer Society Digital Library | *12 FREE downloads* | | *Member pricing* | | *12 FREE downloads* | | *Member pricing* | | *Included* |
| Training webinars | *3 FREE webinars* | | *3 FREE webinars* | | *Member pricing* | | *Member pricing* | | *Member pricing* |
| Priority registration to Computer Society events | ✓ | | | | | | | | |
| Right to vote and hold office | ✓ | | ✓ | | ✓ | | ✓ | | |
| One-time 20% Computer Society online store discount | ✓ | | | | | | | | |

* *Print publications are available for an additional fee. See catalog for details.*

www.computer.org/membership

IEEE computer society

IEEE micro

Published by the IEEE Computer Society
www.computer.org/micro

November/December 2017 • Volume 37 Number 6

## Editorial Office

10662 Los Vaqueros Circle
Los Alamitos, CA 90720
+1 714 821-8380

## Submissions

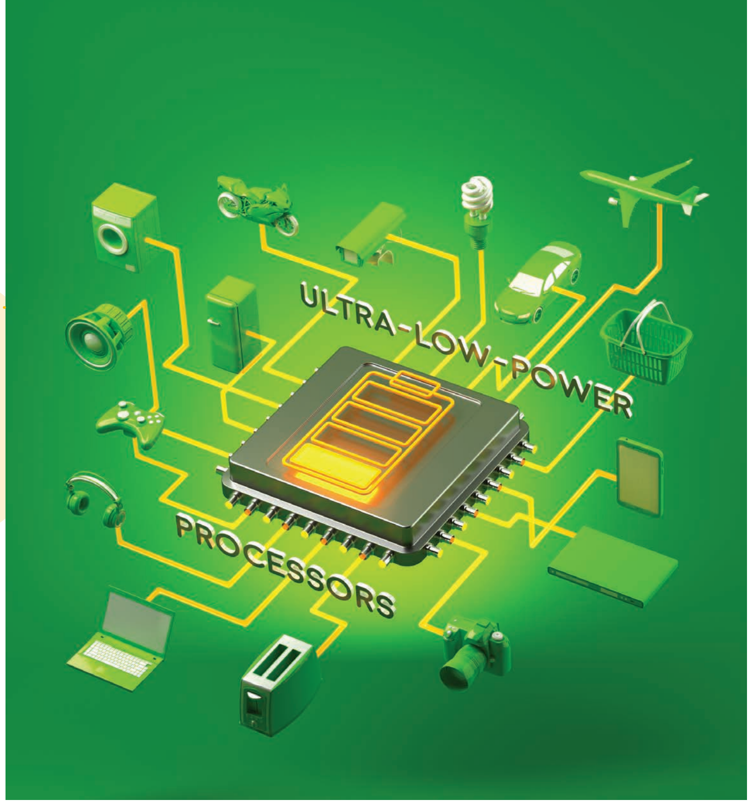
https://mc.manuscriptcentral.com
/micro-cs

***Cover art by Oliver Burston, Début Art***
*spookypooka@virginmedia.com*

For more information on computing topics, visit the Computer Society Digital Library at www.computer.org/csdl.

***IEEE Micro*** (ISSN 0272-1732) is published bimonthly by the IEEE Computer Society. IEEE Headquarters, Three Park Ave., 17th Floor, New York, NY 10016-5997; IEEE Computer Society Headquarters, 2001 L St., Ste. 700, Washington, DC 20036; IEEE Computer Society Publications Office, 10662 Los Vaqueros Circle, PO Box 3014, Los Alamitos, CA 90720.

**Postmaster:** Send address changes and undelivered copies to IEEE, Membership Processing Dept., 445 Hoes Ln., Piscataway, NJ 08855. Periodicals postage is paid at New York, NY, and at additional mailing offices. Canadian GST #125634188. Canada Post Corp. (Canadian distribution) Publications Mail Agreement #40013885. Return undeliverable Canadian addresses to 4960-2 Walker Road; Windsor, ON N9A 6J3. Printed in USA.



## Features

### Ultra-Low-Power Processors

## Departments

## Also in This Issue

# Moore's Law and Ultra-Low-Power Processors

**Lieven Eeckhout**
*Ghent University*

**Welcome to the November/December** 2017 special issue on ultra-low-power processors.

The Internet of Things (IoT) revolution is happening at a rapid pace. Most projections forecast that the number of connected IoT devices will grow exponentially, easily reaching over 100 billion within the next decade. This revolution leads to a vigorous demand for ultra-low-power (ULP) edge computing devices and associated system-on-chip (SoC) architectures. This special issue includes six articles that highlight some of the state-of-the-art research and potentially viable solutions in ULP processors. I suggest you read the guest editors' column for more details about these articles. I want to wholeheartedly thank David Brooks and John Sartori for their excellent work as guest editors of this special issue. I hope you will enjoy reading these articles as much as I did.

There is one article that I would like to highlight here. Mark Bohr and Ian Young talk about CMOS scaling trends and how Intel has been able to keep Moore's law alive over the past decades. This required continuous innovation in materials and device structures to deliver the performance, power, and cost improvements as expected with each technology generation. The article also highlights new device options and technology directions to continue scaling in the near future. The excellent keynote Mark delivered at the ISCA 2017 conference instigated this article, and I'm grateful to Mark and Ian for having taken the time to write up this excellent contribution.

This issue also includes two thought-provoking Expert Opinion articles about challenging topics in our field. In the first column, Bobbie Manne, Bryan Chin, and Steve Reinhardt posit that architects should pursue architectural agility to lower the barriers to developing innovative and disruptive solutions in an unpredictable and rapidly evolving technology landscape as we face new technology limitations. They present several ideas for engineers to integrate agility into both processor and datacenter design.

Reetu Das describes two historical waves in processing in memory (PIM) to combat the memory wall, and then argues for moving computation closer to memory—thereby transforming memory into powerful accelerators—which seems like an appealing and promising vision in an era increasingly dominated by data-intensive workloads.

This issue also includes an award testimonial. David Brooks reports on the 2017 ISCA Influential Paper Award, which was given to "Drowsy Caches: Simple Techniques for Reducing Leakage Power" by Krisztián Flautner, Nam Sung Kim, Steven M. Martin, David Blaauw, and Trevor N. Mudge. This award recognizes the paper published 15 years ago—in this case, 2002—at the ISCA conference that has had the most impact on the field (in terms of research, development, products, or ideas) during the intervening years. The drowsy cache paper made a seminal contribution to power-efficient computing. The paper was written at a time when leakage current was becoming a major concern, especially in large on-chip caches in high-end processors. The paper's key idea was to put parts of the cache into a low-power "drowsy" mode to save energy while retaining the data. Congratulations to the award winners on their groundbreaking research!

Finally, in this issue, Shane Greenstein talks about the "hush-hush norm," which I will let you discover for yourself in the Micro Economics column.

The IEEE Computer Society will be making some changes to how magazine articles are edited. Unfortunately, this means that our current copy editor, Molly Gamborg, will no longer

0272-1732/17/$33.00 © 2017 IEEE

be working with *IEEE Micro*. Molly has been *IEEE Micro*'s copy editor for seven years. I have interacted with her for many years, first as an author and then as the editor in chief for the past three years. I'm sure that many of you have interacted with Molly as well over these many years, as an author or otherwise. Simply said, Molly did an outstanding job, both in terms of copy editing magazine articles and in terms of managing deadlines and schedules for the magazine. I've always been amazed by her performance and professionalism. She really made my job as the editor in chief easy and enjoyable. I feel very fortunate to have worked with Molly. At the same time, I feel saddened that she will no longer be part of the team. Thanks a lot, Molly, for your great service—we will miss you!

With that, I'd like to conclude and wish you a happy reading.

**Lieven Eeckhout** is a professor in the Department of Electronics and Information Systems at Ghent University. Contact him at lieven.eeckhout@ugent.be.

# If You Build It, Will They Come?

**Srilatha Manne**
*Cavium*

**Bryan Chin**
*University of California, San Diego*

**Steven K. Reinhardt**
*Microsoft*

**All hardware companies face** a conundrum. Should they continue the evolutionary trend of their current products, or build riskier products that have the potential for greater reward but carry a higher probability of failure? The safe course, and one that many customers ask for, is the former. However, as Clayton Christensen points out in *The Innovator's Dilemma*, "most companies with a practiced discipline of listening to their best customers and identifying new products that promise greater profitability and growth are rarely able to build a case for investing in disruptive technologies until it is too late."[1]

Computer hardware companies expend enormous resources to successfully improve their products in an evolutionary fashion. Single-threaded processor performance has been improving at a rate of 15 to 20 percent per year by utilizing both process technology and architectural improvements.[2] These improvements, however, are increasingly difficult to achieve. Using data from Moein Khazraee and colleagues,[3] Figure 1 shows that a processor's cost per operation, as defined by a combination of fabrication, nonrecurring engineering (NRE), and packaging costs, has not significantly improved in the past decade. However, performance improvements are flattening out due to power restrictions and the breakdown of Dennard scaling. For instance, Intel is no longer relying on the tick-tock model, which it rode to market dominance for the past decade, due to the declining benefits of process technology scaling.[4]

## Sustaining versus Disruptive Technology

Christensen describes the evolutionary process of improvements using the *sustaining technology S-curve* (see Figure 2). For every successful technology, the performance metric is initially flat during development, rapidly improves for a period of time, and flattens out again when the product and/or technology reaches maturity. Sustaining technologies are dominating the processor industry, and these technologies are reaching a plateau.

Sometimes a disruptive technology with a new S-curve will enter the landscape, as shown in Figure 2. Disruptive technologies do not go head to head with mainstream technologies, but they do have features that a few fringe markets value. Typically, disruptive technologies initially underperform, but then rapidly match and exceed the previous technology. Successful companies not only ride their sustaining S-curves but generate new, disruptive curves to improve performance as the current technology curve flattens out. Microprocessors were once a disruptive technology,[1] and the computing landscape over the past few decades is littered with disruptive technologies, from minicomputers to PCs to smartphones to cloud computing. In all these cases, the disruptive technology yielded worse performance in the near-term when using the same cost function as mainstream technology. However, as Christensen maintains, disruptive technologies eventually redefine how performance is measured.

Recent examples of disruptive technologies in processor architecture include GPUs and Arm servers. GPUs were originally designed for 3D graphics processing, but have made significant inroads first in high-performance computing (HPC) and more recently in machine learning. For applications that are similar to those found in SPECint, GPUs underperform general-purpose processors. However, for targeted HPC applications and machine learning, GPUs are overwhelmingly superior.

Arm processors originally targeted power-constrained embedded domains, but have more recently entered the server market with product offerings from companies such as

0272-1732/17/$33.00 © 2017 IEEE

Cavium and Qualcomm that address multicore throughput computing.[5,6] A new S-curve could develop for these specialized throughput-based server products—enabled by highly parallelizable shared-memory applications—just as it did with HPC and machine learning in the GPU market.

It took the GPU market nearly two decades to make headway outside of graphics applications, and the Arm server market has resulted in several failures. Christensen notes that this commonplace in disruptive markets is where "[it] is simply impossible to predict with any useful degree of precision how disruptive products will be used or how large their markets will be." So, how does one innovate in a rapidly changing technology landscape where the underlying cost function is in flux? How does a company keep up with the necessary and expensive evolutionary changes, yet also prepare for and justify expending valuable resources investigating disruptive technologies that are inevitable?

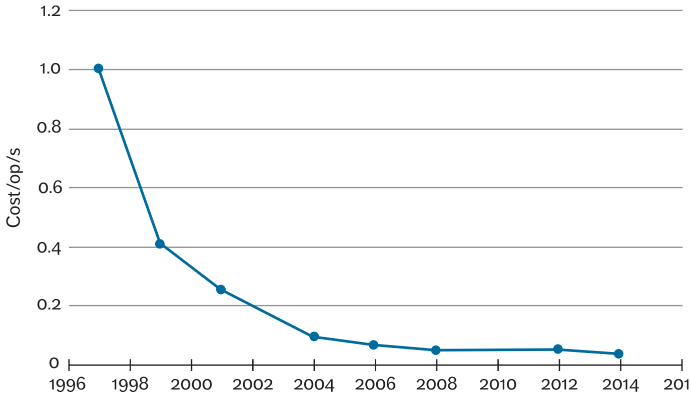


**Figure 1.** Processor computation cost as a function of time. Cost is defined as a combination of fabrication, nonrecurring engineering (NRE), and packaging costs.[3]

## The Case for Agility

Companies and their mainstream customers alike are notoriously bad at predicting what disruptive products will take root in the marketplace. There are many instances of high-profile developments that flopped. For example, it is unlikely you are reading this article on your Apple Newton while listening to music on your Microsoft Zune. Conversely, some disruptive technologies have found success in surprising places such as GPUs. Innovation in a rapidly changing landscape is difficult and prone to failure. Therefore, we posit that architects, rather than trying to predict the future, should pursue agility in order to accelerate innovation while minimizing costs. Hardware companies, architects, and the underlying design methodologies and infrastructure must be nimble enough to deal with disruptive technologies that come from within and outside the current technology landscape. The rest of the article presents some ideas on how this may be accomplished.

## Agile Architecture

In his book *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Business*,[7] Eric Ries writes about software companies that use agile software development strategies. The premise is to deliver prototypes as quickly as possible, even if haphazardly put together, to get early customer feedback. The goal is to use customer feedback to drive product features and direction through a process of continuous development. If you consider how frequently the apps on your phone are updated, or the look and feel of social networking sites evolve, you have seen agile software practices in action.

Facebook, for example, uses agile coding practices. As Kent Beck explains,[8] one of the basic practices at Facebook is reversibility. If a decision is reversible, it does not require the rigorous testing that irreversible decisions require. Code is also released incrementally to a small subset of users, which enables changes to be rolled back with minimal disruption if a problem is found.[9] The challenge for the hardware industry is how to adapt a similar agile methodology without incurring large overheads. We address this challenge in both traditional processor hardware methodologies and innovative methodologies utilized by large computing companies.

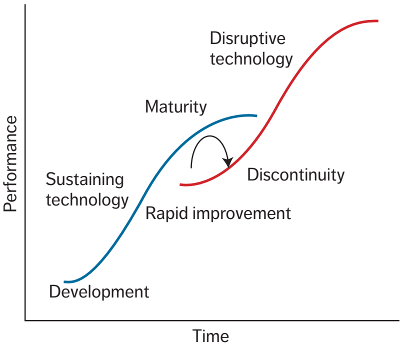


**Figure 2.** The sustaining technology and disruptive technology S-curves.

### Processor Agility

Prior to the ASIC revolution of the past few decades, hardware prototypes were a common means of achieving the rapid development and early feedback cycle. Old technologies such as wire-wrap, breadboards, programmable logic devices (PLDs), and low-cost printed circuit boards (PCBs) enabled hardware companies to quickly build and iterate on products. This methodology is no longer feasible given the complexity and cost of processor development both in terms of engineering time and fabrication costs.[3]

**Automated design methodology and reuse.** Companies today rely on improved design methodologies and reusability to reduce design time and cost. Design methodologies have made great strides in the past two decades, resulting in shorter design cycle times and an expanded product portfolio using the same fundamental components. Most processors, even those designed for high performance, are mostly or completely synthesized. The Arm roadmap has synthesized cores operating at 3 GHz, and AMD, Intel, and IBM extensively use automated tools throughout their design.[10–12] In addition, companies utilize a modular design methodology such that multiple products can be developed using the same basic components.

Both Intel and AMD use their respective base core designs and innovative packaging technologies to build products ranging from low-power mobile parts to multicore server products.[13] Similarly, silicon companies such as Cavium and Nvidia have been able to create a family of devices with varying price/performance points from the same basic design by utilizing flexible chip layouts that let designers vary the number of computational units and/or the amount of on-die memory. Intel has taken this one step further by collaborating with Facebook to develop a specialized version of Broadwell (referred to as *Broadwell-D*) to meet the specific needs of Facebook.[14]

The technologies mentioned so far reduce design cycle time, but there is still significant overhead associated with bringing a chip to production. Post-silicon functional and performance debug is a formidable challenge for modern processors that may encompass multiple sockets, heterogeneous and/or multithreaded cores, many cores combined with multiple levels of memory hierarchy, complex memory coherence and consistency protocols, and extensive power and performance management via on-chip controllers. In addition, modern processors may operate under complex software stacks containing one or more nested virtual environments. For these reasons, even with mostly synthesized methodologies and reuse of existing components, the transition from first silicon to full production part can take up to a year or more.[15]

**Functional verification and bug mitigation.** Post-production bugs are commonplace, and fixing bugs in shipped products often involves errata, metal and full-layer spins, and/or replacing existing silicon. Infamous examples of such bugs are the Pentium FDIV bug,[16] the Haswell/Broadwell transactional memory bug,[17] and the AMD TLB bug.[18] These bugs cost the respective companies millions of dollars in lost revenue, and in AMD's case, contributed to its loss of momentum in the server market. All processors have a large list of errata. The table of known errata in Haswell, for instance, covers six pages.[19]

To meet market needs and address the complexity and cost of post-silicon debug, architects must focus on hardware and software solutions for exposing, analyzing, and mitigating functional and performance bugs. Processor vendors must provide tools that rapidly expose and identify bugs and have systems in place for mitigating these bugs without the need for extensive silicon changes. Efforts such as Arm's hardware debug architecture attempt to standardize the infrastructure so that common tools can be made available to the Arm hardware development ecosystem.[20]

Both software and hardware solutions should be explored for mitigating hardware bugs in the field. On the hardware front, microcode fixes on traditional CISC processors come to mind, as does the PAL (Privileged Architecture Library) code feature of DEC's Alpha processors. A similar technology that might help processor vendors mitigate bugs is virtual machine environments. Much software these days is compiled to an abstract machine. Two examples of such abstraction layers, one current and one historical, are Oracle's Java Virtual Machine (JVM)[21] and IBM's AS/400 Series.[22] If an entire processor is designed to execute only a JVM, then the JVM itself provides the instruction set architecture (ISA) of the machine, and the underlying physical machine may have bugs or features that are invisible to the JVM. The JVM addresses ISA-related bugs. Similarly, more fully specified virtual machine environments, such as VMware's vSphere and Microsoft's Hyper-V, virtualize system aspects of the machine, such as memory management and I/O. Machines such as IBM's AS/400 managed to maintain a stable abstract architecture through multiple generations of hardware. By expecting and architecting for bug discovery, analysis, and mitigation, processor vendors can reduce the number of bugs that reach production silicon, and respond to issues in post-production parts quickly and effectively. This shortens the designer-customer feedback loop and leads to a faster development cycle and improved successor products.

**Performance verification and optimization.** Another critical facet of bringing a processor to production is performance tuning. Processors are designed with dozens of control bits (also referred to as *chicken bits*) to manage system performance. Some chicken bits are exposed to the user (for example, disabling prefetching or simultaneous multithreading mode, or restricting power management), and others are known only by the manufacturer. Regardless, how these bits are set and tuned can have a significant impact on performance. Unfortunately, there are hundreds of these interdependent knobs, and tuning them by hand is impractical. However, self-tuning systems, either integrated into the operating system or as separate tools,[23] that can dynamically adjust these bits according to application needs may be an innovative mechanism for achieving optimal performance. Best of all, these tuners can be deployed on-site, which means they do not gate product release to customers. Finally, the same techniques for fixing bugs via low-level software or implementing a virtual machine can also be used to adapt silicon to new applications. Hardware designers can enable and deploy new instructions and features through the same mechanisms used to patch around bugs. New versions of a JVM implementation, for example, may exploit optimizations that are relevant to new application areas.

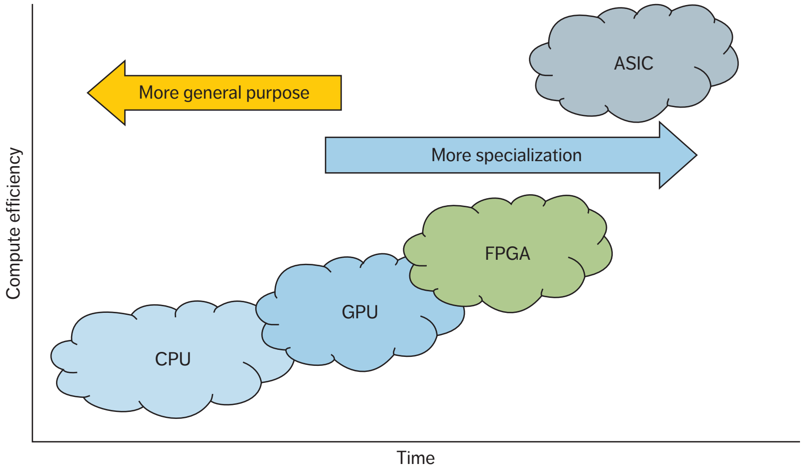


**Figure 3.** Specialization trend over time.

## Computational Agility

So far, we have addressed agility at the processor level. However, with the advent of warehouse-scale systems driven by cloud computing, the processor becomes one piece of a larger computational problem. New companies entering the computing arena include numerous startups and large, established companies from outside the traditional chip design industry, such as Google, Microsoft, and Amazon. Few if any of these companies are choosing to go head-to-head in the general-purpose processor market with traditional designs such as Intel and AMD. Rather, they are achieving agility via specialized devices targeting narrower but highly relevant domains.

**The need for specialization.** The end of Dennard scaling and the slowdown and imminent demise of Moore's law drive the need for specialization, just as they demand agility in processor design. During the steep part of the S-curve for general-purpose processors, specialized architectures were quickly outpaced by these cheaper commodity devices. The slowing rate of improvement in general-purpose designs both creates opportunity for specialized architectures and drives demand, as customers can no longer rely on the commodity market to satisfy their computing needs.

A prerequisite for specialization is identifying an application or application domain narrow enough to benefit from specialization but large enough to justify a specialized device. Focusing on smaller and smaller domains (down to specific applications) increases the amount of potential performance uplift through specialization, while decreasing the potential market. To be successful, the total value created through specialization (roughly speaking, the value per device times the number of devices) must exceed the cost of developing the specialized device. By developing agile methodologies that reduce engineering costs, we can enable specialization for smaller domains and allow specialized devices to emerge sooner in growing markets.

Figure 3 shows the specialization trend over time, starting with CPUs and ending with custom ASICs. Cryptocurrency mining followed this trend,[24] and deep learning, one of the most prominent new markets attracting specialized architectures, is following suit. GPUs offer better performance than CPUs for certain tasks, such as training for AI, whereas state-of-the art field-programmable gate arrays (FPGAs) can outperform standard GPUs for certain computations such as low-precision arithmetic.[25] Finally, custom ASIC accelerators provide the highest performance efficiency.

Multiple startups such as Graphcore, Wave Computing, Nervana (now part of Intel), and Groq are developing or have developed customized deep learning accelerators that occupy the upper right corner of Figure 3. However, one of the earliest and most publicized deep learning accelerators is not from a startup but from an established company without a history of chip design. The Google Tensor Processing Unit (TPU) was developed in a short 15 months.[26] To achieve a rapid production cycle, Google used an older and more stable process technology (28 nm) and existing communication interfaces. The first-generation TPU was for internal use and had computational and memory bandwidth limitations. However, the TPU is now on its second iteration, and it not only supports higher computational capability and memory bandwidth, but will reportedly be made accessible to third parties.[27]

Even in an agile environment, the delay from the initial ASIC concept to fully deployed device is measured in years. Once deployed, ASICs must continue to provide value for multiple additional years before replacement. Thus, an ASIC must accelerate a function that, from the point of conception, will still be valuable four to five years in the future. While some functions, such as compression and encryption algorithms, tend to be stable over these time frames, those in rapidly evolving fields such as deep learning may develop new and different requirements in the interval from design start to deployment. Stable, high-volume accelerators can easily justify an ASIC's higher nonrecurring engineering cost. Because an ASIC design needs larger markets and longer lifetimes, an ASIC accelerator typically includes as much flexibility as designers can afford in the form of configuration parameters, options, and software programmability.

To achieve a more agile acceleration framework, Microsoft took an unusual approach to specialization by focusing on FPGAs rather than ASICs for datacenter acceleration.[28] For a given accelerator design, an FPGA implementation could be several times slower and less energy efficient than an ASIC implementation. However, by using hardware devices that can be reprogrammed after deployment, Microsoft gains agility at the expense of computational efficiency. FPGA-based accelerators not only are tolerant to the changing requirements of a given application, but can be completely retargeted as new applications emerge or demand shifts. An FPGA accelerator design can afford to be less configurable and more customized to specific situations, as the design itself can be incrementally modified after initial deployment to address new circumstances. In this fashion, the FPGA's agility as a platform can be used to recover a portion of the efficiency that it sacrifices to an equivalent ASIC-based design.

FPGAs can also close the gap with ASICs by incorporating larger and more complex hard logic blocks on chip. Current FPGAs include multiply-accumulate units and even full microprocessor cores as hard logic. Researchers have also proposed devices that are mostly hard logic, but with configurable interconnect, referred to as *coarse-grained reconfigurable accelerators* (CGRAs).[29] The line between FPGAs and ASICs is further blurred by integrated multichip packages that incorporate both an FPGA and ASIC die.[30] The ability for customers to specify which ASICs are included in the package provides yet another dimension of flexibility.

**The computational marketplace.** Amazon has also developed hardware for internal consumption from custom routers to chipsets used in its servers.[31] This enables Amazon to optimize the hardware for its specific needs with full control of both the hardware and software stack. Amazon also provides hardware agility to its customers by offering platforms for custom programmable hardware as part of the AWS services plan.[32] The goal is to encourage companies to develop accelerators using Amazon's FPGA framework for internal use and/or sell the resulting computational capability to end customers on the AWS Marketplace. Amazon's EC F1 instances with FPGAs offer two significant benefits for custom solution developers. First, Amazon provides the FPGA hardware, tools, and infrastructure, significantly lowering the cost and convenience threshold for developing customized hardware. Second, Amazon provides a deployment model (via AWS) and a ready marketplace of potential customers for the final product. No longer are hardware developers restricted to products with a large Tier One customer base. They can rapidly develop and deploy niche hardware and test its viability in the AWS computational marketplace with many small customers across the country and the world. The computational marketplace scenario comes closest to achieving the rapid deployment model highlighted in *The Lean Startup*.[7] Finally, if any of these customized solutions become pervasive, they can eventually be reimplemented as an ASIC, as noted by Khazraee,[3] or integrated into a general-purpose processor architecture.

**Standardized ecosystem.** A successful computational marketplace requires standardized interfaces for interacting with accelerators. On the hardware side, current solutions from Amazon, Microsoft, Google, and others rely on PCIe for accelerator integration. PCIe has been the de facto standard for peripherals for many years, and a part of its success can be attributed to having an open standard. However, for processor designers wanting to create specialized accelerators, PCIe may not offer the tightly coupled memory system integration desired or required by the application. Proprietary coherent processor interconnects such as Intel's QPI and AMD's Infinity Fabric offer the memory system integration that a specialized accelerator might require, while Nvidia's NVLink is a proprietary interconnect for GPUs. Nonproprietary standards from different consortia such as OpenCAPI (www.opencapi.org), Gen-Z (www.genzconsortium.org), and CCIX (www.ccixconsortium.com) might also supplement PCIe as these standards evolve. What is clear, from the PCIe example, is that the new standard should be easily licensable and controlled by an open standards organization to enable a level playing field.

While we have thus far emphasized agility in hardware development and deployment, software agility is also a critical requirement. An environment in which hardware capabilities change and evolve rapidly is impossible to use unless low-level software can adapt equally rapidly, while providing stable

APIs to higher-level services so that the bulk of the code base can remain independent of the underlying implementation's details. Software stacks can provide additional agility when they help to automate the mapping of applications to accelerators, and enable hardware bug workarounds to cope with issues that may slip through an accelerated development and testing schedule.

Processor architecture has changed significantly over the past few decades with the advent of multicore designs, design for low power, heterogeneous systems, and many-core processors that can run a hundred or more threads. With cloud computing and the emerging customizable marketplace of products, we are once again witnessing a sea change in the way computing takes place.

In this article, we have made a case for agility because we cannot predict the future with any level of accuracy. We need agility not only for rapid evolution of conventional architecture, but also for lowering the barrier for specialized architectures. As Bill Gates once noted, "We always overestimate the change that will occur in the next two years and underestimate the change that will occur in the next ten. Don't let yourself be lulled into inaction."[33] As architects, we must develop the infrastructure and mindset that enable us to be agile and take risks in order to evolve with a rapidly changing environment and create the next disruptive technology.

## References

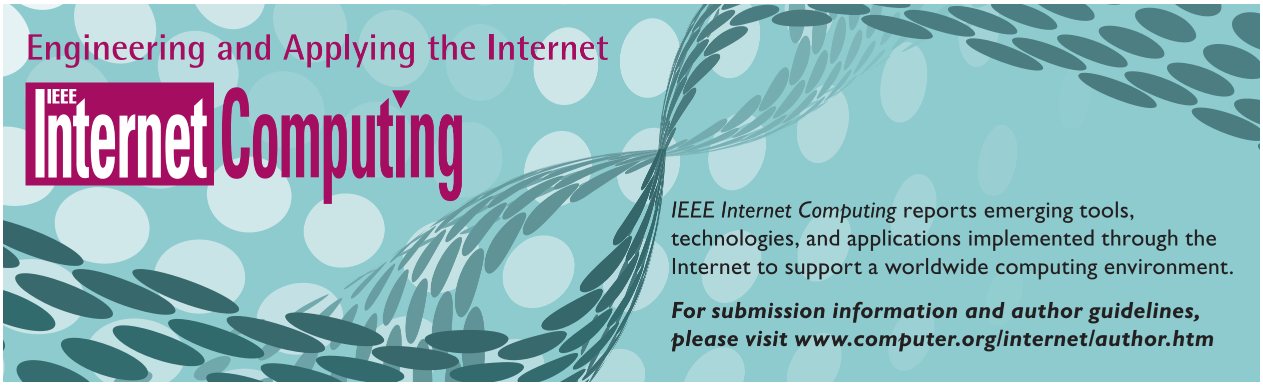
1. C.M. Christensen, *The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail*, Harvard Business School Press, 1997.
2. "A Look Back at Single-Threaded CPU Performance," blog, 8 Feb. 2012; http://preshing.com/20120208/a-look-back-at-single-threaded-cpu-performance.
3. M. Khazraee et al., "Moonwalk: NRE Optimization in ASIC Clouds," *Proc. 22nd Int'l Conf. Architectural Support for Programming Languages and Operating Systems*, 2017, pp. 511–526.
4. J. Hruska, "Intel Formally Kills its Tick-Tock Approach to Processor Development," blog, 23 Mar. 2016; www.extremetech.com/extreme/225353-intel-formally-kills-its-tick-tock-approach-to-processor-development.
5. T.P. Morgan, "Qualcomm Fires ARM Server Salvo, Broadcom Silences Guns," 7 Dec. 2016; www.nextplatform.com/2016/12/07/qualcomm-fires-arm-server-salvo-broadcom-silences-guns.
6. R. Brueckner, "Cavium ThunderX2 Processors Power New Baymax HyperScale Server Platforms," blog, 29 May 2017; http://insidehpc.com/2017/05/cavium-thunderx2-processors-power-new-baymax-hyperscale-server-platforms.
7. E. Ries, *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Business*, Crown Publishing Group, 2011.
8. C. Murphy, "Facebook Guru and Agile Pioneer Kent Beck Reveals the Mind of the Modern Programmer," *Forbes*, 9 Jan. 2017; www.forbes.com/sites/oracle/2017/01/09/facebook-guru-and-agile-pioneer-kent-beck-reveals-the-mind-of-the-modern-programmer.
9. J. Bird, "This Is How Facebook Develops and Deploys Software. Should You Care?" blog, 4 Sept. 2013; http://dzone.com/articles/how-facebook-develops-and.
10. M. Humrick, "Exploring DynamIQ and ARM's New CPUs: Cortex-A75, Cortex-A55," blog, 29 May 2017; www.anandtech.com/show/11441/dynamiq-and-arms-new-cpus-cortex-a75-a55.
11. P. Gelsinger et al., "Such a CAD! Coping with the Complexity of Microprocessor Design at Intel," *IEEE Solid-State Circuits*, vol. 2, no. 3, 2010, pp. 32–43.
12. M. Ziegler, R. Puri, and B. Philhower, "POWER8 Design Methodology Innovations for Improving Productivity and Reducing Power," *Proc. IEEE Custom Integrated Circuits Conf.*, 2014, pp. 1–9.
13. A. Patrizio, "Intel Shakes Up Its Chip Design Process," blog, 23 May 2014; www.itworld.com/article/2699164/hardware/intel-shakes-up-its-chip-design-process.html.
14. V. Rao and E. Smith, "Facebook's New Front-End Server Design Delivers on Performance without Sucking Up Power," blog, 9 Mar. 2016; http://code.facebook.com/posts/1711485769063510/facebook-s-new-front-end-server-design-delivers-on-performance-without-sucking-up-power.
15. M. Abramovici and P. Bradley, "A New Approach to In-System Silicon Validation and Debug," *EE Times*, 16 Sept. 2007; www.eetimes.com/document.asp?doc_id=1276099.
16. "Pentium FDIV Bug," blog; www.cs.earlham.edu/~dusko/cs63/fdiv.html.
17. S. Wasson, "Errata Prompts Intel to Disable TSX in Haswell, Early Broadwell CPUs," blog, 12 Aug. 2014; http://techreport.com/news/26911/errata-prompts-intel-to-disable-tsx-in-haswell-early-broadwell-cpus.
18. K. Kubicki, "Understanding AMD's 'TLB' Processor Bug," blog, 5 Dec. 2007; www.dailytech.com/Understanding++AMDs+TLB+Processor+Bug/article9915.htm.

19. *Desktop 4th Generation Intel Core Processor Family, Desktop Intel Pentium Processor Family, and Desktop Intel Celeron Processor Family*, report 328899-037US, Mar. 2017.
20. "Debug Architecture Overview," *ARM*, 2017; http://developer.arm.com/products/architecture/debug-architecture
21. T. Lindholm et al., *The Java Virtual Machine Specification: Java SE 7 Edition*, 28 Feb. 2013.
22. F.G. Soltis, *Inside the AS/400: Featuring the AS/400e Series*, 2nd ed., 29th Street Press, 1997.
23. T. Morad, *The Era of Self-Tuning Servers*, 7 Feb. 2017; www.hpcadvisorycouncil.com/events/2017/stanford-workshop/pdf/Morad_TheEraOfSelfTuningServers.pdf.
24. P. Jama, "The Future of Machine Learning Hardware," blog, 10 Sept. 2016; http://hackernoon.com/the-future-of-machine-learning-hardware-c872a0448be8.
25. L. Barney, "Can FPGAs Beat GPUs in Accelerating Next-Generation Deep Learning?" blog, 21 Mar. 2017; www.nextplatform.com/2017/03/21/can-fpgas-beat-gpus-accelerating-next-generation-deep-learning.
26. K. Sato, C. Young, and D. Patterson, "An In-Depth Look at Google's First Tensor Processing Unit (TPU)," blog, 12 May 2017; http://cloud.google.com/blog/big-data/2017/05/an-in-depth-look-at-googles-first-tensor-processing-unit-tpu.
27. P. Teich, "Under the Hood of Google's TPU2 Machine Learning Clusters," blog, 22 May 2017; www.nextplatform.com/2017/05/22/hood-googles-tpu2-machine-learning-clusters.
28. A. Putnam et al., "A Reconfigurable Fabric for Accelerating Large-Scale Datacenter Services," blog, 1 June 2014; www.microsoft.com/en-us/research/publication/a-reconfigurable-fabric-for-accelerating-large-scale-datacenter-services.
29. M. Gao and C. Kozyrakis, "HRL: Efficient and Flexible Reconfigurable Logic for Near-Data Processing," *Proc. IEEE Int'l Symp. High Performance Computer Architecture*, 2016, doi: 10.1109/HPCA.2016.7446059.
30. M. Deo, *Enabling Next-Generation Platforms Using Intel's 3D System-in-Package Technology*, white paper WP-01251-1.5, Intel, Aug. 2017.
31. D. Richman, "Amazon Web Services' Secret Weapon: Its Custom-Made Hardware and Network," blog, 19 Jan. 2017; www.geekwire.com/2017/amazon-web-services-secret-weapon-custom-made-hardware-network.
32. "Amazon EC2 F1 Instances, Customizable FPGAs for Hardware Acceleration Are Now Generally Available," blog, 19 Apr. 2017; http://aws.amazon.com/about-aws/whats-new/2017/04/amazon-ec2-f1-instances-customizable-fpgas-for-hardware-acceleration-are-now-generally-available.
33. B. Gates, *The Road Ahead*, Viking Penguin, 1996.

**Srilatha Manne** is a principal hardware architect at Cavium. Contact her at bobbiemanne12@gmail.com.

**Bryan Chin** is a lecturer in the Computer Science and Engineering Department at the University of California, San Diego. Contact him at b5chin@ucsd.edu.

**Steven K. Reinhardt** is a partner hardware engineering manager at Microsoft. Contact him at stever@microsoft.com.



# Blurring the Lines between Memory and Computation

**Reetuparna Das**
*University of Michigan*

**Computer designers have traditionally** separated the roles of storage and computation. Memories stored data. Processors computed them. Is this distinction necessary? A human brain does not separate the two so distinctly, so why should a computer? Before addressing this question, let us start with the well-known memory wall problem.[1]

What is the memory wall in today's context? The memory wall originally referred to the problem of growing disparity in speed between fast processors and slow memories. Since 2005 or so, as processor speed flatlined, memory latency has remained about the same. But as the number of processor cores per chip kept increasing, memory bandwidth and memory energy became more dominant issues. A significant fraction of energy is spent today in moving data back and forth between memory and computing units, a problem that is exacerbated in modern data-intensive systems.

How do we overcome the memory wall in today's computing world that is increasingly dominated by data-intensive applications? For well over two decades, architects have tried a variety of strategies to overcome the memory wall. Most of them have centered on exploiting locality. Here is an alternative: what if we could move computation closer to memory—so much that the line that divides computation and memory starts to blur?

## The First Wave

Researchers discussed processing in memory (PIM) in the 1990s[2–6] (initial suggestions date back to as early as the 1970s[7]) as an alternative solution to scale the memory wall. The key idea was to physically bring the computation and memory units closer together by placing computation units inside the main memory (DRAM). But this idea did not quite take off back then, due to the high cost of integrating computational units within a DRAM die. Another factor may have been the fact that cheaper optimizations were still possible, thanks to Moore's law and Dennard scaling.

The advent of commercially feasible 3D chip stacking technology, such as Micron's Hybrid Memory Cube (HMC),[8] has renewed our interest in PIM. HMC stacks layers of DRAM memory on top of a logic layer. Computational units in the logic layer can communicate with memory through high-bandwidth through-silicon vias. Thanks to 3D integration technology, we can now take computational and DRAM dies implemented in different process technologies and stack them on top of each other.

The additional dimension in 3D PIM allows an order of magnitude more physical connections between the computational and memory units, and thereby provides massive memory bandwidth to the computational units.[9–15] The available memory bandwidth is so high in these systems that a general-purpose multicore processor with tens of cores is a poor candidate to take advantage of 3D PIM. The bandwidth of cheaper conventional DRAM is mostly adequate for these general-purpose processors. Better candidates are customized computational units that can truly take advantage of the abundant memory bandwidth in 3D PIM data-parallel accelerators, such as a GPU, or even better, customized accelerators such as Google's Tensor Processing Unit.[16]

Although 3D PIM is a clear winner in terms of memory bandwidth compared to conventional DRAM, its latency and energy advantages are perhaps exaggerated in literature. 3D PIM brings computation closer to DRAM memory. It has no effect on the energy spent accessing data within DRAM layers, DRAM refresh and leakage, and on-die interconnect in the logic layer, which together happen to be the dominant cost. To be clear, there is some memory latency and energy reduction as it eliminates communication over the off-chip memory channels by integrating computation in 3D PIM's logic

0272-1732/17/$33.00 © 2017 IEEE

layer. However, this benefit is not likely to be a big win and paves a smaller step toward reducing the steep data-movement overheads.[17,18]

## The Second Wave

Although PIM brings computational and memory units closer together, the functionality and design of memory units remains unchanged. An even more exciting technology is one that dissolves the line that distinguishes memory from computational units. Nearly three-fourths of silicon in processor and main memory dies is simply to store and access data. What if we could take this memory silicon and repurpose it to do computation? Let us refer to the resulting unit as *Compute Memory*.

Compute Memory repurposes the memory structures, the ones that are traditionally used only to store data, into active computational units for near-zero area cost. Compute Memory's biggest advantage is that its memory arrays morph into massive vector computing units (potentially, one or two orders of magnitude larger than a GPU's vector units), as data stored across hundreds of memory arrays could be operated on concurrently. Because we do not have to move data in and out of memory, the architecture naturally saves the energy spent in those activities, and memory bandwidth becomes a meaningless metric.

Micron's Automata Processor (AP)[19,20] is an example for Compute Memory. It transforms DRAM structures to a Nondeterministic Finite Automata (NFA) computational unit. NFA processing occurs in two phases: state match and state transition. AP cleverly repurposes the DRAM array decode logic to enable state matches. Each of the several hundreds of memory arrays can now perform state matches in parallel. The state-match logic is coupled with a custom interconnect to enable state transition. We can process as many as 1,053 regular expressions in Snort (a classic network-intrusion detection system) in one go using little more than DRAM hardware. AP can be an order of magnitude more efficient than GPUs and nearly two orders of magnitude more efficient than general-purpose multicore CPUs! Imagine the possibilities if we can sequence a genome within minutes using cheap DRAM hardware.

AP repurposed just the decode logic in DRAMs. Could we do better? In our recent work on Compute Caches,[21,22] we showed that it is possible to repurpose SRAM array bit-lines and sense-amplifiers to perform in-place analog bit-line computation on the data stored in SRAM. A cache is typically organized as a set of sub-arrays; as many as thousands of sub-arrays, depending on the cache level.[23–25] These sub-arrays can all compute concurrently on several hundred thousands of data elements stored in them with little extensions to the existing cache structures, while incurring an overall area overhead of 4 percent. Thus, caches can effectively function as large vector computational units, whose operand sizes are orders of magnitude larger than conventional SIMD units. Of course, it also eliminates the energy spent in moving data in and out of caches. While our initial work supports few useful operations (logical, search, and copy), we believe that it is just a matter of time before we are able to support more complex operations (including comparisons, addition, multiplication, sorting).

Supporting Compute Caches' style-in-place, analog bit-line computing in DRAMs is more challenging. The problem is that DRAM reads are destructive—one reason why DRAMs need periodic refresh. Although in-place DRAM computing may not be possible, an interesting solution is to copy the data to a temporary row in the DRAM[8] and then do bit-line computing. This approach will incur extra copies, but retains the massive parallelism benefits.

Unlike DRAMs, bit-line computing may work well in a diverse set of nonvolatile memory technologies (RRAMs, STT-MRAMs, and Flash). Researchers have already found success in repurposing structures in emerging NVMs to build efficient ternary content-addressable memory (TCAM)[26] and neural networks.[27–29]

Computational memories can be massively data parallel—potentially, an order of magnitude more performance and energy efficient than modern data-parallel accelerators such as GPUs. Such dramatic improvements could have a transformative effect on applications ranging from genome sequencing to deep neural networks. However, capabilities of computational memories may not be as general purpose as GPUs are today, and may impose additional constraints in terms of where data is stored. Application developers may have to rework their algorithms to fully take advantage of Compute Memory. Modern data-parallel domain-specific language frameworks such as CUDA and Tensorflow can be adapted to help these developers. It may also require runtime and system software support to meet computational memory constraints such as data placement.

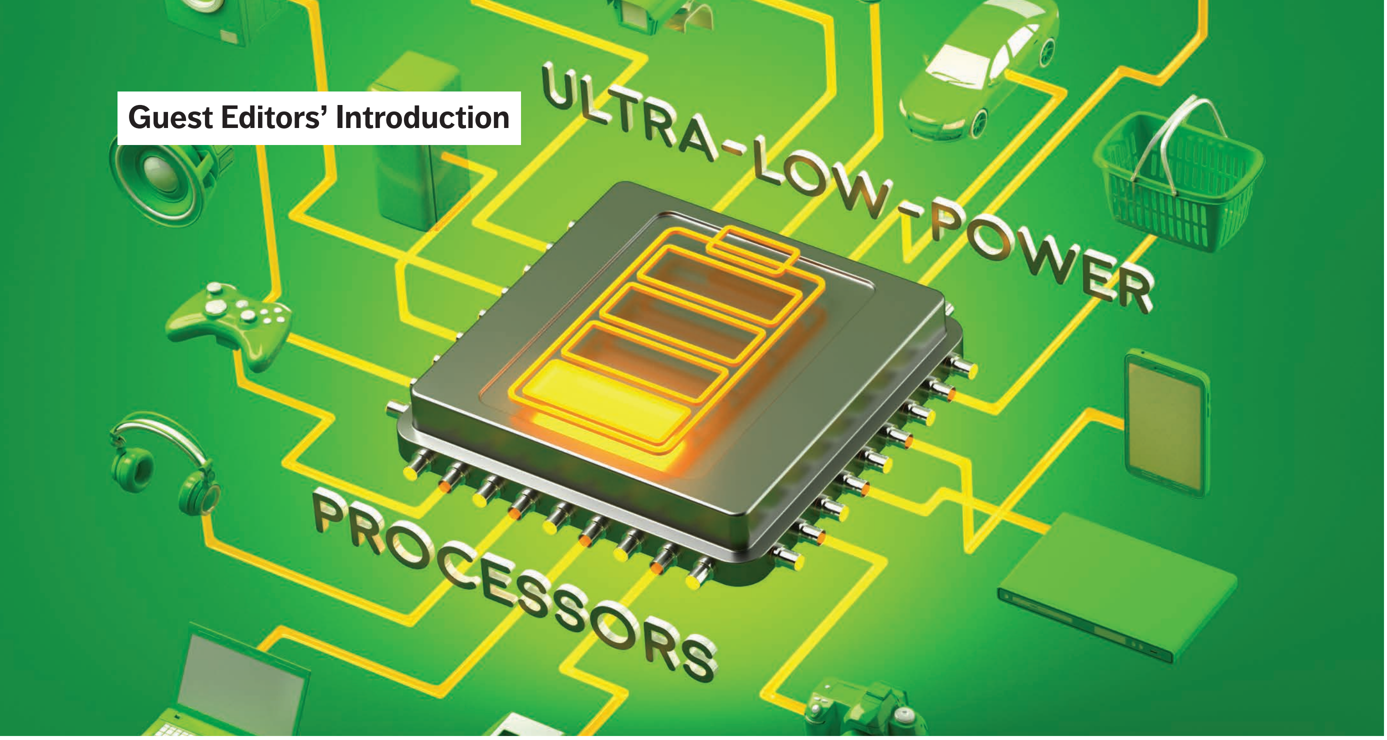
As the general-purpose core's efficiency flatlined over the past decade, both industry and academia have wholeheartedly embraced customization of computational units. It is high time for us to think about customizing memory units as well. While there are many ways that one could think of customizing memory, turning it into powerful accelerators is one of the more exciting avenues to pursue. Until recently, we have viewed computing and memory units as two separate entities. Even within a processor, caches and computational logic have operated as two separate entities that served different roles. The time has come to dissolve the line that separates them.

## References

1. W.A. Wulf and S.A. McKee, "Hitting the Memory Wall: Implications of the Obvious," *SIGARCH Computer Architecture News*, vol. 23, no. 1, 1995, pp. 20–24.
2. M. Gokhale, B. Holmes, and K. Iobst, "Processing in Memory: The Terasys Massively Parallel PIM Array," *Computer*, 1995, vol. 28, no. 4, 1995, pp. 23–31.
3. Y. Kang et al., "FlexRAM: Toward an Advanced Intelligent Memory System," *Proc. Int'l Conf. Computer Design*, 1999, pp. 192–201.
4. P. Kogge, "Execube: A New Architecture for Scaleable MPPs," *Proc. Int'l Conf. Parallel Processing*, vol. 1, 1994, pp, 77–84.
5. M. Oskin, F. Chong, and T. Sherwood, "Active Pages: A Computation Model for Intelligent Memory," *Proc. 25th Ann. Int'l Symp. Computer Architecture*, 1998, pp. 192–203.
6. D. Patterson et al., "A Case for Intelligent RAM," *IEEE Micro*, vol. 17, no. 2, 1997, pp. 34–44.
7. H.S. Stone, "A Logic-in-Memory Computer," *IEEE Trans. Computers*, vol. C-19, no. 1, 1970, pp. 73–78.
8. *Hybrid Memory Cube Specification*, 2014; http://hybridmemorycube.org.
9. J. Ahn et al., "PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture," *Proc. 42nd Ann. Int'l Symp. Computer Architecture*, 2015, pp. 336–348.
10. A. Farmahini-Farahani et al., "NDA: Near-Dram Acceleration Architecture Leveraging Commodity DRAM Devices and Standard Memory Modules," *Proc. IEEE 21st Int'l Symp. High Performance Computer Architecture*, 2015, pp. 283–295.
11. D. Kim et al., "Neurocube: A Programmable Digital Neuromorphic Architecture with High-Density 3D Memory," *Proc. 43rd Int'l Symp. Computer Architecture*, 2016, pp. 380–392.
12. S. Pugsley et al., "NDC: Analyzing the Impact of 3D-Stacked Memory+Logic Devices on MapReduce Workloads," *Proc. IEEE Int'l Symp. Performance Analysis of Systems and Software*, 2014, pp. 190–200.
13. V. Seshadri et al., "RowClone: Fast and Energy-Efficient In-DRAM Bulk Data Copy and Initialization," *Proc. 46th Ann. IEEE/ACM Int'l Symp. Microarchitecture*, 2013, pp. 185–197.
14. D. Zhang et al., "Top-PIM: Throughput-Oriented Programmable Processing in Memory," *Proc. 23rd Int'l Symp. High-Performance Parallel and Distributed Computing*, 2014, pp. 85–98.
15. Q. Zhu et al., "A 3D-Stacked Logic-in-Memory Accelerator for Application-Specific Data Intensive Computing," *Proc. IEEE Int'l 3D Systems Integration Conf.*, 2013, doi:10.1109/3DIC.2013.6702348.
16. N.P. Jouppi et al., "In-Datacenter Performance Analysis of a Tensor Processing Unit," *Proc. 44th Ann. Int'l Symp. Computer Architecture*, 2017, pp. 1–12.
17. K. Bergman et al., *ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems*, DARPA, 2008.
18. B. Dally, "Power, Programmability, and Granularity: The Challenges of ExaScale Computing," *Proc. IEEE Int'l Parallel Distributed Processing Symp.*, 2011, p. 878.
19. Micron Automata Processing; www.micronautomata.com.
20. P. Dlugosch et al., "An Efficient and Scalable Semiconductor Architecture for Parallel Automata Processing," *IEEE Trans. Parallel and Distributed Systems*, vol. 25, no. 12, 2014, pp. 3088–3098.
21. S. Aga et al., "Compute Caches," *Proc. 23rd Int'l Symp. High Performance Computer Architecture*, 2017, pp. 481–492.
22. S. Jeloka et al., "A Configurable TCAM/BCAM/SRAM using 28nm Push-Rule 6T Bit Cell," *Proc. IEEE Symp. VLSI Circuits*, 2015, pp. C272–C273.
23. W.J. Bowhill et al., "The Xeon R Processor E5-2600 v3: A 22 nm 18-Core Product Family," *J. Solid-State Circuits*, vol. 51, no. 1, 2016, pp. 92–104.
24. W. Chen et al., "A 22nm 2.5 MB Slice On-Die L3 Cache for the Next Generation Xeon R Processor," *Proc. IEEE Symp. VLSI Technology*, 2013, pp. C132–C133.
25. M. Huang et al., "An Energy Efficient 32-nm 20-MB Shared On-Die L3 Cache for Intel R Xeon R Processor E5 Family," *J. Solid-State Circuits*, vol. 48, no. 8, 2013, pp. 1954–1962.
26. Q. Guo et al., "Resistive Ternary Content Addressable Memory Systems for Data-Intensive Computing," *IEEE Micro*, vol. 35, no. 5, 2015, pp. 62–71.
27. M.N. Bojnordi and E. Ipek, "Memristive Boltzmann Machine: A Hardware Accelerator for Combinatorial Optimization and Deep Learning," *Proc. IEEE Int'l Symp. High Performance Computer Architecture*, 2016, pp. 1–13.
28. P. Chi et al., "Prime: A Novel Processing-in-Memory Architecture for Neural Network Computation in ReRAM-Based Main Memory," *Proc. 43rd Int'l Symp. Computer Architecture*, 2016, pp. 27–39.
29. A. Shafiee et al., "Isaac: A Convolutional Neural Network Accelerator with In-Situ Analog Arithmetic in Crossbars," *Proc. 43rd Int'l Symp. Computer Architecture*, 2016, pp. 14–26.

**Reetuparna Das** is an assistant professor in the Electrical Engineering and Computer Science Department at the University of Michigan. Contact her at reetudas@umich.edu.

Guest Editors' Introduction



# Ultra-Low-Power Processors

**David Brooks**
*Harvard University*

**John Sartori**
*University of Minnesota*

Society's increasing use of connected sensing and wearable computing has created robust demand for ultra-low-power (ULP) edge computing devices and associated system-on-chip (SoC) architectures. In fact, the ubiquity of ULP processing has already made such embedded devices the highest-volume processor part in production, with an even greater dominance expected in the near future. The Internet of Everything calls for an embedded processor in every object, necessitating billions or trillions of processors. At the same time, the explosion of data generated from these devices, in conjunction with the traditional model of using cloud-based services to process the data, will place tremendous demands on limited wireless spectrum and energy-hungry wireless networks. Smart, ULP edge devices are the only viable option that can meet these demands.

One big area of expansion for ULP processors is the Internet of Things (IoT). Most projections forecast the number of connected IoT devices to grow exponentially, easily reaching over 100 billion within the next decade. Even assuming a conservative ULP power budget of a few milliwatts per device, the total energy consumption of all these connected devices will be over 10 trillion kWh per year. That's more energy than over half of the countries in the world consume in a year. Given the sheer number of devices that will be connected (IoT devices will outnumber humans by more than an

Published by the IEEE Computer Society

order of magnitude), even trying to change or charge all their batteries will become an infeasible task, necessitating more research on energy harvesting to create energy-neutral devices that can fend for themselves by collecting their own energy. Likewise, more research will be needed on novel ways to reduce power dramatically—by an order of magnitude or more, enabling ULP devices to be integrated in more places and in higher quantities. In conjunction with this research on ultra-low power and energy, the fact that all these devices will be connected to the Internet demands more research on energy-efficient security measures. In a world where IoT devices have access to all of our data—personal, health-critical, financial, and all the rest—the attack surface for potential information security leaks becomes larger than ever. With all this critical information entrusted to devices that can barely scrape together enough power to boot up, much less implement a host of security protocols, ensuring information security at ultra-low power and energy levels will be critical.

The articles in this special issue highlight some of the critical research and explore some of the potentially viable solutions that will help to advance the state of the art toward a more power- and energy-efficient future for ULP processing.

## Beyond CMOS

The continuation of CMOS device scaling is of utmost importance to computer architects, and in recent years the perception has been that CMOS scaling has slowed. In "CMOS Scaling Trends and Beyond," Mark T. Bohr and Ian A. Young dispel this notion by showing that through the hard work and ingenuity of device R&D, several new transistor design innovations have been brought to bear on the problem over the past several generations of CMOS technology at Intel. The article also highlights several "beyond-CMOS" technologies that have the potential to complement CMOS by outperforming it in certain niche applications. An example highlighted in the article is Tunnel FETs that can drastically improve the energy-delay product over conventional CMOS. Such devices would be especially attractive for the ULP processors that are the focus of this special issue.

## Implementing a Low-Power Neural Network on Chip

Implementing a neural network in a ULP chip is a challenging feat. Neural networks are one of those applications that require intensive computation that is typically delegated to massively parallel GPGPUs. Nevertheless, in "Low-Power Convolutional Neural Network Processor for a Face-Recognition System," Kyeongryeol Bong and colleagues took on this challenge and fabricated a face-recognition chip based on convolutional neural networks (CNNs) that boasts power consumption of less than a milliwatt for a computation rate of 1 fps. They achieved this low power consumption by splitting the task of face recognition into two stages—face detection, which is performed by a low-power ASIC, and face verification, which is performed by a highly accurate CNN. In their chip, the ULP face-detection circuitry acts as an energy-conscientious gatekeeper for the higher-powered CNN logic, such that the CNN is called on only when needed (that is, when a face has been detected and extracted from an input image). The chip also dynamically adapts its power characteristics using dynamic voltage and frequency scaling based on the number of faces detected to keep power consumption low even under heavy load conditions. The result is a low-power chip that performs a task that's integral to many ULP applications.

## Edge–Cloud Computing

Machine learning is a key component of many IoT systems that must make decisions based on the data they gather in the wild. However, the computationally intense nature of machine learning makes it unsuitable for execution on ULP processors. Typically, massively parallel GPGPUs are used for such computations; however, powering and carrying around a GPU is out of the question for most ULP systems, which are constrained to small form factors, low cost, and ultra-low power and energy budgets. "Flying IoT: Toward Low-Power Vision in the Sky" by Hasan Genc and colleagues explores a computing paradigm in which data are collected at the edge by ULP processors but are processed by high-performance computing resources in the cloud. While this approach enables edge systems to function within their restrictive constraints, it obviously introduces a communication bottleneck. The article explores the proposed tag-team edge–cloud computing paradigm in a stress-test scenario—a drone that requires real-time results for computations performed in the cloud. The authors investigate how to design ULP systems that can meet real-time deadlines while simultaneously meeting requirements for low power, small form factor, and low cost by harnessing cloud computing intelligently.

## Visual IoT

Visual computing at the edge has clear applications to future IoT devices, with applications ranging from security and surveillance to augmented reality devices. Visual data collected through today's high-resolution cameras tend to demand quite high bandwidth, and the number of such cameras is exploding as low-cost image sensors become common. This means that sending all of the visual data in the cloud for computing is impractical, and edge-based solutions are a growing necessity. In "Visual IoT: Ultra-Low-Power Processing Architectures and Implications," Vui Seng Chua and colleagues describe a mixed-mode approach to such systems, ranging from static image feature detection to dynamic video analytic applications. Neural networks are now an essential component to any type of visual computing application, and the article describes challenges and opportunities with neural network hardware accelerator designs for application in the visual IoT realm.

The articles in this special issue highlight some of the critical research and explore some of the potentially viable solutions that will help to advance the state-of-art toward a more power- and energy-efficient future for ULP processing.

## Time-Based Stochastic Computing

Stochastic computing is a potentially promising technology for ULP systems because it allows extreme reductions in system hardware for certain functions. For example, a multiplier, which can be synthesized as thousands of gates in a traditional digital circuit, can be implemented with a single logic gate in a stochastic computing circuit. This is an exciting prospect for applications that are amenable to stochastic processing, such as real-time image or video processing, since they can be supported with hardware that has orders of magnitude smaller area and power requirements than traditional hardware architectures for the applications. However, one of the main drawbacks of existing approaches for stochastic computing in the context of ULP processing is that they reduce power and area but increase energy due to the data encoding used, which represents values as a probabilistic bitstream. This potentially makes stochastic computing infeasible for the vast majority of ULP systems, which are severely energy constrained (such as energy harvesting or battery-powered systems). "An Overview of Time-Based Computing with Stochastic Constructs" by M. Hassan Najafi and colleagues provides an overview of a new time-based encoding that uses pulse-width modulation to harness stochastic computing's strengths—namely, ultra-low power and area—for ULP computing while allowing stochastic computing circuits to reach ultra-low energy targets as well.

### Ultra-Low-Power Security Constructs

IoT systems will be successful when they become a pervasive element in our society. For this to happen, they will become embedded into our daily lives in areas where security and privacy issues are paramount. For example, if life-saving medical equipment or self-driving cars are susceptible to hacking attacks, practical deployments will be slow due to safety and regulatory concerns. IoT systems are susceptible to multiple attack vectors due both to their placement in potentially hostile environments and their network connectivity requirements. Due to cost reasons, it is also not practical to deploy significant hardware resources to maintain security and privacy. In "Hardware Designs for Security in Ultra-Low-Power IoT Systems: An Overview and Survey," Kaiyuan Yang and colleagues explore a range of low-power and low-cost hardware building blocks that can provide the underpinnings for security and privacy at the higher levels. Examples of such blocks include physically unclonable functions (PUFs) that rely on device properties to provide a unique signature that provides an authentication code for a given system. The article outlines a taxonomy of designs that can be used to develop PUFs and describes several practical hardware implementations of PUFs that have been realized in silicon.

We appreciate all the authors who submitted papers to this issue, and we thank the anonymous reviewers for their efforts. We hope readers will enjoy this special issue of *IEEE Micro*.

**David Brooks** is the Haley Family Professor of Computer Science at Harvard University. Contact him at dbrooks@eecs.harvard.edu.

**John Sartori** is an assistant professor at the University of Minnesota. Contact him at jsartori@umn.edu.

# CMOS Scaling Trends and Beyond

**Scaling transistors and following Moore's law have served the industry well for more than 50 years in providing integrated circuits that are denser, cheaper, higher performance, and lower power. This article describes trends in CMOS scaling over the past decade and discusses some of the new device options and technology directions being explored to continue scaling into the future.**

**Mark T. Bohr,**
**Ian A. Young**
*Intel*

Gordon Moore famously predicted in his 1965 paper that the number of components per chip would continue to increase by a factor of two every year.[1] The goals of following Moore's law are to decrease the cost per component and reduce the power consumed per component. In 1975, Moore updated his earlier prediction by forecasting that components per chip would increase by a factor of two every two years, and that this would come from the combination of scaling component size and increasing chip area.[2] Back in 1965, the industry was producing chips using a minimum feature size of approximately 50 μm totaling about 50 components. Today's leading chips use a minimum feature size of approximately 10 nm and incorporate several billion transistors.

Robert Dennard and colleagues described in 1974 a scaling methodology for metal-oxide-semiconductor field-effect transistors (MOSFETs) that would deliver consistent improvements in transistor area, performance, and power reduction.[3] The methodology called for the scaling of transistor gate length, gate width, gate oxide thickness, and supply voltage all by the same scale factor, and increasing channel doping by the inverse of the same scale factor (see Figure 1). The result would be transistors with smaller area, higher drive current (higher performance), and lower parasitic capacitance (lower active power). This method for scaling MOSFET transistors is generally referred to as "classic" or "traditional" scaling and was very successfully used by the industry up until the 130-nm generation in the early 2000s.

For the past 20 years, we have been developing new generations of process technologies on a two-year cadence, and each generation scaled the minimum feature size by approximately 0.7 times to deliver an area scaling improvement of about 0.5 times (see Figure 2). Thus, we have been doubling transistor density every two years. But recent technology generations (such as 14 nm and 10 nm) have taken longer to develop than the normal two-year cadence, owing to increased process complexity and an increased number of photomasking steps. Nonetheless, Intel's 14-nm and 10-nm technologies have provided better-than-normal transistor density improvements that keep us on pace with increasing transistor density at a rate of doubling about every two years.

0272-1732/17/$33.00 © 2017 IEEE

| Device or circuit parameter | Scaling factor |
|---|---|
| Device dimension *tox, L, W* | 1/κ |
| Doping concentration *Na* | κ |
| Voltage *V* | 1/κ |
| Current *I* | 1/κ |
| Capacitance *εA/t* | 1/κ |
| Delay time/circuit *VC/I* | 1/κ |
| Power dissipation/circuit *VI* | 1/κ2 |
| Power density *VI/A* | 1 |

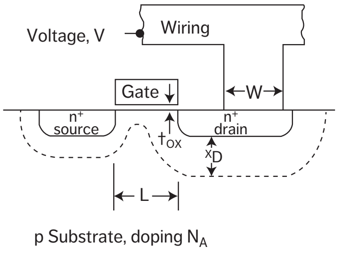


**Figure 1.** Traditional MOSFET scaling as described by Robert Dennard.

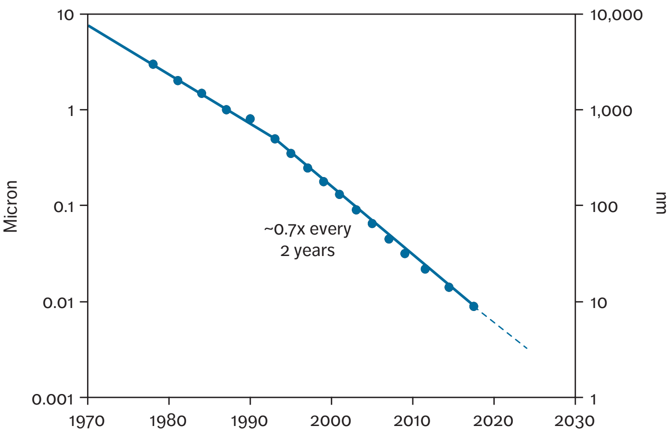


**Figure 2.** Minimum feature size scaling trend for Intel logic technologies.

## Transistor Innovations

As mentioned earlier, traditional MOSFET scaling worked well up until the 130-nm generation in the early 2000s. By that generation, the SiO2 gate oxide thickness had scaled to about 1.2 nm, and electron tunneling through such a thin dielectric was becoming a significant portion of total transistor leakage current. We had reached the limit for scaling transistors using traditional methods, and we needed to start introducing innovations in transistor materials and structure to continue scaling.

One of the first significant innovations was the introduction of strained silicon transistors on Intel's 90-nm technology in 2003.[4] This innovation used tensile strain in *n*-channel MOS (NMOS) transistor channels to increase electron mobility and compressive strain in *p*-channel MOS (PMOS) channels to increase hole mobility (see Figure 3). Tensile strain was induced by adding a high-stress film above the NMOS transistor. Compressive strain was induced by replacing the PMOS source-drain regions with epitaxial SiGe depositions. The resultant increases in electron and hole mobility provided increased transistor drive currents without having to further scale the SiO2 gate oxide thickness. This strained silicon technique has been adopted by all major semiconductor companies and continues to be used on the latest 10-nm technologies.

The need to improve the transistor gate dielectric to continue scaling could not be avoided, and Intel's 45-nm technology in 2007 first introduced high-κ metal gate transistors.[5] The traditional SiO2 gate oxide was replaced by a hafnium-based high-κ dielectric. The high-κ dielectric both reduced gate oxide leakage current and improved transistor drive current. The traditional doped-polysilicon gate electrode was replaced by metal electrodes with separate materials for NMOS and PMOS to provide optimal transistor threshold voltages. The combination of high-κ dielectric and metal gate electrodes (see Figure 4) was a revolutionary process change that provided significant improvements in transistor performance while also reducing transistor leakage current. High-κ metal gate transistors are now universally used on advanced logic technologies.

The next major transistor innovation was the introduction of FinFET (tri-gate) transistors on Intel's 22-nm technology in 2011.[6] Traditional planar MOSFETs had been able to scale transistor gate length down to about 32 nm to deliver good performance and density while also maintaining low off-state leakage. But scaling the gate length below 32 nm was problematic without sacrificing either performance or leakage. A solution was to convert from a planar transistor structure to a 3D FinFET structure in which the gate electrode had better electrostatic control of the transistor channel formed in a tall narrow silicon fin (see Figure 5). This improved electrostatic control provided scaled transistors with steeper sub-threshold slope (see Figure 6a). Steeper sub-threshold slope either provided transistors with lower off-state leakage or allowed threshold voltage to be reduced, which enabled improved performance at low operating voltage (see Figure 6b). Operating integrated circuits at a lower voltage is highly desired in order to reduce active power consumption. All advanced logic technologies now use FinFET transistors

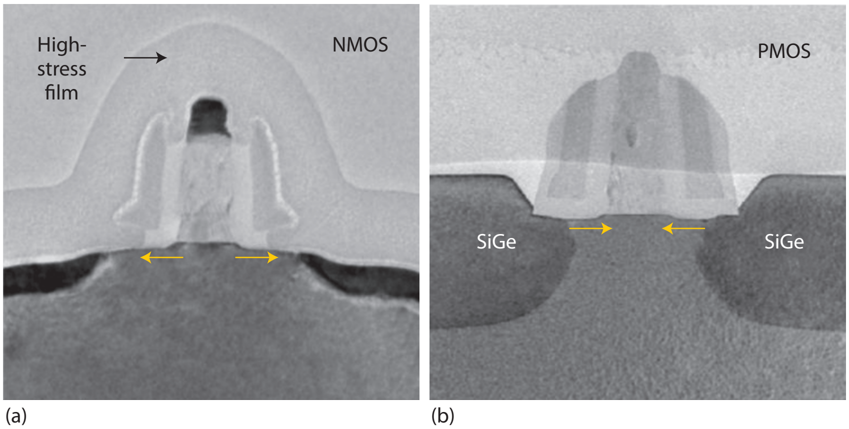


**Figure 3.** Channel strain techniques used on 90-nm generation transistors. (a) NMOS transistor using SiN cap layer; tensile channel strain. (b) PMOS transistor using SiGe source-drain; compressive channel strain.

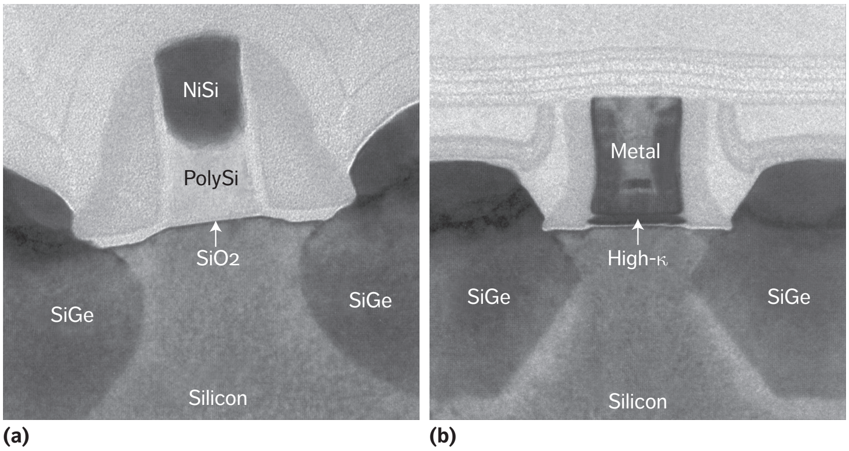


**Figure 4.** Comparison of transistor structures. (a) 65-nm generation transistor using SiO2 dielectric; polysilicon gate electrode. (b) 45-nm generation transistor using hafnium-based dielectric; metal gate electrode.

for their good density and superior low-voltage performance compared to planar transistors. As Figure 7 shows, when traditional MOSFET scaling ran out of steam in the early 2000s, innovations such as strained silicon, high-κ metal gate, and FinFETs were needed, and we must now continually invent new transistor materials and structures to continue scaling.

## Recent Logic Technologies

Intel's 14-nm logic technology started volume production early in 2014. This was Intel's second-generation FinFET technology, and it used advanced features such as 70-nm transistor gate pitch, 42-nm fin pitch, 52-nm interconnect pitch, double patterning techniques, and a 6-T SRAM bitcell area of 0.0588 $\mu m^2$.[7] This technology took longer to develop and get ready for volume manufacturing due to the increased process complexity and mask count: about 2.5 years instead of the normal 2-year cadence. But this technology also provided better-than-normal area scaling. Instead of the 0.5 times area scaling that new technology generations normally provide, Intel's 14-nm technology provided about 0.37 times logic area scaling compared to the previous 22-nm technology (see Figure 8).

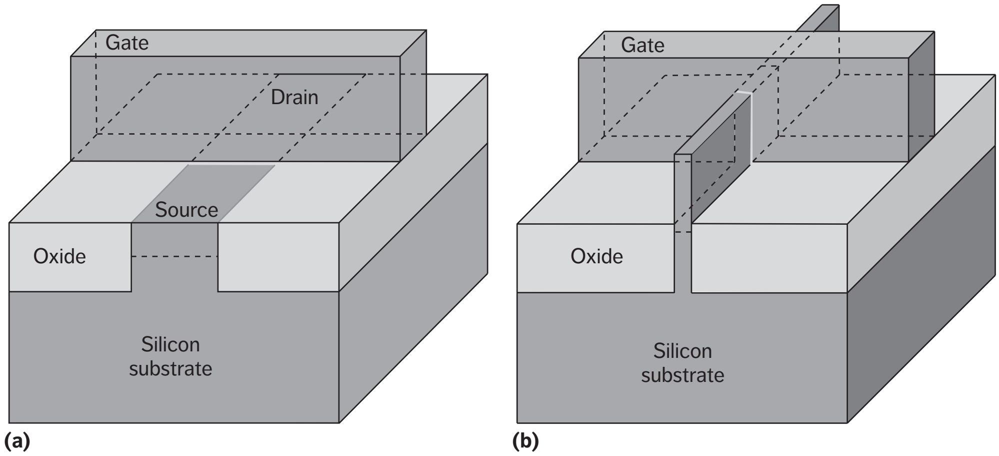


Figure 5. Comparison of transistor structures. (a) Planar transistor. (b) FinFET transistor.

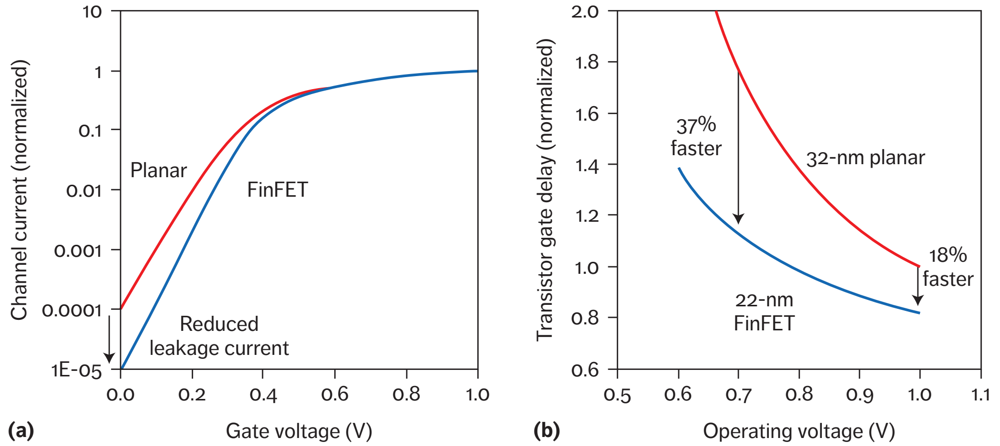


Figure 6. Comparison of planar versus FinFET transistor electrical characteristics. (a) Channel current versus gate voltage. (b) Transistor gate delay versus operating voltage.

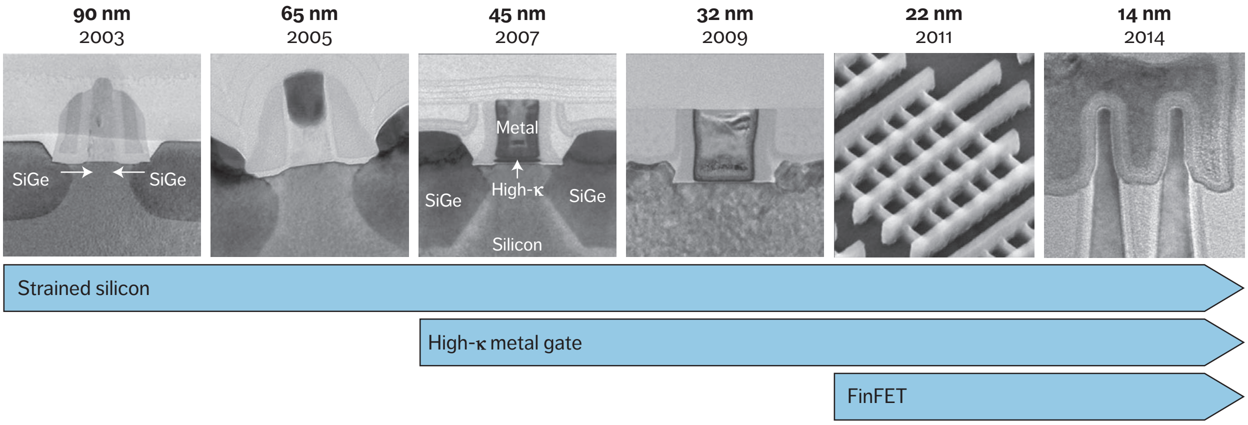


Figure 7. Six generations of Intel transistor innovations used to continue scaling.

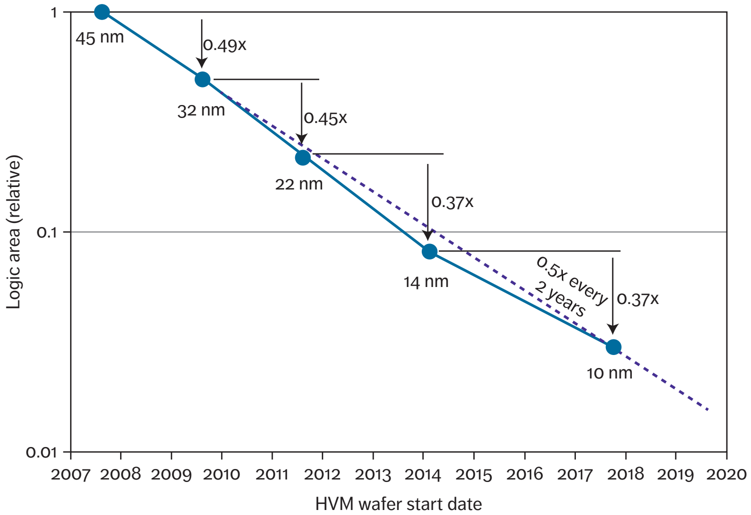


**Figure 8.** Intel's trend for scaling logic circuit area over the past five generations.

Intel's newest 10-nm logic technology is scheduled to start product shipments before the end of 2017. This 10-nm technology introduces some advanced process features such as 54-nm transistor gate pitch, 34-nm fin pitch, 36-nm interconnect pitch, quad patterning techniques, and a 6-T SRAM bitcell area of 0.0312 $\mu m^2$. This technology also introduces some important density-improvement techniques: single dummy gates adjacent to logic cells and the ability to make transistor gate connections directly over active gates. Again, this technology took more than two years to develop and get ready for volume manufacturing due to increased process complexity and mask count, but it also delivers better-than-normal area scaling. The innovative features on this technology deliver about 0.37 times logic area scaling compared to the previous 14-nm generation. As Figure 8 shows, Intel's 14-nm and 10-nm generations each took more than two years to develop, but they also took bigger steps in terms of scaling logic area. As a result, Intel logic technologies continue to deliver improved area scaling at the rate of about 0.5 times every two years.

It's apparent that after more than 50 years we're continuing to scale transistor area, but are we delivering the other promises of Moore's law and Dennard's scaling methodology: lower cost per transistor, higher performance, and lower active power? Figure 9a shows how Intel logic technologies have been scaling transistor area, and Figure 9b shows the trend of increasing wafer cost due to increased process complexity. Figure 9c shows how the cost per transistor continues to come down due to better-than-normal area scaling. Figure 10 shows Intel's trends for improving transistor performance (Figure 10a) and reducing dynamic capacitance to lower active power (Figure 10b). Figure 10c shows how performance improvement divided by active power consumption (performance per watt) continues to improve with each generation. Different products on a given technology can choose to tune the transistor or design to deliver better performance or lower power, depending on what the application values most. Figure 10 also shows the strategy of developing performance-enhanced versions of each generation (for example, 10+ and 10++) to deliver improved performance per watt and extend the life of these technologies.

## Future Device Options

MOSFET transistor researchers are exploring device structure and channel material changes to enable further generations of MOSFET scaling. The MOSFET implemented with stacks of multiple horizontal nanowires (see Figure 11b) is one option that, due to its superior electrostatics, could enable further gate-length scaling beyond what the FinFET (see Figure 11a) can achieve. MOSFETs with III-V semiconductor channel materials are a promising option for realizing a higher-mobility channel than silicon (see Figure 12). This higher mobility can be used either to provide higher drive current and higher performance or to allow the MOSFET to be operated at lower voltage for lower active power.[8]

Lowering the supply voltage of CMOS logic below about 0.5 V leads to a dilemma between logic having high performance and high static leakage current versus logic with lower performance and low leakage current. This is due to the choice of MOSFET threshold voltage and its electron "thermal tail" determined sub-threshold gate voltage swing of 60 mV/decade. One alternative transistor option that operates differently than a MOSFET (and as such could be classified as a *beyond-CMOS device*) is the Tunneling Field Effect Transistor (TFET).[9] The TFET can achieve subthreshold swing smaller than 60 mV/decade (that

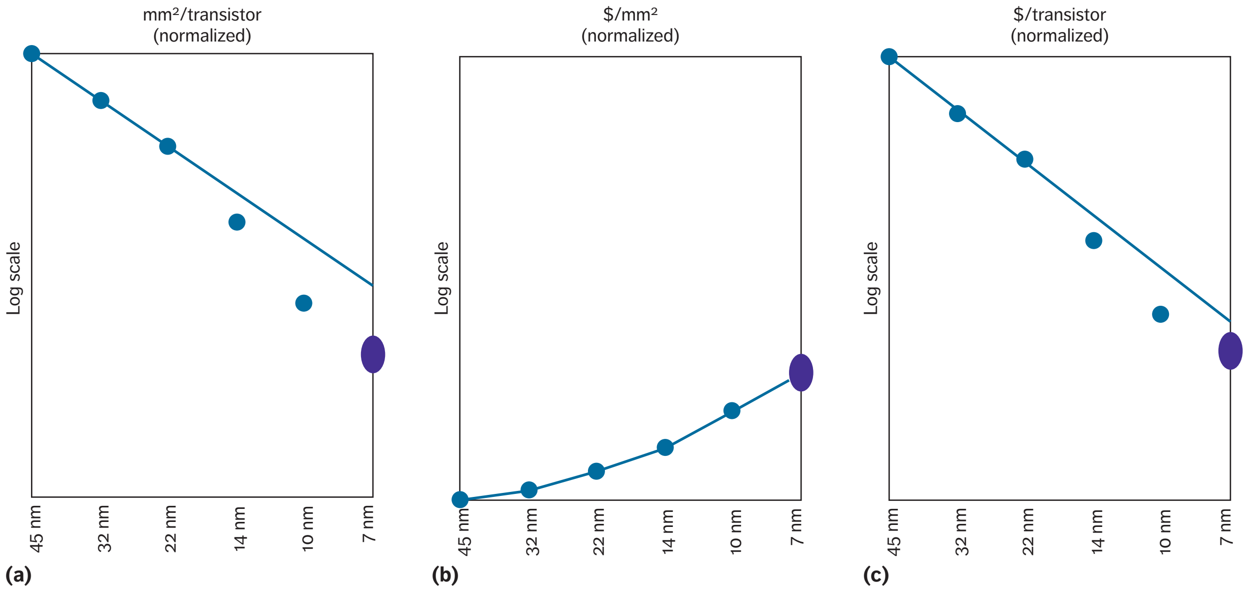


**Figure 9.** Trends for improving logic transistor area and cost per transistor. (a) Area per transistor. (b) Wafer cost. (c) Cost per transistor.

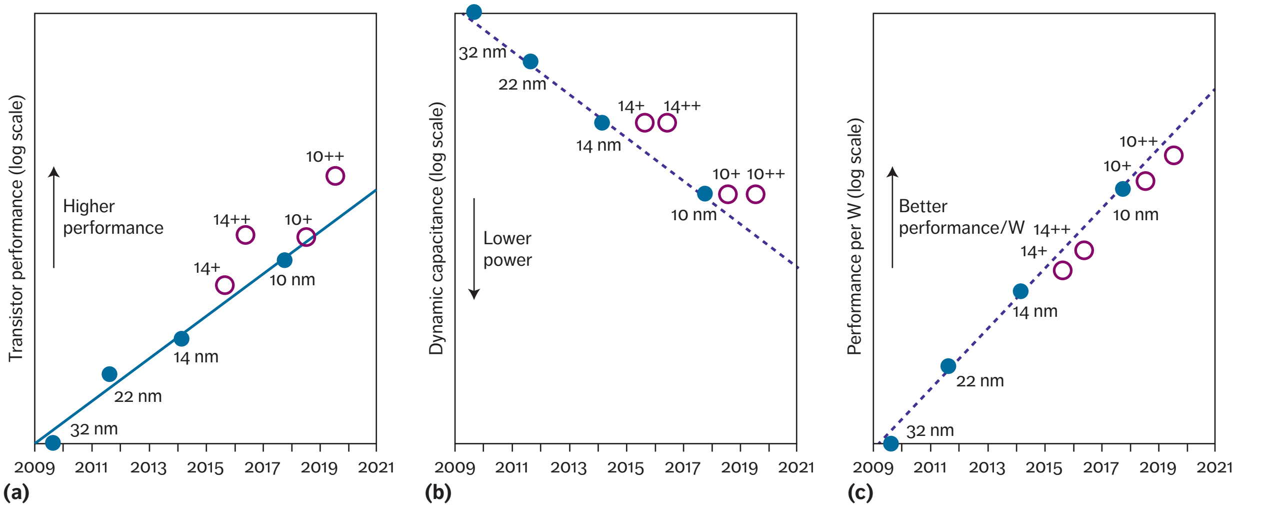


**Figure 10.** Trends for improving transistor performance and reducing active power. (a) Transistor performance. (b) Dynamic capacitance. (c) Performance per watt.

is, steeper current turn-on) and can therefore operate at a lower power supply voltage than a MOSFET. Figure 13 shows drain current versus gate voltage simulation results for nanowire TFETs implemented with different III-V semiconductor materials.

While the success of information technology progress in the past 50 years was based on Moore's law[1,2] scaling and mostly one underlying technology—CMOS transistors—present-day research efforts are exploring logic technologies going beyond CMOS,[10] with an objective to complement CMOS rather than to replace it. The goal of Beyond-CMOS research is to identify and enable an integrated circuit technology that will be more energy efficient than CMOS. If this happens, it will support the continuation of Moore's law.

Beyond-CMOS research efforts have been underway for 10 years, being funded in the US

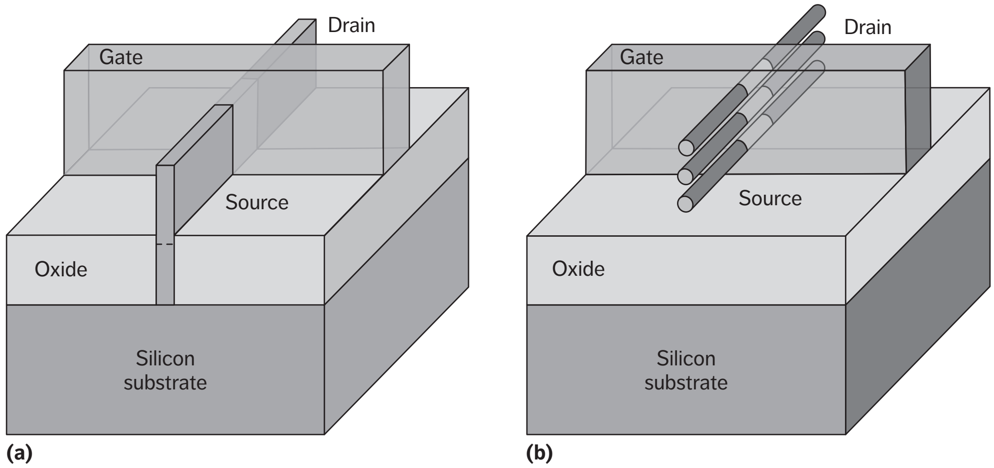


**Figure 11.** Comparison of transistor structures. (a) FinFET transistor. (b) Nanowire transistor.

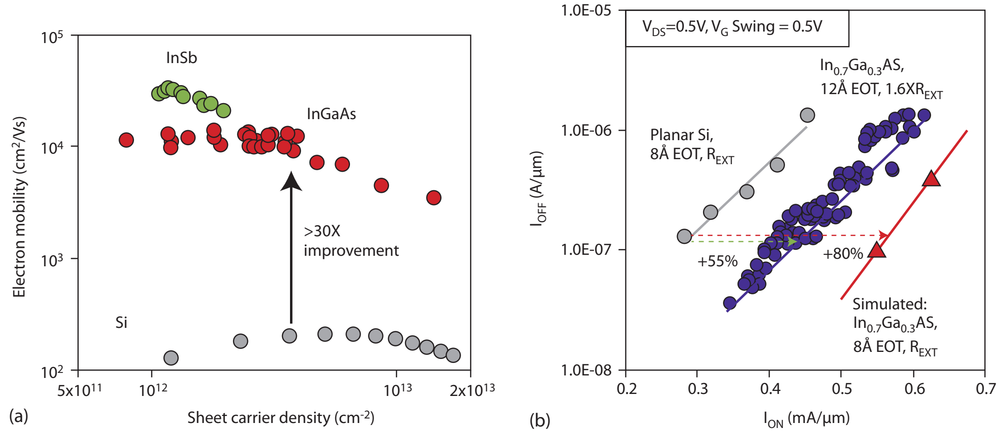


**Figure 12.** Comparison of III-V and silicon transistor electrical characteristics. (a) Electron mobility versus carrier density. (b) Off-current versus on-current.

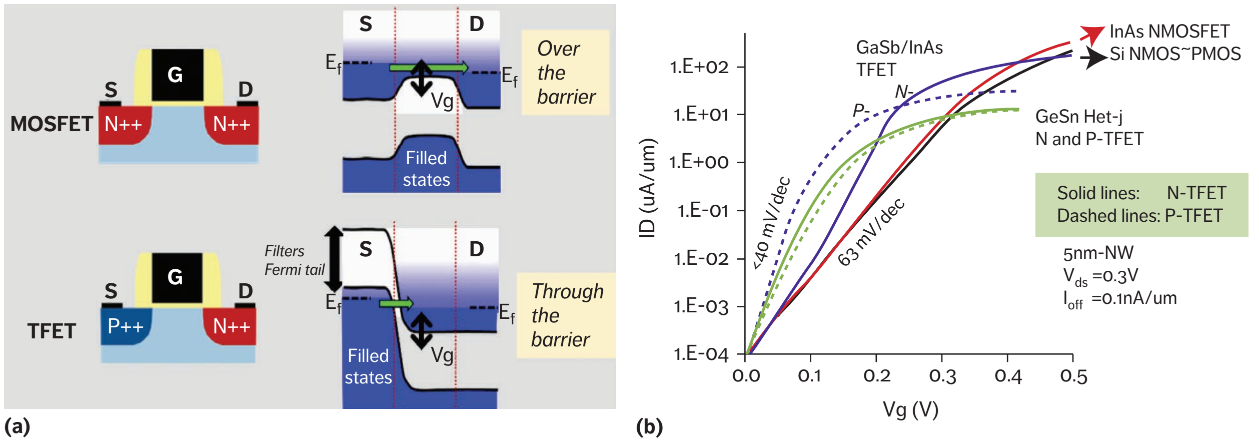
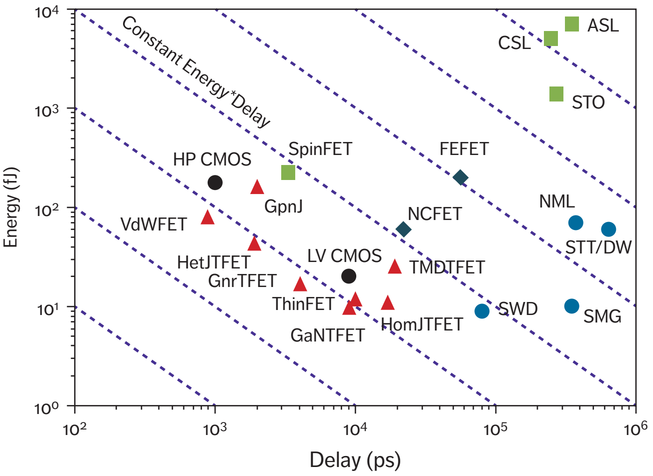


**Figure 13.** Comparison of Tunneling FET and MOSFET transistors. (a) Transistor structures and channel current modulation techniques. (b) Drive current versus gate voltage electrical characteristics.



| | |
|---|---|
| HP CMOS | High-performance CMOS |
| LV CMOS | Low-voltage CMOS |
| VdWFET | Van der Waals FET |
| HomJTFET | Homojunction III-V TFET |
| HetJTFET | Heterojunction III-V TFET |
| GnrTFET | Graphene nanoribbon TFET |
| ThinFET | 2D hetero interlayer TFET |
| GaNFET | GaN TFET |
| TMDTFET | Trans metal dichalchogenide |
| GpnJ | Graphene pn junction |
| FEFET | Ferroelectric FET |
| NCFET | Negative capacitance FET |
| SpinFET | Spin FET |
| ASL | All spin logic |
| CSL | Charge spin logic |
| STT/DW | Spin torque domain wall |
| SMG | Spin majority gate |
| STO | Spin torque oscillator |
| SWD | Spin wave device |
| NML | Nanomagnetic logic |

**Figure 14.** Simulated switching energy and delay for a 32-bit arithmetic logic unit circuit for CMOS and for various beyond-CMOS device options.

in large part via the Semiconductor Research Corporation (SRC).[11] The expectation of this industry–university research consortium 10 years ago was that this field would produce a computing technology that is better than CMOS for the majority of its applications. Reality showed that, among many impressive proposals and demonstrations, none of them beat CMOS. However, they do possess many valuable features, such as low-power operation and non-volatility. Thus, the current vision is that beyond-CMOS circuits will replace CMOS in some critically important computation or information processing applications. They would be monolithically integrated with CMOS on the same chip or packaged together in a multichip module.

Another expectation was that beyond-CMOS circuits would not require any MOSFETs as part of their operation and could maybe even eliminate any charge currents in the quest for energy efficiency. This did not come true: a thorough circuit analysis reveals that a MOSFET transistor is needed to supply power and for clocking and control of the logic circuit operation. However, this does not preclude pursuing the key direction of beyond-CMOS research, which is to discover and invent computation that can operate at significantly lower supply voltages than CMOS to enable dramatic improvements in energy efficiency.

To this end, beyond-CMOS benchmarking[12,13] (see Figure 14) was helpful in evaluating the potential of various materials and devices to implement computing technologies. It enabled the setting of expectations for power and performance and revealed some pathways for improvement. Experimental demonstrations have not yet achieved the theoretical modeling projections put forward in the benchmarking. One reason is that each computing technology requires solving numerous fabrication challenges.[14]

The various materials implementations of the TFET (see Figure 13) have shown that they have improved energy-delay product (and therefore power and performance) over the future CMOS technology node that they are benchmarked against (see Figure 14): the *International Technology Roadmap for Semiconductors* prediction in 2011 of the 2018 CMOS node. With a potential three-times improvement in energy-delay product over CMOS, this is starting to be an interesting device option, and it does not require a drastic change in circuit design for logic while it offers some additional circuit functionality.

The spintronic devices in Figure 14 operate with a wide range of switching energy and at slower switching speed compared to CMOS. The spintronic devices that match the best CMOS switching energy use magnetoelectric materials to do the switching of nanomagnets. Although they are slower than CMOS, they have the added benefit of being non-volatile. Non-volatility in the logic device has the potential to provide energy efficiency benefits by taking advantage of it in the computing microarchitecture.

A historic similarity for beyond-CMOS research is fitting—the disruption of bipolar transistors for computing logic by CMOS.[15] The latter had the advantage of lower power, but it was slower than bipolar and was much more difficult to manufacture. We believe that the same drive toward lower-power computing should compel technologists to solve implementation problems for beyond-CMOS computing. One should understand that a 100-times improvement in the energy-delay product (which is equivalent to more than four generations of historic Dennard-era CMOS scaling) will justify its integration for computer and information processing systems. As research into beyond-CMOS continues, it is going to be critical that researchers focus on the leading options and eliminate the less attractive ones. To do this will require all levels of benchmarking analysis covering materials, devices, circuits, and computing architectures.[16]

Transistor scaling, and in particular MOSFET scaling, has served our industry well for more than 50 years by providing new generations of integrated circuit technology that simultaneously provided improved density, higher performance, reduced power consumption, and lower cost per transistor. At times, transistor scaling was provided by the use of simple evolutionary techniques, but at other times more revolutionary technology changes were required, such as switching from bipolar to MOSFET transistors, and more recently by implementing high-κ metal gate and FinFET transistors. Furthermore, 14-nm and now 10-nm generations have continued to deliver the promises of Moore's law for improved density, performance, power, and cost.

Scaling of the MOSFET transistor will continue for future CMOS generations as far as researchers can see by exploiting the options in device structure and channel materials. Beyond-CMOS research into quantum nanoelectronics or nanomagnetics is aimed at inventing and developing another integrated circuit technology that offers improved power and performance. This will happen at the appropriate time when it can be integrated onto CMOS in a manufacturing process that offers lower cost per function and improved power and performance.

## References

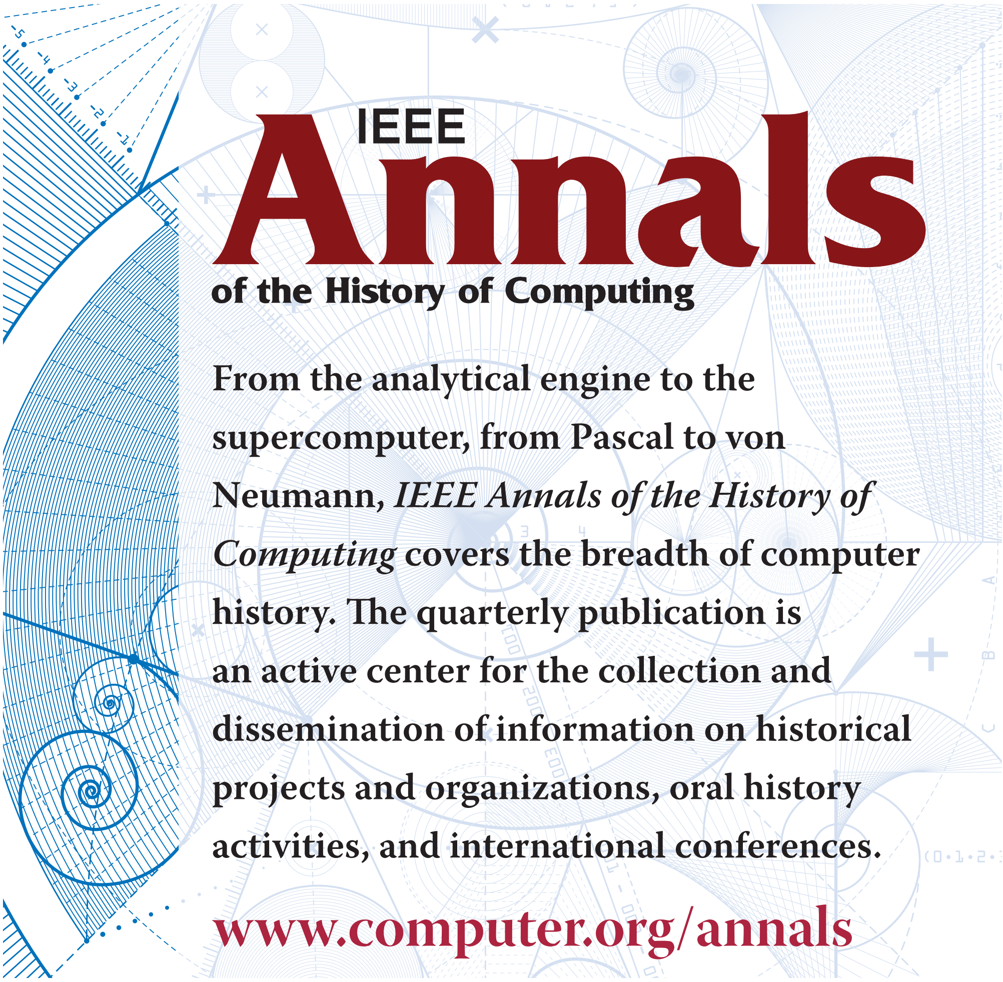
1. G. Moore, "Cramming More Components onto Integrated Circuits," *Electronics*, vol. 38, no. 8, 1965, pp. 114–117.
2. G. Moore, "Progress in Digital Integrated Electronics," *IEEE Int'l Electron Devices Meeting Technical Digest*, 1975, pp. 11–13.
3. R. Dennard et al., "Design of Ion-Implanted MOSFETs with Very Small Physical Dimensions," *IEEE J. Solid State Circuits*, vol. 9, no. 5, 1974, pp. 256–268.
4. T. Ghani et al., "A 90nm High Volume Manufacturing Logic Technology Featuring Novel 45nm Gate Length Strained Silicon CMOS Transistors," *IEEE Int'l Electron Devices Meeting Technical Digest*, 2003, pp. 978–980.
5. K. Mistry et al., "A 45nm Logic Technology with High-k + Metal Gate Transistors, Strained Silicon, 9 Cu Interconnect Layers, 193nm Dry Patterning, and 100% Pb-free Packaging," *IEEE Int'l Electron Devices Meeting Technical Digest*, 2007, pp. 247–250.
6. A.C. Auth et al., "A 22nm High Performance and Low-Power CMOS Technology Featuring Fully-Depleted Tri-gate Transistors, Self-Aligned Contacts and High Density MIM Capacitors," *Proc. Symp. VLSI Technology*, 2012, pp. 131–132.
7. S. Natarajan et al., "A 14nm Logic Technology Featuring 2nd Generation FinFET Transistors, Air-Gapped Interconnects, Self-Aligned Double Patterning and a 0.0588um2 SRAM Cell Size," *IEEE Int'l Electron Devices Meeting Technical Digest*, 2014, pp. 71–74.
8. R. Kim, U.E. Avci, and I.A. Young, "Comprehensive Performance Benchmarking of III-V and Si nMOSFETs (Gate Length = 13 nm) Considering Supply Voltage and OFF-Current," *IEEE Trans. Electron Devices*, vol. 62, no. 3, 2015, pp. 713–721.
9. U.E. Avci et al., "Energy Efficiency Comparison of Nanowire Heterojunction TFET and Si MOSFET at Lg = 13 nm, Including P-TFET and Variation Considerations," *IEEE Int'l Electron Devices Meeting Technical Digest*, 2013, pp. 33–36.
10. W.M. Holt, "1.1 Moore's Law: A Path Going Forward," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2016, pp. 8–13.
11. J.J. Welser et al., "The Quest for the Next Information Processing Technology,"

*J. Nanoparticle Research*, vol. 10, 2008, pp. 1–10.

12. D.E. Nikonov and I.A. Young, "Overview of Beyond-CMOS Devices and a Uniform Methodology for their Benchmarking," *Proc. IEEE*, vol. 101, no. 12, 2013, pp. 2498–2533.
13. D.E. Nikonov and I.A. Young, "Benchmarking of Beyond-CMOS Exploratory Devices for Logic Integrated Circuits," *IEEE J. Exploratory Solid-State Computational Devices and Circuits*, vol. 1, 2015, pp. 3–11.
14. I.P. Radu et al., "Spintronic Majority Gates," *IEEE Int'l Electron Devices Meeting Technical Digest*, 2015, p. 32.5.1–32.5.4.
15. S. Borkar, "Electronics Beyond Nano-Scale CMOS," *Proc. 43rd ACM/IEEE Design Automation Conf.*, 2006, pp. 807–808.
16. I.A. Young and D.E. Nikonov, "Principals and Trends in Quantum Nano-Electronics and Nano-Magnetics for Beyond CMOS Computing," to be published in *Proc. European Solid-State Device Research Conf.*, 2017.

**Mark T. Bohr** is an Intel Senior Fellow in the Logic Technology Development group at Intel. His research interests include scaling logic transistors, interconnects, and memory cells. Bohr received a master's degree in electrical engineering from the University of Illinois. Contact him at mark.bohr@intel.com.

**Ian A. Young** is an Intel Senior Fellow in the Components Research group at Intel. His research interests include novel embedded memory and quantum nanoelectronic and nanomagnetic devices for energy-efficient integrated circuits beyond-CMOS. Young received a PhD in electrical engineering from the University of California at Berkeley. Contact him at ian.young@intel.com.



# Low-Power Convolutional Neural Network Processor for a Face-Recognition System

**The authors propose a low-power convolutional neural network (CNN)-based face recognition system for user authentication in smart devices. The system comprises an always-on functional CMOS image sensor (CIS) for imaging and face detection, and a low-power CNN processor (CNNP) for face verification. Implemented in 65-nm CMOS technology, the system consumes 0.62 mW to evaluate one face at 1 fps and achieves 97 percent accuracy.**

**Kyeongryeol Bong, Sungpill Choi, Changhyeon Kim, Hoi-Jun Yoo**
*Korea Advanced Institute of Science and Technology*

With the increase in the number of smart devices per person,[1] researchers are investigating always-on face recognition for these devices to recognize, authenticate, and interact with users. Compared to fingerprint authentication, which requires users to make contact with each device, face authentication has an advantage in its camera-based operation, which provides a nonintrusive way to unlock multiple devices. Moreover, in the Internet of Things (IoT) era, in which every device should be intelligent and interact with its user, always-on face recognition is considered an essential functionality.

The most challenging part of realizing face-based unlock for battery-powered wearable devices such as smart watches is achieving low power consumption. Typically, always-on systems comprise two stages: always-on event detection and event-driven processing.[2] The always-on event detection of face recognition includes image capturing and face detection, and the event-driven processing includes face verification. Because face verification should have high accuracy to prevent unauthorized users from getting permission, a convolutional neural network (CNN) becomes an essential element to satisfy the accuracy requirement, and it results in much more power consumption in the event-driven tasks compared to that in the always-on tasks.[3,4] However, the always-on tasks may have more impact on the

0272-1732/17/$33.00 © 2017 IEEE

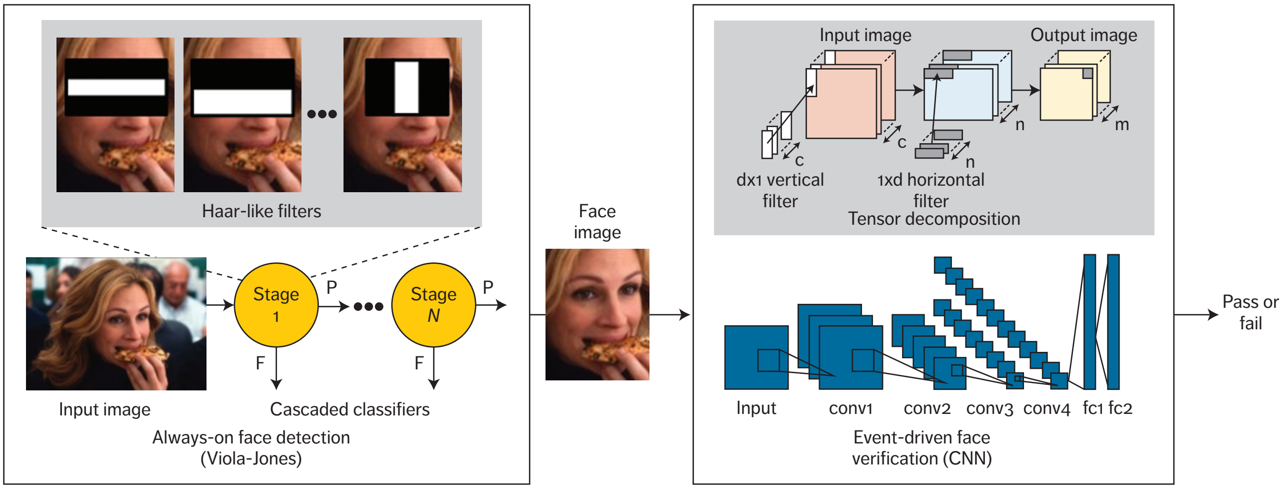


Figure 1. Overall processing flow. Face detection and verification using the Viola-Jones algorithm and convolutional neural network (CNN).

battery life according to the frequency of events; therefore, the power consumption of both stages should be independently minimized with dedicated architectures.

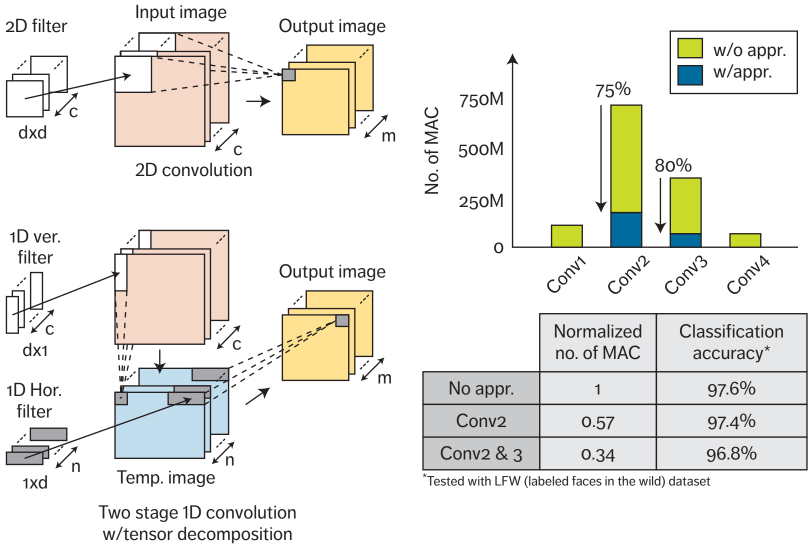
Previously, researchers proposed a low-power image sensor for always-on applications[5] and several hardware accelerators[4,6] for face recognition. However, in their system architecture, an image sensor chip and a digital processor chip were separated, and the digital processor performed the entire processing for both face detection and face verification. Hence, for the always-on tasks, the system architecture should deal with the imaging and face-detection processing independently, and the entire image data continuously generated from the always-on image sensor should be transferred to the digital processor to check whether a face is present. For event-driven face verification, researchers achieved insufficient accuracy due to the use of hand-crafted features,[6] and adopted CNN and proposed a dedicated hardware with dynamic voltage, accuracy, and frequency scaling for energy efficiency.[4]

In this article, we propose an always-on face recognition system to achieve low power consumption with high accuracy.[3] For always-on imaging and face detection, we propose a functional CMOS image sensor (CIS) architecture in which a face-detection accelerator is integrated with an image sensor in a single chip to reduce the chip-to-chip communication and remove frame buffer by transferring the face region-of-interest images only when faces exist. Moreover, we consider the use of column-level processing in the functional CIS to manage the required on-chip static RAM (SRAM) size during face-detection processing and to improve energy efficiency. For event-driven face verification, we present a CNN processor (CNNP). Because workload varies dynamically with the number of faces in a given input scene, dynamic voltage and frequency scaling (DVFS) from nominal voltage to near-threshold voltage is realized to minimize power consumption. In addition, we adopt tensor decomposition to reduce the convolutional layers' workload,[7] and the CNNP architecture based on transpose-read SRAM (T-SRAM) reduces the power consumption of using the tensor decomposition by enabling efficient local memory access.

## Overall Processing Flow

Figure 1 shows the proposed face recognition system's overall processing flow. We use the Viola-Jones algorithm for face detection,[8] and face verification adopts CNN to generate face descriptors for input faces and a simple classifier to classify the descriptors.

When an image frame is captured, subwindows are evaluated at different positions by sliding a subwindow in the face-detection stages. At this time, we evaluate multiple scales of subwindows to detect faces of various sizes presented in the scene. The Viola-Jones algorithm comprises a number of cascaded classifying stages,



| | Normalized no. of MAC | Classification accuracy* |
|---|---|---|
| No appr. | 1 | 97.6% |
| Conv2 | 0.57 | 97.4% |
| Conv2 & 3 | 0.34 | 96.8% |

*Tested with LFW (labeled faces in the wild) dataset

**Figure 2.** Tensor decomposition. Workload reduction by tensor decomposition and resulted accuracy degradation in face verification.

each of which contains several Haar-like filters. The classifying stages are processed in order, and in each stage, the given windows are passed or rejected based on the results of the Haar-like filters. The Haar-like filters are composed of black and white rectangular regions, and the result is given by comparing the intensity summation over those two regions. After the whole windows enter the first stage, only the windows passed can go forward to the next stage, and the windows that passed the last stage are classified as faces.

In face verification, the CNN's feedforward operation is performed for every detected face. This work used tensor decomposition for the convolutional filters[7] to reduce the CNN's large computational workload. As Figure 2 shows, a convolutional layer with a $d \times d \times c \times m$ filter is approximated by two convolutional layers with vertical dx1xcxn filters and horizontal 1xdxnxm filters. In our test, the tensor decomposition enables two to three times workload reduction, while the accuracy degradation is managed to be less than 1 percent based on the LFW dataset.[9]

## Functional CIS Architecture for Always-On Face Detection

Figure 3a shows the previous system architecture with a separated image sensor and digital processor. When performing the always-on tasks, including image capturing and face detection, in this architecture, even if the captured image frame contains no face, the whole image data should be streamed to the digital processor to check whether a face is present. In addition, it often requires a frame buffer to store the image frame until the processor completes the face-detection processing. Hence, the power reduction using a face-detection accelerator in a digital processor has been limited in reducing the overall power consumption of the always-on tasks, because the chip-to-chip communication must always be fully turned on, regardless of the existence of a face.

Figure 3b shows the proposed system architecture using the functional CIS to resolve this issue. This architecture integrates the face-detection accelerator with the image sensor in a single chip. Then, the functional CIS performs all of the always-on tasks, and it can control its output data to be only the region-of-interest images with detected faces, while turning off the CNN chip and its I/O when there is no event coming from the always-on functional CIS. Also, the amount of data generated by the functional CIS is significantly reduced from the entire image frame to the partial face images, and the CNN chip usually can keep this data with on-chip memory not accessing the off-chip frame buffer.

Figure 4 shows the overall architecture of the functional CIS. It comprises an image sensor,

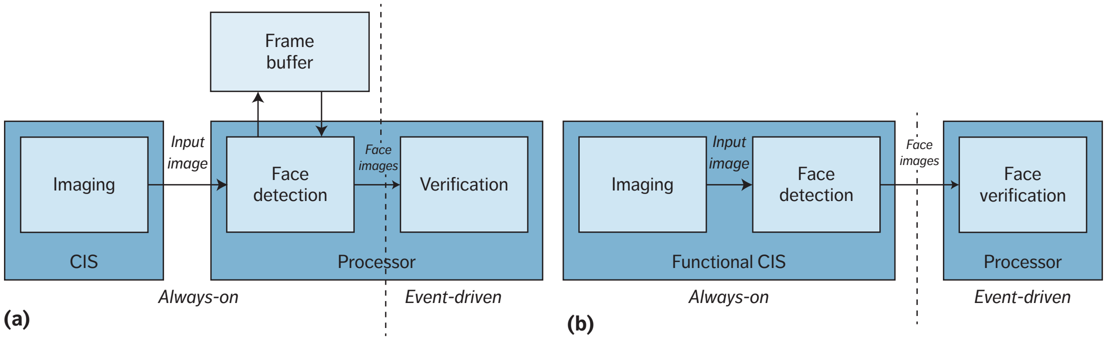


**Figure 3.** System architecture for face recognition: (a) previous system architecture with a separated image sensor and digital processor; (b) proposed system architecture with a functional CMOS image sensor (CIS).

a column-level analog face-detection unit (AFDU), a digital face-detection unit (DFDU), and a controller. Following the rolling shutter operation,[10] the pixel array of the image sensor is read out in a row-by-row manner with a fixed time interval. While the image sensor generates a row, it is stored in the analog memory of the AFDU, and the AFDU processes the first few stages of the cascaded classifiers. After that, only the passed subwindows are converted to the digital domain and moved to the window memory of the DFDU, and the remaining stages are handled by the DFDU. To sum up, the face-detection processing in the functional CIS is performed through two steps (see Figure 5a). This is to improve the energy efficiency of Haar-like filtering operations and to adjust the memory size required for the face-detection processing to the on-chip memory size of the functional CIS.

The conventional approach to process Haar-like filtering is based on the integral image.[8] Although it requires significant initial processing effort to generate the integral image, it makes the intensity summation over a rectangular block to be three add/sub operations, which is simple and energy efficient when processing a large number of Haar-like filters. However, as Figure 5b shows, greater than 50 percent and greater than 90 percent of the input windows are rejected until the first and the third stages of the cascaded classifier, respectively, and the energy efficiency when using the integral image is decreased for these early-rejected windows. The AFDU can improve it by processing the Haar-like filtering with direct summation of pixel intensities without any initial processing, which is more energy efficient when processing a small number of Haar-like filters. As Figure 5c shows, when combining the AFDU and the DFDU, the energy consumption first has the same value as AFDU, and after the third stage in this case, it follows the DFDU. Due to the energy consumption at the AFDU, the accumulated energy consumption after the third stage is larger than the case of using only the DFDU. However, because the

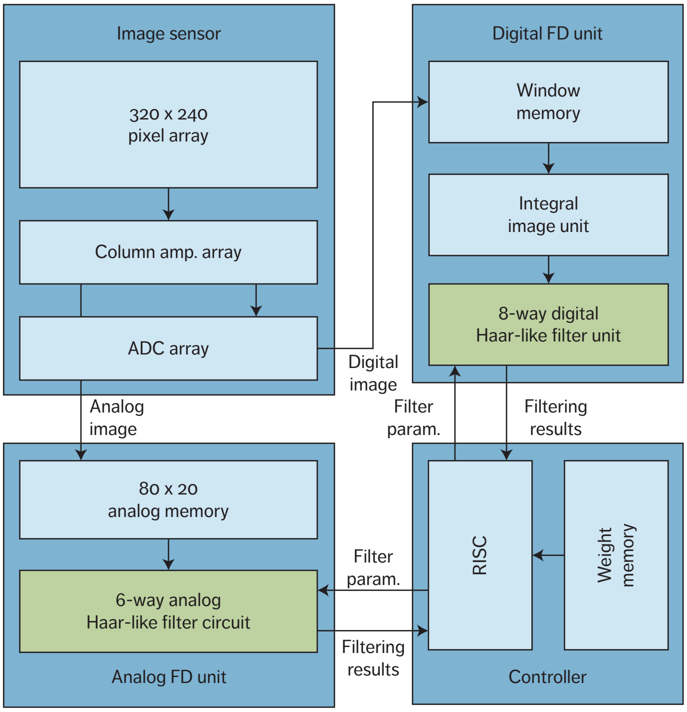


**Figure 4.** Functional CIS architecture. The integration of an image sensor and face-detection (FD) units in a single chip.

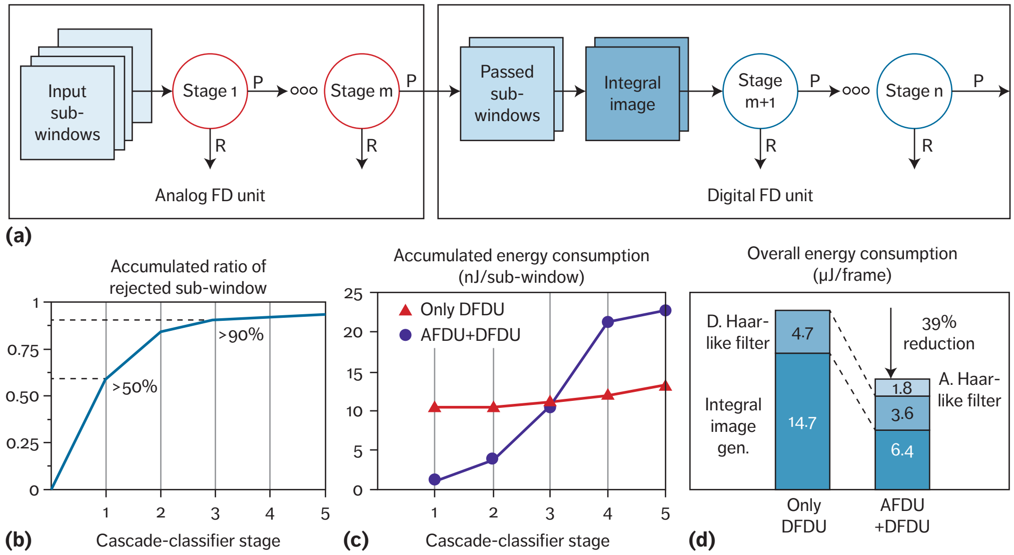


**Figure 5.** Face detection in the functional CIS. (a) Processing flow with the analog face-detection unit (AFDU). Implementation results: (b) the ratio of the rejected subwindows, (c) energy consumption for one subwindow, and (d) energy consumption for one frame.

number of the subwindows rejected until the third stage is much larger than the rest, the overall energy consumption is improved by 39 percent (see Figure 5d).

The number of classifying stages processed by the AFDU is usually determined on the basis of the energy consumption. However, the AFDU can process more stages than the energy-optimal point to control the data size of the passed subwindows, which varies dynamically depending on the input image. Plain background scenes make a few subwindows after going through the AFDU by the energy-optimal stage, but complex foreground scenes generate several hundreds of the passed subwindows. Because there is a limit to the number of the subwindows that can be stored in the on-chip memory—40 subwindows in this work—the DFDU's window memory can be overflowed when the AFDU transfers more subwindows than the DFDU finishes processing. To prevent this, the RISC monitors the available space in the window memory and the number of passed subwindows at the AFDU and extends the boundary stage over the energy-optimal point until enough space becomes available in the memory. The RISC can monitor and reconfigure the AFDU by accessing its memory-mapped registers.

## CNNP Architecture for Event-Driven Face Verification

Figure 6a shows the overall architecture of the CNNP. The CNNP comprises 4 × 4 processing elements and local distributed memory. The 4 × 4 processing elements are interconnected by a mesh-type network, and the boundary processing elements are connected to the external interfaces.

Figure 6b shows the processing element's detailed block diagram. The local SRAM can fetch 32 words per cycle to a register file. Each convolution unit of the processing element includes a 16-way SIMD MAC datapath, in which the input weights are shared. While the partial sums of the convolution are accumulated at the same column, the input operands of the MAC operation are given by loading a row vector of an input feature map from the register file or shifting the existing row by one column. When shifting, the operand at the headmost column can be transferred to the neighbor processing element to connect multiple processing elements for various sizes of feature maps. In a single processing element, four convolutional units are integrated, and the CNNP with a 4 × 4 processing element array can support 1,024 MAC operations per cycle.

Because the number of faces varies dynamically depending on input scenes, the

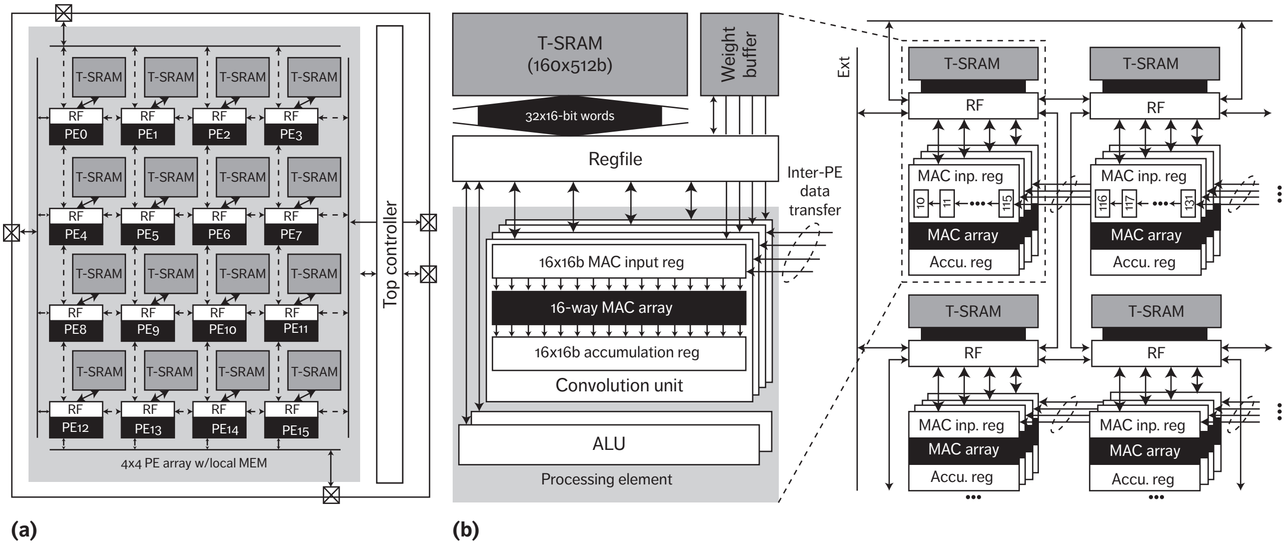


**Figure 6.** Convolutional neural network processor (CNNP) architecture. (a) 4 × 4 processing elements with local distributed memory. (b) Processing element block diagram.

face-verification workload also changes, and the CNNP doesn't have to operate with its maximum throughput. To minimize the power consumption when a small number of faces is given, the CNNP adopts DVFS. Because the CNN's processing cycles are deterministic, the DVFS mode of the CNNP is decided by the latency requirement and the number of faces detected. In this way, the CNNP's DVFS mode is changed at the frame rate of the face-recognition system. If there is no face, the CNNP is fully turned off.

In addition, as explained, this work adopted tensor decomposition to reduce the workload for processing the convolutional layer. When processing the horizontal filters, a single SRAM access can complete the convolution operation, because the direction of the horizontal filters is matched to the direction of row feature vectors. However, SRAM cannot read column feature vectors at once, whose elements are connected to the same bit line, and the vertical filtering must fetch it through multiple SRAM accesses. As Figure 7b shows, although tensor decomposition decreases the latency, the SRAM's activity factor—hence, the dynamic power consumption—increases due to the inefficient memory accesses during vertical filtering. This work utilizes T-SRAM, which has two read modes: normal read for accessing a row vector and transpose read for accessing a column vector.[3] Thanks to the T-SRAM, the inefficient memory access during the vertical filtering can be resolved by the transpose-read mode, and 21 percent more energy is saved with the tensor decomposition. As a result, the total energy consumption for processing a convolutional layer is decreased by 78 percent.

## Implementation Results

Figure 8 shows the chip photograph of the functional CIS and the CNNP fabricated using 65-nm CMOS technology and the performance summary. The functional CIS and the CNNP occupy 3.30 × 3.36 mm$^2$ and 4 × 4 mm$^2$ die area, respectively. At a 1 frame per second (fps) framerate, the functional CIS consumed 24 to 96 μW for imaging and face detection, depending on the number of faces in input scenes. The CNNP operates at 0.46 to 0.8 V supply voltage with 5 to 100 MHz operating frequency, and the peak power consumption with maximum processing element utilization is 5.3 and 211 mW, respectively.

Based on the voltage and the frequency pairs in Figure 9a, five DVFS modes are decided for the CNNP (see Figure 9b). The DVFS mode is selected to handle the faces detected from the input image frame within the latency requirement. At the fastest operating mode, the

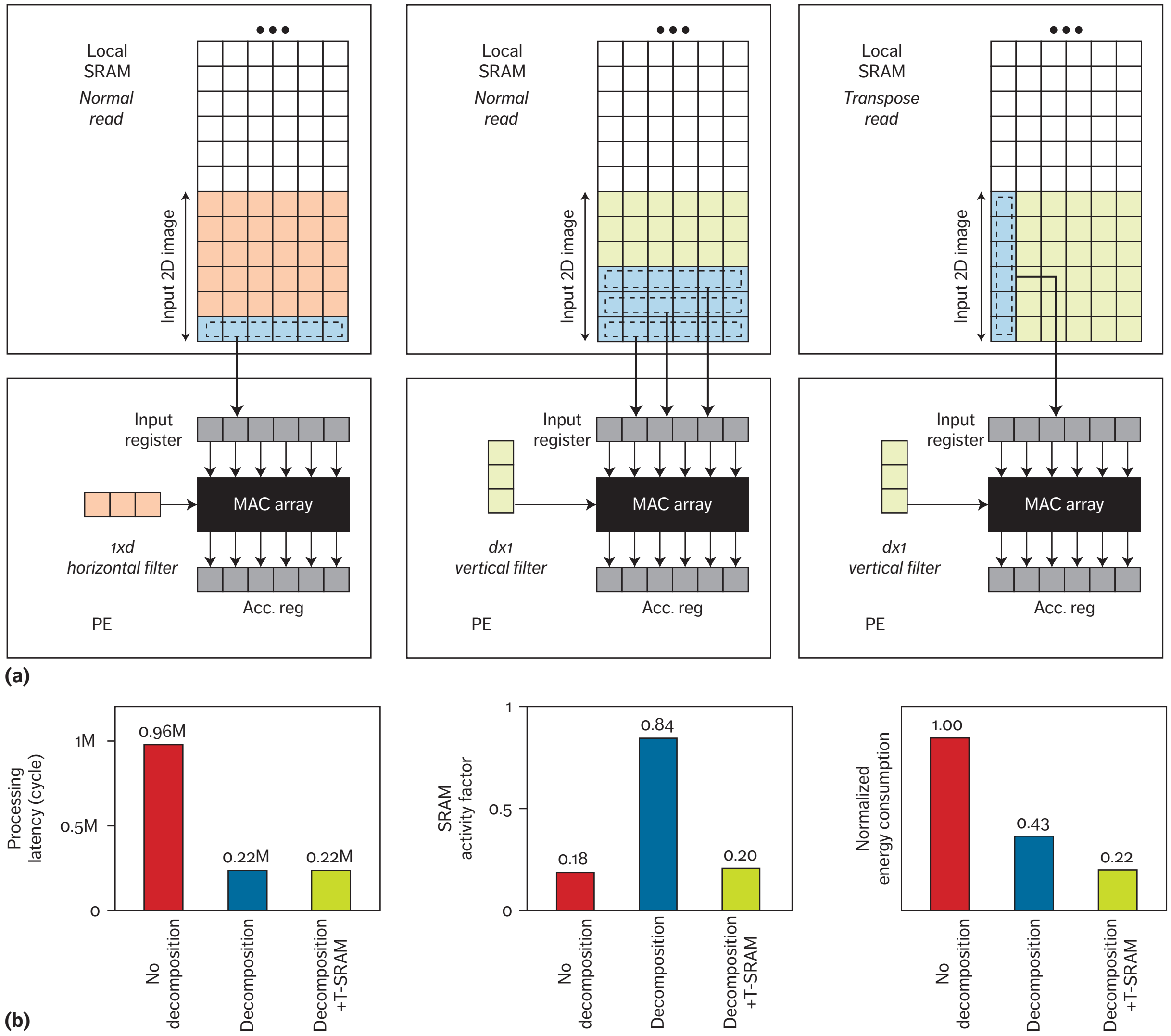
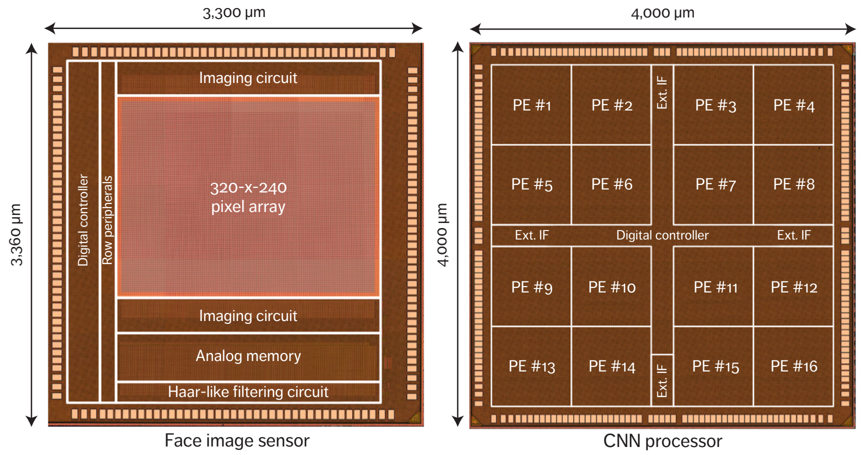


**Figure 7.** Tensor decomposition in the CNNP with the T-SRAM. (a) Convolution of horizontal and vertical filters with normal read and transpose read modes. (b) Implementation results: processing latency, SRAM activity factor, and normalized energy consumption.

CNNP can evaluate at most 75 faces during 1 second latency. The slowest operating mode (5 MHz operating frequency at 0.46 V supply voltage) is determined by the minimum energy point, and at this condition, the CNNP consumes 0.6 mJ to evaluate one face image with the target CNN, which requires 1.26 GMAC (giga multiply-accumulate operation) for four convolutional layers and one fully connected layer. In this evaluation, the target CNN is approximated by the tensor decomposition to have 0.72 GMAC operations, and the energy efficiency is 1.2 TOPS/W, whereas the effective energy efficiency considering the original workload is 2.1 TOPS/W. The approximated CNN causes 0.2 percent accuracy degradation and achieves 97.4 percent accuracy at the LFW dataset.

We proposed a functional CIS integrated with an always-on face detector and a CNNP to realize always-on face recognition for user authentication of smart devices.

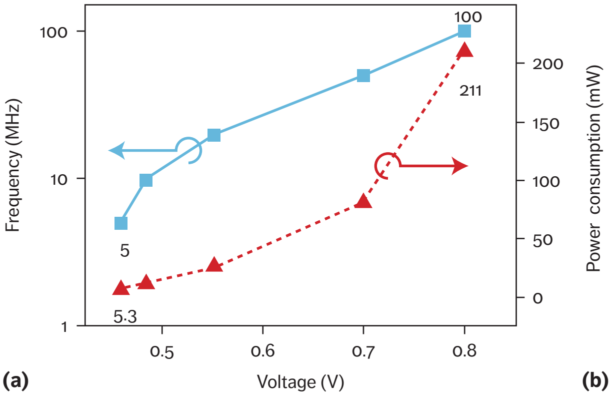
The functional CIS had two stages that reject the unnecessary workload for the following



| | |
|---|---|
| Technology | 65-nm 1P8M logic CMOS |
| Die area | 3,300 x 3,360 µm$^2$ |
| Pixel array | 320 x 240 |
| Pixel size | 7 x 7 µm$^2$ |
| Supply voltage | Analog: 2.5 V, digital: 0.5 ~ 0.8 V |
| Clock frequency | Anlalog: 50 MHz,<br>Digital: 5 ~ 100 MHz |
| Power consumption | 24 ~ 96 µW at 1fps |

| | |
|---|---|
| Technology | 65-nm 1P8M logic CMOS |
| Die area | 3,300 x 3,360 µm |
| Supply voltage | 0.46 ~ 0.8 V |
| Clock frequency | 5 ~ 100 MHz |
| Peak power consumption | 211 mW (0.8 V, 100 MHz) |
| | 5.3 mW (0.46 V, 5 MHz) |
| Energy efficiency | 2.11 nJ/cycle (0.8 V, 100 MHz) |
| | 1.06 nJ/cycle (0.46 V, 5 MHz) |

**Figure 8.** Chip photograph and performance summary.



(b)

| Mode | Voltage | Frequency | Max no. of faces at 1 fps |
|---|---|---|---|
| 1 | 0.80 V | 100 MHz | 75 |
| 2 | 0.70 V | 50 MHz | 37 |
| 3 | 0.55 V | 20 MHz | 15 |
| 4 | 0.48 V | 10 MHz | 7 |
| 5 | 0.46 V | 5 MHz | 3 |

**Figure 9.** CNNP: (a) voltage versus frequency; (b) dynamic voltage and frequency scaling modes.

stages: one is the AFDU, which determines the workload of the data converter and the DFDU, and the other is the functional CIS itself, which regulates the activity of the CNNP and its I/O. Although the concept of having an analog-domain rejecting stage for always-on image sensors at the front end still has many challenges on its robustness, scalability, and applicability for more advanced detection algorithms, this kind of rejecting stage will find its use in emerging always-on devices that require minimal power consumption over various input scenarios.

In addition, the CNNP opened the use of CNN for low-power applications along with the low-power techniques such as tensor decomposition with the T-SRAM and DVFS. However, deeper CNNs with more computational workload have been showing better accuracy in various applications, and future CNN processors will have better energy efficiency.

## Acknowledgments

This work was supported by an Institute for Information & Communications Technology Promotion (IITP) grant funded by the Korean government (MSIP) (no. 2016-0-00207).

## References

1. D. Evans, *The Internet of Things: How the Next Evolution of the Internet Is Changing Everything*, white paper, Cisco IBSG, Apr. 2011.
2. G. Andrews and L. Przywara, *Keeping Always-On Systems on for Low-Energy Internet-of-Things Applications*, report, Cadence Design Systems, 2015.
3. K. Bong et al., "A 0.62mW Ultra-Low-Power Convolutional-Neural-Network Face-Recognition Processor and a CIS Integrated with Always-On Haar-Like Face Detector," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2017, pp. 248–249.
4. B. Moons et al., "ENVISION: A 0.26-to-10 TOPS/W Subword-Parallel Dynamic-Voltage-Accuracy-Frequency-Scalable CNN Processor in 28nm FDSOI," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2017, pp. 246–247.
5. J. Choi et al., "Always-On CMOS Image Sensor for Mobile and Wearable Devices," *IEEE J. Solid-State Circuits*, vol. 51, no. 1, 2016, pp. 130–140.
6. D. Jeon et al., "A 23mW Face Recognition Accelerator in 40nm CMOS with Mostly-Read 5T Memory," *Proc. IEEE Symp. VLSI Circuits*, 2015, pp. C48–C49.
7. M. Jaderberg et al., "Speeding Up Convolutional Neural Networks with Low Rank Expansions," arXiv:1405.3866, 2014.
8. P. Viola et al., "Rapid Object Detection using a Boosted Cascade of Simple Features," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition*, 2001, doi:10.1109/CVPR.2001.990517.
9. G.B. Huang et al., *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*, tech. report 07-49, College of Information and Computer Sciences, Univ. Massachusetts, Amherst, Oct. 2007.
10. M. Bigas et al., "Review of CMOS Image Sensors," *Microelectronics J.*, vol. 37, no. 5, 2006, pp. 433–451.

**Kyeongryeol Bong** is a PhD student at the Korea Advanced Institute of Science and Technology (KAIST). His research interests include low-power vision SoCs, especially deep neural network accelerators and functional CMOS image sensors. Bong received an MS in electrical engineering from KAIST. He is a student member of IEEE. Contact him at krbong@kaist.ac.kr.

**Sungpill Choi** is a PhD student at the Korea Advanced Institute of Science and Technology (KAIST). His research interests include low-power and memory-efficient architecture design for mobile vision SoCs, especially object segmentation, stereoscopic, and deep learning. Choi received an MS in electrical engineering from KAIST. He is a student member of IEEE. Contact him at cisipi@kaist.ac.kr.

**Changhyeon Kim** is a PhD student at the Korea Advanced Institute of Science and Technology (KAIST). His research interests include low-power SoC design, especially parallel processors for artificial intelligence and machine learning algorithms. Kim received an MS in electrical engineering from KAIST. He is a student member of IEEE. Contact him at ch.kim@kaist.ac.kr.

**Hoi-Jun Yoo** is a full professor in the Department of Electrical Engineering at the Korea Advanced Institute of Science and Technology (KAIST). He is also the general chair of the Korean Institute of Next Generation Computing. His research interests include computer vision SoC, body area networks, and biomedical devices and circuits. Yoo received a PhD in electrical engineering from KAIST. He is coauthor of several books, including *Embedded Systems* (Wiley, 2012) and *Ultra-Low-Power Short-Range Radios* (Springer, 2015). He has received numerous awards, including the Electronic Industrial Association of Korea Award, the Hynix Development Award, the Korea Semiconductor Industry Association Award, and the Order of Service Merit from the Korean Ministry of Public Administration and Security. He is a Fellow of IEEE. Contact him at hjyoo@kaist.ac.kr.

IEEE computer society
Now there's even more to love about your membership...

myCS
HOME
CONTACT
LOGIN
myCS
Your subscriptions. Your way.
LEARN MORE
Computer
BIG DATA
Computer
PHYSIOLOGICAL COMPUTING
NO ADDITIONAL FEE

# Flying IoT: Toward Low-Power Vision in the Sky

**The authors study cognitive computer vision in a new design space called the Flying IoT. They investigate ultra-low-power computing challenges on a state-of-the-art platform involving a commercial micro UAV. To improve the drone's performance while maintaining low power, they propose a sensor-cloud architecture to process its computer vision algorithms with simple software optimizations that enable the drone to consume less power than cutting-edge embedded processors while achieving better performance.**

**Hasan Genc, Yazhou Zu**
*University of Texas at Austin*

**Ting-Wu Chin**
*Carnegie Mellon University*

**Matthew Halpern, Vijay Janapa Reddi**
*University of Texas at Austin*

The Internet of Things (IoT) is becoming a major paradigm, enabling applications in industries from smart cities[1] to health care[2] and dietary assessment.[3] IoT devices embed sensors and low-power processors into the physical world, allowing a rich set of information to be gathered by a diverse set of devices. IoT devices are becoming so common that they are expected to make up 18 billion of the 29 billion connected devices worldwide by 2021.[4] The widespread deployment of IoT devices necessitates that we understand the challenges of designing solutions for these next-generation platforms.

IoT devices are on the cusp of a revolution as they start to embrace some form of intelligence via cognitive or deep learning computing capability to provide intelligent end user services. For instance, smart swim watches seek to distinguish between swim strokes, and digital home assistants seek to understand human voices and language. Drones are an emerging form of new IoT devices, flying in the sky with full network connectivity capabilities. Intelligent drones with cognitive computing skills need the capability to automatically recognize and track objects to free users from the tedious task of controlling them, all

0272-1732/17/$33.00 © 2017 IEEE

of which must be performed within the power-constrained environment of a Li-Po battery.

In this article, we study a cognitive drone platform, a domain we refer to as the *Flying IoT*, to quantify the power and performance characteristics of cognitive applications on these emerging mobile devices. Specifically, we study an application called *Follow the Leader*, which automatically detects, tracks, and follows a moving human target. The application is centered on a machine learning task called *object detection*, which is its most computationally intensive kernel. The ability to perform basic computer vision tasks like object detection is a necessary step toward new and intelligent applications such as sports photography and package delivery. Cognitive applications are often very computationally intensive, which makes them difficult to run on embedded computers that have low-power, lightweight, small-size design requirements. For example, state-of-the-art machine-learning models for image classification, such as ResNet, require gigaflops to process a single image. Computing these models in real time requires many ALUs in a processor, increasing chip size and consequently leading to larger heatsinks and larger, heavier devices, which increases the drones' take-off weight and decreases their flight time.

The challenge of performing object detection on drones is to balance performance and power efficiency. To operate successfully, Follow the Leader must detect a person multiple times per second, or it could lose the person it is attempting to track. This real-time performance requirement is difficult to satisfy on extremely low-power CPUs, or even GPUs, even for simple shallow machine learning models. For complex multiclass deep models like convolutional neural networks (CNNs), desktop or server-level processors are required. So, extremely low-power, low-performance processors alone are not sufficient without hardware specializations, but hardware specialization introduces design complexity and nonrecurring engineering costs.

As an alternative to high-performance, low-power hardware specialization approaches, we investigate a general-purpose software paradigm to sustain low power consumption and small processor form factor. Using a range of off-the-shelf low-power to high-performance processors, we show that it is impractical to simultaneously achieve real-time performance and low power when executing a cognitive drone application. Therefore, we propose a sensor-cloud architecture to partition data collection and processing between the edge and the cloud. We characterize a drone application that runs on the i.MX6, a low-power, low-performance system, and on the TX1, a high-power, high-performance system. Both of these computational systems are typically found on existing drone platforms. Neither can provide the necessary performance and energy efficiency to make our application viable. Subsequently, we characterize the effects that incorporating the cloud has on the performance, power, and energy consumption of the drone application on these platforms. We show that by offloading complex object-detection models to the cloud, we can improve performance while minimizing edge power to negligible levels. The sensor-cloud architectures sustained low power on an energy-constrained drone platform, while delivering the real-time performance needed by continuous object detection and decision making.

Finally, we demonstrate how various common software-level optimizations, such as image downsampling and lossy compression, can trade small accuracy loss for significant performance and energy efficiency improvements.

## Experimental Setup

In this section, we provide an overview of our drone platform, the processors we evaluate, and the Follow the Leader workload under study.

### Drone Computing Platform

For a typical Flying IoT computing platform, we use the i.MX6, a low-power, low-performance ARM Cortex A-9 processor that comes with the 3DR Solo, a commercially available state-of-the-art hobbyist drone. The 3DR Solo is designed in a fashion typical to many autonomous drones, and its processor is indicative of the computational performance one might expect to find on a typical Flying IoT platform. The i.MX6 is powerful enough to enable only simple flight operations such as instructing a drone to start taking pictures once it has reached a certain height.

To evaluate higher-performance edge processors, we also attach a state-of-the-art embedded processor, the Jetson TX1, to the drone in place of the i.MX6. The TX1 is equipped with a GPU that is typically more power-hungry than a CPU. However, the performance per watt of a GPU for object-detection tasks, which we use in this study, is much higher than that of a typical CPU.[5] Thus, we can expect that the TX1 consumes less power to run our applications within our performance constraints than a CPU capable of achieving the same performance would consume. The TX1 is also equipped with a Cortex-M processor. This CPU, which runs an Ubuntu Linux system, is responsible for kernel and GPU management, whereas the GPU runs tasks such as object detection.

### Cloud Server

We envision future cloud computing systems to support drone services. The cloud server we use is equipped with a 3.6 GHz Intel i7 processor and a GeForce GTX 1080 Ti graphics card and is connected to the drone through Wi-Fi. In the future, emerging 5G networks are expected to guarantee Wi-Fi speeds[6] while becoming more energy efficient than 4G mobile data networks.[7] To run our cognitive workloads, we installed our server with CUDA 8.0, cudNN v5.1, OpenCV 3.1, and the Caffe deep learning framework.

### Power Measurements

It is difficult to measure the power consumption of the cores of the i.MX6 and the TX1. Therefore, we instead measure their full-board power by using the DroneKit software library's power module for the i.MX6 and by using INA monitors that can be accessed programmatically for the TX1.[8] Board power, in this context, refers to the power consumed by all components on the carrier boards of the i.MX6 and TX1, including CPUs, GPUs, and network communication components such as Wi-Fi modules. For this study, we do not consider the power consumption of a drone's mechanical components, such as its motors, as this study focuses primarily on the power consumed by drones for computations and network communication.

The DroneKit library function we use to measure the i.MX6's power returns the total power consumed by not just the i.MX6's carrier board but also by drone electronics such as GPS sensors and IMUs. Therefore, to remove the idle power consumed by these other components from our readings, we subtract all DroneKit power readings by the idle power consumption of the drone, and we add to it the idle power consumption of the i.MX6,[9] which lets us estimate its power consumption.

### Cognitive Workload

To quantify the performance of our computing platforms, we developed a cognitive drone application that is typical of the Flying IoT domain. We describe the application, the dataset it runs on, and its computational kernels below.

**Application.** We create a *Follow the Leader* application where our drone detects and follows a moving human target in real time. Many smart drones, in domains from security to sports photography, must be capable of following human targets, whether to record video of a quickly moving athlete or to monitor a suspicious individual in a crowd. To detect targets, our application runs object-detection algorithms on images taken from the drone's cameras. The drone then flies along the horizontal plane, centering its target in the middle of the drone's field of view. An autonomous drone application has strict real-time requirements, because it must fetch, analyze, and react to sensory data quickly enough to avoid its moving targets from exiting its field of view. In this work, we set a real-time performance goal of 10 frames per second (fps).

**Dataset.** To ensure a controlled test environment, we evaluate the drone application indoors, with the drone stationary, replacing its camera input with "positive test" images from the INRIA Person dataset (http://pascal.inrialpes.fr/data/human).[10] The drone reads images from the dataset and treats them as inputs from its own camera, attempting to fly toward the people in those images. In practice, the application's image inputs would be dynamically changing, whereas the images from the INRIA dataset are static. Regardless, the proposed evaluation model is sufficient, because the application's performance does not change based on whether or not the images it is operating on are related to the images that came before. The object-detection algorithms, for example, run at the same speed regardless.

One limitation of using a static image dataset is that it does not precisely represent the performance of a physical camera. To overcome this issue, we convert the images in the dataset to the BMP format for experiments involving the i.MX6, and we keep the original PNG format for experiments involving the TX1. This minimizes the difference between the time spent decoding images from the dataset and the time that would have been spent fetching images from physical cameras to within about 0.04 seconds.

**Cognitive algorithms.** To allow our Follow the Leader application to detect targets, we evaluate both multiclass and single-class object-detection algorithms. Multiclass object detectors can detect multiple different types of objects in a single image, whereas single-class object detectors are designed to detect only a single type of object at a time.

We pick two state-of-the-art CNN multiclass object detectors that are trained end-to-end from raw image pixels—Faster R-CNN and YOLO—and two single-class object detectors that are trained using hand-crafted features—Haar cascade classifiers and histogram of oriented gradients (HOG) detectors. For Faster R-CNN, we use the official Python implementation along with the pretrained model, ZF. For YOLO, we use the official implementation and the pretrained model trained with the COCO dataset. For the Haar and HOG single-class object detectors, we used the CUDA implementation in OpenCV 2.4.13 when GPUs were available, and the CPU implementation when they were not. Methods to improve CNN efficiency, such as pruning, are outside the scope of this article; our focus, instead, is on evaluating the performance and power of preexisting models and algorithms.

## Performance and Power Characterization of Object Detection on the Drone

We characterize the performance and power of different processors running Follow the Leader on the edge. We show that edge processors cannot achieve real-time performance for this application without consuming excessive amounts of power. Drones typically have quite constrained battery capacities, which severely limits their flight time. Excessive power consumption by processors can reduce the amount of time they are in the air even further. Software optimizations such as downsampling can improve performance and power but still fall far short of satisfying the power and performance goal.

When running on the drone, Follow the Leader's workflow can be broken into several pipeline stages: fetching images, preprocessing images, detecting objects in those images, and taking action to move the drone toward detected targets. These stages, taken together, constitute a single frame of our application, and we quantify the performance of our application by the number of frames that are executed every second.

### Object Detection on Low-Power i.MX6 Processor

The i.MX6 on the drone is a low-power, low-performance chip typical of battery-constrained Flying IoT devices. We characterize its power and performance and show that although the i.MX6 satisfies a low-power device's power budget, its performance is far too low to provide real-time cognition.

**Performance.** As Table 1 shows, when Follow the Leader runs locally on the i.MX6, it fails to achieve anything close to our real-time goal of 10 fps, owing to the time taken to perform object detection. The application runs at less than 0.1 fps with single-class object detectors due to the i.MX6's simple in-order microarchitecture and its low 1 GHz frequency. Object detection, a key task, is the overwhelming bottleneck, suggesting that machine learning algorithms are the performance limiters in cognitive Flying IoT applications.

**Algorithm optimizations.** Reducing image resolution via downsampling enhances performance without noticeably affecting the accuracy of our object detectors.[11] By approximating the original images with smaller-sized images, downsampling can simplify the computation task of the algorithm and shorten computing time. As Figure 1a shows, downsampling by some scaling factor greatly improves Follow the Leader's performance on the i.MX6 because it reduces the time spent on object detection.

Table 1. Performance breakdown when running Follow the Leader on a low-power computer (i.MX6).

| Algorithm | Fetch (%) | Preprocess (%) | Analyze (%) | Act (%) | Overall (s) | Frames per second |
|---|---|---|---|---|---|---|
| Haar | 0.50 | 0.18 | 99.30 | 0.01 | 18.24 | 0.05 |
| HOG | 0.76 | 0.28 | 98.93 | 0.02 | 11.17 | 0.09 |

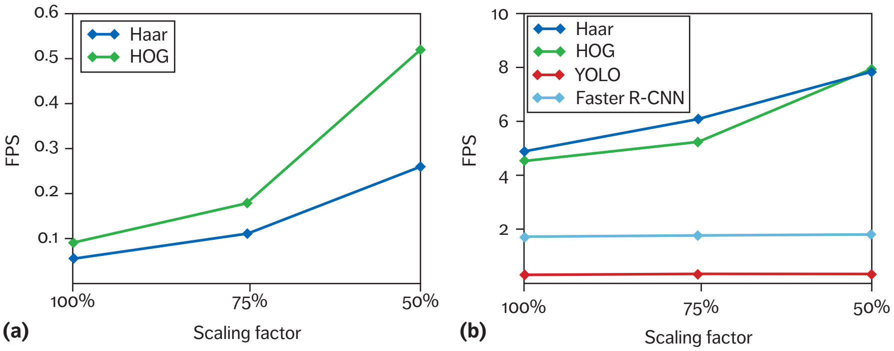


**Figure 1.** Performance variation when downsampling images. (a) Performance versus downsampling for i.MX6. (b) Performance when reducing resolution for TX1. A scaling factor of 100 percent means retaining the original image, whereas scaling 50 percent means scaling the image down to half of its original size.

However, it still fails to reach even one fps, let alone anything close to our real-time target.

**Power and energy.** The i.MX6 consumes little instantaneous power, but because it takes so long to process frames in the application, the total energy that it spends per frame is quite large (see Figure 2a). The i.MX6 consumes between 0.9 and 1.7 W of instantaneous power when running Follow the Leader with Haar and HOG object detection. This low instantaneous power consumption is almost negligible, which would have made the i.MX6 the perfect candidate for cognitive drone platforms, but its low computing performance (see Figure 1a) decreases the energy efficiency so much that a full 25 J is required to process a single image in some circumstances.

Fortunately, downsampling images can improve the energy efficiency of Follow the Leader on the i.MX6 by four to six times (see Figure 2a), although computing performance is still too low to allow real-time operation.

## Object Detection on High-Performance TX1 GPU

Ultra-low-power CPU architectures like the i.MX6 do not satisfy the performance needs of real-time object detection. Therefore, we explore a higher performance embedded system on chip (SoC) as an alternative and analyze the power and performance tradeoffs. We run the object-detection algorithms on the Nvidia Jetson TX1 because its GPU provides higher performance for these applications and is more power efficient than a CPU. At the time the research was conducted, the Nvidia Jetson TX2 platform was yet to be released.

**Performance.** As Table 2 demonstrates, our application performs much better on the TX1, because its GPU can execute the object-detection phase much faster than the i.MX6's CPU can. However, CNN-based detectors such as YOLO and Faster R-CNN still occupy over 90 percent of the total application time, and the TX1 still fails to achieve the 10 fps real-time performance.

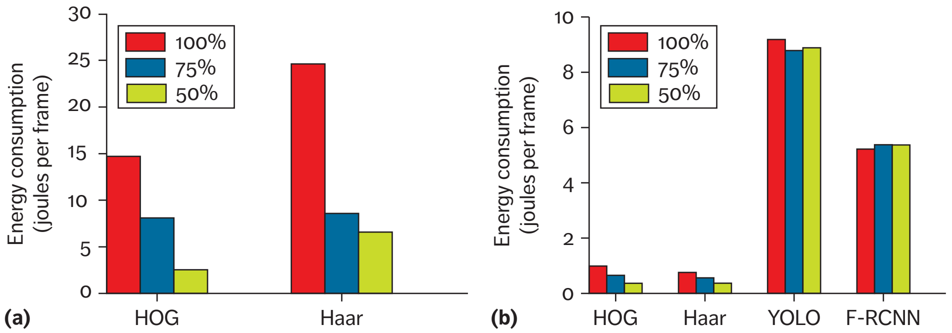


**Figure 2.** Energy consumption per frame: (a) i.MX6 and (b) TX1. The scaling factors are shown in the figure, where 100 percent means the original figure size is not scaled down. Note that the *y*-axis of (a) is scaled to three times the *y*-axis of (b).

**Table 2. Performance breakdown when running Follow the Leader on a high-performance computer (TX1).**

| Algorithm | Fetch (%) | Preprocess (%) | Analyze (%) | Act (%) | Overall (s) | Frames per second |
|---|---|---|---|---|---|---|
| Haar | 24.65 | 0.07 | 74.02 | 1.25 | 0.21 | 4.79 |
| HOG | 22.35 | 0.07 | 76.50 | 1.09 | 0.23 | 4.41 |
| YOLO | 1.76 | 0.01 | 98.20 | 0.02 | 3.25 | 0.31 |
| F-RCNN | 8.45 | 0.07 | 91.28 | 0.21 | 0.67 | 1.49 |

As earlier, we attempt downsampling to increase our application's frame rate. As Figure 1b shows, Haar and HOG's performance increases to around 10 fps, which is on par with the 10-fps target we set for real-time decision making.

However, downsampling does not enhance the performance of our CNN models, because the models evaluate input images of a fixed size, which is set when the models are being trained.

**Power and energy.** The TX1's higher performance comes at the cost of higher instantaneous power consumption compared to the i.MX6, but its energy efficiency is far better than that of the i.MX6. Figure 2b shows the TX1's energy consumption per frame with different algorithms and downsampling ratios. Our application's power consumption on the TX1 is roughly 3 to 4.5 W when using Haar and HOG object detectors, about two to three times the power consumption of the i.MX6. When running YOLO, the TX1 again consumes approximately 3 W, but when running Faster R-CNN, the TX1's instantaneous power consumption shoots up to approximately 9 W.

Regardless of how much more instantaneous power the TX1 consumes compared to the i.MX6, it is still much more energy efficient, consuming 6 to 15 times less energy to process each frame of the application when running HOG and 15 to 32 times less energy when running Haar. Even then, however, the TX1's larger form factor, increased take-off weight, and greater idle power consumption might cause significant power drains for smaller, more battery-constrained drones such as nano aerial vehicles; such drones require a computing platform with a small size and low idle power consumption for practical operation.

Table 3. Performance breakdown on the sensor-cloud system using unscaled images and the PNG format on a low-power computer (i.MX6).

| Algorithm | Fetch (%) | Preprocess (%) | Compress (%) | Transmit (%) | Decompress (%) | Analyze (%) | Act (%) | Total time (s) | Frames per second |
|---|---|---|---|---|---|---|---|---|---|
| Haar | 5.15 | 15.78 | 53.03 | 16.84 | 1.85 | 7.17 | 0.18 | 0.52 | 1.94 |
| HOG | 5.75 | 15.96 | 54.71 | 18.06 | 1.80 | 3.53 | 0.18 | 0.50 | 1.98 |
| YOLO | 4.56 | 14.15 | 50.18 | 16.70 | 1.33 | 12.90 | 0.18 | 0.56 | 1.78 |
| Faster R-CNN | 5.16 | 13.40 | 48.44 | 15.04 | 1.45 | 16.35 | 0.16 | 0.59 | 1.70 |

Downsampling reduces HOG and Haar power and energy consumption because a reduction in the size of the image inputs reduces the number of operations to compute, but for the CNN-based method, downsampling does not seem to have any effect. With software downsampling, GPU-equipped SoCs such as the TX1 can approach real-time performance for single-class object detection, but they still fail to approach real-time performance for the multiclass detectors.

## Sensor-Cloud Architecture for Low-Power Real-Time Object Detection

We propose a sensor-cloud system to bring server-level computational capability to low-power IoT devices such as drones. In a sensor-cloud system, computationally intensive tasks are offloaded to the cloud while data collection tasks are done at the edge. We modify our Follow the Leader application so that it offloads object detection, our bottleneck stage, to the cloud, enhancing our application's performance while maintaining a low power consumption. In fact, with software optimizations such as compression, even low-performance CPUs like the i.MX6 can achieve the performance of the TX1 on the edge.

The sensor-cloud application's workflow is represented by the following pipeline stages: the drone fetches images, preprocesses them, compresses them, and transmits them to the cloud. The cloud then decompresses the images, analyzes them using object-detection algorithms, and sends the results back to the drone. Finally, the drone takes action based on the response from the cloud.

Real-time applications such as Follow the Leader cannot tolerate long communication delays between drones and the cloud. Thus, our performance characterization takes into account the time delays associated with network communication in the application. One assumption we make is that there is no failure in network connectivity.

### Performance Measurement

As Table 3 shows, the i.MX6 shows significant speed improvements when running Follow the Leader on a sensor-cloud architecture. The single class detectors ran 22 to 39 times faster than they did in the i.MX6's non-cloud implementation, and as an added improvement, the sensor-cloud system was able to use the multiclass detectors that the i.MX6 could not implement on its own.

The TX1 experiences a relative performance improvement of nine and two times over its non-cloud implementation when running YOLO and Faster R-CNN, respectively. However, it experiences performance deterioration when using single-class detectors. This is because the time saved by the cloud's increased processing capability is partly nullified by the time taken to transmit data to and from the edge.

We also find that for the sensor-cloud system, the performance bottlenecks are image compression and transmission, rather than object detection. This is because the GPU in the cloud is powerful enough to run object-detection

Table 4. Performance breakdown on the sensor-cloud system using unscaled images and the PNG format on a state-of-the-art computer (TX1).

| Algorithm | Fetch (%) | Preprocess (%) | Compress (%) | Transmit (%) | Decompress (%) | Analyze (%) | Act (%) | Total time (s) | Frames per second |
|---|---|---|---|---|---|---|---|---|---|
| Haar | 25.61 | 0.17 | 16.83 | 37.03 | 12.95 | 6.94 | 0.39 | 0.26 | 3.81 |
| HOG | 28.11 | 0.18 | 16.62 | 39.36 | 12.47 | 2.88 | 0.39 | 0.27 | 3.72 |
| YOLO | 17.79 | 0.12 | 12.22 | 36.63 | 8.99 | 23.92 | 0.32 | 0.36 | 2.77 |
| Faster R-CNN | 21.61 | 0.14 | 14.57 | 39.96 | 10.81 | 12.53 | 0.38 | 0.30 | 3.30 |

algorithms quickly. We find that compression is more important for the i.MX6 because its CPU spends about half its time in this phase. On the TX1, compression runs faster because its CPU core has higher single-thread performance.

Because compression and transmission are the bottlenecks in the sensor-cloud system, we optimize our application by exploring faster, more compact, lossy compression formats to reduce both compression and network transmission time.

For our initial experiments, in Tables 3 and 4, we compress our images into the lossless PNG image format. A lossy compression algorithm, on the other hand, could improve our application's performance, because lossy algorithms are typically faster and create smaller data packages to transmit. We investigate the JPEG format, which is an extremely popular lossy format for images. JPEG images can be saved in any "quality" from 100 to 0 percent. As quality decreases, compression ratios improve, but less information is preserved. Lowering JPEG quality from 100 to 60 percent does not noticeably reduce the accuracy of our four object-detection algorithms.[11]

As Figure 3 shows, we compare the speed of our Follow the Leader application under various conditions: using the lossless PNG format; using the lossy JPEG format at 100, 80, and 60 percent quality; and using the uncompressed BMP format to send data. We find that the PNG format, the BMP format, and the JPEG format at 100 percent quality have the lowest performance, because they incur high transmission penalties. The JPEG format at 80 and 60 percent quality, on the other hand, is faster.

Software optimizations such as image downsampling and lossy compression vastly improve the performance of Follow the Leader, allowing it to reach fps rates that were impossible in the edge-only implementation. We maximize performance by downsampling our images by 50 percent and then compressing them using the JPEG format at 60 percent quality. At the highest speeds, we are able to surpass our 10 fps goal for single-class detectors (Haar and HOG) on the TX1 and to approach very close to the real-time goal with single-class detectors on the i.MX6.

Both the i.MX6 and the TX1 achieve similar speeds with multiclass detectors (YOLO and Faster R-CNN) at the highest optimization levels, because as our optimizations become more aggressive, the bottleneck for our application when using multiclass detectors increasingly becomes the speed at which the cloud server itself can process images. However, although both the i.MX6 and the TX1 fail to reach real-time performance (that is, 10 fps) with multiclass detectors, the TX1 achieves performance improvements of 4 to 15 times over its non-cloud implementation.

## Power and Energy

In addition to improving performance, switching to a sensor-cloud system can potentially reduce power and energy consumption at the edge (see Figure 4). On the i.MX6, the instantaneous power consumption falls only slightly, but the performance improvement brought

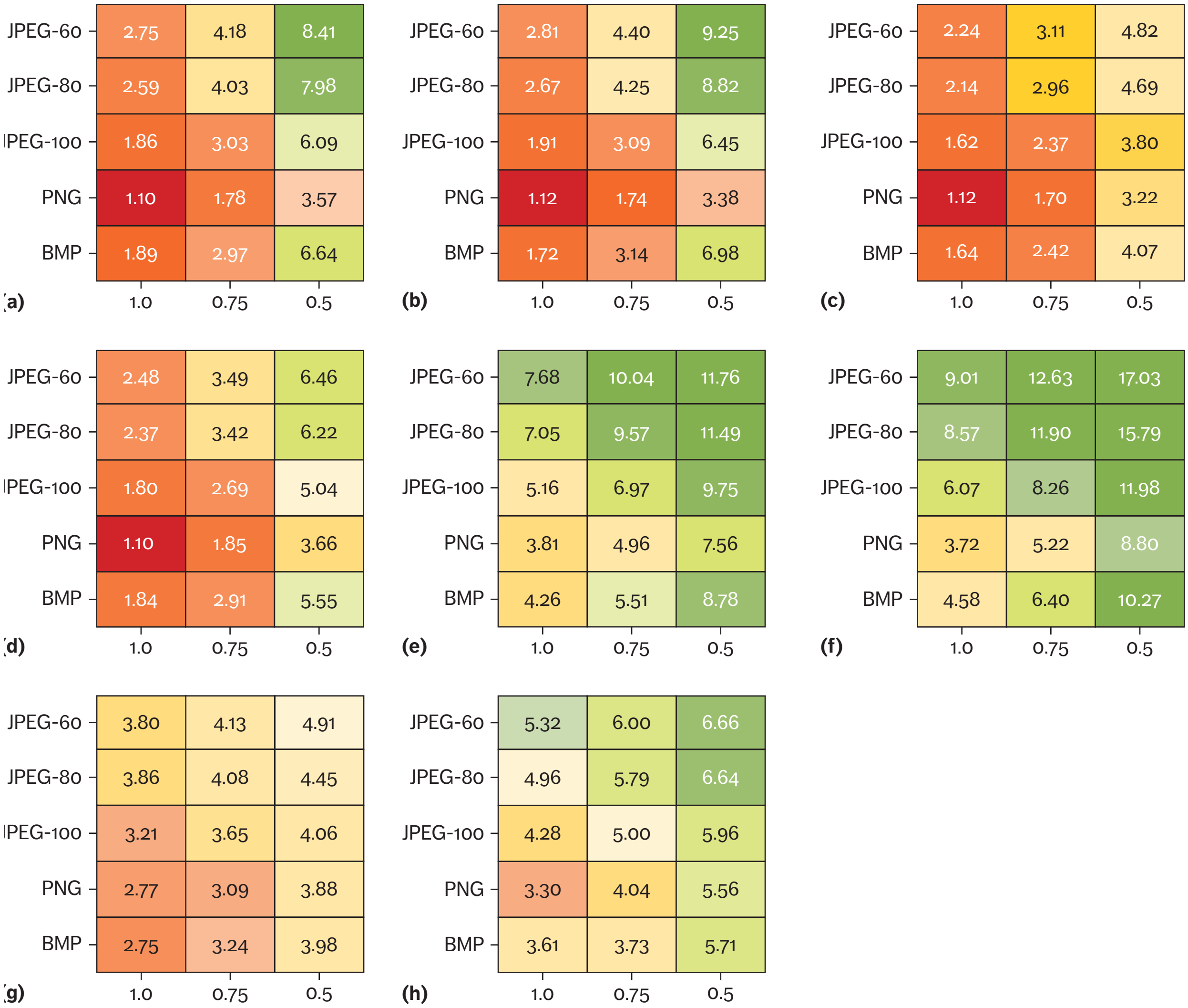


**Figure 3.** Frame rate versus compression algorithm and resolution. The numbers in the boxes represent frames per second (fps). Sensor-cloud performance increases significantly with JPEG compression and downsampling. (a) Haar (i.MX6); (b) HOG (i.MX6); (c) YOLO (i.MX6); (d) Faster R-CNN (i.MX6); (e) Haar (TX1); (f) HOG (TX1); (g) YOLO (TX1); (h) Faster R-CNN (TX1).

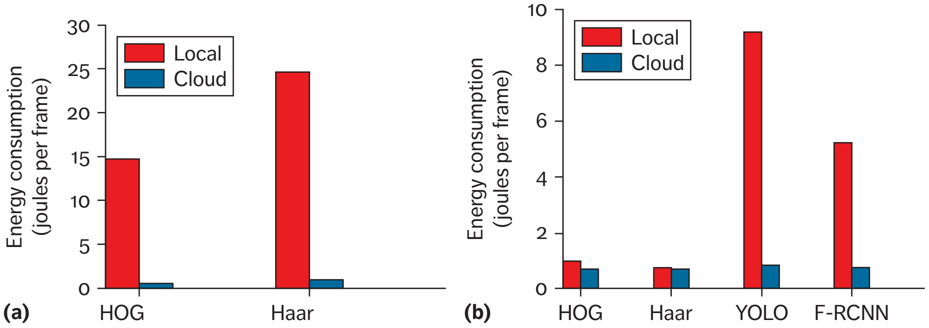


**Figure 4.** Sensor-cloud architecture reduces the energy consumed per frame for the Follow the Leader application: (a) i.MX6 and (b) TX1. Images are unscaled and compressed using PNG. Note that the *y*-axis of (a) is scaled to three times the *y*-axis of (b).

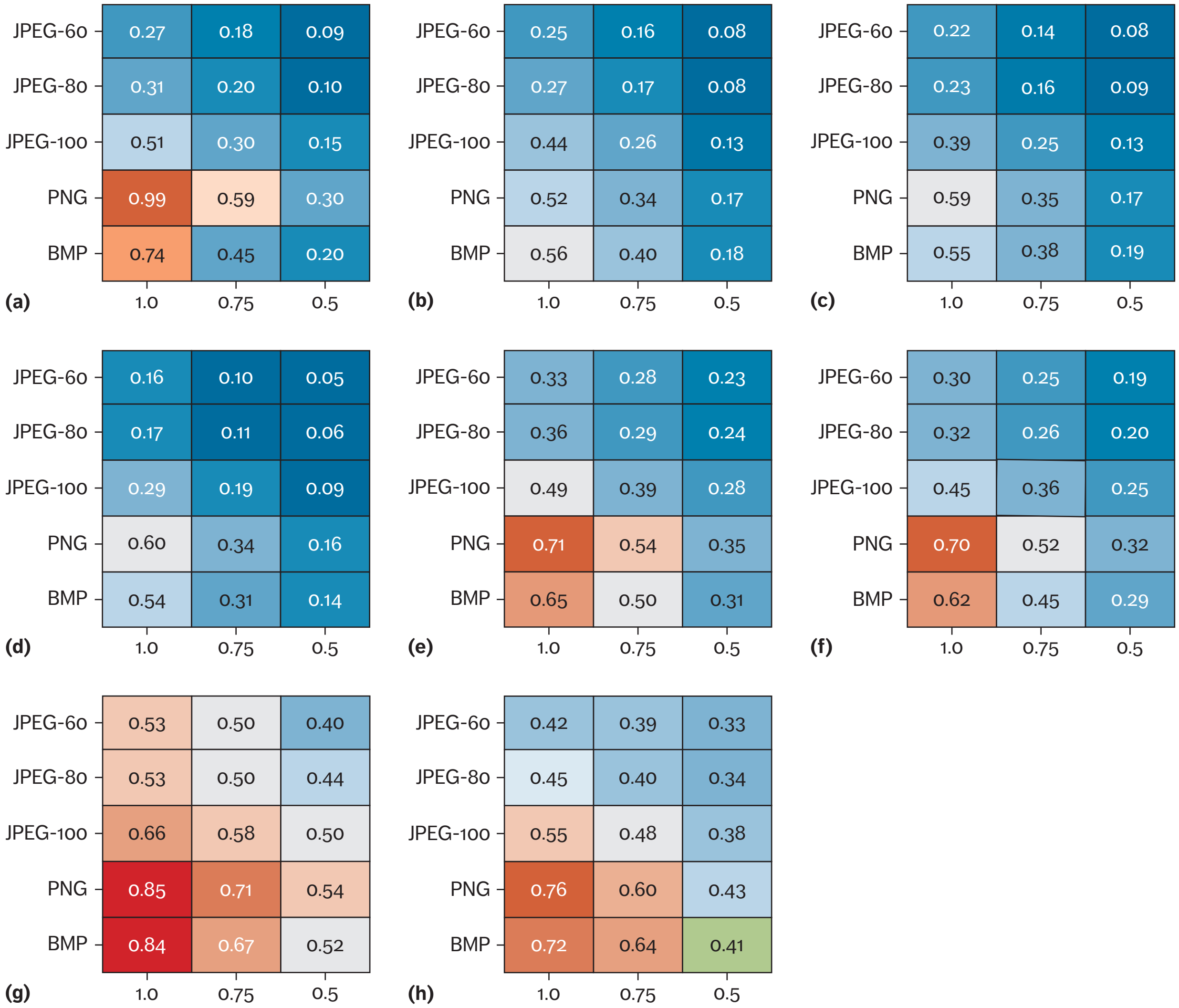


**Figure 5.** Energy consumption per frame (in joules per frame) versus compression algorithm and scaling factor. The numbers in the boxes represent the energy consumed per frame of the application. Compression and downsampling improve the system's energy efficiency. (a) Haar (i.MX6); (b) HOG (i.MX6); (c) YOLO (i.MX6); (d) Faster R-CNN (i.MX6); (e) Haar (TX1); (f) HOG (TX1); (g) YOLO (TX1); (h) Faster R-CNN (TX1).

by the cloud causes our energy consumption per frame to plummet to below 1 J per frame. The TX1's instantaneous power consumption falls by 0.5 to 2 W for the Haar, HOG, and YOLO detectors, and by a full 6.4 W for Faster R-CNN, because the application no longer uses the GPU. These instantaneous power savings help reduce the energy consumed per frame for all detectors, but especially for YOLO and Faster R-CNN, which consume 6 to 11 times less energy to process frames. Typically, in a sensor-cloud system, GPUs would not be installed on the edge. We can see from these results that utilizing GPUs in the cloud instead of on the edge can yield significant power and energy savings.

Our previous software optimizations do not significantly affect the instantaneous power consumption of our computing platforms. However, the optimizations allow the application to process many more frames per second while consuming approximately the same amount of power. Thus, in Figure 5, we can see that both the i.MX6 (Figures 5a through 5d) and the TX1 (Figures 5e through 5h) save significant amounts of energy on every frame they process. The i.MX6, in particular, consumes less than 0.1 J per frame under our most aggressive optimizations, which, as Figure 3 shows, are sufficient to bring our application to near-real-time performance with single-class detectors.

With appropriate compression and downsampling optimizations, sensor-cloud architectures can reduce edge power significantly compared to placing all the computation on the edge. Meanwhile, sensor-cloud architectures enhance application performance to almost the level of the real-time target by offloading computationally intensive software kernels to the cloud.

The Internet of Things is entering a new paradigm where devices on the edge need both cognitive capability and the ability to interact directly with their environments in real time. Although our work demonstrates that sensor-cloud architectures can accelerate Flying IoT applications to near-real-time performance, it also exposes some of the challenges associated with them.

The performance of sensor-cloud applications is significantly limited by the speed at which they can compress and transmit data. Currently, compression on drone processors is typically done by CPUs. However, by developing specialized compression accelerators, researchers can alleviate the pressure put on CPUs, dramatically improving performance.

Furthermore, it will be important to investigate the implications of network stability on drone applications that utilize the cloud. Our study uses a Wi-Fi network to connect the drone to the cloud, but it will be worthwhile to look into the impact of 4G and future 5G networks on the speed and energy efficiency of drone applications.

## References

1. W.F. Domoney et al., "Smart City Solutions to Water Management using Self-Powered, Low-Cost, Water Sensors and Apache Spark Data Aggregation," *Proc. 3rd Int'l Renewable and Sustainable Energy Conf.*, 2015, pp. 1–4.
2. G. Demiris et al., "Senior Residents Perceived Need of and Preferences for Smart Home Sensor Technologies," *Int'l J. Technology Assessment in Health Care*, vol. 24, no. 1, 2008, pp. 120–124.
3. H.-C. Chen et al., "Saliency-Aware Food Image Segmentation for Personal Dietary Assessment using a Wearable Computer," *Measurement Science and Technology*, vol. 26, no. 2, 2015, p. 025702.
4. *Ericsson Mobility Report*, Ericsson, Nov. 2016.
5. V. Campmany et al., "GPU-Based Pedestrian Detection for Autonomous Driving," *Procedia Computer Science*, vol. 80, 2016, pp. 2377–2381.
6. J.G. Andrews et al., "What Will 5G Be?" *IEEE J. Selected Areas in Comm.*, vol. 32, no. 6, 2014, pp. 1065–1082.
7. A. Abrol and R.K. Jha, "Power Optimization in 5G Networks: A Step Towards Green Communication," *IEEE Access*, vol. 4, 2016, pp. 1355–1374.
8. "Jetson/TX1 Power Monitor," *eLinux*, Oct. 2016; www.elinux.org/Jetson/TX1_Power_Monitor.
9. *i.MX 6Solo Power Consumption Measurement*, report no. AN4715, Freescale Semiconductor, June 2013.
10. N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition*, vol. 1, 2005, pp. 886–893.
11. H. Genc et al., "Optimizing Sensor-Cloud Architectures for Real-Time Autonomous Drone Operation," *Sensors to Cloud Architectures Workshop*, 2017; http://3nity.io/,vj/downloads/publications/hasan17drones.pdf.

**Hasan Genc** is an undergraduate student in the Department of Electrical and Computer Engineering at the University of Texas at Austin. His research interests include computer architecture, embedded systems, and mobile computing. Contact him at hngenc@utexas.edu.

**Yazhou Zu** is a PhD candidate in the Department of Electrical and Computer Engineering at the University of Texas at Austin. His research interests include resilient and energy-efficient processor design and management. Zu received a BS in microelectronics from Shanghai Jiao

Tong University of China. Contact him at yazhou.zu@utexas.edu.

**Ting-Wu Chin** is a PhD student in the Department of Electrical and Computer Engineering at Carnegie Mellon University. His research interests include machine learning, optimization, approximate computing, and energy-aware computing. Chin received his MS in computer science from National Chiao Tung University in Taiwan. Contact him at tingwuc@andrew.cmu.edu.

**Matthew Halpern** is a PhD student in the Department of Electrical and Computer Engineering at the University of Texas at Austin. His research interests include runtime systems and computer architecture for mobile computing. Halpern has a BS in electrical and computer engineering from the University of Texas at Austin. Contact him at matthalp@utexas.edu.

**Vijay Janapa Reddi** is an associate professor in the Department of Electrical and Computer Engineering at the University of Texas at Austin. His research interests span the definition of computer architecture, including software design and optimization, to enhance mobile quality of experience and improve the energy efficiency of high-performance computing systems. Janapa Reddi received a PhD in computer science from Harvard University. Contact him at vj@ece.utexas.edu.

# Visual IoT: Ultra-Low-Power Processing Architectures and Implications

**This article describes three key implications in ultra-low-power visual edge processing: the constrained data footprint, limited power-efficient computation, and difficulties processing large-scale data. The authors review three case studies—small-scale visual recognition for digits and characters, medium-scale visual recognition for hand gestures, and large-scale visual processing requiring video summarization—to show that co-designing algorithms and architectures for ultra-low-power processing in edge devices helps address the key challenges.**

**Vui Seng Chua, Julio Zamora Esquivel, Anindya S. Paul, Thawee Techathamnukool, Carlos Flores Fajardo, Nilesh Jain, Omesh Tickoo, Ravi Iyer**
*Intel*

Cameras are widely used in multiple applications ranging from security (surveillance and monitoring), entertainment (recording of public and personal events such as sports and music), and, more recently, interactive environments (augmented and merged reality) and robotics and drones (navigation, delivery, interaction, and assistance). In the visual Internet of Things (IoT),[1] critical challenges must be addressed because of the large bandwidth needs of visual data and the tradeoff between computing and communication. One approach is to move the visual data from an ultra-low-power edge device to a higher-performance platform (gateway or cloud) for computation. Another approach is to accomplish a lot of the processing at the edge device in order to conserve visual communication bandwidth. There are also hybrid approaches that partition overall work intelligently between the edge and the cloud to optimize overall efficiency. In this article, we focus on the key implications of performing visual processing on ultra-low-power edge devices and examine how to achieve the best efficiency for realistic end-to-end usages.

Visual processing at the edge device has to work within the ultra-low-power constraints, which typically manifest themselves in three types of challenges: the amount of static RAM (SRAM) available on-die is severely constrained due to leakage and dynamic power,

0272-1732/17/$33.00 © 2017 IEEE

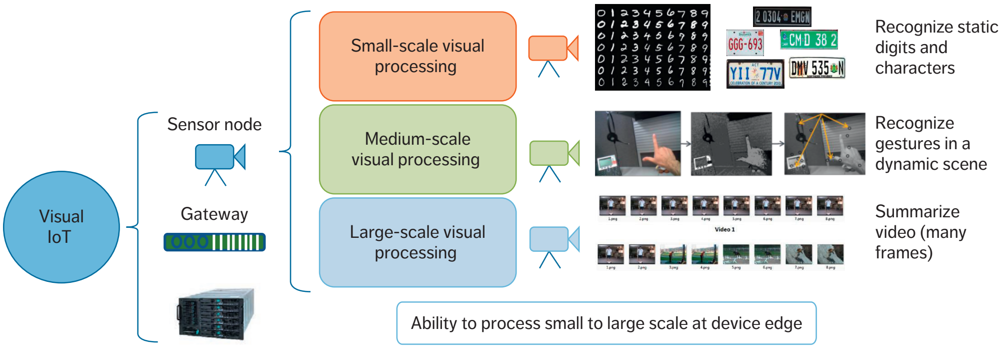

Visual IoT
Sensor node
Gateway
Small-scale visual processing
Medium-scale visual processing
Large-scale visual processing
Recognize static digits and characters
Recognize gestures in a dynamic scene
Summarize video (many frames)
Ability to process small to large scale at device edge

**Figure 1.** Visual Internet of Things (IoT) usage scenarios: from small-scale to large-scale visual processing at the edge device.

the computational processing is constrained because it is typically on an ultra-low-power microcontroller class core, and the availability of significant storage on the edge device is also constrained due to form factor and power limitations. To understand and address such processing challenges for visual edge devices, we examine three levels of visual processing: small-scale visual processing, in which the edge device must recognize static digits and characters, but needs to accomplish this task within a data footprint constraint; medium-scale visual processing, in which the edge device must recognize static hand gestures from a human within a computational power constraint; and large-scale visual processing, in which the edge device must assist in video summarization, which involves finding salient frames in a video stream and ensuring that the salient frames are retained to provide sufficient coverage and representation of the incoming visual stream over time. Our key observation from these case studies is that co-designing software and hardware approaches can yield significant efficiency improvements, which are required for ultra-low-power edge processing.

## Visual IoT Overview

Figure 1 shows an end-to-end visual IoT platform comprising edge devices, communication gateways, and visual cloud servers. We focus on the edge device from an ultra-low-power processing perspective and examine three scales of processing below.

### Small-Scale Visual Processing

Typically, in a visual IoT usage, cameras are positioned to detect or recognize specific objects or attributes (for example, license plate recognition or reading highway signs). Current image-recognition pipelines are gateway or cloud based and thus the classification pipeline does not consider limitations on device memory and computation. The goal of these solutions is to build highly accurate models that classify images with a deep layer neural network (NN) architecture.[2,3] However, running the same pipeline on a small form factor edge device requires several optimizations, from input image scaling to new transforms, feature size compression, effective memory reuse, and optimum model size for classification. We will walk through the entire flow for digit and character recognition and show how NN implementations need to be optimized to fit within an edge device footprint.

### Medium-Scale Visual Processing

In medium-scale visual processing, we consider the problem of achieving a robust and light algorithm implementation for edge devices to naturally interact with users using gesture (hand-pose) recognition. We will describe the entire processing flow from segmentation to feature extraction to classification of hand poses. We show that such processing is challenging because it needs to be accomplished on a power-constrained edge device with limited computational and memory capabilities. We show that achieving hand-pose recognition within such a device requires acceleration, and we outline a novel NN accelerator that accomplishes the end goal of power efficiency.

### Large-Scale Visual Processing

Effective consumption of large video data leads to the need for efficient video summarization applications that can filter out salient parts of a video. The video summarization pipeline generally comprises feature extraction, a similarity measure, and key frame selection (see, for example, previous work by Shayok Chakraborty and colleagues[4]). To perform video summarization on mobile and wearable environments, the video summarization pipeline's computational complexity is a bottleneck for real-time implementation. We focus our attention on speeding up a key bottleneck of the summarization, namely histogram of oriented gradients (HOG) processing. We outline a HOG acceleration approach that can make such processing power efficient on edge devices for online video summarization.

## Visual IoT: Small-Scale Processing

In this section, we describe an image-recognition workload that we developed using two-layer, fully connected feed-forward NN for the digit and character recognition of English alphabets. This image recognition had to be accomplished on a power-constrained edge device, and we used the Arduino 101 as a reference platform for this work. To address the data footprint constraints, we introduced the following major optimizations:

- Define new image transforms to eliminate the need for convolution.
- Resample the input image to the smallest size possible (that is, 20 pixels × 20 pixels) and still get satisfactory classification performance.
- Identify a feature extraction method that can operate on images of this small size and reduce the number of features per image without compromising too much accuracy.
- Optimize the number of neurons in various layers. Develop a hierarchical ensemble design for classification. In other words, break the entire classification into several smaller tasks, design one weak classifier for each task, and, finally, combine outputs to generate a consolidated final result.

Through these optimizations, our implementation managed to recognize VGA quality images (feature extraction and classification) at 8 frames per second (fps) on Arduino 101 with SRAM of less than 20 Kbytes.

### Recognition Pipeline and System Components

Figure 2a summarizes the major building blocks of the image-recognition pipeline running on Arduino 101. We put together a low-resolution image-recognition system using an Arduino 101 board and VGA Arducam camera shield. The board contains an Intel Curie module that has two tiny cores, an x86 (Quark) and a 32-bit ARC architecture core, both clocked at 32 MHz. The Curie module has 80 Kbytes of SRAM, but the Arduino Sketch exposes only 24 Kbytes of SRAM available for this workload. We evaluated the performance using the MNIST and MSFT datasets.[5]

### Image Transforms

The captured image first undergoes downsampling to generate a 20-×20-pixel image. We employed linear interpolation for simplicity and observed no major performance degradation. The RGB image is converted to grayscale and scaled to keep the value range uniform. We avoided any image filtering or transforms that involve convolution operations.

### Feature Engineering

We carried out feature engineering to meet the limited system resource requirements while maintaining good accuracy. Taking motivation from the HOG developed by Navneet Dalal and Bill Triggs,[6] we used this feature extractor because it focuses on the shape of the characters, maintains size and rotational invariance, and can control computational complexity and the feature vector dimension size for recognition tasks. The feature extraction method involves evaluating weight-normalized local histograms on the gradient angle of the image pixels over a grid of overlapping sliding windows. The algorithm computes intensity gradients from raw pixels along both the $x$ and $y$ dimensions of each image and derives 2D gradient magnitude and orientation matrices. The entire image is then divided into small spatial regions (the size of each region controls the granularity or resolution of the features) within which the local histograms are formed over gradient orientations. Instead of simple counting over each angular bin, a weighted bilinear transform has been performed to develop the

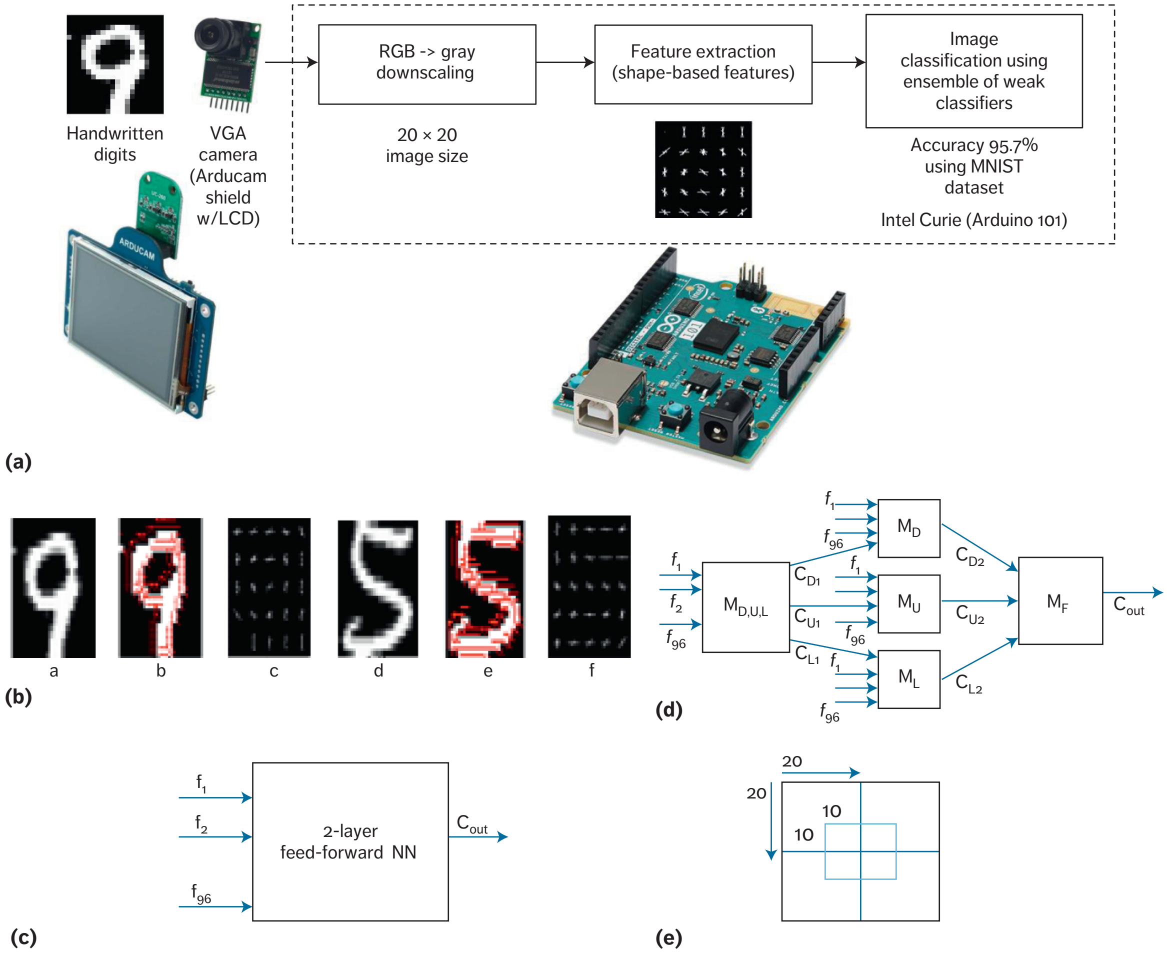


**Figure 2.** Small-scale image-recognition pipeline and edge-optimized image classification models. (a) Major building blocks and pipeline. (b) Feature extraction process for digits 9 and 5. Ensemble model design for (c) MD1, (d) MD2, and (e) MD3.

histogram, wherein the weight is governed by the function of gradient magnitudes. Finally, the histogram values are normalized using the Euclidian norm to make the features scale invariant. After experimentations, we decided to include the size of spatial regions of six pixels along *x* and *y* with 50 percent overlap and divided the whole 180-degree angular region to six bins, each covering 30 degrees of angular space. The dimension of the feature vector obtained from the entire image using these parameters is 96. Figure 2b demonstrates the features extraction process for two examples.

## Model Design and Optimization

We investigated three model designs to solve the problem of digit and character recognition on Arduino 101.

**Model design 1 (MD1).** We started by designing a single two-layer fully connected feed-forward NN model to classify all digits and letters in one attempt (see Figure 2c). The NN has 96 input nodes (feature dimension per image) and 62 output nodes (10 digits + 26 uppercase letters + 26 lowercase letters). We constrained the number of hidden layers to one to keep it within the computational envelope of an ultra-low-power edge device. In a typical fully connected NN design, model parameters involve weights and biases between the input-to-hidden and hidden-to-output layers. We used a five-fold cross-validation method to determine the optimum number of hidden nodes that provides the average best accuracy. The number of hidden nodes we found suitable for this case is 250; then we started seeing an

Table 1. Benchmark results on accuracy, latency, and memory.

| Performance measures | MD1 | MD2 | MD3 | Comp* |
|---|---|---|---|---|
| Digits (accuracy) | 97.9% | 95.1% | 95.0% | 99.0% |
| Uppercase (accuracy) | 80.0% | 78.6% | 78.2% | 82.0% |
| Lowercase (accuracy) | 73.0% | 70.2% | 70.1% | 75.0% |
| Capture/transforms (processing time) | 2,765 ms | 2,774 ms | 2,768 ms | N/A |
| Feature extraction (processing time) | 118 ms | 118 ms | 118 ms | N/A |
| Classification (processing time) | 3.06 ms | 3.9 ms | 7.8 ms | N/A |
| Model size | 150 Kbytes | 18 Kbytes | 6.4 Kbytes | N/A |
| Total pipeline size | 160 Kbytes | 23.8 Kbytes | 12.3 Kbytes | N/A |

** Best-known published numbers.*

overfitting effect on the validation set. Additionally, we also used regularization to limit the growth of NN weights. This model performs satisfactorily, but the recognition pipeline needed more than 150 Kbytes of RAM, which we do not have on Arduino 101.

**Model design 2 (MD2).** This design was our first step toward an ensemble/hierarchical architecture. We used weak classifiers operating at various stages and aggregated their results to produce the final classification decision. Figure 2d demonstrates the design. Each feature vector ($f_1 \ldots f_{96}$) from an input image is passed to the first model ($M_{\mathrm{D,U,L}}$), which classifies it among three classes: digit (D), uppercase letter (U), or lowercase letter (L). The second stage comprises three models: digit ($M_{\mathrm{D}}$), uppercase letter ($M_{\mathrm{U}}$), and lowercase letter ($M_{\mathrm{L}}$). On the basis of the decision in the first stage, the corresponding model is chosen in the second stage—that is, $M_{\mathrm{D}}$ is chosen if the $M_{\mathrm{D,U,L}}$ output ($C_{\mathrm{D1}}$) shows the highest likelihood of a digit. The task of $M_{\mathrm{D}}$ is to perform classification among 10 digits (0 to 9) and pass on likelihoods to the third model, $M_{\mathrm{P}}$, which applies a simple post-processing filter and does the final classification decision based on the highest likelihood. The pipeline works in a similar way for characters. Models $M_{\mathrm{D,U,L}}$, $M_{\mathrm{D}}$, $M_{\mathrm{U}}$, and $M_{\mathrm{L}}$ are all two-layer NNs with only 20 nodes at the hidden layer. The advantage of this design is that we can keep each model size much smaller (that is, less than 1/10th the size of the model described in MD1). The MD2 design is about 24 Kbytes, which is the upper threshold that the Arduino OS allows. The slight downside is additional latency for loading models into RAM.

**Model design 3 (MD3).** We generated a modularized ensemble design with even less RAM usage compared to MD2. The input image breaks down into five spatial regions (four corners and one center, each 10 pixels wide), as shown in Figure 2e, and a separate model has been trained for classifying each region. In other words, when comparing with MD2, each NN model ($M_{\mathrm{D,U,L}}$ through $M_{\mathrm{L}}$) is now a combination of five smaller NN models. From a higher level, MD3's architecture and functionality are the same as those of MD2; the only change is the size and number of models (5 to 10 hidden nodes per model). We reduced each model's size to 25 percent in MD2, and our entire MD3 pipeline from capture to the recognition result takes only 12.3 Kbytes, which sufficiently fits into Curie.

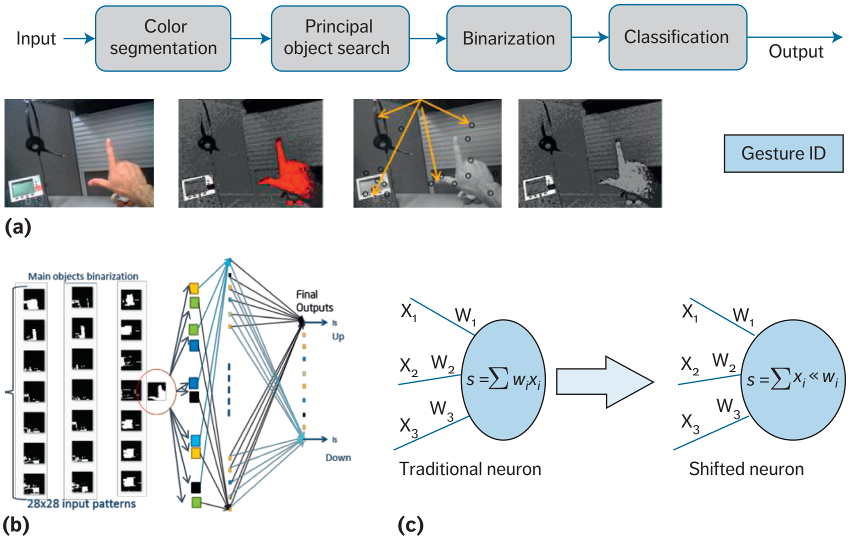


**Figure 3.** Recognition flow for gesture recognition and optimized neural network (NN) classification models. (a) Pipeline overview. (b) Classification model. (c) Optimized NN approach.

Classification time increases due to an increase of models, but MD3 excels in terms of the lowest memory footprint.

### Results

Table 1 summarizes our results and shows that our implementation (MD3) takes 12.3 Kbytes of SRAM, which is within the allowable range of the Arduino 101 Sketch OS. The average latency of the entire inference pipeline from capture to classification display on LCD runs in less than 3 seconds.

## Visual IoT: Medium-Scale Recognition

In this section, we investigate the ultra-low-power processing implications for gesture (hand-pose) recognition.

### Processing Flow for Gesture Recognition

As Figure 3a shows, our gesture algorithm comprises four main steps. Step one (color segmentation) is responsible for extracting the image color histogram and clustering image pixels based on color. Each layer contains one or more objects, including one layer that contains the background. To perform the segmentation, our algorithm maps the image to the hue-saturation (HS) color space, creates a histogram image in the HS space mapping every pixel color into a cell of the histogram, and uses an iterative method to cluster pixels. Step two (principal object search) uses *k*-means segmentation to identify principal objects in each layer. Step three (binarization) performs binarization and normalization of each piece to generate a collection of candidate patterns, and step four (classification) uses a convolutional neural network (CNN) to recognize the embedded patterns. Figure 3b shows the hand-pose classification pipeline.

### Neuromatch: Optimized Neural Network

For our pipeline, the CNN tends to be the biggest bottleneck. More specifically, layer three of the CNN (a fully connected layer) consumes the most power. From this observation, we designed a hardware accelerator for the NN. Our proposed artificial NN implementation, Neuromatch, reduces the memory footprint of a traditional NN by approximately eight times, avoids the use of double-precision floating units, eliminates the use of transcendental functions, and reduces to zero the number of multiplications in the feed-forward layers, thus reducing power and improving performance.

**Shifted neuron.** Our Neuromatch implementation replaces a traditional neuron with a "shifted neuron" using bit shifts instead of multiplication (see Figure 3c). By using this shifted neural network (SNN) approach, we

Table 2. Accuracy and performance: digit and hand pose.

| Performance measures | Digit | Hand pose |
|---|---|---|
| Testing size | 10,000 | 100 |
| Floating-point NN accuracy | 99% | 99% |
| SNN accuracy | 96% | 97% |
| SNN performance | 52 µs | 52 µs |

can reduce the neuron complexity in terms of power and performance. Additionally, the use of bit shifts instead of multiplications reduces the memory footprint to store the weights, because now we are storing the number of bit positions to shift the input instead of storing typical floating-point data.

**Sigmoid function approximation.** In Neuromatch, the output of the shifted neuron is then input to the activation function. Sigmoids are the most widely used activation function. For Neuromatch, we used a linearly approximated sigmoid function. The basis of such approximations is similar to previous approaches.[7]

**Hardware implementation.** We implemented the Neuromatch hardware subsystem on a state-of-the-art process node, and our synthesis results show that the implementation is efficient with a small footprint and sub-mW power consumption. We evaluated Neuromatch's performance for algorithms for handwritten digit, hand-pose, and voice commands recognition. The complete subsystem where Neuromatch was tested integrates the optimized NN solution with a microcontroller core, a convolution hardware accelerator, and an interconnection fabric. For hand-pose recognition, the 29-×-29 pixel images were passed to the convolution accelerator as a first stage; once the convolution was finished, the resulting vector of 1,250 data words was the input for Neuromatch. To train and test the handwritten digit-recognition algorithm, we used the available MNIST dataset. The data used for the hand-pose-recognition algorithm was collected locally. We compared Neuromatch's accuracy against a traditional NN implementation using the same network configuration and the same input vectors. The performance metric considered in these results is the time consumed by Neuromatch to classify the provided data, with a clock frequency of 100 Mhz. Table 2 shows the accuracy and performance results. Neuromatch can match the accuracy results of much heavier floating-point NN implementations for chosen applications. Neuromatch accomplishes this with much lower computational and power costs.

## Visual IoT: Large-Scale Summarization

In this section, we investigate the low-power processing implications of HOG processing for video summarization.

### HOG and HOG-LX for Video Summarization

Because visual IoT devices tend to capture a lot of video, it is challenging to determine salient parts that might be worth viewing or analyzing later. Summarizing videos is an important problem to address, and it is challenging because it requires understanding the changes across many frames in a long-running video as well as the minimum number of frames to best represent it. An example summarization pipeline consists of feature extraction on each frame, followed by an analysis of similarity across frames, and then selection of the key frames that best represent the video.

HOG is one of the most popular and fundamental image features and a basis for other higher-level and more complex feature sets.[8,9] Although the HOG features have been optimized for object-recognition tasks, scene descriptions using HOG tend to be computationally costly. As Figure 4a shows, for summarization, HOG tends to be the most costly operation in terms of computations. Many hardware accelerators for HOG have been developed for real-time object detection.[10] Most of the previous architecture studies on HOG have been focused on achieving real time up to 1080p, especially for object-detection applications.[11] To reduce the original HOG's

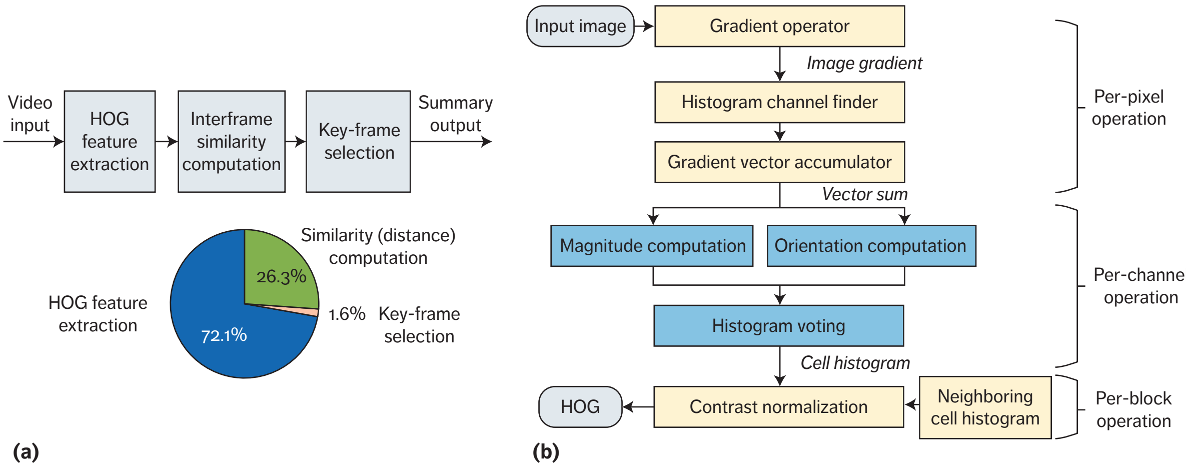


**Figure 4.** Video summarization pipeline and our low-complexity histogram of oriented gradients (HOG-LX) approach. (a) Breakdown of video summarization flow. (b) HOG-LX for efficient video summarization.

computations while minimizing performance degradation as a scene descriptor, we proposed a low-complexity HOG (called HOG-LX) as an efficient front end to many computer vision applications. It reorganizes the order of computations and creates operations per histogram channel. The design of the HOG-LX scheme in Figure 4b is focused on hardware acceleration, which can be leveraged for low-power, low-cost products, including wearable and mobile platforms. The complete detailed architecture of HOG-LX is available in previous work.[12] Here, we enumerate some of the optimizations made to the HOG computation.

**Accelerated histogram binning.** Each cell computes an orientation histogram from image gradients within an $n \times n$ cell region. We replaced expensive per-pixel computations with a new scheme that converts the per-pixel binning into operations per histogram channel. For example, with a cell size $n = 8$ and histogram channels $m = 9$, $\times 64$ operations ($n^2$) are reduced to $\times 9$ operations ($m$).

**Histogram channel finding.** We developed a fast channel finder that avoids the expensive *arctan* operation used in the original HOG implementations. It minimizes the number of comparisons using the symmetry of tangent, thereby decreasing multiplications and memory requirement for boundary tangent values.

**Per-channel computation.** We optimize per-channel operations by first determining channels and constituent gradients, then combining gradients belonging to the same channels through a vector sum, which results in a single gradient per channel. The remaining expensive operations are then performed only on a single vector.

**Simplification for lightweight computations.** From various experimental case studies, we found that the level of sophistication in HOG can be adjusted depending on applications, especially for scene saliency, in which HOG is used as a global scene descriptor.[12] To eliminate nonhardware-friendly operations, HOG-LX chooses four simplifications: skip the bilinear voting in the histogram binning, no arctangent computation in the histogram binning, use L1 norm of a gradient vector sum in the voting, and use L∞ norm in the contrast normalization.

**Data-reusable scanning for efficient memory usage.** To reduce the memory storage requirements at a given time, we divide the input image into vertical tiles with overlapping border cells. The novel tiled memory scanning keeps histograms of horizontally adjacent cells of a single tile instead of keeping an entire image in the local memory. The tile width can be adjusted to the local memory size as much as needed under memory-constrained conditions.

## HOG-LX Analysis and Evaluation

We used ModelSim, a Verilog-based tool for simulation, to evaluate HOG-LX's efficiency.[12] For 1080p with 30 fps video using a 20-cell width tile, HOG-LX can achieve 1.17 gigaoperations per second (GOPs) with 0.85 Mbits per second of memory bandwidth. The original HOG requires 15.12 GOPs and 3.07 Gbits per second of memory bandwidth. For accuracy comparison between HOG and HOG-LX, we used the UCF-101 action dataset[13] and observed similar discrimination. See previous work for a detailed analysis of the HOG-LX benefits.[12]

In this article, we focused on the ultra-low-power processing implications for edge devices in the visual IoT domain. We showed that the challenges in computational and memory needs while staying within the processing power constraints can be addressed through intelligent co-design of algorithms and models as well as accelerators. We believe the data and the findings in this article will be useful to researchers and architects working on visual IoT technologies and solutions going forward.

## Acknowledgments

The authors thank Teahyung Lee, Myung Hwangbo, Tanfer Alan, and Shayok Chakraborty for their contributions to the video summarization.

## References

1. R. Iyer and E. Ozer, "Visual IoT: Architectural Opportunities and Challenges," *IEEE Micro*, vol. 36, no. 6, 2016, pp. 45–49.
2. A. Krizhevsky, I. Sutskever, and G.E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Proc. 25th Int'l Conf. Neural Information Processing Systems*, 2012, pp. 1097–1105.
3. D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column Deep Neural Networks for Image Classification," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2012, doi:10.1109/CVPR.2012.6248110.
4. S. Chakraborty, O. Tickoo, and R. Iyer, "Adaptive Keyframe Selection for Video Summarization," *Proc. IEEE Winter Conf. Applications of Computer Vision*, 2015, pp. 702–709.
5. T.E. Campos, B.R. Babu, and M. Varma, "Character Recognition in Natural Images," *Proc. 4th Int'l Conf. Computer Vision Theory and Applications*, vol. 2, 2009, pp. 273–280.
6. N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition*, vol. 1, 2005, pp. 886–893.
7. A. Hesham, K. Memy Curtis, and Barrie R. Hayes-Gill. "Piecewise Linear Approximation Applied to Nonlinear Function of a Neural Network," *IEE Proc.-Circuits, Devices and Systems*, 1997, pp. 313–317.
8. L.J. Li et al., "Object Bank: An Object-Level Image Representation for High-Level Visual Recognition," *Int'l J. Computer Vision*, vol. 107, no. 1, 2014, pp. 20–39.
9. P. Dollar et al., "Fast Feature Pyramids for Object Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2014, pp. 1532–1545.
10. R. Kadota et al., "Hardware Architecture for HOG Feature Extraction," *Proc. 5th Int'l Conf. Intelligent Information Hiding and Multimedia Signal Processing*, 2009, doi:10.1109/IIH-MSP.2009.216.
11. K. Takagi et al., "A Sub-100-Milliwatt Dual-Core HOG Accelerator VLSI for Real-Time Multiple Object Detection," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing*, 2013, doi:10.1109/ICASSP.2013.6638112.
12. T. Lee et al., "Low-Complexity HOG for Efficient Video Saliency," *Proc. Int'l Conf. Image Processing*, 2015, doi:10.1109/ICIP.2015.7351505.
13. K. Soomro, A.R. Zamir, and M. Shah, *UCF101: A Dataset of 101 Human Action Classes from Videos in the Wild*, tech. report CRCV-TR-12-01, Center for Research in Computer Vision, Univ. Central Florida, 2012.

**Vui Seng Chua** is a software engineer at Intel, where he works on a cloud-to-edge smart computing system that involves sensor fusion, computer vision, and machine learning. Chua received an MEng in mechatronics engineering from the University of Nottingham (Malaysia campus). Contact him at vui.seng.chua@intel.com.

**Julio Zamora Esquivel** is an R&D research scientist at Intel. His research interests include computer vision, machine learning, robotics, and geometric algebra. Zamora Esquivel received a PhD in computer vision from Cinvestav (Center for Research and Advanced Studies of the National Polytechnic Institute, Mexico). Contact him at julio.c.zamora.esquivel@intel.com.

**Anindya S. Paul** is a senior research scientist at Intel, where he's working on machine learning and deep learning based solutions on platform analytics. Paul received a PhD in electrical engineering from OGI School of Science and Engineering at Oregon Health & Science University. Contact him at anindya.s.paul@intel.com.

**Thawee Techathamnukool** is an experienced software engineer. His research interests include 3G/4G/LTE wireless telecommunication, machine learning on the IoT, and wearable and embedded software development. Techathamnukool received an MS in computer science from Southern Illinois University at Carbondale. He completed the work for this article while at Intel. Contact him at mrthawee@gmail.com.

**Carlos Flores Fajardo** is a HW engineer at Intel, where he's working on SoC architecture and design, wireless communications systems, and FPGA emulation/prototyping. Flores Fajardo received an MEng in electronic design at ITESO University, Mexico. Contact him at carlos.a.flores.fajardo@intel.com.

**Nilesh Jain** is a principal engineer in the Data Center Group at Intel. His research interests include machine learning applications at the edge and in the cloud and system architecture and technologies that improve power and performance. Jain received an MS in computer engineering from the Oregon Graduate Institute, Portland. Contact him at nilesh.jain@intel.com.

**Omesh Tickoo** is a research manager in Intel Labs. His research interests include next-generation algorithms and platform solutions for human-computer interaction using computer vision and associated sensing modalities. Tickoo received a PhD in multimedia transmission over wireless networks from Rensselaer Polytechnic Institute. Contact him at omesh.tickoo@intel.com.

**Ravi Iyer** is an Intel Fellow and the director of datacenter technologies in the Data Center Group at Intel. His research interests include developing efficient architectures and solutions for emerging workloads, including visual computing and AI. Iyer received his PhD in computer science from Texas A&M University. He is also an IEEE Fellow. Contact him at ravishankar.iyer@intel.com.

# An Overview of Time-Based Computing with Stochastic Constructs

**Computing on time-based data is a recent evolution of research in stochastic computing (SC). As with SC, complex functions can be computed with low area cost, but the latency and energy efficiency are favorable compared to computations on conventional binary radix. This article reviews the design and implementation of arithmetic operations on time-encoded signals and discusses the advantages, challenges, and potential applications.**

**M. Hassan Najafi, Shiva Jamali-Zavareh, David J. Lilja, Marc D. Riedel, Kia Bazargan, Ramesh Harjani**
*University of Minnesota, Minneapolis*

Stochastic computing (SC), a paradigm first introduced by W.J. Poppelbaum[1] and Brian Gaines[2] in the 1960s, has received considerable attention in recent years, particularly after Weikang Qian and colleagues reintroduced the concept to the electronic design automation community.[3,4] It has since been explored as a potential paradigm for emerging technologies and "post-CMOS" computing. SC systems have very low area cost. This generally translates to low power consumption, making the paradigm interesting for ultra-low-power processing systems.

In SC systems, logical computation is performed on random bitstreams called *stochastic numbers* (SNs). Two representations are used:

- In the *unipolar* representation, each real valued number $x$ ($0 \leq x \leq 1$) is represented by a sequence of random bits, each of which has probability $x$ of being 1 and probability $1 - x$ of being 0.
- In the *bipolar* representation ($-1 \leq x \leq 1$), each bit in the stream has a probability $(x + 1)/2$ of being 1 and $1 - (x + 1)/2$ of being 0.

For example, 10011, 10101, and 11100 are all SNs representing 0.60 in the unipolar and 0.2 in the bipolar representations.

SC offers some intriguing advantages over conventional binary radix. Complex functions can be implemented with simple hardware. This enables the design of low-area and

0272-1732/17/$33.00 © 2017 IEEE

low-power arithmetic units. For instance, multiplication can be performed with a single AND gate, and scaled addition can be formed with a single multiplexer unit. Also, SC provides tolerance to soft errors (that is, bit flips),[4] timing errors,[5] and clock skew.[6] The obvious disadvantage of SC is the latency. A stochastic representation is exponentially longer than conventional binary radix. This translates to long operation times, particularly if high accuracy is required.[7] Long bitstreams can be compensated for, to some extent, by shortened clock cycles. Nevertheless, long latencies translate into high energy consumption and so offset any gains made by simplified hardware.

This article explores an evolution of the concept of SC. Instead of encoding data in space, as random bitstreams, we encode values in time. The time encoding consists of periodic signals, with the value encoded as the fraction of the time that the signal is in the high (on) state compared to the low (off) state in each cycle. We call these *pulse-width modulated* (PWM) signals (see Figure 1).

Our approach is motivated by the observation that, as technology has scaled and device sizes have gotten smaller, the supply voltages have dropped while the device speeds have improved.[8] Control of the dynamic range in the voltage domain is limited; however, control of the length of pulses in the time domain can be precise.[8,9] Encoding data in the time domain may be more accurate and efficient than converting signals into binary radix.

This time-based representation is an excellent fit for low-power applications that include time-based sensors, such as image processing circuits in vision chips. Converting a variety of signals from an external voltage to a time-based representation can be done much more efficiently than a full conversion to binary radix. This enables a savings of at least 10 times in power at the outset.[10]

By exploiting pulse width modulation, signals with specific probabilities can be generated by adjusting the frequency and duty cycles of the PWM signals. These signals can be treated as inputs to the same logical structures used in stochastic computation, with the value defined by the duty cycle. This observation is motivated by noting that the stochastic representation is a uniform, fractional representation. All that matters in terms of the value that is computed is the fraction of time that the signal is high.[6] For example, if a signal is high 68.7 percent of the time, it is evaluated as 0.687 (see Figure 1).

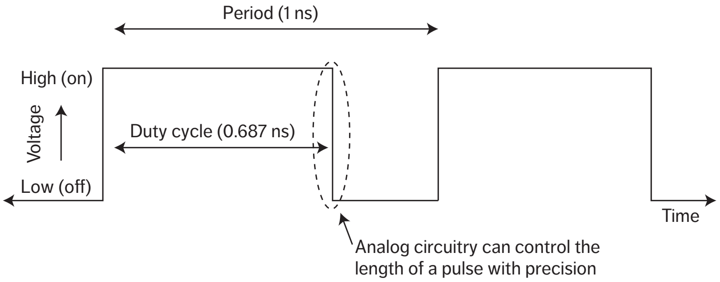

Period (1 ns)
High (on)
Voltage
Duty cycle (0.687 ns)
Low (off)
Time
Analog circuitry can control the length of a pulse with precision

**Figure 1.** Encoding in time with a periodic analog signal. The value represented is the fraction of the time that the signal is high in each cycle—in this case, 0.687.

This article reviews a transformative new idea: a technique for performing computation on time-encoded analog values directly with ordinary CMOS digital logic.[10] This is related to work on a deterministic approach to SC.[10–12] We have shown that, if properly structured, computation on deterministic bitstreams can be performed with the same circuits as are used in SC, yielding the following benefits:

- Unlike stochastic methods, our deterministic methods produce completely accurate results, not approximations, with no errors or fluctuations.
- The cost of generating deterministic streams is a small fraction of the cost of generating streams from random or pseudorandom sources.
- Most importantly, the latency is reduced by a factor of $1/2^n$, where $n$ is the equivalent number of bits of precision in binary.

Computation on signals encoded in time is directly analogous to this deterministic approach to SC. In this article, we review the performance of different stochastic operations for data processing of inputs generated by a sensing circuit; such data is time-encoded with PWM signals. We discuss the advantages, challenges, and potential applications for computation on such time-encoded signals.

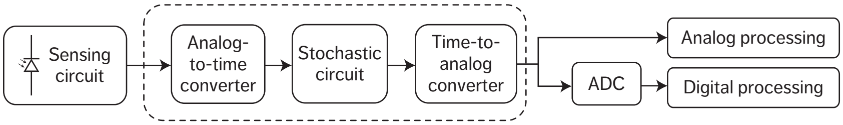


**Figure 2.** Time-based computing with stochastic constructs. An ATC converts the sensed data to a time-encoded pulse signal. The converted signal is processed using the stochastic circuit, and the output is converted back to a desired analog format using a TAC.

## Time-Based Encoding of Stochastic Numbers

Conventionally, the inputs to stochastic circuits are random bitstreams. Sensing circuits, such as image sensors, convert the sensed data (for example, light intensity) to an analog voltage or current. The voltages or currents are then converted to digital form, as binary radix, with costly analog-to-digital convertors (ADCs). Finally, stochastic bitstream generators, consisting of random number generators (that is, linear-feedback shift registers) and comparators, are used to convert the data from binary radix format to stochastic bitstreams.[4]

Recent work has demonstrated low-cost converters that directly convert sensed data from analog form to stochastic bitstreams.[13,14] These greatly reduce the hardware footprint and power consumption of the front end of stochastic circuits. Nevertheless, due to the long latency of operating on random bitstreams, the overall energy consumption—defined as the integral of power consumption over time—remains high. In particular, when high accuracy is needed, the length of stochastic bitstreams becomes prohibitive (for example, more than 1,024 cycles). Even with a higher working frequency, the latency is high; this makes stochastic processing of digital bitstreams inefficient in terms of energy.

However, with sensors that produce time-encoded outputs, which in turn become inputs to the SC circuit, we can work directly with these analog signals instead of converting them into digital bitstreams. This results in a significant saving in energy at the front end. Another compelling advantage is the improvement in the processing time. By using time-encoded signals, the total processing time can be reduced to a time equal to only one clock cycle.[12] The precision of the computation now depends on the precision of the PWM signal in time, rather than the length of the bitstream. Experimental results on image processing applications show up to 99 percent speedup in performance and 98 percent saving in energy dissipation when processing time-encoded signals instead of conventional digital bitstreams[10,12]

Figure 2 shows the flow of computing on time-encoding signals. Assuming that the sensing circuit's output is in voltage or current form, an analog-to-time converter (ATC) circuit (that is, a PWM signal generator) is used to convert the sensed data to a time-encoded pulse signal. This circuit is very low cost, both in terms of hardware area and energy consumption (approximately 30 $\mu m^2$ and 0.08 pJ, respectively, for 1 GHz frequency, when supplying the converter with an external clock source). The converted signal is processed using the same circuit constructs as are used in SC. The output is converted back to a desired analog format using a time-to-analog converter (TAC). This is simply a voltage integrator.

The implementation cost of an ATC, which consists of an analog comparator, a ramp generator, and a clock generator, is a function of its frequency. Increasing the frequency (and thus decreasing the period of the PWM signal) increases the implementation cost of the comparator and ramp generator, but lowers the cost of the clock generator (for example, a lower number of inverters in a ring oscillator leads to a higher oscillation frequency). For frequency ranges of lower than 3 GHz, the clock generator has the dominant cost and so increasing the frequency lowers the total implementation cost of the ATC. However, care must be taken because increasing the frequency lowers the effective number of bit (ENOB) of time-based representation, which might then decrease the accuracy of the computation. For comparable accuracy levels, the synthesis results in our previous work show a 40 percent hardware cost reduction when replacing the conventional SN generator with ATCs in image-processing applications.[10]

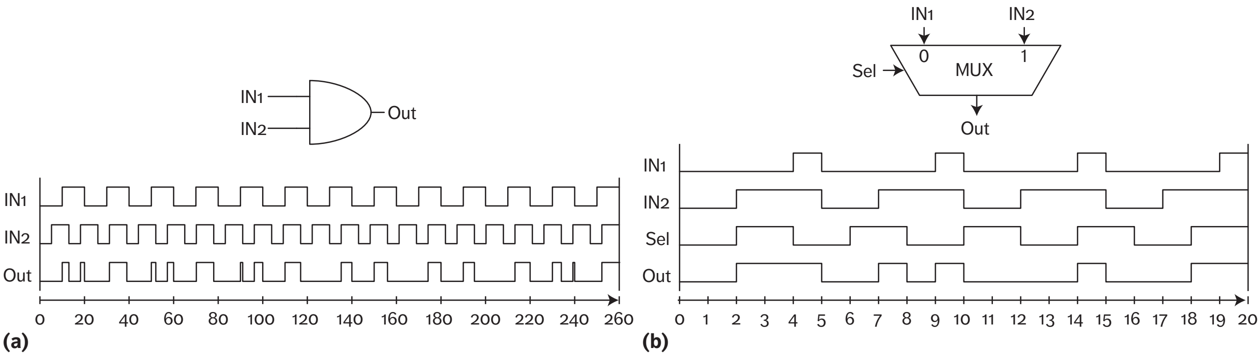


**Figure 3.** Examples of stochastic operations with independent time-encoded inputs. (a) Multiplying two time-encoded pulse-width modulated (PWM) signals using an AND gate. IN1 represents 0.5 with a period of 20 ns, and IN2 represents 0.6 with a period of 13 ns. The output signal represents 0.30 (78 ns/260 ns), the expected value from multiplication of the inputs. (b) Scaled addition using a multiplexer (MUX). IN1 and IN2 represent 0.2 and 0.6 with a period of 5 ns, and Sel represents 0.5 with a period of 4 ns. The output signal represents 0.40 (8 ns / 20 ns), the expected value from the scaled addition of the inputs.

## Independence in Stochastic Circuits

Stochastic operations can be divided into two main categories with respect to correlation between their inputs: operations that require independent (that is, uncorrelated) inputs, and operations that require highly correlated inputs. Multiplication and scaled addition and subtraction are the most common stochastic operations that require independent inputs for correct functionality. An AND gate multiplies two unipolar SNs only if its inputs are independent bitstreams. A multiplexer (MUX) connected to two SNs as the main inputs and another SN as the select input accurately performs scaled addition and subtraction only if the select input is independent of the two main inputs. (Note, however, that the main inputs need not be independent of each other.)

With time-encoded PWM signals, we set the duty cycle to be the value represented. For operations that require independent inputs, such as multiplication using an AND gate or scaled addition using a MUX, PWM signals that are not harmonically related must be used.[10] To see why, consider connecting two PWM signals with the same duty cycle and the same frequency to the inputs of an AND gate. This produces an output equal to the inputs and not the product of the values. Inharmonic frequencies are selected for the input signals, and the operation is run for the least-common multiple (LCM) or multiples of the LCM of the period of the input signals, to produce highly accurate results. Figure 3 shows examples of performing multiplication and scaled addition using time-encoded PWM signals.

Three properties are exclusive to the operations with independent time-encoded inputs:

- *Property 1.* Each independent input must have a frequency inharmonic to the frequencies of other independent inputs. A separate clock source is, therefore, required for each independent input.
- *Property 2.* Increasing the number of independent inputs increases the operation time. The period of the output signal and so the operation time equals the product of the periods (1/frequency) of the independent time-encoded inputs. Thus, by increasing the number of independent inputs, the circuit must run for a longer time to produce accurate results.
- *Property 3.* The accuracy of operations is inversely proportional to the frequency of input signals. Although increasing the frequency lowers the operation time, it decreases the ENOB in representing the input values and so the accuracy in the computations.

Compared to conventional bitstream-based SC, time-encoding the inputs can significantly improve the processing time and hardware area and power cost, and so the energy consumption of operations that require independent inputs.

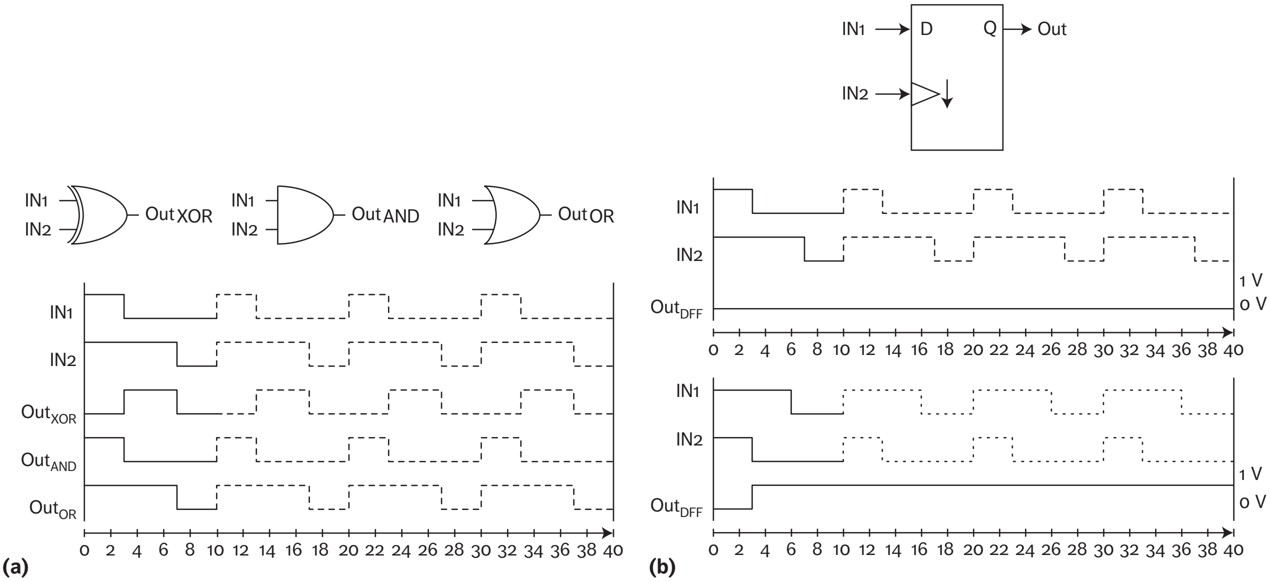


**Figure 4.** Examples of stochastic operations with correlated time-encoded inputs. (a) Performing stochastic absolute-valued subtraction, minimum, and maximum operations on two synchronized PWM signals: IN1 represents 0.3 and IN2 represents 0.7. Both PWM signals have a period of 10 ns. (b) Comparing stochastic numbers (SNs), represented by synchronized PWM signals, using a D-type flip-flop: (up) IN1 < IN2, and thus Out = 0; (down) IN1 > IN2, and thus Out = 1.

## Correlation in Stochastic Circuits

The second category of stochastic operations includes those that require highly correlated inputs. An XOR gate implements absolute-valued subtraction $|x_1 - x_2|$ when it is supplied with highly correlated inputs—that is to say, where the two input streams have maximum overlap in their 1s.[15] As an example, connecting S1 = 11101 and S2 = 10001, two correlated stochastic streams representing 4/5 and 2/5, to the inputs of an XOR gate produces S3 = 01100, the expected value for absolute-valued subtraction. This operation is particularly useful in stochastic implementation of image-processing algorithms, such as Robert's cross-edge detection algorithm.[16]

An AND gate with independent inputs works as a multiplier. However, with highly correlated inputs, it gives the *minimum* of the two stochastic streams. An OR gate supplied with highly correlated streams gives the *maximum* of the two stochastic streams. Thus, a basic sorting unit can be constructed with only an AND and an OR gate: supplied with two correlated inputs, it produces the smaller of the two values on one output line, and the greater of the two on the other. Such a low-cost implementation of sorting can save orders of magnitude in hardware resources and power when compared to the costs of a conventional binary implementation. Such circuits are important for applications such as the median filtering noise-reduction algorithm.[17]

Comparison of SNs is another common operation in stochastic circuits. A low-cost stochastic comparator using a simple D-type flip-flop was proposed in our previous work.[12] For correct functionality, the inputs of the flip-flop must be correlated. For a digital representation, all 1s in each stream must be placed together at the beginning of the stream. The first SN should be connected to the D input, and the second one should be connected to the falling edge triggered clock input. The output of comparing two SNs, N1 and N2, will be 0 if IN1 < IN2, and 1 otherwise.

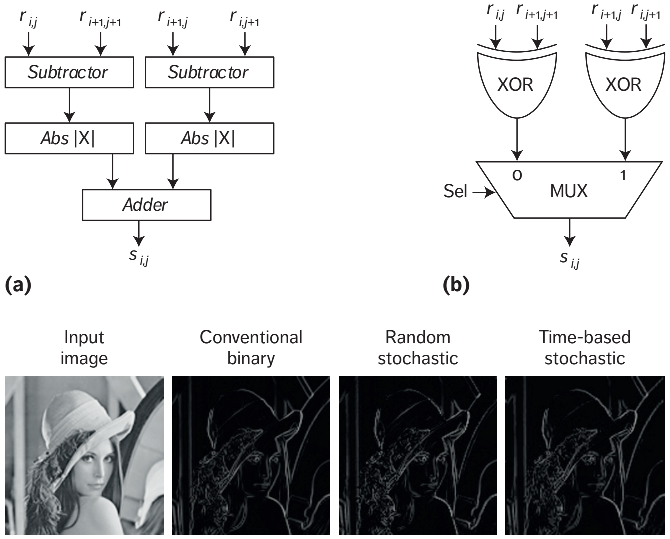
When representing SNs with time-encoded PWM signals, high correlation or maximum overlap is provided by satisfying two requirements: choosing the same frequency for the signals, and having maximum overlap between the high parts of the signals. For example, two PWM signals that have the same frequency, and each has the high part located at the beginning or end of each period, are called “correlated” or “synchronized” signals.[12] Figure 4a shows two synchronized PWM signals and the outputs of performing the stochastic absolute-valued subtraction, minimum, and maximum operations on these. Note that the expected output is produced after a single cycle of the PWM input signals. Continuing the operations for additional cycles (the dotted lines) does not improve the accuracy of the results.

Figure 4b also shows two possible cases of comparing SNs, represented by PWM signals using a D-type flip-flop. When IN1 is smaller than IN2, the falling edge of the PWM signal representing N2 causes the flip-flop to sample a low-level signal, and thus logical-0 is produced at the output. When N1 is greater than N2, the PWM signal representing N1 is still at a high level when the falling edge of IN2 occurs. So, logical-1 will be produced at the output of the flip-flop.

The exclusive properties of operations with correlated time-encoded inputs include the following:

- *Property 1.* The output of performing stochastic operations on synchronized PWM signals is ready after running the operation for only one period of the input signals. As Figure 4 shows, the fraction of time each output signal is high is the same in all periods of each output signal. In such cases, continuing the operation for additional periods (the dotted lines in the figures) does not change the value or, most importantly, the accuracy of the output.
- *Property 2.* In contrast to stochastic operations with independent inputs that needed time-encoded signals with inharmonic frequencies, the inputs of correlated operations must have the same frequency. Thus, only one source, generating one clock signal, suffices.

Similar to operations that require independent inputs, by time-encoding of inputs, the processing time, area, and power cost, and consequently, energy consumption of operations that require highly correlated inputs can all be greatly reduced when compared to those of the conventional bitstream based processing.



| | Conventional binary | Random stochastic | Time-based stochastic |
|---|---|---|---|
| Mean error (%): | 0.00 | 1.49 | 1.28 |
| Area (μm²): | 1,626 | 960 | 596 |
| Delay (ns): | 0.78 | 87.0 | 1.02 |
| Power (μW): | 1,415 | 2,813 | 943 |
| Energy (pJ): | 1.10 | 244 | 0.96 |

(c)

**Figure 5.** The Robert’s cross edge-detection circuit: (a) conventional binary implementation, (b) core stochastic logic, and (c) synthesis results and the results of processing a 128 × 128 sample input image using the binary design, a stochastic design with 256-bit random SNs, and time-based SNs. (For details of the implementations, see our previous work.[10])

## Applications

Growth in digital and video imaging cameras, mobile imaging, biomedical imaging, robotics, and optical sensors has spurred demand for low-cost, energy-efficient circuits for image processing. Prior work on SC has shown this computing paradigm’s potential in low-cost implementation of image and video-processing algorithms. Image processing based on time-encoded signals could have significant impact in this application area, particularly when power constraints dominate. Time-encoded, mixed-signal processing can be performed on the same chip, with analog-to-time conversion followed by logical computation on the time-encoded signals, using stochastic constructs.

Figure 5 shows the conventional binary implementation and the core stochastic logic for the Robert’s cross edge-detection algorithm. The figure summarizes the synthesis results, which are based on a 45-nm gate library. Two sets of numbers are reported: one for a stochastic design, processing 256-bit random streams, and one for a time-based design. Both of these designs share the same core logic, shown in Figure 5b. The conventional bitstream-based stochastic design uses the random stochastic stream generator proposed by Qian and colleagues.[4] The time-based design uses the ATC proposed in our previous work[10] for time-encoding the inputs.

Considering the critical path of the core stochastic logic as the minimum allowed period of the signals when time-encoding the input data, 0.51 ns is selected as the period of the four main inputs and 0.34 ns is selected for the period of the select input. (For more details on choosing the period of the time-encoded signals, see our previous work.[10]) As Figure 5 shows, the time-based design has significantly lower area and power costs than the conventional binary and stochastic designs. The processing time and the energy consumption are also dramatically improved.

Mixed-signal design is attractive for VLSI implementations of neural networks (NNs) for reasons of speed and energy efficiency. Also, mixed-signal solutions do not suffer from the quantization effects that arise with analog-to-digital conversion. NNs are computationally complex, which makes them a good candidate for processing with low-cost stochastic logic. Digital bitstream-based processing of data in stochastic NN often requires running for more than 1,000 clock cycles to achieve an accuracy close to that of conventional deterministic fixed-point binary designs, which then leads to high energy consumption. Time-based SC has the potential to mitigate these costs, offering energy-efficient designs. Unlike conventional SC, the computations can be completely accurate with no random fluctuation. The approach could have a significant impact in the design of near-sensor NN accelerators.

## Challenges

Time-based computing is a mixed-signal technology that combines an analog representation in time with digital processing, using stochastic constructs. In this section, we briefly discuss different challenges in the development and application of method.

### Analog Noise

Recent work has shown that by properly structuring digital bitstreams, completely deterministic computation can be performed with stochastic logic.[11] The results are completely accurate with no random fluctuations. Due to the mixed-signal nature of time-based processing, computations on time-encoded signals are susceptible to noise; one cannot promise 100 percent accuracy. Analog noise cannot be completely eliminated from signals and therefore from computation. By careful design of ATC and TAC, and by choosing appropriate frequencies, however, the error can be made very low (less than 0.001 percent mean absolute error).

### Resolution

The resolution in time-based processing is limited by noise, rather than by the length of bitstreams, as it is with SC. While there is no limit in the resolution of SNs represented by digital bitstreams, the resolution in our time-encoded approach is limited by the maximum ENOB of the ATC (that is, the PWM generator). For a minimum frequency of 10 MHz, current ATCs can achieve a maximum ENOB of 11 to 12 bits.

### Truncation

With time-encoded signals, operations should run for a specific amount of time to produce correct results. For operations with independent inputs, this time equals the product of the period of the input signals; for operations with correlated inputs, it equals the period of the input signals. Running the operation for longer or shorter than the required time results in truncation error.[10] In contrast, stochastic bitstreams have the property of *progressive precision*, meaning that short subsequences of an SN can provide low-precision estimates of its value.[16] The longer the stream runs, the more precise the value. Given enough time, the output converges to the expected correct value, and consequently, the truncation error is generally low.

### Synchronization

Operations using synchronized PWM signals are limited to only the first level of logic in a circuit. Providing the required synchronization—that is, having maximal overlap between the high part of the input signals—is difficult to achieve for the second and higher logic levels.

A naive solution is to convert the output of each level back to an analog format, then perform an analog-to-time conversion and feed this to a higher level. However, this naive method decreases the accuracy and is costly in terms of latency, area, and energy.

### Skew

The synchronization must be perfect in operations that require synchronized inputs. On-chip variations or noise sources affecting clock generators can result in deviations from the expected period, phase shift, or slew rate of the signals. Different delays for AND and OR gates, for example, can be a source of significant skew in implementing sorting-based circuits. The skew in each stage is propagated to the next, resulting in a considerable skew error for large circuits. Mitigating the skew by delaying some signals is complex and costly, and may offset gains in area and power.

### Rotation

Relatively prime stream lengths, clock division, and rotation were three methods explored by Devon Jenson and Marc Riedel for processing bitstreams deterministically.[11] Choosing inharmonic frequencies for the time-encoded signals corresponds to the "relatively prime" method in Jenson and Riedel.[11] A high-frequency time-encoded PWM signal is connected to the select input of the MUX in previous work by Najafi and Lilja[12] for an accurate scaled addition operation. This approach corresponds to the "clock division" method in Jenson and Riedel.[11] In their "rotation" method,[11] digital bitstreams are stalled for one cycle at powers of the stream length, causing each bit of one bitstream to see each bit of the other stream exactly once. Considering the high working frequency of time-based SC, stalling PWM signals for a very short and precise amount of time might not be possible.

### Sequential Circuits

Sequential finite-state machine (FSM)-based approaches exist for implementing complex functions with SC.[18,19] These methods depend on randomness in different ways than combinational methods do. It is not clear how to translate these sequential constructs to deterministic computation on time-based PWM signals.

Computation on time-based encodings offers significant advantages over both deterministic and conventional stochastic approaches. It generally results in circuits that are much less costly in terms of area and power, particularly for applications where the inputs are presented in analog voltage or current form. The savings in the analog-to-time conversion step compared to a full analog-to-digital conversion are significant. Accordingly, the approach is a good fit for low-power real-time image-processing circuits, such as those in vision chips. In future work, we will develop an ultra-low-power video-processing unit using the discussed time-based processing approach. We also use this processing approach in a low-cost, energy-efficient implementation of convolutional NNs and near-sensor NN accelerators

### Acknowledgments

This work was supported in part by National Science Foundation grant no. CCF-1408123. Any opinions, findings and conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

### References

1. W.J. Poppelbaum, C. Afuso, and J.W. Esch, "Stochastic Computing Elements and Systems," *Proc. Jt. Computer Conf.*, 1967, pp. 635–644.
2. B.R. Gaines, "Stochastic Computing Systems," *Advances in Information Systems Science*, J. Tou, ed., Springer, 1969, pp. 37–172.
3. W. Qian and M. Riedel, "The Synthesis of Robust Polynomial Arithmetic with Stochastic Logic," *Proc. 45th ACM/IEEE Design Automation Conf.*, 2008, pp. 648–653.
4. W. Qian et al., "An Architecture for Fault-Tolerant Computation with Stochastic Logic," *IEEE Trans. Computers*, vol. 60, no. 1, 2011, pp. 93–105.
5. A. Alaghi et al., "Trading Accuracy for Energy in Stochastic Circuit Design," *J. Emerging Technologies in Computing*

*Systems*, vol. 13, no. 3, 2017, pp. 47:1–47:30.

6. M.H. Najafi et al., “Polysynchronous Stochastic Circuits,” *Proc. 21st Asia and South Pacific Design Automation Conf.*, 2016, doi:10.1109/ASPDAC.2016.7428060.
7. J. Hayes, “Introduction to Stochastic Computing and Its Challenges,” *Proc. 52nd ACM/EDAC/IEEE Design Automation Conf.*, 2015, pp. 1–3.
8. *International Technology Roadmap for Semiconductors 2.0*, 2015; www.itrs2.net/itrs-reports.html.
9. G.W. Roberts and M. Ali-Bakhshian, “A Brief Introduction to Time-to-Digital and Digital-to-Time Converters,” *IEEE Trans. Circuits and System-II*, vol. 57, no. 3, 2010, pp. 153–157.
10. M.H. Najafi et al., “Time-Encoded Values for Highly Efficient Stochastic Circuits,” *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 5, 2017, pp. 1–14.
11. D. Jenson and M. Riedel, “A Deterministic Approach to Stochastic Computation,” *Proc. 35th Int’l Conf. Computer-Aided Design*, 2016, pp. 102:1–102:8.
12. M.H. Najafi and D.J. Lilja, “High-Speed Stochastic Circuits using Synchronous Analog Pulses,” *Proc. 22nd Asia and South Pacific Design Automation Conf.*, 2017, pp. 481–487.
13. D. Fick et al., “Mixed-Signal Stochastic Computation Demonstrated in an Image Sensor with Integrated 2D Edge Detection and Noise Filtering,” *Proc. IEEE Custom Integrated Circuits Conf.*, 2014, pp. 1–4.
14. N. Onizawa et al., “Analogto-Stochastic Converter using Magnetic Tunnel Junction Devices for Vision Chips,” *IEEE Trans. Nanotechnology*, vol. 15, no. 5, 2016, pp. 705–714.
15. A. Alaghi and J. Hayes, “Exploiting Correlation in Stochastic Circuit Design,” *Proc. IEEE 31st Int’l Conf. Computer Design*, 2013, pp. 39–46.
16. A. Alaghi, C. Li, and J. Hayes, “Stochastic Circuits for Real-Time Image-Processing Applications,” *Proc. 50th ACM/IEEE Design Automation Conf.*, 2013, pp. 1–6.
17. M. Hassan Najafi et al., “Power and Area Efficient Sorting Networks using Unary Processing,” to be published in *Proc. IEEE 35th Int’l Conf. Computer Design*, 2017.
18. B.D. Brown and H.C. Card, “Stochastic Neural Computation I: Computational Elements,” *IEEE Trans. Computers*, vol. 50, no. 9, 2001, pp. 891–905.
19. M.H. Najafi et al., “A Reconfigurable Architecture with Sequential Logic-Based Stochastic Computing,” *ACM J. Emerging Technologies in Computing Systems*, vol. 13, no. 4, 2017, article 57.

**M. Hassan Najafi** is a PhD candidate and research assistant at ARCTiC Labs in the Department of Electrical and Computer Engineering at the University of Minnesota, Minneapolis. His research interests include stochastic and approximate computing, computer-aided design of integrated circuits, low-power design, and fault-tolerant system design. Najafi received an MSc in computer architecture from the University of Tehran, Iran. Contact him at najaf011@umn.edu.

**Shiva Jamali-Zavareh** is a PhD student in the Analog Design Laboratory in the Department of Electrical and Computer Engineering at the University of Minnesota, Minneapolis. Her research interests include analog front ends, data converters, RF circuit design, and stochastic computing. Jamali-Zavareh received an MSc in microelectronic circuit design from Aalto University, Finland. Contact her at jamal036@umn.edu.

**David J. Lilja** is the Schnell Professor of Electrical and Computer Engineering at the University of Minnesota, Minneapolis, where he also serves as a member of the graduate faculties in computer science, scientific computation, and data science. His research interests include computer architecture, high-performance parallel processing, computer systems performance analysis, approximate computing, computing with emerging technologies, and storage systems. Lilja received a PhD in electrical engineering from the University of Illinois at Urbana-Champaign. He is a Fellow of IEEE and the American Association for the Advancement of Science (AAAS). Contact him at lilja@umn.edu.

**Marc D. Riedel** is an associate professor of electrical and computer engineering with the

University of Minnesota, Minneapolis, where he is a member of the Graduate Faculty of biomedical informatics and computational biology. His research interests include logic synthesis, stochastic computing, and DNA computing. Riedel received a PhD in electrical engineering from Caltech. He has received the Charl H. Wilts Prize for the Best Doctoral Research in Electrical Engineering at Caltech, the Best Paper Award at the Design Automation Conference, and the US National Science Foundation CAREER Award. Contact him at mriedel@umn.edu.

**Kia Bazargan** is an associate professor in the Department of Electrical and Computer Engineering at the University of Minnesota, Minneapolis. His research interests include stochastic computing, FPGA architectures and applications, and physical design for FPGAs. Bazargan received a PhD in electrical and computer engineering from Northwestern University. He received the US National Science Foundation Career Award. He was an associate editor of *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* and a guest editor of *ACM Transactions on Embedded Computing Systems*. He is a senior member of the IEEE Computer Society. Contact him at kia@umn.edu

**Ramesh Harjani** is the Edgar F. Johnson Professor with the Department of Electrical and Computer Engineering at the University of Minnesota, Minneapolis. His research interests include analog/RF circuits for wired and wireless communications. Harjani received a PhD in electrical engineering from Carnegie Mellon University. He is the cofounder of Bermai, a start-up developing CMOS chips for wireless multimedia applications. Contact him at harjani@umn.edu.

# Hardware Designs for Security in Ultra-Low-Power IoT Systems: An Overview and Survey

**This article presents a survey of state-of-the-art hardware designs optimizing the tradeoffs between security, power, and costs in ultra-low-power systems like the Internet of Things. The authors analyze the connections between hardware specs and system demands to bridge the gap between research conducted in different communities. They also identify open problems in designing future ultra-low-power and secure hardware.**

**Kaiyuan Yang**
*Rice University*

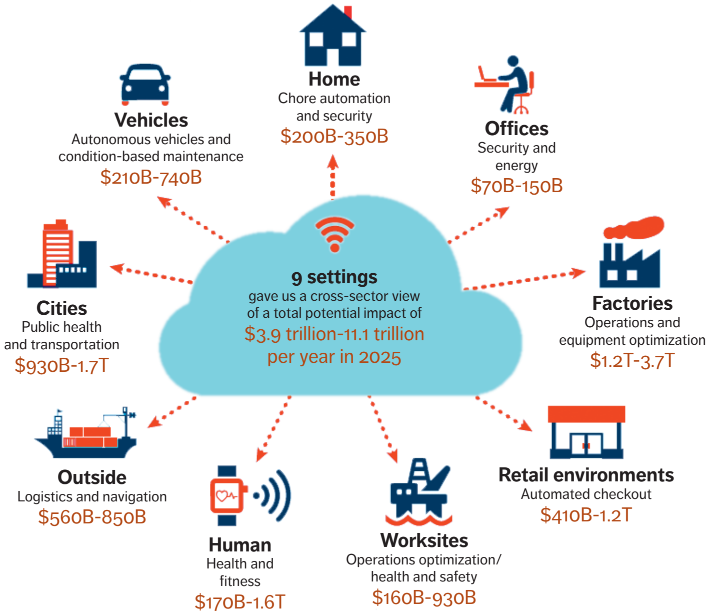
**David Blaauw,**
**Dennis Sylvester**
*University of Michigan*

The emergence of the Internet of Things (IoT) and pervasive computing are expected to enable physical things in the world to collect, process, and exchange data over the Internet. The blending of physical and cyber worlds will open up opportunities to revolutionize healthcare, transportation, infrastructure, and manufacturing industries (see Figure 1). The fundamental technology enablers are ubiquitous ultra-low-power (ULP) and ultra-low-cost edge devices equipped with sensors, actuators, computers, and network connectivity. In particular, ULP systems that can operate on batteries or even on harvested energy for years will enable many disruptive applications, such as implanted and wearable medical and fitness devices, environmental monitors for ecosystem study and protection, and industrial applications.

At the heart of all the ubiquitous applications in Figure 1 is a huge amount of personal, sensitive, or confidential data to be processed and transmitted. Therefore, security and privacy issues are among the most important challenges faced by this technology. Securing these ULP systems poses additional difficulties beyond conventional computer system and network security due to strictly limited computing resources, stringent power budgets, severe cost pressures, and the devices' physical accessibility to attackers. Security within ULP systems must be improved and optimized at every system stack. This article focuses on the lower stacks of the system, namely the hardware building blocks.

Published by the IEEE Computer Society

0272-1732/17/$33.00 © 2017 IEEE



Home
Chore automation and security
$200B-350B
Vehicles
Autonomous vehicles and condition-based maintenance
$210B-740B
Offices
Security and energy
$70B-150B
Cities
Public health and transportation
$930B-1.7T
9 settings
gave us a cross-sector view of a total potential impact of
$3.9 trillion-11.1 trillion per year in 2025
Factories
Operations and equipment optimization
$1.2T-3.7T
Outside
Logistics and navigation
$560B-850B
Human
Health and fitness
$170B-1.6T
Worksites
Operations optimization/ health and safety
$160B-930B
Retail environments
Automated checkout
$410B-1.2T

**Figure 1.** Internet of Things (IoT) applications and their expected market share. (Source: McKinsey Global Institute.)

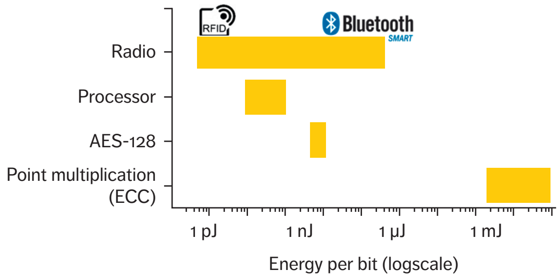


**Figure 2.** Energy efficiency of common IoT building blocks.

The identification and authentication of physical items are among the most fundamental requirements for secure IoT systems. Common examples include RFIDs for supply chain management, smart cards for owner verification, and wireless sensor nodes for secure data transmission. Identification can be established with any form of public identifier, such as physical marks or electronic IDs stored in nonvolatile memory (NVM) devices. Comparatively, authentication is much more demanding; it requires one entity of a protocol (the verifier) to be assured of the claimed identity of the other entity (the prover)—that is, to distinguish genuine physical things from counterfeited ones and prevent impersonation attack in networks. One-way or mutual authentication should be implemented depending on the targeted applications. A common approach to authentication relies on challenge-and-response protocols, in which the verifier asks a question and the prover must provide a valid answer to be authenticated. The questions and answers are agreed on in advance. The most common implementation of such a protocol is based on cryptographic primitives and secret keys. However, implementing these two primitives in IoT devices faces challenges of severe power and cost budgets, as well as physical attacks ranging from direct probing to side-channel monitoring. Therefore, a new security primitive aiming at secure key storage and lightweight authentication, called *physically unclonable function* (PUF), has emerged in recent years. The essential idea of PUF is to employ manufacturing variations as entropy sources to generate a random mapping function unique to each fabricated instance, which was first envisioned with optical and silicon demonstrations in 2002.[1,2] Over the years, however, it has been shown that PUFs face several critical issues related to their reproducibility, physical security, and vulnerability to modeling attacks.

In addition to identification and authentication of IoT devices, the secrecy and integrity of sensitive data being transmitted within the network, usually wirelessly, are also critical. Until now, cryptographic primitives have been the only practical methods to achieve the security requirements. High power consumption is the main challenge to implementing all kinds of cryptographic primitives in IoT devices. Figure 2 shows the energy costs of encryption compared to other building blocks of a typical IoT system. Similar to entity authentication, these designs face vulnerability to physical attacks. Attackers can exploit power and electromagnetic (EM) radiation information of the physical implementation to reveal the secret keys.

In this article, we present a survey of the state-of-the-art hardware designs optimizing the tradeoffs between security, power, and costs (including design and manufacturing). Most of the designs have silicon prototypes and measurement results. Open questions in designing future ULP secure hardware are discussed as well.

## Cryptography-Based Entity Authentication

Figure 3 shows common challenge-and-response authentication protocols using cryptographic

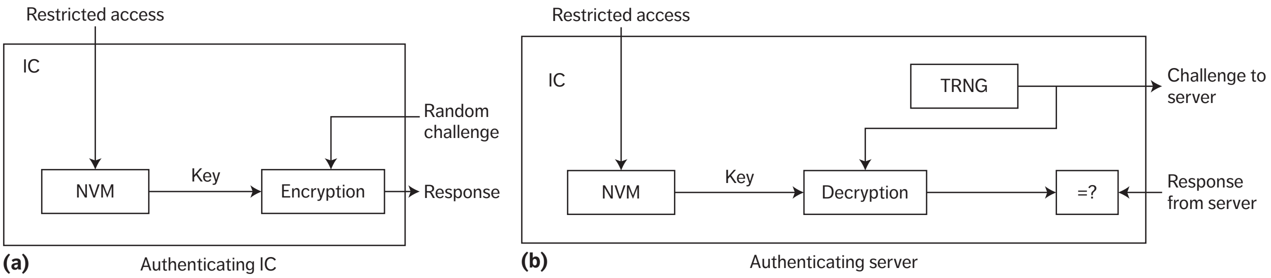


**Figure 3.** Basic authentication protocol using cryptographic primitives.[3]

primitives and secret keys. There are two types of solutions: block ciphers or hash functions with secret keys shared by the verifier and prover, and public-key ciphers with secret keys protected by provers. Both of these require hardware blocks for random number generation, cryptographic computation, and secret key storage. Novel implementations of these blocks are necessary to keep them within power and cost budgets while keeping them secure from potential attacks.

## Secure Crypto Engine

Cryptographic primitives such as block ciphers, public-key ciphers, and hash functions are the most widely used building blocks in secure systems. As Figure 2 shows, running these algorithms in software is not practical for ULP processors in IoT devices, because of the large latency, low energy efficiency, and limited memory space. Therefore, hardware accelerators for ULP devices are critical to reducing overall power consumption, but the available power and area budget limit the use of existing high-throughput and high-efficiency accelerators targeting server applications.[4] Recent research efforts have focused on developing lightweight accelerators for cryptographic algorithms that consume less power and area without the loss of energy efficiency. We selected the most widely used block cipher, Advanced Encryption Standard (AES), for a case study. Many of the design techniques can be adopted by other algorithms as well.

**Lightweight AES engine.** AES is a block cipher working on 128-bit input blocks. It takes 10/12/14 rounds for 128/196/256 key lengths. Within each round, the data is processed in four steps, including AddRoundKey (mixing input data and round key by XOR), SubBytes (nonlinear operation based on an SBox), ShiftRow (cyclically shifting the four rows by 1/2/3/4 bytes), and MixColumn (modular polynomial multiplication with a constant array). Early design efforts focused on high-speed designs using pipelined and loop-unrolled architectures. One of the fastest and most efficient designs was developed by Sanu Mathew and colleagues.[4] However, the specs of such designs are not suitable for ULP systems, and their architectures have delay overhead when used in cipher modes with feedbacks (such as CBC-MAC), which are widely used for authenticated encryption protocols suitable for IoT systems.

One of the earliest lightweight AES engines with complete measurement results and significantly improved power and area costs was presented in 2006.[5] It uses an 8-bit iterative datapath to save area and power, and it employs an SBox calculated in a composite field $GF(2^4)^2$ in runtime, instead of directly storing the $GF(2^8)$ lookup table in the ROM. This technique was first introduced in 2001,[6] and was optimized over the years to achieve better efficiency and a smaller footprint. Almost all recent AES designs have adopted SBox in the composite field, and it is shown to be beneficial to both ULP and high-performance designs. Mathew and colleagues performed an exhaustive search of optimal polynomials to construct the composite field,[7] presenting a 22-nm lightweight AES engine using only 1,947 gates for encryption. The results show that the worst polynomial choice will have 30 percent area overhead compared to the best one. Another innovation is removing the high power and area costs associated with ShiftRow byte permutations.[7] Mathew and colleagues moved this step to the start of each round by rescheduling the input data loaded to the data registers according to the ShiftRow rules.

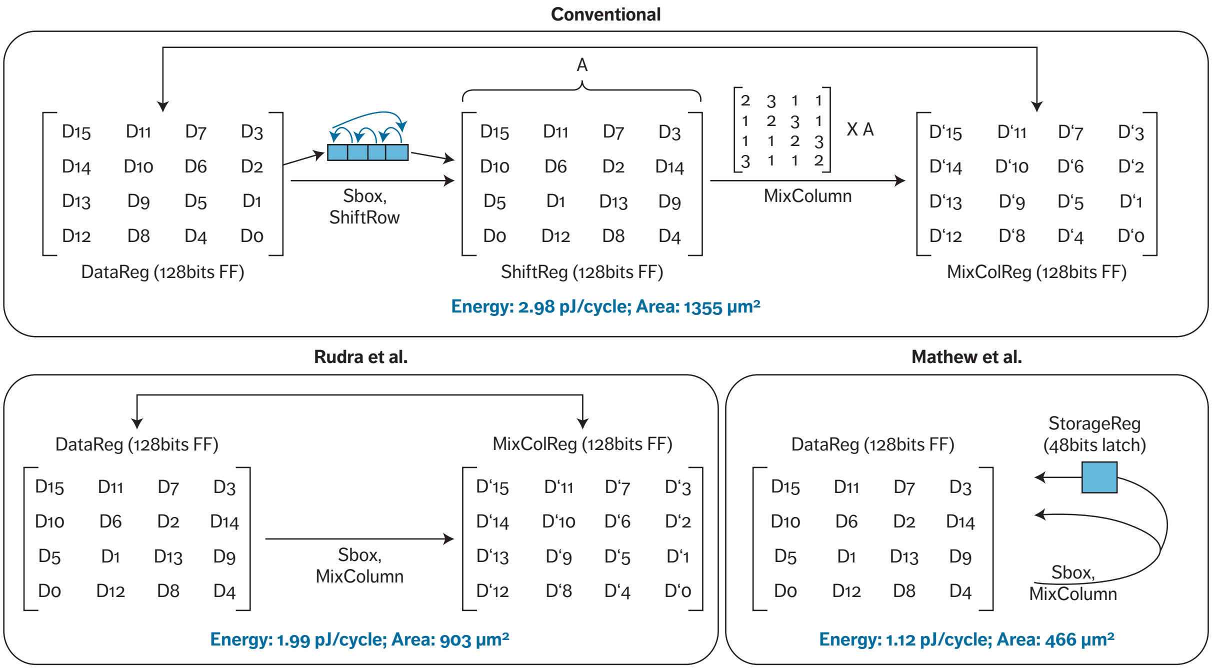


**Figure 4.** Register elimination in the Advanced Encryption Standard (AES) datapath.[8]

Yiqun Zhang and colleagues proposed the latest lightweight AES design,[8] observing that intermediate registers take about 50 percent of total power and area in an 8-bit iterative AES design. They proposed the following techniques to reduce the number of registers (see Figure 4):

- removing ShiftRow, similar to Mathew's work[7];
- reducing MixColumn registers from 128 to 48 bits by rescheduling the data update sequence;
- replacing data, key, and intermediate registers with latches to save area and power; and
- optimizing shift registers by shifting only one-hot addresses instead of all the data registers.

All of these techniques exploit the fixed and known data access patterns of cipher computations and can be applied to other ciphers. Table 1 summarizes the design metrics of the aforementioned AES accelerators.

**Resistance to side-channel attacks.** At the same time, research has shown that physical implementations of secure ciphers can leak information about the secret keys being used for encryption. Researchers have proposed various attack algorithms to break the key from power consumption and EM radiations. Differential power analysis (DPA) is a powerful attack that does not require knowledge about the detailed implementation of the victim hardware.[9] This type of noninvasive attack is a growing concern for IoT devices because adversaries can easily get hold of a device and measure its power and EM information with low-cost devices. This is in contrast to invasive attacks that require expensive equipment to deploy. Therefore, these side-channel attacks are more likely to target low-cost commercial systems that represent the majority of IoT devices.

There are two categories of defense against side-channel attacks: relying on new physical implementations, and using algorithm-level random masking. Although both serve the purpose of reducing the signal-to-noise ratios adversaries can get, they have distinct properties. Physical hiding can be evaluated only heuristically through measurements, but it can be helpful to defend against almost any attack algorithms. On the other hand, algorithmic masking by adding random variables and transforming computations can be provably secure

Table 1. Performance summary of state-of-the-art lightweight AES accelerator.

| Design specifications | P. Hamalainen et al. (EUROMICRO 06)[5] | | S. Mathew et al. (JSSC 15)[7] | | Y. Zhang et al. (VLSI 16)[8] | |
|---|---|---|---|---|---|---|
| Technology | 130 nm | | 22 nm | | 40 nm | |
| Voltage (V) | N/A | | 0.9 | 0.43 | 0.9 | 0.47 |
| Power (mW) | 17.98 | 3.9 | 13 | 0.45 | 4.39 | 0.1 |
| Throughput (Mbps) | 232 | 104 | 432 | 83.6 | 494 | 46.2 |
| Efficiency (pJ/b) | 77.5 | 37.5 | 31 | 5.38 | 8.85 | 2.24 |
| No. of gates | 3,200 | 3,900 | 1,947 | | 2,228 | |

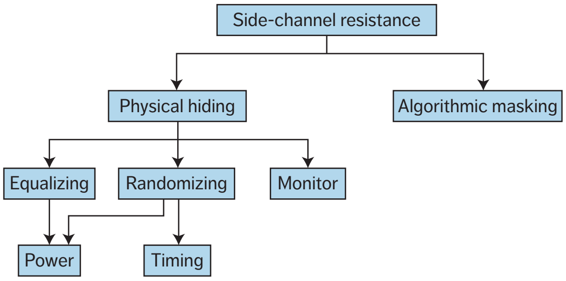


Figure 5. Taxonomy of side-channel defenses.

against certain types of attacks, but the determined nature of the masking algorithms makes them potentially vulnerable to higher-order attacks. Figure 5 shows a taxonomy of defenses. We focus on the progress of physical hiding in this article.

Physically hiding side-channel information leakage has been approached in different ways, including using logic gates that consume the same power for different transitions, a power management unit that randomizes the power consumption seen from outside the chip, and on-chip monitors detecting malicious probing. Ideally, the first approach can have the strongest protection against side-channel attacks by closing the source of side-channel information leakage. However, differential logic cells such as Differential Cascode Voltage Switching Logic and even specially optimized logic gates like Sense Amplifier Based Logic[10] cannot fully equalize the current consumptions of different transitions because of parasitics. Therefore, we can evaluate these designs' security levels only in terms of the signal-to-noise ratio, or by the number of measurements to disclosure of the secret keys. These differential logic gates are also more complicated to design with and consume more than twice the energy compared to conventional CMOS logic gates.

To reduce the power consumption, researchers adopted charge-recycling adiabatic logic, which was originally developed for high-efficiency and use-differential logic states, for resistance to side-channel attacks. The concept was first proposed in 2006,[11] and was demonstrated with a complete silicon implementation until 2015.[12] The results show that the adiabatic AES core requires 200+ times more power traces to find the correct key in a DPA attack, and it consumes only 70 percent of the power, compared to a baseline implementation with standard CMOS logic gates. However, the area overhead is about twice that of the baseline area. Thus, adiabatic logic might be the best option for now to physically hide side-channel leakage. The power overhead can be almost negligible, even compared with optimized AES designs, but the area overhead caused by differential cells and off-chip inductors can prevent their application in low-cost devices.

The second category of defense targets equalizing or randomizing the power

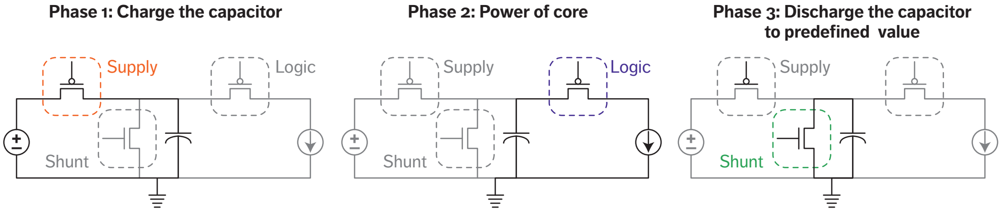

Phase 1: Charge the capacitor
Supply
Logic
Shunt
Phase 2: Power of core
Supply
Logic
Shunt
Phase 3: Discharge the capacitor to predefined value
Supply
Shunt

**Figure 6.** Operating principles of switched-capacitor current equalizer.[13]

consumption measured externally. It aims at defending against adversaries with limited resources and motivations to physically probing the chip for side-channel attacks. As discussed previously, this represents the major scenario that a side-channel attack will be carried out. One of the earliest proposals in this direction involves the use of a switched-capacitor current equalizer (see Figure 6).[13] The equalizer has three phases, controlled by closing one of the three switches to recharge the capacitor, supply power, and discharge the capacitor to a predefined level. Three equalizers work in a staggered fashion to ensure continuous operation of the crypto core. As expected, keys are not disclosed even after 10 million measurements, when only the equalizer's power input is exposed to adversaries. This design incurs 33 percent power overhead and 25 percent area overhead to the baseline. To further reduce the power overhead, a recent effort adds a control loop randomization block into an integrated buck voltage regulator to randomize, instead of equalizing the power drawn from the external source. This design adds a mere 5 percent power overhead and 103 gates area overhead to the baseline while being able to resist Correlation Power Analysis (an improved version of conventional DPA attack) and Test Vector Leakage Assessment.[14]

These defenses are effective only against power side-channel attacks. Researchers have shown that EM radiation can leak as much information and EM probes can even collect localized data to reduce noise.[15] To defend against EM attacks, researchers proposed an EM probe monitor based on a LC oscillator implemented on top of a protected circuit.[16] A coil made by top-layer metal is used as a sensor for EM probes. It is based on the observation that EM probes getting close to the chip surface will reduce the inductance of the coil and therefore can be detected by monitoring the frequency of an LC oscillator built with the coil as L. Calibration and referencing techniques are developed for the monitor to detect probes greater than 0.1 mm away from the chip surface. This EM monitor adds 9,000 $\mu m^2$ area overhead and consumes 17 $\mu W$ in 180-nm CMOS.

## Random Number Generation

Random numbers are critical to cryptographic systems to prevent replay attacks and key guesses. Two types of random number generators are widely used: *pseudorandom number generators* (PRNGs), which use a fixed algorithm and initial random seed to generate a sequence of numbers that can be approximated as random numbers; and *true random number generators* (TRNGs), which harvest entropy from physical noise sources, do not require an initial seed, and do not present any periodicity. Although many PRNGs are designed to be indistinguishable by adversaries from a truly random sequence without knowing the input seed, the seed's security and randomness become a concern in IoT devices because of the limited randomness the device can use (such as user input) and the physical accessibility by attackers. On the other hand, the main issues with TRNGs are high power, high cost, and potential vulnerability to external disturbance and attack.

An intuitive approach to on-chip TRNGs is to amplify resistor thermal noise directly and quantize it into digital bits.[17] However, such a design requires several high-performance analog blocks to mitigate nonideal effects (for example, comparator offset or reference variation) that lead to biased, low-entropy outputs. These designs generally consume higher static power, occupy larger area, and have less technology portability. Therefore, recent research

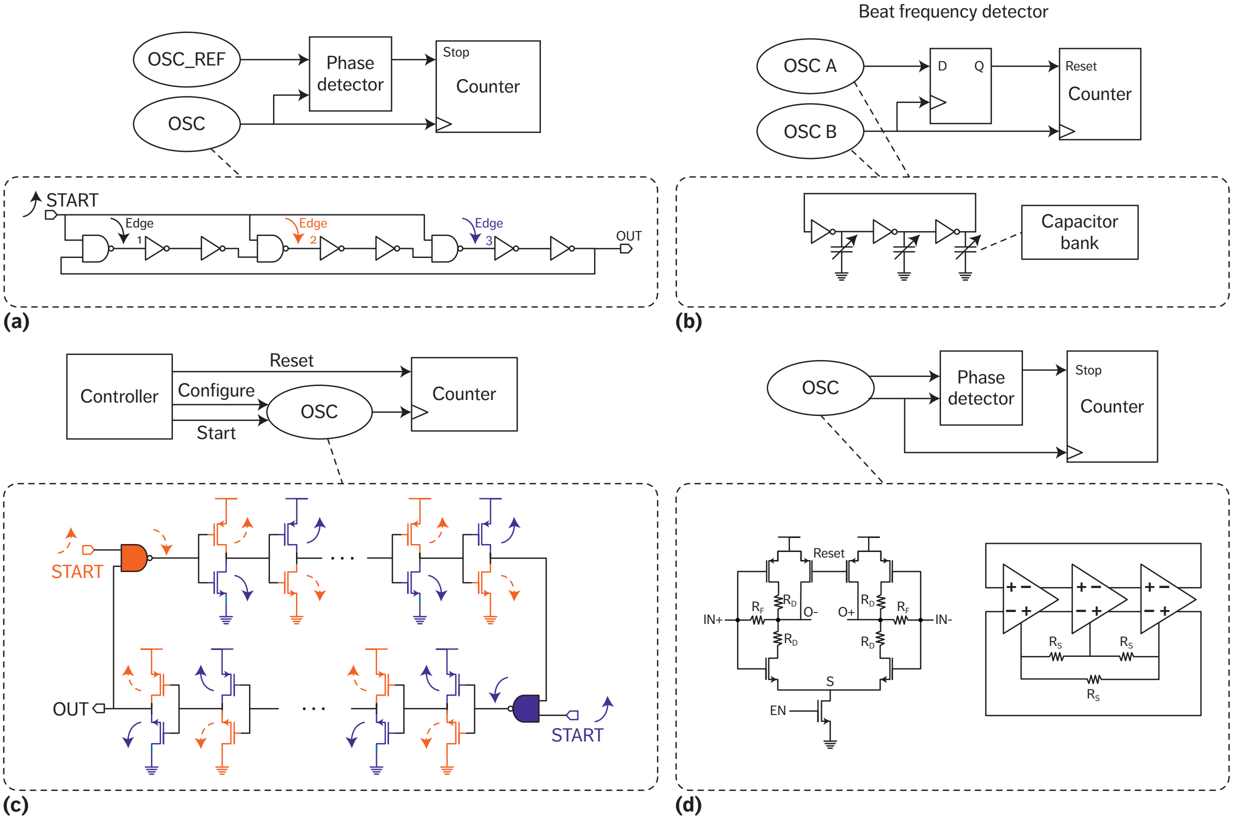


Figure 7. Simplified diagrams for state-of-the-art true random number generators (TRNGs), based work by the following authors: (a) K. Yang et al. (ISSCC 14)[23]; (b) Q. Tang et al. (CICC 14)[24]; (c) K. Yang et al. (JSSC 16)[25]; and (d) E. Kim et al. (ISSCC 17).[26]

efforts have focused on digital implementations of TRNG that exploit thermal noise in metastable circuits[18,19] and oscillators.[20] While metastability-based TRNGs offer high speed and efficiency because of the fast transition between metastable and stable states, the transition is easily affected by device variations and environmental conditions so that the generated bits are deterministically biased without complicated postprocessing or calibration steps.[21] Comparatively, oscillator-based TRNGs offer a simpler design to achieve higher raw entropy at the cost of slower speed. They have also been found to be vulnerable to a supply injection attack that injection-locks the oscillator in a TRNG to an external oscillator to reduce its jitter.[22] State-of-the-art TRNGs have focused on improving the speed and efficiency of oscillator-based TRNGs while achieving high-entropy raw outputs[23–26] and providing lightweight quality check and entropy improvement to existing TRNGs.[27]

The keys to further improving TRNGs are to decouple the output bias from process variation and environmental conditions, to automatically stop TRNG operation, and to implement a runtime quality check for TRNGs. This leads to the development of edge-chasing TRNGs that use the phase differences of multiple oscillations in one or multiple oscillators for random number generation. Kaiyuan Yang and colleagues introduced this concept in 2014,[23] in which three edges were injected into the same ring oscillator to oscillate independently, as shown in Figure 7a. In these designs, all three oscillations have exactly the same frequency because they happen in the same physical oscillator, but they have different phases because of their initial phase and independently accumulated noise. The fluctuation of phase differences among the three oscillations is determined by random noise and accumulated over time. Therefore, given enough time, two of the three oscillations will meet and cancel each other because of their opposite phase. The time it takes for this first-hit-and-collapse event to happen is decided by random noise and therefore used as the proposed TRNG's entropy source. To achieve a uniform distribution of output bits, it has been shown that if the first-hit-and-collapse time is quantized into small enough bins, the least significant bits of the time can be a good approximation to a uniform distribution. Various designs have adopted similar techniques over the years to convert a distribution of time into uniform bits.[23–26] Because this design is not affected by process variations,[23] it can be synthesized with commercial standard cells and placement-and-routing tools (with manually defined rules) and was verified in both 65-nm and 28-nm CMOS technologies. However, this design does not provide runtime quality check and self-tuning capabilities to avoid entropy degradation and denial-of-service attacks caused by supply injection at certain frequencies. The authors proposed to use low-pass filters to protect the TRNG core against supply injection attacks,[23] which can also be applied to most other designs.

A different design with a similar concept was proposed in 2014.[24] The authors used the chasing time of two oscillations in different oscillators with precalibrated small delay difference. As Figure 7b shows, OSC_A is calibrated during start-up to be slightly faster than OSC_B, and they are started simultaneously until OSC_A runs one more cycle and overtakes OSC_B. In this way, the chasing time is bounded within a smaller range, with the average value decided by the deterministic delay difference, so that the TRNG speed is more constant compared to our previous work[23] and provides a knob for tuning the amount of entropy being accumulated.

Yang and colleagues proposed an improved design[25] that combined these previous approaches[23,24] by integrating two oscillations into one even-stage ring oscillator to save area and power. As Figure 7c shows, the two injected edges travel different paths in the even-stage oscillator, which emulates the two oscillators in Tang's work[24] and avoids the complexity of detecting the chase time, because the two oscillations will cancel each other when they meet. In Tang's work,[24] the calibration of the two oscillators is done with capacitor banks, occupying a relatively large area and offering limited resolution. A different calibration technique is employed by Yang and colleagues,[25] who use the intrinsic process variations for fine-grained tuning. Multiple copies of identically designed delay cells are implemented in parallel, and a random search will go over different configurations to find one that falls into the desired operating range. This approach is simpler and requires less area, but can take more trials to set up than the binary search in Tang's work.[24] Yang also provides an analytical model assuming a normal distribution of jitter,[25] which also applies to Tang's design.[24] The chasing of two oscillations is modeled by a random walk with constant drift, and the first-hit-and-collapse time is shown to follow an inverse Gaussian distribution. Mean and variance of the distribution can also be solved by analytical expressions, which helps researchers understand and optimize these designs. These two values are also directly related to the TRNG's operating conditions, which can be used as monitors of the physical random generation process to improve the TRNG's robustness to environmental variations and even deliberate supply injection attacks. Yang and colleagues describe a runtime calibration loop based on these monitors and experimentally verify its robustness against –40 to 120°C and 0.6 to 0.9 V variations.[25] They also show that supply injection attacks can be monitored and thwarted by retuning the oscillator to run at a different frequency.

Eunhwan Kim and colleagues offer the latest improvement aiming at resistance against supply injection attack and simplified startup process (see Figure 7d).[26] They eliminate the precalibration step[25] by using differential delay cells and forcing the differential paths to start with the same phase. The time for the transition from 0° to the normal 180° phase difference is affected by random noise, and the average time is decided by resistor Rs in Figure 7d, similar to previous work.[24,25] The multiple resistors are the key to the robustness against process variations and power injection attack by adding feedback and limiting oscillation amplitude in delay cells. Table 2 provides a summary of the TRNGs.

Although the raw entropy of TRNGs has been significantly improved, postprocessing algorithms to further improve and guarantee

Table 2. State-of-the-art low-power TRNGs.

| Design specifications | K. Yang et al. (ISSCC 14)[23] | | Q. Tang et al. (CICC 14)[24] | K. Yang et al. (JSSC 16)[25] | E. Kim et al. (ISSCC 17)[26] |
|---|---|---|---|---|---|
| Technology | 28 nm | 65 nm | 65 nm | 40 nm | 65 nm |
| Voltage (V) | 0.9 | 0.9 | 0.8 | 0.9 | 1.08 |
| Power (mW) | 0.54 | 0.046 | 0.13 | 0.046 | 0.289 |
| Throughput (Mbps) | 23.16 | 2.8 | 2 | 2 | 8.2 |
| Efficiency (pJ/b) | 23 | 57 | 66 | 23 (11 at 0.6 V) | 36 |
| Area ($\mu m^2$) | 375 | 960 | 6,000 | 836 | 920 |
| Operating voltage range | N/A | N/A | 0.8 to 1.2 | 0.6 to 1 | 1.08 to 1.44 |
| Resistance to power injection attack | Need filter | Need filter | N/A | Yes | Yes |
| Pretuning | No | No | Yes | Yes | No |

Table 3. Comparison of postprocessing methods for TRNG.

| Design specifications | AES-CBC | SHA-256 | S.K. Mathew et al. (JSSC 16)[27] |
|---|---|---|---|
| Full-entropy throughput | 7 cycles/bit | 7.25 cycles/bit | 8 cycles/bit |
| No. of gates | 32,000 | 19,000 | 4,900 |
| Energy/full-entropy bit | 49 pJ | 34 pJ | 9 pJ |

randomness are of great interest to commercial products to avoid both the potential critical failure of the entropy source and strong physical attacks, and to adopt legacy TRNG designs. According to a National Institute of Standards and Technology recommendation,[28] block ciphers like AES in CBC-MAC mode and certain HMAC functions are ideal for conditioning random bits. For example, the complete TRNG system in Intel CPUs targeting desktop and server applications includes on-chip health and wellness tests and conditioning circuits based on counter-mode AES.[21] However, these designs require significantly larger area and power than the TRNG entropy source core, rendering them unsuitable for ULP IoT and wearable devices.[27] Intel recently developed a lightweight TRNG system employing three independent entropy sources based on metastability and the Barak-Impagliazzo-Wigderson randomness extractor with an 8-bit datapath.[27] The prototype in 14-nm CMOS costs only a fraction of power and area compared to conventional approaches (see Table 3) while maintaining close-to-ideal Shannon entropy and min-entropy across 0.4 to 0.95 V supply variations. The efficiency can be further improved to 3 pJ/bit when operating it at near-threshold 0.4 V.

In summary, a number of novel TRNG designs have been proposed in recent years that achieve better randomness with smaller area, lower power, less complexity, and better design portability. Researchers have also studied intentional attacks such as power supply injection. However, the quality of the generated random bits has been verified only with certain statistical tests; more theoretical analysis and modeling of the designs are expected to facilitate the adoptions of these designs and the development of future low-power, high-entropy TRNGs. At the same time, a plethora of existing randomness extraction algorithms used in cryptography should be studied and optimized for lightweight TRNG conditioning.

## Secret-Key Storage

For authentication and encryption, it is necessary to securely store a digital key on chip. In this section, we describe two types of secret-key storage using nonvolatile memory and PUFs. We present recent progress in the design of both types of key storage.

**Nonvolatile memory.** Conventionally, the secret keys are usually stored in on-chip or stand-alone NVMs, including one-time programmable memory (such as ROM, electronic fuse, and antifuse) and nonvolatile random-access memory (such as electrically erasable programmable read-only memory and flash memory). However, a wide range of invasive (depackaging and probing) and semiinvasive (depackaging only) attacks can be used to read the data stored in these memories. Additionally, most of these memories require extra fabrication steps, which is not desirable for low-cost IoT systems and which cannot scale together with CMOS technologies. In responding to these needs, new memory technologies and designs are introduced for security applications.

In 2017, TSMC reported an antifuse technology using only standard 10-nm FinFET transistors.[29] The data is programmed by gate oxide breakdown during the enrollment phase. Each memory cell comprises just two FinFETs with 0.028 $\mu m^2$ area in 10-nm technology. For side-channel resistance, each bit is stored in two cells with complimentary values so that power consumption during read will not reveal information about the stored keys. This technique also improves the read margin for reliability. This design has overcome most of the security and cost concerns of using conventional NVMs for key storage. The only potential drawbacks are that it cannot destroy its own storage when being tampered, and it can be easily duplicated once a key is exposed. Additionally, it is still vulnerable to certain high-resolution, high-accuracy invasive and semi-invasive attacks, but should be secure enough for most applications.

**Weak PUF for key generation.** In addition to new NVMs, a drastically different key storage method has emerged over the years that relies on hardware-intrinsic process variations to generate and store secret keys. This concept was first proposed for chip identification,[30] but has been renamed as "weak PUFs" as a category of the prevailing PUF concept[2] and increasingly used for security. Because the keys are not stored in digital formats and are sensitive to invasive attacks, they are believed to be more secure than conventional key storage solutions. At the same time, PUFs are completely designed with CMOS transistors so that they can benefit from technology scaling and be easily migrated to different technologies, ranging from cost-sensitive to high-performance applications.

Almost all weak PUFs are designed with a differential structure to generate a response by comparing the characteristics of a differential pair, such as the voltage, current, and delay. Because process variation follows a normal distribution, PUFs are likely to have unreliable responses when the difference between the two arms is small. To solve this problem for security applications, helper data is generated during the initial enrollment phase and is used by the reproduction unit to recover the correct response in subsequent authentication sessions. Jeroen Delvaux and colleagues provide an in-depth overview of helper data algorithms to ensure reproducibility and uniformity of PUF key generation.[31] By replacing NVM in Figure 2 with weak PUF and a reproduction unit, we can achieve authentication using weak PUF. Although error-correction codes are widely used to ensure reliable key generation,[32–34] the information leakage and power and area costs associated with the correction are not negligible. To push the boundary for optimization, researchers have been building

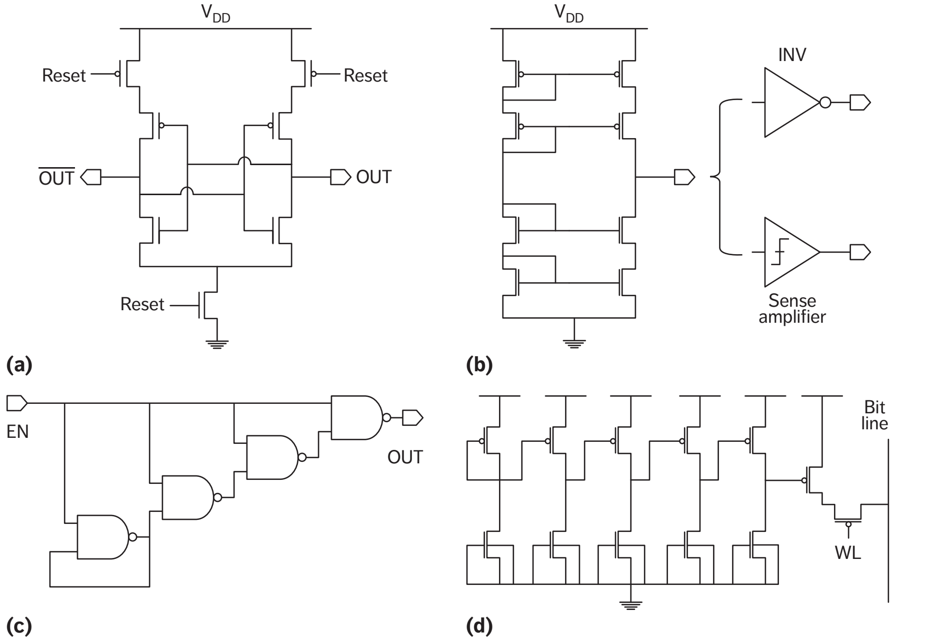


**Figure 8.** Circuit diagrams of state-of-the-art weak physically unclonable functions (PUFs), based on the following work: (a) S. Mathew et al. (ISSCC 14)[33] and Y. Su et al. (JSSC 08)[36]; (b) A. Alvarez et al. (ISSCC 15)[37]; (c) B. Karpinskyy et al. (ISSCC 16)[34]; (d) K. Yang et al. (ISSCC 17).[38]

custom PUF cells that outperform conventional designs using static RAM (SRAM) and oscillators in every aspect. This section presents some state-of-the-art custom PUF cell designs that can potentially alleviate the concerns about unreliable PUF responses.

Output reproducibility across process, voltage, and temperature (PVT) variations and density of the array are two critical metrics directly related to the security and cost of a PUF. The most popular PUF designs use the random power-up state of standard SRAM array[35] because SRAM IPs are widely available and already included in many systems. However, some off-the-shelf SRAMs are biased toward the "1" state,[35] and therefore require postprocessing to improve uniformity. On the basis of the same working principle as SRAM, researchers have designed custom PUF cells based on cross-coupled inverters,[33,36] which include reset switches that bring the structure to a metastable state for evaluation (see Figure 8a). Without further postprocessing techniques, these custom PUF cells do not achieve much better stability (around 6 to 8 percent bit-error rate [BER] at nominal condition) and occupy a larger cell area. However, they don't require manipulation of the power rail and can save a significant amount of power.

Recent custom PUFs with new circuit structures significantly improve the reproducibility and uniformity of native PUF outputs while saving area and power. In 2015, Anastacia Alvarez and colleagues presented a current mirror-based PUF cell (see Figure 8b).[37] Having a static operation and local quantization to PUF output greatly suppresses the noise effects on bit reproducibility, achieving around a 0.3 percent BER (that is, more than 20 times improvement over the SRAM PUF), but at the cost of a large cell. To further reduce area and improve reproducibility, Bohdan Karpinskyy and colleagues introduced a PUF cell using serially connected NAND gates.[34] As Figure 8c shows, the first stage has input and output pins shorted, which forces its output to stay at the transition voltage of the NAND gate. This design does not have an explicit differential structure, but the comparison happens between the transition voltage of the first and second stage. The large gain of the NAND gate around the transition voltage is used to reliably amplify their difference to digital PUF outputs. However, this PUF cell is sensitive to supply voltage and could experience

Table 4. State-of-the-art weak PUFs.

| Design specifications | | S. Mathew et al. (ISSCC 14)[33] | Y. Su et al. (JSSC 08)[36] | A. Alvarez et al. (ISSCC 15)[37] | B. Karpinskyy et al. (ISSCC 16)[34] | K. Yang et al. (ISSCC 17)[38] |
|---|---|---|---|---|---|---|
| Technology | | 130 nm | 22 nm | 65 nm | 45 nm | 180 nm |
| PUF cell area/bit ($F^2$) | | 1,092 | 9,628 | 6,036 | 2,613 | 553 |
| Total area/bit ($F^2$) | | 1,767 | N/A | ~36,450 | N/A | 843 |
| Native unstable bits (no. of evaluations) | | N/A | 30% (5,000) | 1.73% (400) | N/A | 1.67% (2,000) |
| Bit-error rates (nominal condition) (%) | | 3.04 | 8.3<br>0.97* | N/A | 0.1† | 0.13 |
| Tested operating conditions | Temperature (°C) | 0 to 80 | 25 to 50 | 25 to 85 | −25 to 85 | −40 to 120 |
| | Supply (V) | 0.9 to 1.2 | 0.7 to 0.9 | 0.7 to 1 | N/A | 0.8 to 1.8 |
| Bit errors per 10°C (%) | | 0.68 | N/A | 0.47 | 0.15 | 0.2 |
| Bit errors per 0.1 V (%) | | 1.82 | 0.49* | 1.27 | N/A | 0.2 |
| PUF core energy (fJ/bit) | | 930 | 13 | 15 | N/A | 11.3 at 1.2 V<br>1.51 at 0.8 V |
| Normal inter-PUF Hamming distance | | 0.506 | ~0.49 | 0.5014 | 0.498 | 0.499 |

** After stabilizing techniques including burn-in, 15-bit temporal majority voting, and dark bits masking. † With 2-bit glitch detector to remove.*

a large short-circuit current during operation. The latest PUF design extends Karpinskyy's ideas by replacing the NAND gates with two-transistor amplifiers (see Figure 8d).[38] This amplifier is biased at the deep subthreshold region to achieve a very high gain (more than 40) with ultra-low power consumption (about 5 pW). Similar to Karpinskyy's work,[34] the difference of switching voltages between the first and second stages is amplified by four cascading two-transistor amplifiers. The designers add two more transistors the same as the 8T SRAM cell's read port for reading, so that the PUF can be arranged in a crossbar array for maximum density, throughput, and efficiency. Table 4 provides a summary of the PUFs' design specs.

Although the advances in PUF cell designs help release the burden on error correction and uniformity, these postprocessing techniques are indispensable. At the same time, researchers have shown that with advanced backside imaging systems and Focused Ion Beams, PUF outputs of 600 nm SRAM cells can be read and edited by adversaries with standard university failure analysis equipment.[39] There is no theoretical barrier to apply the same attack to the more-advanced silicon PUFs mentioned here, although the ultra-low power consumption can potentially increase the requirements on imaging.[37,38] Even though this cloning attack is demonstrated with only 600-nm PUF, researchers should rethink the physical security of PUFs, especially when compared with new NVMs like the 10-nm antifuse discussed earlier. The main difference between the PUFs we studied and NVMs is the former's volatile nature, which is contradictory to reproducibility and intentionally removed in NVMs. For future weak PUF–based solutions, efforts should be made not only on lower costs with reliable operation, but also on capabilities to detect physical attacks actively or passively and respond to these attacks by using the volatile nature.

## Hardware Designs for PUF-Based Entity Authentication

PUFs are usually categorized into strong and weak PUFs.[40] Both types of PUFs can be modeled as a challenge-and-response function. The difference between them is related to the scalability of the function. Weak PUFs usually have a challenge space linearly related to their area so that only a limited number of challenge-and-response mappings is possible in practice. Strong PUFs, on the other hand, provide a large challenge space that usually exponentially increases with PUF area, and therefore a huge number of mappings can be generated. Because of a large number of random mappings, strong PUFs can be used in challenge-response protocols for authentication as well as for key generation, whereas a weak PUF can be used only for reusable secret keys. This section focuses on using strong PUFs for lightweight authentications.

If the challenge-and-response mappings are truly random, strong PUFs are ideal for secure and lightweight authentication for ULP devices. However, the limited number of random variables in strong PUF circuits and a relatively simple combination of these random variables cannot remove the correlations between different challenge and response pairs. This issue was first envisioned in the original silicon PUF proposal[1] and later proved to be a very effective attack against the most popular arbiter-based strong PUF.[41] Figure 7 shows the arbiter-based PUF,[42] which uses two delay lines with *N* multiplexers in between to reconfigure the two delay paths. The *N* inputs to multiplexers are used as challenges to the PUF, and the racing of the two delay paths (judged by an arbiter) is used for PUF outputs. This design proposes an effective method to create an exponential challenge space with limited resources. However, because the challenge-and-response function can be approximated with a linear model, machine learning algorithms like linear regression and evolution strategies can easily find the random variables in PUF with a few challenge-response pairs (CRPs).[41] To defend against modeling attack and overcome other non-idealities in existing strong PUFs, researchers have suggested several authentication protocols using strong PUFs. Delvaux and colleagues present an excellent description and comparison of 19 strong PUF protocols in literature.[3] Eight of them are identified as promising solutions and categorized into two groups. Protocols in the first group work in a similar fashion as weak PUF-based authentication (see Figure 3). Cryptographic computation is still required to improve security, and all the strong PUF versions can be simplified to weak PUF versions. Researchers claim that strong PUFs are more secure against physical attacks, because a modeling attack is required to duplicate the device. This requires a longer attack time, but the strong PUF versions are not theoretically more secure than the weak PUF versions. The burden returns to the strong PUF implementation. The second group of potential protocols (PUF obfuscation) is closer to the original challenge-and-response PUF proposal[2] and keeps the lightweight property by eliminating cryptographic primitives. To satisfy these requirements, slender PUF,[43] noise bifurcation PUF,[44] and lockdown protocol[45] all require the use of a benign model of the strong PUF stored in the server and a TRNG on chip. For these protocols, the design of PUF circuits is even more complicated, because they require modeling by the owner and resistance to modeling attacks. This is possible only by using compound PUFs that have access to the internal simple PUFs during initial enrollment.

As you can see, protocol designs solve only part of the problem; better strong PUF designs are necessary to complement the protocols and help achieve better protection against practical attackers. The fundamental contradiction and tradeoff in strong PUFs are between complexity of the function and reproducibility. When the function is more complicated and nonlinear, a small perturbation of the random variables will significantly change the final response. This seems an impossible mission, because only a combination of them can achieve strong security—for example, a reproducible NVM key and complicated cryptographic primitives (assuming no physical attacks). Fortunately, we can optimize the PUF design in two directions following the two groups of authentication protocols.

### Reproducible but Learnable PUFs

One direction is following the second group by constructing a compound PUF. Researchers have shown both empirically[41] and theoretically[46] that XORing the outputs of enough independent learnable PUFs can make the computing requirement intractable for practical attackers. However, the noise associated with each PUF is accumulated in the XOR PUF, which limits the number of PUFs that can be XORed. According to the accurate modeling of the BER of the XOR PUF (equation 6 in work by Meng-Day Yu and colleagues[45]), it can be approximated as the number of PUFs multiplying the BER of each PUF, when the BER of each PUF is small. Therefore, by reducing the BER of a single PUF to half, the number of PUFs can be doubled in XOR PUF while keeping the same false-acceptance rate and false-rejection rate. In addition, the time and training data required to perform a modeling attack on XOR PUF is growing exponentially with the number of XORs. Thus, the goal here is to improve the BER of a single PUF, not considering the complexity of the mapping function.

A recent progress in this direction replaces the delay lines in an arbiter PUF with a ring oscillator.[47] Delay cells are configurable by input challenges to create a large challenge space. Similar to the edge-chasing TRNG,[25] two edges are inserted to opposite positions of an even-stage oscillator and chase each other. During this process, the mismatch between them is increasing linearly with time, while noise is increasing as a square root function of time. In this way, the PUF's entropy source (process variation) is amplified relative to noise to achieve better reproducibility. In addition, the time for the chasing to finish can be measured by a counter and used to indicate the amount of mismatch between the two edges. This in-situ monitor can accurately monitor the PUF under a varying environment and make decisions about the confidence of this specific CRP. By using it, unreliable responses can be excluded in runtime to significantly reduce the BER. The measurement results of a 40-nm prototype show that the BER can be reduced to less than $10^{-8}$ when 30 percent of the CRPs are discarded.[47] To further lower power, improve efficiency, and improve the BER, delay cells are biased at the near-threshold region. Future work in this direction must carefully consider side-channel attacks, which have been shown to be effective against slender PUF, controlled PUF, and XOR PUFs by using reliability of response and power side channels.[48,49] Although the concern has been alleviated by enforcing the number of accesses to a PUF in lockdown protocol,[45] other defenses are worth investigating.

### Difficult-to-Learn PUFs

The second direction is related to the first group of strong PUF protocols. Improving a single PUF's resistance to modeling attacks will increase the difficulty of attacking these protocols with combined physical and modeling attacks. However, the reproducibility of these new PUFs must be kept low enough to avoid other problems. Two designs targeting modeling-resistant strong PUFs were published in 2017.[50,51] In the former, reconfigurable subthreshold transistors connected in serial and parallel are used to create a nonlinear mapping.[50] In the latter, challenges are changed to sequences of inputs and the PUF is changed from combinational logic to sequential logic for nonlinearity.[51] The design is based on a commercial 6T SRAM array initialized to its power-up value, similar to power-up SRAM PUF.[35] By sequentially shorting different rows, the final state of the last accessed row depends on all the previously accessed rows and the access sequence. By choosing more rows from an array, a large challenge space can be achieved. Both works show similar resistance to linear regression and SVM-based machine learning attacks with up to 10,000 training data, while keeping comparable BERs compared to conventional SRAM and arbiter PUFs.[50,51] These designs are promising but require more rigorous attacks that are designed with knowledge of the PUF.

## Data Security

The demands for data security can be achieved only by cryptographic primitives, including symmetric key ciphers for encryption, hash functions for integrity, and public key ciphers for signature and key exchange. We have discussed these systems' building blocks, including cipher engines, random number generators, and key storage.

For resource-constrained devices, ASIC accelerators can provide the best possible efficiency, power, and area. Many defenses against side-channel attacks are also easier to integrate together with ASIC implementations.

However, one thing to notice is that there will be a wide range of IoT devices and communication standards in the ecosystem, and therefore flexibility of the security algorithm and protocol is important. This represents a different optimization space compared to a pure ASIC design that can exploit fixed operations. Some recent works propose the use of in-memory computing[52] and a flexible-bit-width Galois Field arithmetic logic unit with SIMD instructions[53] to accelerate the most demanding computation in many cryptographic and even error-correction algorithms. They achieve 5 to 20 times improvement over software implementation and within a few times to state-of-the-art ASIC designs.

ULP devices are expected to support a wide range of new and disruptive applications like the IoT. The power and cost budget and physical attack threats demand new hardware and system designs to ensure the security of these devices. In this article, we discussed the need for better hardware blocks to support entity authentication and data security. We presented a survey of recent hardware designs matching these needs in order to show the state of the art. We also identified open problems and future directions for ULP hardware designs for security. We showed that the selection of protocols and hardware design is strongly dependent on specific applications—for example, systems that already require encryption engines are more suitable for weak PUF-based authentication protocols. Also, certain stereotypes about the physical security of PUFs and NVMs need to be reconsidered and studied because of new attacks and defenses. m■

## References

1. B. Gassend et al., "Silicon Physical Random Functions," *Proc. 9th ACM Conf. Computer and Communications Security*, 2002, pp. 148–160.
2. R. Pappu, "Physical One-Way Functions," *Science*, vol. 297, no. 5589, 2002, pp. 2026–2030.
3. J. Delvaux et al., "A Survey on Lightweight Entity Authentication with Strong PUFs," *ACM Computing Surveys*, vol. 48, no. 2, 2015, p. 26:1–26:42.
4. S.K. Mathew et al., "53 Gbps Native Composite-Field AES-Encrypt/Decrypt Accelerator for Content-Protection in 45 nm High-Performance Microprocessors," *IEEE J. Solid-State Circuits*, vol. 46, no. 4, 2011, pp. 767–776.
5. P. Hamalainen et al., "Design and Implementation of Low-Area and Low-Power AES Encryption Hardware Core," *Proc. 9th EUROMICRO Conf. Digital System Design*, 2006, pp. 577–583.
6. A. Rudra et al., "Efficient Rijndael Encryption Implementation with Composite Field Arithmetic," *Cryptographic Hardware and Embedded Systems*, Ç.K. Koç, D. Naccache, and C. Paar, eds., Springer, 2001, pp. 171–184.
7. S. Mathew et al., "340mV-1.1V, 289 Gbps/W, 2090-Gate NanoAES Hardware Accelerator with Area-Optimized Encrypt/Decrypt GF(2^4)^2 Polynomials in 22 nm Tri-Gate CMOS," *IEEE J. Solid-State Circuits*, vol. 50, no. 4, 2015, pp. 1048–1058.
8. Y. Zhang et al., "A Compact 446 Gbps/W AES Accelerator for Mobile SoC and IoT in 40nm," *Proc. IEEE Symp. VLSI Circuits*, 2016.
9. P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," *Advances in Cryptology*, 1999, pp. 388–397.
10. K. Tiri, M. Akmal, and I. Verbauwhede, "A Dynamic and Differential CMOS Logic with Signal Independent Power Consumption to Withstand Differential Power Analysis on Smart Cards," *Proc. 28th European Solid-State Circuits Conf.*, 2002, pp. 403–406.
11. M. Khatir et al., "A Secure and Low-Energy Logic Style using Charge Recovery Approach," *Proc. ACM/IEEE Int'l Symp. Low Power Electronics and Design*, 2008, pp. 259–264.
12. S. Lu, Z. Zhang, and M. Papaefthymiou, "1.32GHz High-Throughput Charge-Recovery AES Core with Resistance to DPA Attacks," *Proc. Symp. VLSI Circuits*, 2015, pp. C246–C247.

13. C. Tokunaga and D. Blaauw, "Secure AES Engine with a Local Switched-Capacitor Current Equalizer," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2009, p. 64–65,65a.
14. B.J. Gilbert Goodwill et al., "A Testing Methodology for Side-Channel Resistance Validation," *Proc. Non-invasive Attack Testing Workshop*, 2011.
15. K. Gandolfi, C. Mourtel, and F. Olivier, "Electromagnetic Analysis: Concrete Results," *Cryptographic Hardware and Embedded Systems*, 2001, pp. 251–261.
16. N. Miura et al., "A Local EM-Analysis Attack Resistant Cryptographic Engine with Fully-Digital Oscillator-Based Tamper-Access Sensor," *Proc. Symp. VLSI Circuits*, 2014.
17. C.S. Petrie and J.A. Connelly, "A Noise-Based IC Random Number Generator for Applications in Cryptography," *IEEE Trans. Circuits and Systems I: Fundamental Theory and Applications*, vol. 47, no. 5, 2000, pp. 615–621.
18. S.K. Mathew et al., "2.4 Gbps, 7 mW All-Digital PVT-Variation Tolerant True Random Number Generator for 45 nm CMOS High-Performance Microprocessors," *IEEE J. Solid-State Circuits*, vol. 47, no. 11, 2012, pp. 2807–2821.
19. C. Tokunaga, D. Blaauw, and T. Mudge, "True Random Number Generator with a Metastability-Based Quality Control," *IEEE J. Solid-State Circuits*, vol. 43, no. 1, 2008, pp. 78–85.
20. M. Bucci et al., "A High-Speed Oscillator-Based Truly Random Number Source for Cryptographic Applications on a Smart Card IC," *IEEE Trans. Computers*, vol. 52, no. 4, 2003, pp. 403–409.
21. M. Hamburg, P. Kocher, and M.E. Marson, *Analysis of Intel's Ivy Bridge Digital Random Number Generator*, tech. report, Cryptography Research, 2012.
22. A.T. Markettos and S.W. Moore, "The Frequency Injection Attack on Ring-Oscillator-Based True Random Number Generators," *Cryptographic Hardware and Embedded Systems*, Springer, 2009, pp. 317–331.
23. K. Yang et al., "A 23Mb/s 23pJ/b Fully Synthesized True-Random-Number Generator in 28nm and 65nm CMOS," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2014, pp. 280–281.
24. Q. Tang et al., "True Random Number Generator Circuits Based on Single- and Multi-phase Beat Frequency Detection," *Proc. IEEE Custom Integrated Circuits Conf.*, 2014, pp. 1–4.
25. K. Yang, D. Blaauw, and D. Sylvester, "An All-Digital Edge Racing True Random Number Generator Robust Against PVT Variations," *IEEE J. Solid-State Circuits*, vol. 51, no. 4, 2016, pp. 1022–1031.
26. E. Kim, M. Lee, and J.J. Kim, "8Mb/s 28Mb/mJ Robust True-Random-Number Generator in 65nm CMOS based on Differential Ring Oscillator with Feedback Resistors," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2017, pp. 144–145.
27. S.K. Mathew et al., "μRNG: A 300–950 mV, 323 Gbps/W All-Digital Full-Entropy True Random Number Generator in 14 nm FinFET CMOS," *IEEE J. Solid-State Circuits*, vol. 51, no. 7, 2016, pp. 1695–1704.
28. M.S. Turan et al., *Recommendation for the Entropy Sources Used for Random Bit Generation*, report 800-90B, Nat'l Inst. Standards and Technology, 2016.
29. S.Y. Chou et al., "A 10 nm 32Kb Low-Voltage Logic-Compatible Anti-fuse One-Time-Programmable Memory with Anti-tampering Sensing Scheme," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2017, pp. 200–201.
30. K. Lofstrom, W.R. Daasch, and D. Taylor, "IC Identification Circuit using Device Mismatch," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2000, pp. 372–373.
31. J. Delvaux et al., "Helper Data Algorithms for PUF-Based Key Generation: Overview and Analysis," *IEEE Trans. Computer-Aided Design of Integrated Circuits Systems*, vol. 34, no. 6, 2015, pp. 889–902.
32. R. Maes, A.V. Herrewege, and I. Verbauwhede, "PUFKY: A Fully Functional PUF-Based Cryptographic Key Generator," *Cryptographic Hardware and Embedded Systems*, 2012, pp. 302–319.
33. S.K. Mathew et al., "A 0.19pJ/b PVT-Variation-Tolerant Hybrid Physically Unclonable Function Circuit for

100% Stable Secure Key Generation in 22nm CMOS," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2014, pp. 278–279.

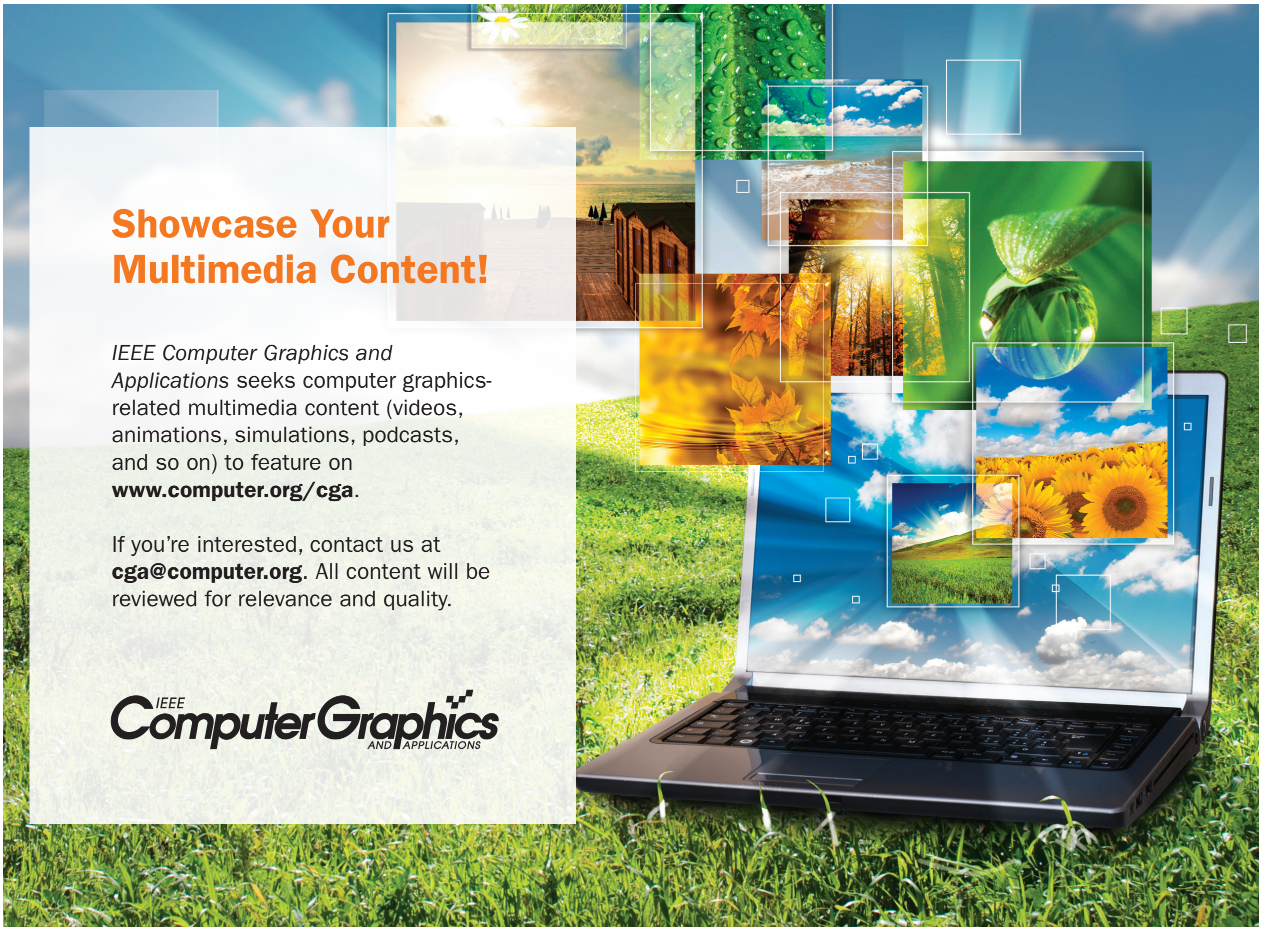
34. B. Karpinskyy et al., "Physically Unclonable Function for Secure Key Generation with a Key Error Rate of 2E-38 in 45 nm Smart-Card Chips," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2016, pp. 158–160.
35. D.E. Holcomb, W.P. Burleson, and K. Fu, "Power-Up SRAM State as an Identifying Fingerprint and Source of True Random Numbers," *IEEE Trans. Computers*, vol. 58, no. 9, 2009, pp. 1198–1210.
36. Y. Su, J. Holleman, and B.P. Otis, "A Digital 1.6 pJ/bit Chip Identification Circuit Using Process Variations," *IEEE J. Solid-State Circuits*, vol. 43, no. 1, 2008, pp. 69–77.
37. A. Alvarez, W. Zhao, and M. Alioto, "15fJ/b Static Physically Unclonable Functions for Secure Chip Identification with <2% Native Bit Instability and 140x Inter/Intra PUF Hamming Distance Separation in 65nm," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2015, pp. 256–257.
38. K. Yang et al., "A 553F$^2$ 2-Transistor Amplifier-Based Physically Unclonable Function (PUF) with 1.67% Native Instability," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2017, pp. 146–147.
39. C. Helfmeier et al., "Cloning Physically Unclonable Functions," *Proc. IEEE Int'l Symp. Hardware-Oriented Security and Trust*, 2013, pp. 1–6.
40. C. Herder et al., "Physical Unclonable Functions and Applications: A Tutorial," *Proc. IEEE*, vol. 102, no. 8, 2014, pp. 1126–1141.
41. U. Rührmair et al., "Modeling Attacks on Physical Unclonable Functions," *Proc. 17th ACM Conf. Computer and Communications Security*, 2010, pp. 237–249.
42. J.W. Lee et al., "A Technique to Build a Secret Key in Integrated Circuits for Identification and Authentication Applications," *Proc. Symp. VLSI Circuits*, 2004, pp. 176–179.
43. M. Majzoobi et al., "Slender PUF Protocol: A Lightweight, Robust, and Secure Authentication by Substring Matching," *Proc. IEEE Symp. Security and Privacy*, 2012, pp. 33–44.
44. M.-D. Yu et al., "A Noise Bifurcation Architecture for Linear Additive Physical Functions," *Proc. IEEE Int'l Symp. Hardware-Oriented Security and Trust*, 2014, pp. 124–129.
45. M.D. Yu et al., "A Lockdown Technique to Prevent Machine Learning on PUFs for Lightweight Authentication," *IEEE Trans. Multi-Scale Computer Systems*, vol. 2, no. 3, 2016, pp. 146–159.
46. F. Ganji, S. Tajik, and J.-P. Seifert, "Why Attackers Win: On the Learnability of XOR Arbiter PUFs," *Trust and Trustworthy Computing*, 2015, pp. 22–39.
47. K. Yang et al., "A Physically Unclonable Function with BER $<10^{-8}$ for Robust Chip Authentication using Oscillator Collapse in 40nm CMOS," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2015, pp. 254–255.
48. G.T. Becker, "On the Pitfalls of Using Arbiter-PUFs as Building Blocks," *IEEE Trans. Computer-Aided Design of Integrated Circuits Systems*, vol. 34, no. 8, 2015, pp. 1295–1307.
49. G.T. Becker, "The Gap Between Promise and Reality: On the Insecurity of XOR Arbiter PUFs," *Cryptographic Hardware and Embedded Systems*, 2015, pp. 535–555.
50. X. Xi et al., "Strong Subthreshold Current Array PUF with 265 Challenge-Response Pairs Resilient to Machine Learning Attacks in 130nm CMOS," *Proc. IEEE Symp. VLSI Circuits*, 2017, pp. C268–C269.
51. S. Jeloka et al., "A Sequence Dependent Challenge-Response PUF using 28nm SRAM 6T Bit Cell," *Proc. IEEE Symp. VLSI Circuits*, 2017, pp. C270–C271.
52. Y. Zhang et al., "Recryptor: A Reconfigurable In-Memory Cryptographic Cortex-M0 Processor for IoT," *Proc. IEEE Symp. VLSI Circuits*, 2017, pp. C264–C265.
53. Y. Chen et al., "A Programmable Galois Field Processor for the Internet of Things," *Proc. 44th Ann. Int'l Symp. Computer Architecture*, 2017, pp. 55–68.

**Kaiyuan Yang** is an assistant professor in the Department of Electrical and Computer

Engineering at Rice University. His research interests include energy-efficient integrated circuit and system design and hardware security. Yang received a PhD in electrical engineering from the University of Michigan, Ann Arbor. He is a member of IEEE. Contact him at kyang@rice.edu.

**David Blaauw** is a professor in the Department of Electrical Engineering and Computer Science at the University of Michigan. His research interests include the design of millimeter-scale computing systems and energy-efficient near-threshold computing. Blaauw received a PhD in computer science from the University of Illinois at Urbana-Champaign. He is an IEEE Fellow. Contact him at blaauw@umich.edu.

**Dennis Sylvester** is a professor in the Department of Electrical Engineering and Computer Science at the University of Michigan. His research interests include the design of millimeter-scale computing systems and energy-efficient near-threshold computing. Sylvester received a PhD in electrical engineering from the University of California, Berkeley. He is a cofounder of Ambiq Micro, a fabless semiconductor company developing ultra-low-power mixed-signal solutions for compact wireless devices. He is an IEEE Fellow. Contact him at dmcs@umich.edu.



# 2017 International Symposium on Computer Architecture Influential Paper Award

**David Brooks**
*Harvard University*

**The International Symposium on** Computer Architecture (ISCA) has a tradition of awarding the ACM SIGARCH/IEEE-CS TCCA Influential ISCA Paper Award at the conference each year. This award is conferred on the authors of a paper from the ISCA conference that occurred 15 years prior and which had a substantial impact on the field in terms of research impact and/or industrial influence. The selection process starts by soliciting nominations by members of the current year's ISCA Program Committee. The top papers are then voted on by the full PC (excepting conflicts). The results of this vote are conveyed to a selection committee comprising the current ISCA PC Chair (David Brooks), the ACM Special Interest Group on Computer Architecture (SIGARCH) Chair (Sarita Adve), and the IEEE Computer Society Technical Committee on Computer Architecture (TCCA) Chair (Dean Tullsen). The award includes an honorarium for the authors and a certificate.

At ISCA 2017, the award was presented to the authors of the ISCA paper published in 2002 titled "Drowsy Caches: Simple Techniques for Reducing Leakage Power." The paper was written by Krisztián Flautner, Nam Sung Kim, Steven M. Martin, David Blaauw, and Trevor N. Mudge. At the time of publication in 2002, the computer architecture community was beginning to realize that power concerns would be a major problem for future high-performance and mobile microprocessors. However, most attention was focused on dynamic power consumption, rather than the static, or leakage, power that was growing in importance. In fact, at the time it was projected that leakage power would dominate total power consumption below the 90-nm technology node. "Drowsy Caches" was one of the first papers to address this growing problem area.

Like most influential papers in computer architecture, the key idea of the Drowsy Cache is simple. The authors observe that for fixed periods of time, most cache accesses occur on a small subset of cache lines. The Drowsy Cache is designed to take advantage of this property by splitting the cache lines into active and drowsy states. In the active mode, the cache line can be accessed as normal. In the drowsy mode, the supply voltage to the cache line is reduced to the point where the leakage current is significantly reduced, but the voltage is maintained at a level that allows data retention. A small performance hit is incurred when moving between the drowsy and active states, so the paper proposes architectural policy mechanisms that can be implemented to move lines between the states. The paper shows that with simple policy mechanisms, up to 90 percent of the cache lines can be in the drowsy state without impacting the overall performance by more than 1 percent. The drowsy approach contrasts with prior Gated $V_{DD}$ techniques that turn off cache lines completely, resulting in state loss and the need to fetch the data from lower levels of the memory hierarchy.

A groundbreaking aspect of the paper was the strong collaborative effort between computer architecture and circuit design. This is reflected both in the list of authors and the Drowsy Cache design itself. Memory circuits are notoriously difficult to design because of the tradeoff between array density and susceptibility to process variation and on-chip noise. Thus, one concern with the Drowsy Cache approach is that the techniques needed to create the drowsy mode would be unreliable or require significant chip area. The paper provides comprehensive circuit diagrams to explain how the memory circuits need to be modified to support drowsy operation. The paper also includes detailed HSPICE simulations demonstrating cross-talk analysis of internal nodes of the memory and the expected leakage savings benefits. At the same time, the previously proposed Gated $V_{DD}$ techniques required somewhat complex control algorithms to maintain correctness due

0272-1732/17/$33.00 © 2017 IEEE

to the loss of state from completely disabling the cache lines. The Drowsy Cache paper describes a relatively simple architectural policy mechanism and evaluates the overall energy savings and performance impact on both in-order and out-of-order microprocessor cores using state-of-the-art architectural simulation approaches across a range of benchmarks. In this regard, the paper is a model for researchers working at the interface between computer architecture and circuit design.

The Drowsy Caches paper has had a substantial impact on the research community and has been cited more than 1,000 times as of September 2017. One can also see the influence of the Drowsy Cache work in modern microprocessors that implement aggressive power optimizations in the cache hierarchy. For example, the Intel Xeon Processor 7100 includes leakage power management in the L3 cache design. The design uses sleep transistors that allow fine-grained control of leakage power in the cache subarray blocks with wake-up counters that can be programmed to balance switching and leakage power. Clearly, the Drowsy Cache paper has withstood the test of time and is a worthy recipient of the 2017 SIGARCH/TCCA Influential Paper Award.

**David Brooks** is the Haley Family Professor of Computer Science at Harvard University. Contact him at dbrooks@eecs.harvard.edu.

# The Hush-Hush Norm

**Shane Greenstein**
*Harvard Business School*

**Mainstream writers do not** discuss online sex and porn for fear of touching unseen landmines that offend readers. It is part of a phenomenon that I call the *hush-hush norm*. There are permeable boundaries between rebellious and mainstream hackers, and between porn and mainstream content providers. Yet, the mainstream press discusses all of it as if sex and porn do not exist.

That is, until recently. Some crusading US lawmakers introduced legislation for amending the Communications Decency Act, aiming at interrupting activities that enable human trafficking. The changes aspire to place more responsibility on those who host content. While well-meaning, the effort upends carefully calibrated understandings at many online firms, who fear unintended consequences.

This is a new effort in an old debate in policy or business circles. For years, debates were tied up in abstract knots, dominated by lawyers with an interest in the nuances of free speech and censorship and the legal boundaries of questionable behavior.

Why act now? Because, as any Internet denizen knows, some corners of the Internet have grown more salacious, vulgar, and boorish. Just talk to any parent. It is too easy for children's curiosity to lead them to the sleazy online square, and every parent now worries whether a child has enough sense to handle a disingenuous text. What is a parent to do—keep them off YouTube for fear of much worse?

Look, here is where I am going. I have occasionally listened to these debates and, as a market analyst, noticed the lack of economics. Specifically, a range of economic institutions grew up around the hush-hush norm. The norm served one purpose years ago, and today it serves another. Although the hush-hush norm got us into this mess, it will not get us out. Its role needs to be identified and brought to light so that appropriate actions get taken now.

In case it is not obvious, those last few paragraphs serve as a warning. The content of this column is not suitable for children or, for that matter, Puritans. And one more warning: this column will have failed if some part of this situation does not make you angry.

## The Gray Zone

Start with something obvious: there is a lot of porn on the Internet. It comes in a vast variety of flavors and fantasies and genders. Speaking as a non-lawyer, and merely as a rule of thumb, most porn is legal in the US if a site issues appropriate warnings and stays far away from minors and prostitution.

If the hush-hush norm had not reduced news coverage, market analysts in the past would have said something like this: Online porn competed for sales against salacious VCR tapes, live shows, subscription magazines, and revenue in hotel rooms for "adult entertainment." More recently, and after decades of this competition, the price for online porn is quite low, often free, and it has taken plenty of market share from offline sales.

Data suggests the online market for porn and sex is, at most, a niche market. As part of a research project about surfing, a colleague and I examined online visitor behavior for the top 10,000 most popular US websites in 2008 and 2013, and we could not miss the porn sites. We found online sex comprised approximately 7 to 8 percent of websites, and users spent about 2 to 3 percent of their time on such sites. Also, many households spent no time at such sites.

Those numbers imply two things relevant to today's topic. On the one hand, online sex cannot support much online ad spending. Related, subscriptions can't amount to much money—merely a rounding error on total e-commerce revenue. On the other hand, online sex has to have an outsized influence on the web. There is so much available for web crawlers to find. That means search engines regularly must make decisions about how to classify the activity, and whether to sell ads for it.

0272-1732/17/$33.00 © 2017 IEEE

The hush-hush norm does shape what search engines do. The earliest "pay-for-placement" schemes in search engines did not ignore porn. They tried to make money selling porn providers' ads. That approach made a little money and temporarily raised a lot of attention with investors. However, it did not work out so well: Many of those ads annoyed users, who were uninterested in this niche. The users stopped coming, so did the advertisers, and those sites eventually closed.

As a young firm, Google adopted a policy consistent with the hush-hush norm. It banned ads linked to porn, just as it had banned ads for alcohol, smoking, and gambling. Yet, Google did not ban anything from its organic search service, showing links for anything users clicked. The rationale: Some users wanted those links, just privately, and it was not Google's job to censor. Hush-hush. Eventually, and not trivially, Google also made revenue on ads to those users.

That approach solved one problem but created another, since it did not meet the needs of families. In their book, *How Google Works* (Hachette Book Group, 2014), Eric Schmidt and Jonathan Rosenberg describe developing algorithms to recognize and filter pornography. As it turned out, those filters worked well enough and quelled any call for change.

That basically describes where the US settled in the prior decade. Google's filters seemed to affirm the belief—to which the Valley is predisposed—that clever technology could fix any issue, even with sex.

## The Status Quo

The hush-hush norm prevented a robust public conversation about the status quo. Seth Stephens-Davidowitz's best-selling book *Everybody Lies* (Harper-Collins, 2017) offers a good place to start understanding. The book tried to break through the shroud of nondiscussion. This book won't tell you much about the dark side of sex market. Rather, it focuses on large-scale societal-level patterns by presenting and analyzing the vast range of Google search requests related to private topics.

As it turns out, many users ask Google questions they would never share in public, especially about sex. That reveals a lot about society. Many desire and feel things they do not express publicly. No reader can walk away from the data in this book without realizing that a complex sexual world lives behind the hush-hush norm. More to the point, despite varied and complex private lives, many businesses made money directly or indirectly off porn without ever saying so. That removed pressure to alter what grew up around the norm.

Google's filters seemed to affirm the belief—to which the Valley is predisposed—that clever technology could fix any issue, even with sex.

For many years, online porn has been a small fraction of the hosting, carrier, and content traffic. Firms in those markets often labeled the source as "miscellaneous" on their income statements. Most financial analysts learned to interpret, and everyone simply carried on. Only carriers complained about the situation, particularly when pirated porn clogged capacity. While the complaints had some merit, they also were a bit disingenuous. Plenty of legit porn also clogged capacity at the same time.

Many entrepreneurs also adapted to the norm. Many pitched their firm as if sex did not exist.

There are just too many examples to enumerate, so take this rule of thumb: If a new app or online site had a strong visual or video-sharing capability, and no religious branding, then sometimes the entrepreneur built a sexual angle into the business. You just had to ask about it privately. If there was one, the founder knew what it was, and if not, the founder would say so, too.

This is an explanation, not an excuse. I am not saying this was a good or bad strategy, or morally corrupt or enlightened. My only point: this is how the norm worked. Some tried to make money, and everyone carried on in public as if sex did not exist.

(You might respond that a few years ago Tinder removed all pretense. Yes, they did. Is it a trend? This is a niche market, so I doubt it is. Let's move on.)

Firms also benefited from imitating technical advance in porn. That pattern arose because, long story short, many "lead users" and technological "pioneers" have touched porn. The presence of lead users in porn goes back to VCR tapes and bulletin boards, and more recently, peer-to-peer software, which moved data-intensive salacious videos between users. In the present era, hackers developed innovations in buffering, compression, and rendering of video streaming and applied these innovations to pirated material and, um, sexual services.

How do mainstream firms benefit? Firms have assigned employees to "analyze" technical advance in porn and "borrow" the useful parts, sometimes from open source communities.

To be frank, while others have told me about this "borrowing," I do not know how widespread it is. It might be impossible to ever know. No mainstream firm has ever publicly crowed about learning from these lead users. It is hush-hush, after all.

Nonetheless, the foregoing leads to a sarcastic aside: The next time you watch a great basketball highlight on your browser, try not to think about who performed the test drives for that sharp picture.

## Freedom's Limits

Today we live in a world where porn remains just a click away, and so do virtual red-light districts, as well as activities much worse. Not talking about it just lets problems fester.

First of all, every parent knows the filters have flaws. An airtight filter interferes with browsing. With anything less than airtight, any sufficiently clever teenager can find what they are looking for.

Let me digress with a short editorial right here. Let's not blame technology for human behavior. Clever teenagers found a way before the Internet, too. And I say this as a parent: nothing substitutes for a frank conversation between parents and children. (It is not easy being a parent now, and never has been.)

Notice the root of the problems—namely, technical success. Modern search technology is simply too good at finding everything. There are degrees of sleazy libertine exploitation that never used to be available to a young person's fingertips.

Notice the root of the problems—namely, technical success. Modern search technology is simply too good at finding everything.

In plain language: Search engines make it too damned easy for a young and nontechnical user to find this stuff.

Let's also put this in perspective. While it is not an everyday problem, this is a place where even a little bit is too much. You would not take your child or younger sibling to an X-rated movie, so why tolerate it during Internet surfing?

Let's also recognize why this is a difficult legal problem. For all intents and purposes, adults can exercise freedom. Legal lines need to be drawn, and those are not always bright lines.

Here is a mild example of the issue. There are large numbers of sites for escort services and masseuses. Some are legit, but many merely offer a thin veil on prostitution. Just try explaining this to your child when they run across such a site by accident.

Craigslist's experience illustrates a related problem. For many years, personal ads allegedly served as a home for prostitution, and Craigslist had repeated run-ins with law enforcement. Eventually, Craigslist adopted more restrictive terms of service and banned the illegal ads.

Alas, the results are unsatisfying. Many of the ads in those sections today still are unsuitable for innocent readers, to put it euphemistically. Moreover, much of the illegal explicit activity merely moved elsewhere, such as Backpage, which now receives most of the official ire, allegedly for facilitating prostitution by minors. And Craigslist and Backpage want every parent to keep their child off the site? Uh huh. Good luck with that.

Let's not forget malware, which varies between annoying and destructive. Much originates from porn sites. The hush-hush norm makes this problem more difficult to address. After all, Yelp does not accumulate ratings for porn sites, and it is not about to start a list of bad sites.

Some readers will point out that such lists exist in the security community, and technically adept users know how to act. Yes, but let's be realistic. Most mainstream users are not that adept, and many do not even know how to ask.

It is possible to continue with additional examples of fraud, but for the sake of brevity, let's get to the worst of these examples. I am not entirely certain when or why a few criminals involved in sex trafficking lost all sense of shame and raised the profile on their activities. It did not happen all at once, but—very long story short—it seems to be another example where no good deed went unpunished in technology.

It started with good intentions, as an antidote to crackdowns in repressive regimes. That motivated additional technical advances in protecting privacy—for example, better VPNs and encryption (among other inventions). Tor disassociated the browser from an IP address, hiding a surfer's location and identity.

At the same time, a set of shameless participants, now virtually anonymous, started developing markets for international drug dealing on the dark web. Along with it came child pornography and exploitive human trafficking. All along, some block chain exchanges turned a blind eye, laundered electronic money, and left no traceable identity. And so it grew: the dark web began to contain some of worst examples of online human depravity.

A few years ago, one of the places for illegal commerce, the Silk Road, became too big for law enforcement to ignore. The authorities managed to close it. Remarkably, two new places, AlphaBay and Hansa Market, quickly emerged. Again, authorities closed them. Again, and recently, this market has managed to recreate itself. Don't believe me? Just go to Reddit or 4chan or plenty of other places and search.

That description leaves out plenty of detail, but that should be enough to get the idea. The scope of modern technology makes human depravity available to every online participant in the dark web, and it is becoming increasingly accessible in the regular web.

More broadly, while legal rules and social norms created private spaces for some online users to pursue their niche interests, those same norms have fostered something else—thriving sleazy markets that seem difficult to stamp out.

Why amend the Communications Decency Act? To many, it appears the Internet is managed by technically adept firms that—dare I say it—lack more leadership. Pointedly, where are the restrictive terms of service to ban content that contributes to child porn and the international drug trade? Moralizing is easy: what decent human being refuses to try to stop this type of depravity in his or her own backyard? There are many enablers, so there is no need to point at any one of them in particular. Can a law compel any of them to care?

Now I will editorialize. I have been studying technology my entire professional career. Like most technologists, I take pride in technical ingenuity, and for years I believed extensions to the technical frontier resulted in unalloyed gains. But the more I study this situation, the more I question the presumption about "unalloyed." It is not possible to take pride in the illegal parts of online sex. These actions do not improve the human condition.

More to the point, the web developed with unbridled degrees of unquestioned license. Now some bad actors have catalyzed attempts to end that discretion. Frankly, I see the point in the suggested restrictions. Enough is enough. There is no good reason to allow a decent society to put up with this crap any further.

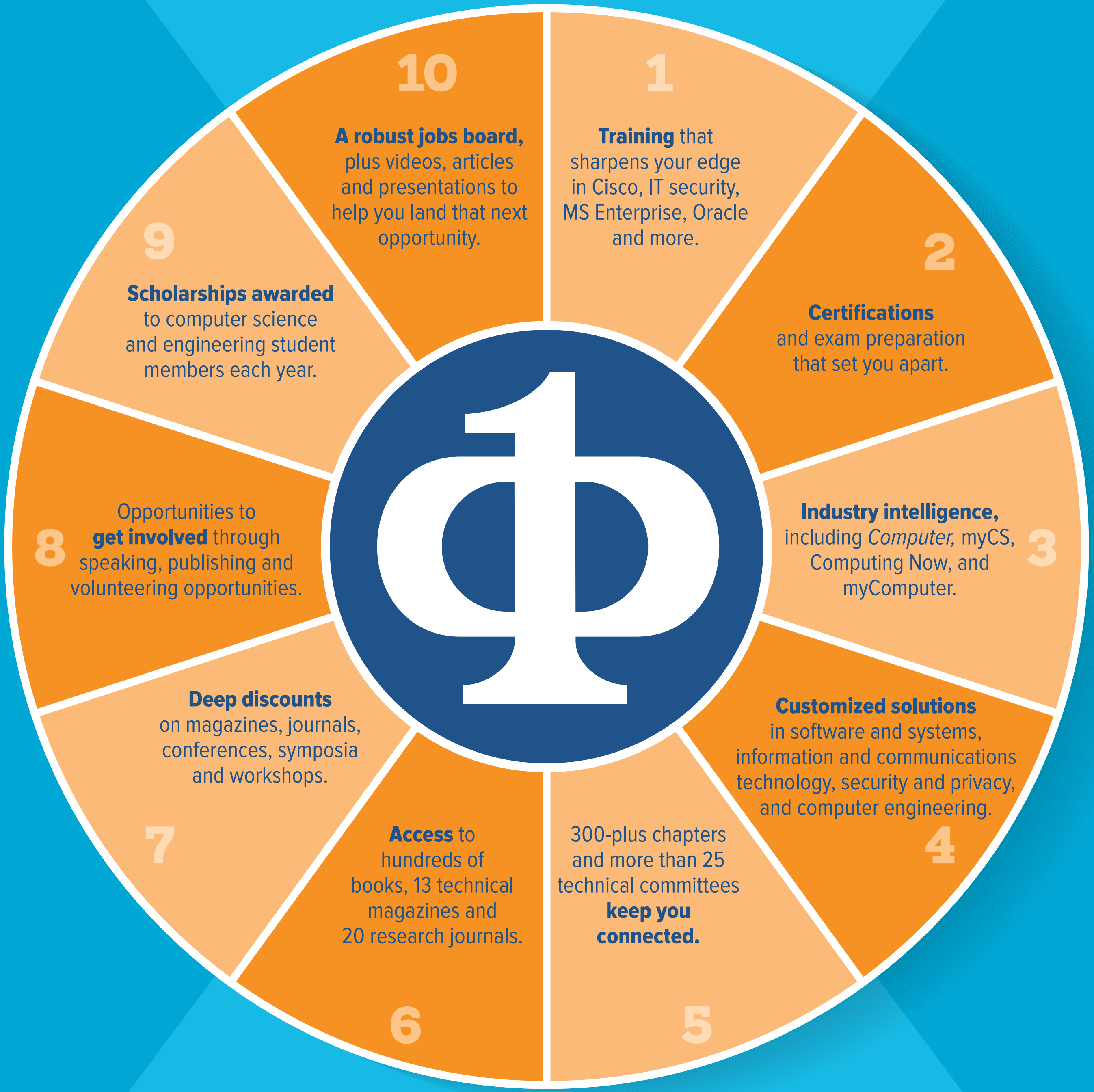
Let me say it another way. If the Valley's management cannot be bothered to take responsibility, then a bunch of crusading legislators in DC will act. I would rather see the Valley's management preempt the legislation, wouldn't you? What are they waiting for?

**Shane Greenstein** is a professor at the Harvard Business School. Contact him at sgreenstein@hbs.edu.

# IEEE COMPUTER SOCIETY: Be at the Center of It All

IEEE Computer Society membership puts you at the heart of the technology profession—and helps you grow with it.

**Here are 10 reasons why you need to belong.**



1 **Training** that sharpens your edge in Cisco, IT security, MS Enterprise, Oracle and more.

2 **Certifications** and exam preparation that set you apart.

3 **Industry intelligence,** including *Computer,* myCS, Computing Now, and myComputer.

4 **Customized solutions** in software and systems, information and communications technology, security and privacy, and computer engineering.

5 300-plus chapters and more than 25 technical committees **keep you connected.**

6 **Access** to hundreds of books, 13 technical magazines and 20 research journals.

7 **Deep discounts** on magazines, journals, conferences, symposia and workshops.

8 Opportunities to **get involved** through speaking, publishing and volunteering opportunities.

9 **Scholarships awarded** to computer science and engineering student members each year.

10 **A robust jobs board,** plus videos, articles and presentations to help you land that next opportunity.

IEEE Computer Society—keeping you ahead of the game. Get involved today.

**www.computer.org/membership**

IEEE computer society

# COMPSAC 2018

**Tokyo, Japan**

**July 23-27**

***Staying Smarter in a Smartening World***

# Call for Papers

COMPSAC is the IEEE Computer Society Signature Conference on Computers, Software and Applications. It is a major international forum for academia, industry, and government to discuss research results and advancements, emerging challenges, and future trends in computer and software technologies and applications. The theme of COMPSAC 2018 is Staying Smarter in a Smartening World.

Computer technologies are producing profound changes in society. Emerging developments in areas such as Deep Learning, supported by increasingly powerful and increasingly miniaturized hardware, are beginning to be deployed in architectures, systems, and applications that are redefining the relationships between humans and technology. As this happens, humans are relinquishing their roles as masters of technology to partnerships wherein autonomous, computer-driven devices become our assistants. What are the technologies enabling these changes? How far can these partnerships go? What will be our future as we deploy more and more “‘things” on the Internet of Things – to create smart cities, smart vehicles, smart hospitals, smart homes, smart clothes, etc.? Will humans simply become IoT devices in these scenarios and if so, what will be the social, cultural, and economic challenges arising from these developments? What are the technical challenges to making this all happen – for example, in terms of technologies such as Big Data, Cloud, Fog, Edge Computing, mobile computing, and pervasive computing in general? What will be the role of the ‘user’ as the 21st Century moves along?

COMPSAC 2018 is organized as a tightly integrated union of symposia, each of which will focus on technical aspects related to the ”smart” theme of the conference. The technical program will include keynote addresses, research papers, industrial case studies, fast abstracts, a doctoral symposium, poster sessions, and workshops and tutorials on emerging and important topics related to the conference theme. A highlight of the conference will be plenary and specialized panels that will address the technical challenges facing technologists who are developing and deploying these smart systems and applications. Panels will also address cultural and societal challenges for a society whose members must continue to learn to live, work, and play in the environments the technologies produce. Authors are invited to submit original, unpublished research work, as well as industrial practice reports. Simultaneous submission to other publication venues is not permitted. All submissions must adhere to IEEE Publishing Policies, and all will be vetted through the IEEE CrossCheck Portal.

**Standing Committee Chair: Sorel Reisman, California State University, USA**
**Steering Committee Chair: Sheikh Iqbal Ahamed, Marquette University, USA**

**General Chairs: Shinichi Honiden (NII, Japan)**
**Roger U. Fujii, Fujii Systems, 2016 IEEE Computer Society Prelident**

**Program Chairs in Chief:**
**Jiannong Cao (Hong Kong Polytechnic University, Hong Kong)**
**Stelvio Cimato (University of MIlan, Italy)**
**Yasuo Okabe (Kyoto University, Japan)**
**Sahra Sedighsarvestani (Missouri University of Science & Technology, USA)**

**Workshop Chairs: Kenichi Yoshida (University of Tskuba, Japan)**
**Ji-Jiang Yang (Tsinghua University, China)**
**Hong Va Leong (Hong Kong Polytechnic University, Hong Kong)**
**Chung Horng Lung (Carleton University, Canada)**

**Local Chair: Hironori Washizaki (Waseda University, Japan)**

**Important Dates**
Workshop proposals
Due date: 15 October 2017
Notification: 15 November 2017

Main Conference papers
Due date: 15 January 2018
Notification: 31 March 2018

Workshop papers
Due date: 10 April 2018
Notification: 1 May 2018

Camera Ready and Registration
Due date: May 15, 2018