Impact of Future Technologies on Architecture

This article presents position statements and a question-and-answer session by panelists at the 4th Workshop on Computer Architecture Research Directions. The subject of the debate was new technologies and their impact on future architectures.

Trevor Mudge
University of Michigan

Frederic T. Chong
University of Chicago

Igor L. Markov
University of Michigan

Resit Sendag
University of Rhode Island

Joshua J. Yi
Dechert

Derek Chiou
University of Texas at Austin

The third minipanel from the 2015 Workshop on Computer Architecture Research Directions, held in conjunction with the 42nd International Symposium on Computer Architecture, featured two experts discussing different technologies and their impact on future architectures. What are these new technologies, where are they going, and what is going to happen to them? More specifically, this panel focused on the following questions:

- Is quantum computing likely to have a wide impact on the development of computing machines, or will it remain of interest only to computer science theoreticians?
- Will on-chip optical interconnect become practical?
- How would you, the panelists, characterize what is happening to Moore's law and Dennard scaling?
- Will approximate computing find a place in computers? How will the user be made aware of it?
- What new memory technologies are the most promising, and how will they change the way we design memory systems and computers?

The first expert panelist was Fred Chong, the Seymour Goodman Professor of Computer Architecture in the Department of Computer Science at the University of Chicago. Chong argued in favor of the promise of new technologies—an optimistic view. The second panelist, Igor Markov, a professor of electrical engineering and computer science at the University of Michigan, took a more pessimistic position and outlined the limitations of the new technologies.

Each panelist had 10 minutes to present his position statement, after which the moderator, Trevor Mudge, asked the panelists a few questions. The floor then opened up to the audience to ask questions.

Frederic T. Chong: Emerging Technologies through Rose-Colored Glasses

Emerging technologies will play a key role in addressing the gap between the needs of big data and Moore's law improvements to CMOS. At one extreme, quantum computation offers both the largest gains and the most risk. Early quantum machines with 5 to 7 bits were built 15 years ago.[1] Recently, there has been significant investment and engineering to develop quantum computation. The D-Wave machine is an example, which gives a notion of the engineering progress and the scale of machines that can be built. Over the last couple of years, there has been quite substantial investment by Google, Microsoft, and IBM. The federal government

Published by the IEEE Computer Society

has also invested $700 million into superconducting devices. Interestingly, to some extent, the size of quantum devices or quantum computers that have been built has been a little bit artificially limited by the funding that we receive.

I think we are at the cusp of some great scaling in terms of quantum machines and quantum devices. In the next few years, the goal is to build machines on the order of 100 quantum bits and then scale basically exponentially past that.

The big questions with quantum computation are, What can be physically built? What is the scaling progress that has been made? What is it good for? But, most important of all, what quantum algorithms do we have? That is probably the most difficult aspect of it. The National Institute of Standards (NIST) has a comprehensive catalog of all the known quantum algorithms, called the Quantum Algorithm Zoo (http://math.nist.gov/quantum/zoo). Figure 1 shows the number of quantum algorithms that existed over time, including variations on certain algorithms. It shows that the number of quantum algorithms has grown quite substantially over the years. Around the year 2000, we were not very high up on the curve, and there was a lot of discussion as to where this was going. Our DARPA program manager at the time, Mike Foster, was quite optimistic. He said, "Well, this is still early in the development of quantum algorithms. We still have a lot more to do."

There has been a lot of interest in quantum computation in terms of cryptography, but more recently, there is some optimism that we will have algorithms that are good for searching, machine learning, and other kinds of pattern optimization problems. There is a lot more to do here, but there is cause for optimism.

Quantum Devices Also Offer Promising Improvements in Classical Computations

One thing that is very interesting in the near term is the use of quantum devices and quantum coherence for classical circuits and latency-critical computations. HP has designed an optical router that uses quantum coherence.[2] It consists of an array of resonators, which are coherent. It performs optical packet routing by doing global corner-to-corner routing using the quantum coherence of those devices in a very low-latency manner. This example illustrates that there is a large, unexplored area of what we could do with quantum computation. For classical tasks, there is a potentially interesting impact.

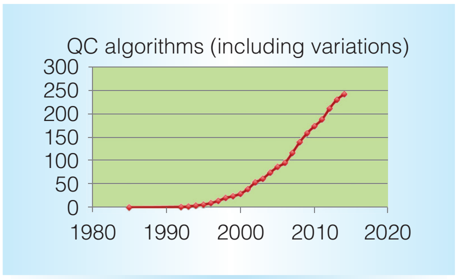

Figure 1. Number of quantum algorithms over time. Data taken from the Quantum Algorithm Zoo (http://math.nist.gov/quantum/zoo).

Impact of Emerging Technologies on Future Architectures

Systems and architectures will need to adapt to the unique properties of new devices. There has been a lot of work in looking at using 3D geometries for different kinds of high-density, high-bandwidth memory systems. I also think there will be some impact in terms of persistent memories. In many persistent memory technologies, there are some unusual dynamic tradeoffs, particularly in terms of the density, endurance, and persistence of storage. As such, there could be entirely different designs for memory systems in a way that we do not expect.

Impact on Computation Models

Building machines with future technologies will require a paradigm shift at the user and compiler levels and some explicit tolerance to variation and errors at the device level. Emerging technologies—even CMOS technologies—will require that we have error- and variation-tolerant computation models exposed to the user and the application. The challenge will be bounding the desired and achievable error. There are a few successful high-level approximation examples. FlexJava provides software-level encoding of the errors that can be tolerated from approximate computing.[3] In the area of big data, approximate

HADOOP has recently shown that, given a statistical solution, it can use sampling to generate an approximate result with a bounded error.[4]

I believe that for big data or high-dimensional problems that generate big data, the energy and time constraints will prevent us from covering the entire space. From an application point of view, we may expect to see a requirement for approximation. One way of achieving approximation is sampling. For example, importance sampling is a very common technique that is used in computational science and data analysis. I think that given this confluence of technology and application space, approximate computation will become critical for large-scale computations.

Igor L. Markov: Fundamental Limits to Computation and What to Do about Them

The death march of Moore's law motivates us to discuss obstacles to computing and ways to circumvent them. Currently, there are a variety of emerging technologies including 3D circuits, as well as exotic possibilities such as quantum computing and carbon nanotubes. But revolutionary new opportunities often harbor grave limitations. For example, the promise of asymptotic runtime improvements through quantum computing may be offset by heavy implementation constants and very narrow markets. More modest steps may offer a viable path to higher-performance computing. In any case, new breakthroughs will require concerted improvements in switching elements, memories, interconnect, full-chip optimization, architecture platforms, compilers, and runtime support. Betting on such emerging technologies is exciting and risky, but one thing is clear: the free ride on the back of Moore's law is over, and researchers must now distinguish worthwhile directions from dead ends.

Limits on Manufacturing

Over the last 40 years, the electronics industry accomplished amazing feats of integration by putting many system components on a single chip. For example, an iPad3 would have qualified as a top 300 supercomputer in 1993, in terms of Linpack performance. Although there have been improvements in architecture and circuit optimization, most of this progress has been due to materials. But the progress due to materials is coming to an end. The immediate obstacle is optical lithography, which is an engineering and cost problem; one could use electron beams for manufacturing, but that is extremely slow and extremely costly.

Limits on Interconnects and Transistors

Jeffrey Davis and his colleagues classified the limits of devices and interconnects as fundamental, material, device, circuit, and/or system.[5] For example, Dennard scaling is a circuit and system trend.[6] Dennard scaling theory shows how to keep power consumption of semiconductor ICs constant while increasing their density. Dennard scaling, however, broke down about 10 years ago.[4] Moore's law, which is more of a device-scaling trend, continues, but without the same large-scale improvements that it enjoyed in the past. At present, it is clear that Moore's law will continue scaling down to 7-nm features and perhaps even 2 nm, but probably not below that. After that, there are atomic limits that are more fundamental. Transistors are limited by their smallest feature—the width of the gate dielectric—which recently reached the size of several atoms. With such small features, a few missing atoms could alter transistor performance, and manufacturing variation makes all transistors slightly different in ways that are difficult to predict.

Energy-Time Limits

One of the main obstacles to the improvement of modern electronics is the management of system power and energy.[7] Generally, the faster the computation, the more energy it consumes. But actual power/performance tradeoffs depend on the physical scale. Analysis of fundamental limits reflects available energy resources, properties of the physical space, power-dissipation constraints, and energy waste.

In the 1960s, Yakir Aharonov and David Bohm studied how much energy was needed to reliably switch 1 bit.[8] Considering 0 and 1 as different energy levels, the smaller the energy gap between them, the more time is spent to switch. To switch faster, the energy gap has to be adjusted. Numerically, this is expressed by the Heisenberg uncertainty principle in the energy-time form, but the technologies that we use today are nowhere close to the Heisenberg uncertainty for switching between 0 and 1. At first, quantum computing and working with individual quantum bits might seem energy efficient. However, the quantum devices used for quantum computing currently do not scale to large sizes, are energy inefficient at the system level, rely on fragile components, and require heavy fault-tolerance overhead.[9]

In a 1959 talk, Richard Feynman suggested that there was "plenty of room at the bottom," which forecasted the miniaturization of electronics. Today, with relatively little physical room left, there is plenty of energy at the bottom. If this energy is tapped for computing, how can the resulting heat be removed? Recycling heat into mass or electricity seems to be ruled out by energy conversion limits and the acceptable thermal envelope. Individual quantum devices now approach energy limits for switching, whereas nonquantum devices remain orders of magnitude away.[4] Such energy considerations suggest yet another obstacle to simulating quantum physics on conventional computers (abstract models aside). Quantum computers can be instrumental, but fault-tolerance overhead offsets their potential benefits in practice, and empirical evidence of quantum speedups has not been compelling so far.[10,11] Moreover, in terms of computational complexity, quantum computers cannot attain significant advantage for many types of problems. The lack of consistent general-purpose speedup limits the benefits of several emerging technologies in mature applications with diverse algorithmic steps, for example, computer-aided design and Web search.

Asymptotic Space-Time Limits

The theoretical limits on energy and power are very loose, and practical technologies might never approach some of them. Reasonably tight limits are rare. David Fisher employed asymptotic runtime estimates to study limits to parallelism and dimensionality (for example, 2D, 3D), the latter of which makes his work more interesting.[12] He assumed a sequential computation with $T(n)$ steps for input size of n, and limits the performance of its parallel variants that can use an unbounded d-dimensional grid of parallel processors communicating at a finite speed (for example, bounded by the speed of light). He found that the parallel runtime requires $T(n)^{1/(d+1)}$ steps. This result undermines the N-fold speedup assumed in Gustafson's law for N processors on appropriately sized input data.[13]

Fisher's results indicate a relatively tight limit in that many parallel computations today are limited by several forms of communication and synchronization. As wires get slower relative to gates at each new technology node, interconnects are responsible for most of the circuit delay.[14] Exploring Fisher's results in two and three dimensions shows that a sequential computation with $T(n)$ steps requires $T(n)^{1/3}$ steps in 2D and $T(n)^{1/4}$ in 3D. If the time to execute is $T(n)^{1/3}$ steps in 2D, then 3D integration asymptotically reduces t to $t^{3/4}$. This speedup is significant but not dramatic and requires scaling in all three dimensions, which is hard and expensive with today's die-stacking technologies with relatively thick through-silicon vias (TSVs). Monolithic 3D integration seems to be a more viable solution. It enables the fabrication of 3D ICs with multiple transistor layers and ultra-dense vertical connectivity between layers.

Questions

This section presents an edited transcription of the question-and-answer session.

Quantum Computers

Trevor Mudge: Is quantum computing likely to have a wide impact on the development of computing machines, or will it remain of interest only to computer science theoreticians?

Fred Chong: The big questions with quantum computation are what can be physically built, what is the scaling progress that can be made, and what is it good for? What quantum algorithms do we have? If you look at the number of quantum algorithms that have existed over time, you'll see it has been growing quite substantially. I think there has been a lot of interest in quantum computation in terms of cryptography, but more recently there's some optimism that we'll have algorithms that are good for search machine learning and other kinds of pattern optimization problems. There's a lot more to do here, but there's cause for optimism.

Igor Markov: There has been amazing progress on quantum simulation in the last 10 to 15 years. Systems that Feynman wanted to simulate, and for which he suggested quantum computers, can now be simulated very quickly on nonquantum computers. But, of course, not everything can be simulated, and interesting questions remain. Photosynthesis is one such example. If we understood photosynthesis better and were able to generate energy from the sun more efficiently, that would be great. Such understanding could be developed with the help of physically accurate simulation, which has been challenging because photosynthesis involves quantum physics. But one way or another, this is currently a science challenge, not an engineering one.

I don't believe that in 10 years there will be quantum computers on anyone's desk. I don't think quantum computers will do any useful engineering tasks—but maybe some science-related tasks. First of all, and most fundamentally, the qubits are flaky. They decohere quickly. You need to restore them, and the necessary error correction requires much larger circuits. For example, for Shor's algorithm (number factoring) you need circuits having many thousands of qubits, which are completely unrealistic today. Fred is very optimistic about where quantum computers could be, assuming there will be sufficient funding. Even if you assumed that quantum computers improved at a Moore's law rate—which is a huge assumption and completely unreasonable right now—in 10 years, scaling 2 times every 18 months; you still miss the stage where Shor's algorithm could be used. And even if you get this algorithm to work and get all the science done, how big of a market is there for Shor's algorithm?

How about quantum machine learning? Google and Microsoft mention this as a promising direction, but I do not see any evidence that suggests that. It's a good science project, but you cannot assume that this will work commercially. A while back, an algorithm from MIT claimed to solve large linear systems on a quantum computer, but it doesn't actually solve linear systems. It only allows you to answer some questions about the solutions of linear systems, which is a lot less compelling. In machine learning, you're dealing with a lot of data, and, currently, quantum computers are not great with a lot of data; they answer yes-or-no questions with a high degree of certainty. So, I'm negative on the engineering aspects of quantum computing, but as far as science goes, this is all worthwhile.

Mudge: What about the development of D-Wave machines as examples of quantum computers?

Chong: Quantum computation is one of the few technologies that we have that promise of some sort of exponential gain to deal with the exponential gap that we had in the last decade. Quantum computing is a high-risk, high-payoff kind of proposition.

There's been a fair amount of hype around this D-Wave machine, which is actually a quantum machine and not exactly a digital quantum computer, but I think it still gives some notion of the engineering progress that's been made and the scale of machines that can be built.

Markov: First of all, the D-Wave is sort of a running joke in the field. What can the D-Wave computer currently do, as compared to conventional computers? Even *The Economist* magazine opined on this: a quantum computer would be both slower and faster at the same time. D-Wave employs physicists that are doing a lot of great physics. Beyond D-Wave, good theoretical progress is made in the field of quantum information processing. But as far as conventional computing goes, a laptop can simulate what D-Wave chips can do and can finish the task faster. This point is worth repeating: whatever the D-Wave chip does, a laptop can literally simulate it faster, and if you have GPGPUs, FPGAs, or multiprocessing systems, you get four to five times the speedup. Maybe D-Wave will come up with a better chip, but at the moment, it's their challenge, not ours.

Mudge: Okay Fred, what was your reaction to Igor's less-than-positive statements about D-Wave and some of the new experiments in quantum computing?

Chong: There certainly has been a lot of skepticism about D-Wave in terms of the practical utility of the machine. You can think about quantum computation as sort of a contest of two exponentials. You have certain algorithms that have exponential gain, but you also have potentially exponential cost in controlling the coherence and noise of the machines. One solution is lots of engineering to break down the exponential costs of making the machines. There are a few end runs that people are looking at. There are topological codes and topological machines. A technology switch may dramatically reduce the amount of error correction you need, and there are some physical techniques that people use to control noise. It's clearly an expensive proposition in terms of the number of quantum bits and energy, as Igor mentions, but I think there's quite a lot of effort in terms of controlling those overheads. It's hard to say where that will go.

Approximate Computing

Mudge: Will approximate computing find a place in computers? How will the user be made aware of it?

Chong: I think emerging technologies will have a large effect on computer models and in conjunction with application and data trends. Emerging technologies—even CMOS technology—require that we have error- and variation-tolerant computer models exposed to the user and the application. The challenge, of course, is to bound the desired and achievable errors of those devices.

There's been a lot of work in this area—for example, Flex Java. When we look at big data problems or high-dimensional problems that generate big data, energy- and time-constrained computation simply cannot cover the entire space. From an application point of view, there will be a requirement for approximation. One method of approximation is sampling. A very common technique that's being used in this domain is the notion of importance sampling in computational science and in data analysis. The problem with approximate computation is that if you have a fair amount of discontinuities, then you have to find the discontinuities and provide more precision. I think there should be some interesting areas here where we can look at the semantics or perhaps the program semantics of applications and try to reverse-engineer those discontinuities. Given this confluence of technology and application space, approximate computation will become critical for large-scale computations.

Markov: Let's explore the relationship between approximate and quantum computations. The most promising application for quantum computers is quantum simulation, and there are good questions in science that could be answered by fast approximate simulation. Consider the D.E. Shaw Anton computer, which is a nonquantum supercomputer custom designed for molecular dynamics. It doesn't account for quantum effects at all because the designers didn't see enough market for that. They studied the market carefully and were well-funded, but they didn't want to build such functionality. This tells me that there isn't much of a market for such quantum simulations. Quantum approximation heuristics are basically algorithms that don't promise to solve something exactly, but they often produce good results quickly. That's an interesting idea. There were some claims recently showing better approximation by a quantum algorithm than the best-known conventional algorithm. But guess what happened. People who worked on the classical approximation algorithms for the same problem said, "Aha! I can do better." And they did, in about six months, so this competition is still ongoing.

Audience Member: What kind of approximate computing do you think is going to become successful, if ever?

Chong: Well, if you're in a serious constraint situation, which could be, for example, that you have a very high-dimensional computation that you just will never have enough budget to deal with either power- or time-wise, you're going to have some form of approximation. If we can get a better balance on the desired and achieved errors on computations, then that will become ubiquitous. Also, as some of these technologies become adopted, we are going to have some inherent variability in error that we have to deal with.

Markov: I'm not an architect by training. My training is in CAD. I have some background in theory of computation and in algorithms. To me as an outsider, it looks like approximate computing is a poor choice of name for what you guys are doing. The associations that come up when you say "approximate computing" include approximation heuristics—for example, CAD tools perform circuit partitioning by approximation. They don't necessarily find optimal solutions. Searching for optimal solutions would be too slow. Such well-established approximate computation is unrelated to what is done in architecture today. Another example: quantum computing. It is probabilistic and also approximate in giving correct results only with some probability. Is this considered approximate computing? No. So, what is approximate computing? It is changing the bit-width of the datapath from 16 bits to 14 bits, and that's been studied before in the CAD community. There are algorithms for sizing datapaths while ensuring required precision—some of the research challenges that Fred mentions. They have been addressed, but not in an exhaustive way that users would find easy to exploit. If you call it datapath sizing, then it looks more convincing.

New Memory and Other Emerging Technologies

Mudge: What new memory technologies are the most promising and how will they change the way we architect memory systems and computers?

Chong: There certainly has been a lot of work in looking at 3D geometries and different kinds of high-density, high-bandwidth memory systems. One thing that I found very interesting is there's certainly going to be some impact in terms of persistent memories. What I've found in many persistent memory technologies is there's some unusual dynamic tradeoffs that you can make. Density, endurance, and persistence of storage can be traded with the time to do writes and reads of those devices, so depending on your application, there can be entirely different designs for memory systems in a way that we don't expect. For example, Hewlett Packard is making a big bet on their new memory architecture, which looks like a 3D block and is designed for read-mostly big data. This storage-class memory will run at near-DRAM speeds but at a lower cost. There is not going to be much write or endurance problems with their memory technology, and they're making a big bet on lots of memory storage for density purposes.

Markov: 3D circuits have not seen much adoption so far, because scaling in three dimensions is hard and the technology is expensive. The main benefits of 3D ICs today are in improving manufacturing yield, increasing I/O bandwidth, and combining 2D ICs. This may work for memory-on-memory stacking, but just putting together a chip on a chip means that the communication between the layers becomes relatively slow. A better solution is monolithic 3D integration, which enables the fabrication of 3D ICs with multiple transistor layers and ultra-dense vertical connectivity. Building monolithic circuits with three-dimensionally integrated transistors results in shorter wires, which helps tackle wire delay problems. Furthermore, stacked device layers increase the number of transistors per unit area without requiring costly feature size reduction. This, in turn, helps tackle cost issues and scaling.

Stanford University has recently built a chip that demonstrated monolithic 3D integration without stacking multiple 2D chips. Their chip has active layers of silicon, resistive RAM, and carbon nanotube FETs (CNT-FETs). CNT-FETs leverage extraordinary carrier mobility in semiconducting carbon nanotubes to use interconnect more efficiently by improving drive strength, while reducing supply voltage. Many technology limits, from fabrication of carbon nanotubes to their integration into silicon, have been overcome to facilitate monolithic 3D integration.

As for memory, future systems will probably need to use multiple memory technologies, such as the emerging nonvolatile ReRAM and STT-RAM, and the traditional DRAMs. Each has different kinds of drawbacks. Some do not withstand many read-write cycles, and some are not very dense. They need to be combined in a memory hierarchy with very high-density interconnect, which monolithic 3D integration can facilitate.

Audience member: I wanted to know if you think that carbon nanotubes are a good solution for making switches. How can we manipulate these carbon nanotubes? They are very difficult to manipulate and expensive to power. Is there any solution, or is there any real processor that has been made using these carbon nanotubes? Or just make one single feature out of it?

Markov: Yes, the Stanford group made a full processor. This processor was even demonstrated driving a hand that would shake people's hands. It was a small MIPS processor that was completed within the last two years. A big practical challenge was sorting out metallic nanotubes apart from semiconducting nanotubes; this was done by burning metallic nanotubes with high voltage.

Chong: Are those chemically fabricated?

Markov: They are grown chemically on the surface of quartz and then transferred mechanically onto silicon at low temperature, which supports monolithic 3D.

Audience member: Igor, can you comment on the state of affairs in emerging memory technologies? I mean, is PCM, STT RAM, the HP stuff, is this part of the science done and it's just a matter of getting the yields up and commercializing it, or are there still real challenges of adoption?

Markov: Yes. There are challenges technology-wise, circuit-wise, full-chip-optimization-wise, and in terms of architecture. There are challenges and opportunities that you can address. These individual memories have some serious problems. For example, some memories don't support a large number of read-writes, so you don't want to put them into caches. You can put them into main memory with some fallback solutions. Other memories can withstand multiple cycles, but they're not as dense, so they make good caches. You need to redesign the entire cache and memory hierarchy to use emerging technologies.

Chong: Those are sort of the architectural aspects. But is this production-ready, or do you see this in the next coming years?

Markov: It does look like it will be production-ready soon. So far, the attempts were to replace a single type of memory in use now with a single emergent type, and that is not the right way to go because new hardware has new drawbacks.

Mudge: Look at DRAMs. You could fabricate gigabits for a few dollars. That's pretty impressive. It seems to me these new technologies need a niche to get started. Flash had MP3 players as their niche that allowed them to reach volume production. Today they have become the most common storage medium except for disk.

Markov: Compared to Flash and even DRAM, these new memories can be denser. DRAM has known scaling problems beyond around 8 nm; the capacitors don't scale any lower. Emerging technologies have better scaling, plus they're nonvolatile, so you get something that is lower power than DRAM and denser than Flash.

Chong: I think the nonvolatility is really going to become important, but as Igor mentioned, if you're trying to beat DRAM on a level playing field, that's not going to happen. You have to have some sort of architectural application or niche, as Trevor mentioned. There're some really interesting architectural tradeoffs you can make in these devices. For example, lifetime is a problem in resistive RAM, but if you write slowly to these devices you get a quadratic advantage in lifetime. An architectural design where we write back to this last-level cache as a delayed write-back can give you that advantage. There are other things you can do, such as iterative writes, that give you multilevel storage into single bits so you can essentially trade exponential time for exponential storage. There are a lot of things that you can do that you can't do with traditional storage devices, and the question is whether those can be used profitably at the current architectural level. MICRO

References

1. L.M.K. Vandersypen et al., "Experimental Realization of Shor's Quantum Factoring Algorithm Using Nuclear Magnetic Resonance," *Nature*, vol. 414, 2001, pp. 883–887.
2. C. Santori et al., "Quantum Noise in Large-Scale Coherent Nonlinear Photonic Circuits," *Physical Rev. Applied* 1, 2014; http://arxiv.org/pdf/1402.5983.pdf.
3. J. Park et al., "FlexJava: Language Support for Safe and Modular Approximate Programming," *Proc. 10th Joint Meeting Foundations Software Eng.*, 2015, pp. 745–757.
4. I. Goiri et al., "ApproxHadoop: Bringing Approximations to MapReduce Frameworks," *SIGARCH Computer Architecture News*, vol. 43, no. 1, 2015, pp. 383–397.
5. J.A. Davis et al., "Interconnect Limits on Gigascale Integration (GSI) in the 21st

Century," *Proc. IEEE*, vol. 89, no. 3, 2001, pp. 305–324.

6. M.A. Bohr, "30 Year Retrospective on Dennard's MOSFET Scaling Paper," *IEEE Solid-State Circuits Society Newsletter*, vol. 12, no. 1, 2007, pp. 11–13.
7. *International Technology Roadmap for Semiconductors*, 2016; www.itrs.net.
8. Y. Aharonov and D. Bohm, "Time in the Quantum Theory and the Uncertainty Relation for Time and Energy," *Physical Rev.*, vol. 122, no. 5, 1961, pp. 1649–1658.
9. M.A. Nielsen and I.L. Chuang, *Quantum Computation and Quantum Information*, Cambridge Univ. Press, 2011.
10. T.F. Rønnow et al., "Defining and Detecting Quantum Speedup," *Science*, vol. 345, no. 420, 2014; https://arxiv.org/pdf/1401.2910v1.pdf.
11. S.W. Shin et al., "How 'Quantum' is the D-Wave Machine?" 2014; http://arxiv.org/pdf/1401.7087v2.pdf.
12. D. Fisher, "Your Favorite Parallel Algorithms Might Not Be as Fast as You Think," *IEEE Trans. Computers*, vol. 37, no. 2, 1988, pp. 211–213.
13. D.A. Padua ed., *Encyclopedia of Parallel Computing*, Springer, 2011.
14. R. Shelar and M. Patyra, "Impact of Local Interconnects on Timing and Power in a High Performance Microprocessor," *IEEE Trans. CAD*, vol. 32, no. 10, 2013, pp. 1623–1627.

Trevor Mudge is the Bredt Family Professor of Computer Science and Engineering at the University of Michigan, Ann Arbor. His research interests include computer architecture, programming languages, VLSI design, and computer vision. Mudge received a PhD in computer science from the University of Illinois, Urbana–Champaign. Contact him at tnm@umich.edu.

Frederic T. Chong is the Seymour Goodman Professor of Computer Architecture in the Department of Computer Science at the University of Chicago. His research interests include emerging technologies for computing, computer security, and sustainable computing. Chong received a PhD in electrical engineering and computer science from the Massachusetts Institute of Technology. Contact him at chong@cs.uchicago.edu.

Igor L. Markov is a professor in the Department of Electrical Engineering and Computer Science at the University of Michigan. His research focuses on computers that make computers. Markov received a PhD in computer science from the University of California, Los Angeles. Contact him at imarkov@eecs.umich.edu.

Resit Sendag is a professor of electrical and computer engineering at the University of Rhode Island. His research interests include microarchitecture, memory systems, and simulation techniques. Sendag received a PhD in electrical engineering from the University of Minnesota, Minneapolis. Contact him at sendag@ele.uri.edu.

Joshua J. Yi is a patent litigation associate at Dechert. His research interests include microarchitecture, reliability, variation-tolerant processor design, and performance methodology. Yi received a PhD in electrical engineering from the University of Minnesota, Minneapolis, and a JD from the University of Texas at Austin. Contact him at joshua.yi@dechert.com.

Derek Chiou is a partner hardware architect at Microsoft and an associate professor at the University of Texas at Austin. His research interests include accelerating datacenter applications and infrastructure; rapid system design; and fast, accurate simulation. Chiou received a PhD in electrical engineering and computer science from the Massachusetts Institute of Technology. Contact him at derek@ece.utexas.edu.