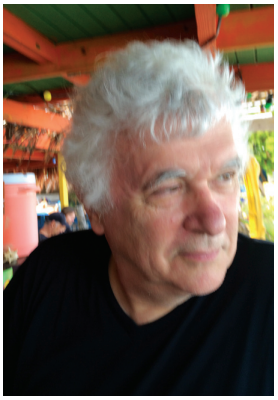

Thoughts on Winning the 2014 Eckert–Mauchly Award

Trevor Mudge
University of Michigan

I'd like to thank Erik Altman and Lieven Eeckhout for the opportunity to summarize some of the remarks I made in my Eckert–Mauchly Award acceptance speech at the 41st International Symposium of Computer Architecture in June 2014. I am very honored to have received the Eckert–Mauchly award, but, as many of you know, I have supervised 50 PhD students, and it would be improbable if I didn't acknowledge their large contribution to work that sometimes gets attributed to me. My principal hope is that I have played the role of constructive critic so that their work has been improved or, better, has taken a novel turn that perhaps they didn't foresee. As you may have guessed, the aspect of academia, and research in academia in particular, that I most enjoy is the opportunity to work with graduate students on research problems.

Coming to America

I received my undergraduate degree from the Applied Physics and Mathematics program at the University of Reading in England. In the fall of 1969 I moved to the Computer Science Department at the University of Illinois in Urbana–Champaign on an English Speaking Union scholarship (I'm monolingual, so my options were limited). It was an eventful time in the United States: the Vietnam War draft soon started, and some of my new classmates were sent to war. On the computer front, Illinois was one of the few universities that built computers. It is also the alma mater of over a quarter of the Eckert–Mauchly winners. The Illinois tradition of building experimental computers started with ORDVAC (Ordnance Discrete Variable Automatic Computer), and its twin, Illiac I. When I got there, Illiac II had just been decommissioned, and Illiac III and IV were under construction.

The faculty was made up of mathematicians, physicists, and electrical engineers. Computer science was not well defined then. There were more courses in quantum mechanics than software. My first research assistantship in computer science was to design digital circuits for experimental digital systems. I also worked on Illiac III. It was built with asynchronous logic like its predecessor, Illiac II. There was a strong focus on asynchronous logic designs, and some of the basic theory was created by the faculty, David Muller (of C-element fame) in particular.

Illiac III was one of the most interesting machines of its day. It was designed primarily as a single-instruction, multiple-data (SIMD) pattern-processing machine, with a novel addition: a pattern articulation unit that received images from a 2D array of photo diodes and processed them in parallel. Each photo diode had a bit serial processor behind it and a shift register so that several images could be stored and pairwise image operations could be performed. It was similar to the later Massively Parallel Processor (MPP), a supercomputer built by Goodyear (the blimp people). In the '80s, many of the ideas in the MPP reappeared in massively parallel supercomputers like Thinking Machine's Connection Machine.

My first computer architecture class was taught by David Kuck, who pioneered many of the ideas we now take for granted in the parallel processing world. Despite enjoying the class and deciding to work in the area, I got a C grade, which for all practical purposes was a failing grade—not very auspicious. I worked with Ted Poppelbaum for my master's degree and Gerry Metze for my PhD—I believe Gerry was the advisor to the largest number of Eckert–Mauchly winners. I developed a hardware design language for my PhD thesis that translated down to asynchronous logic. I learned a lot, but mostly never to use asynchronous logic.

The University of Michigan

In 1977, I started as an assistant professor at the University of Michigan in the ECE department, where I've been since—a remarkable example of lack of imagination. Ann Arbor was a pleasant change from east central Illinois.

I gravitated to dataflow machines, particularly the work of Jack Dennis's group at the Massachusetts Institute of Technology. They started from asynchronous logic, and his group used the same composable asynchronous modules as I

Published by the IEEE Computer Society

had used in my thesis. Asynchronous logic naturally leads to dataflow because it is event driven. Using these modules, Dennis's group had created an elegant implementation of the CDC 6600 scoreboard. My thesis did something similar, but with a design language, and used the example of Tomasulo's algorithm, which I viewed as a data-driven machine.

The 1980s

During the first half of the '80s I became enamored with the Intel 432 and capability machines in general. The 432 was another heroic attempt by Intel to replace the x86 line. It was an unmitigated disaster. It gave the reduced-instruction-set computing (RISC) movement (beginning at that time) abundant ammunition to support a minimalist philosophy of keeping CPU design simple. Support for capabilities does not have to be inimical to simplicity. The basic concept of the 432 was impressive and far ahead of its time. In addition to capabilities, it supported seamless, fault-tolerant multiprocessing on interconnected multiple cores. This was 30 years ago. It could have been groundbreaking if it had been implemented sensibly. For example, it was often possible to experience six levels of indirection to access a 32-bit integer.[1] Given current concerns with privacy and security, it may well be worthwhile to revisit capabilities.

Also in this time period, Mead and Conway started a revolution in teaching integrated circuit design. An important component was support for the fabrication of chips in "multiproject wafers" through MOSIS (supported by NSF and DARPA). For the first time, students could design and fabricate their own prototype integrated circuits. It resonated with my background at Illinois of building prototypes. It led to a new wave of computer hardware companies. Most notable were the Berkeley and Stanford spin-offs of the RISC machines, but there were many others.

On the architecture side, we at Michigan obtained an early *n*-cube—a 64 processor connected as a hypercube.[2] This was one of the many massively parallel machines that borrowed from microprocessor technology and included the Cosmic Cube, Intel's iPSC, and the Connection Machine mentioned earlier. These took over the high-performance computing space. It was clear that traditional supercomputers were no longer trendsetters as they had been in the era of the Cray 1. Rather, the rise of the microprocessor and its associated volumes meant that much of the innovation was now found in the microprocessor world. As a result, my interests moved back into this area, particularly with the advent of cell phones in the 1990s, where high-performance requirements added the challenge of stringent energy requirements.

High-performance Gallium Arsenide (GaAs) computers

By the late '80s and early '90s, Michigan had built a significant ability to produce chips. Thanks to my then-colleague Rich Brown, we had replaced early design tools, mostly produced by researchers, with commercial tools from Mentor Graphics and later Cadence and Synopsis. This meant we had a stable industrial-strength design environment. We initiated some ambitious projects. The first was a MIPS-like processor that was implemented in GaAs. The attraction of GaAs was its high electron mobility, which translated into much faster switching times—2 to 3 times that of silicon. We were successful in producing a 200 MHz processor when most processors were operating at 50 MHz.[3] Unfortunately, DEC demonstrated a 200 MHz Alpha at about the same time. It was clear that silicon would not soon be replaced. We missed a key point: GaAs operated at 1 V when silicon was at 3.3 or 5 V. We should have sold it as a low-power technology. No one was interested in low power then: chip manufacturers were selling frequency. We designed and fabricated a follow-up chip. It was a much more complex out-of-order machine, which didn't work properly. We were too ambitious. It reinforced for me the need to keep hardware mechanisms as simple as possible.

The chip design work inspired a lot of related research. We developed new static-timing techniques for latch-based designs, and I started to look at processor interconnect, particularly crossbars. I convinced three undergraduates (one was Kunle Olukotun) in our VLSI class to fabricate a crossbar in 3-micron technology. It was clear from the layout, which we structured as a static RAM layout, that area was not an issue. Their $O(n^2)$ growth was not a concern for $n < 100$, and they offered a way to build tightly coupled computing nodes with a flat memory space that avoided the headaches of nonuniform-memory-access machines. I have continued with this "obsession" and recently convinced my colleagues to fabricate a series of $64 \times 64 \times 128$ interconnects.[4]

I also continued a line of research into traditional microarchitecture that goes back to my early papers inspired by Bob Keller and the dataflow work mentioned earlier. I was interested in getting performance without too much complexity. This led to ideas like "run-ahead" and the use of larger logical register files. To do this, a group of my graduate students—Dave Greene, Matt Postiff, Dave Oehmke, and Kris Flautner—built MIRV, a compiler to improve register file usage, which included link time register optimization. This line of work eventually led to a scheme for virtualizing registers called the virtual context architecture. Among other things, it allowed register windows to be extended, simplifying function calls and multithreading.[5]

Late 1990s energy-aware computing

In the late 1990s, my interests changed from high performance to energy aware. This was largely due to conversations I had with Bob Colwell, who was responsible for the highly successful Intel P6 core. He suggested that a few of us put together a workshop on computer architecture to discuss the impact of power. At ISCA 29 in Barcelona, Dirk Grunwald, Bobbie Manne, and I organized the "Power Driven Microarchitecture Workshop." Given my new interest in power, and with a sabbatical coming up for the 1999–2000 academic

year, it seemed natural to contact ARM about a sabbatical. ARM by then was gaining a reputation as a designer of low-power processors. I called Mike Muller, their CTO, out of the blue and asked him if they had a place for me. Surprisingly, he said yes, and I spent much of that year at ARM in Cambridge, England.

Upon my return from Cambridge, I started the ARM lab at Michigan. I also summarized my thoughts about the importance of power in processor design in an article for *Computer* called "Power: A First-Class Architectural Design Constraint."[6] Looking back, it seems trite, but at that time the leading processor manufacturers were still selling clock frequency. Herbert Stein noted, "If something cannot go on forever, it will stop," and this indeed happened to clock frequency.

My research also focused on lowering power and energy in processors. In 2002, together with Nam Sung Kim, Kris Flautner, and David Blaauw, I developed the idea of drowsy caches—a way to retain state while lowering voltage on accessed cells in a cache. Shortly afterward, David Blaauw, Todd Austin, and I developed the idea of Razor as a way to save power by lowering voltage below the usual design margins—the circuits were operating on the hairy edge, so Razor seemed an apt description. It also guarded against variability. A prototype was fabricated, and a number of patents ensued. Interested readers can see some of them at my website (http://web.eecs.umich.edu/~tnm). The ARM engineers showed interest and greatly cleaned up the original idea. A new transition detector was created, and the hazard detection logic was shown to be unnecessary. David Blaauw and his students continue to create new, improved versions.

Energy efficiency proved to be a fruitful area of research, and several other projects were started in the decade since my first sabbatical at ARM (I liked ARM so much I returned for another sabbatical seven years later). Through my association with ARM, I became interested in mobile phones. Initially I was interested in making the baseband processor programmable. It had traditionally been implemented as an application-specific integrated circuit (ASIC) for power reasons (key parts still are). It seemed to me to be the ultimate challenge for power-aware computing—tens of giga operations per second in a subwatt envelope. I, together with several others, wrote a paper called "Mobile Supercomputers" that outlined that challenge.[7] Subsequently, baseband processors have become more programmable to support changing standards and algorithms. We produced several paper designs that later were influential in commercial spin-offs from ARM.[8]

The search for techniques that could reduce power also led me to investigate 3D die stacking, and with my then graduate student Tae-ho Kgil, I looked at it in the context of servers. This led to a series of papers on PICO servers.[9] We later extended this to incorporate Flash as a way to further reduce power. More recently, we started to look at operating logic at much lower voltage levels—a regime we called near threshold computing (NTC). This naturally combined with 3D stacking because NTC reduces thermal worries. To test the idea, a group of us, led by Ron Dreslinski and Dave Fick, designed and fabricated a 128-core NTC multiprocessor, Centip3De, that stacked processors, caches, and DRAMs.[10] It was a large undertaking that stretched our design environment because of the lack of 3D tools and the size of the design.

To conclude, much of my research has been informed by the impact of technology on computer design. My thesis is that the field of computer architecture offers new challenges as the implementation technology changes, which it does at an incredible rate. The number of challenges has also grown because more requirements have to be satisfied. It's not just about pure performance—power, and more recently, size, have emerged as design constraints. New technologies are being proposed constantly. Only a tiny few will end up being useful—it's impossible to pick winners early on. Their characteristics will lead to new possibilities for revolutionary computing machines. It's an exciting time. MICRO

References

1. T. Mudge et al., "Object-Based Computer Architectures," *Proc. Conf. Information Sciences and Systems*, 1983, pp. 733–741.
2. J. Hayes et al., "A Microprocessor-based Hypercube Supercomputer," *IEEE Micro*, Oct. 1986, pp. 6–17.
3. M. Upton et al., "A 160,000 Transistor GaAs Microprocessor," *Proc. Int'l Solid-State Circuits Conf.*, vol. 36, 1993, pp. 92–93.
4. R. Dreslinski et al., "Swizzle Switch: A Self-Arbitrating High-Radix Crossbar for NoC Systems," *Hot Chips* 24, 2012, pp. 380–403.
5. D. Oehmke et al., "How to Fake 1000 Registers," *Proc. 38th Ann. IEEE/ACM Symp. Microarchitecture*, 2005, pp. 7–18.
6. T. Mudge, "Power: A First Class Architectural Design Constraint," *Computer*, vol. 34, no. 4, 2001, pp. 52–57.
7. T. Austin et al., "Mobile Supercomputers," *Computer*, vol. 37, no. 5, 2004, pp. 81–83.
8. M. Woh et al., "From SODA to Scotch: The Evolution of a Wireless Baseband Processor," *Proc. 41st IEEE/ACM Int'l Symp. Microarchitecture*, 2008, pp. 152–163.
9. T. Kgil et al., "PicoServer: Using 3D Stacking Technology to Enable a Compact Energy Efficient Chip Multiprocessor," *Proc. 12th Int'l Conf. Architectural Support for Programming Languages and Operating Systems*, 2006, pp. 117–128.
10. D. Fick et al., "Centip3De: A 3930 DMIPS/W Configurable Near-Threshold 3D Stacked System with 64 ARM Cortex-M3 Cores," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2012, pp. 190–191.

Trevor Mudge is the Bredt Family Professor of Engineering in the Electrical Engineering and Computer Science Department at the University of Michigan. Contact him at tnm@umich.edu.