What's Next?

R. Stanley Williams | Hewlett Packard Labs

The end of Moore's law could be the best thing that has happened in computing since the beginning of Moore's law. Confronting the end of an epoch should enable a new era of creativity by encouraging computer scientists to invent biologically inspired devices, circuits, and architectures implemented using recently emerging technologies.

We all knew it was coming, but no one that I knew guessed it would take so long. In 1990, I said it would be 2000 because of the transparency of the gate oxide. In 2000, I said it would be 2010 because of the prohibitive cost of a fab. At some point, the fundamental physical limit of the size of an atom would ensure a hard stop somewhere. Now, the consensus appears to be that the final usable size limit of a field-effect transistor's gate length will be 5 nm,[1] which is essentially the width of nine silicon crystal unit cells. Given the industry's present trajectory, that generation of CMOS will be in production within the next decade. There are still those who predict that 5-nm CMOS will be too expensive, too power hungry, or not reliable enough to achieve the ubiquity and impact of prior generations. I've given up betting against the ingenuity of determined and creative engineers: 5-nm CMOS is inevitable.

Maintaining exponential growth in a technological metric for more than 50 years has been little short of miraculous, but that achievement came at a cost. The effort to scale silicon CMOS overwhelmingly dominated the intellectual and financial capital investments of industry, government, and academia, starving investigations across broad segments of computer science and locking in one dominant model for computers, the von Neumann architecture. Process shrink prowess overwhelmed innovation at the systems level, forcing the market to adapt to a fixed technology rather than having technologies support market needs.

I started to prepare for the end of Moore's law 20 years ago, and what I've realized is that no simple technology shift will put us on a new exponential scaling curve for the next half-century. Every advance

 1521-9615/17/$33.00 © 2017 IEEE Copublished by the IEEE CS and the AIP

from now on might require significant discoveries and inventions across the entire breadth of physical, biological, and computer sciences, or essentially a new miracle every two years. The brutally hard work is just beginning, and in many ways that makes now the most exciting time to be working in the fields of electronics and computing since the days of Shannon, Turing, Bardeen, and von Neumann.

Can't We Just?

Endless discussions and countless workshops have attempted to address the "What's next?" question. Can't we find a replacement for CMOS transistors that we can just drop into our existing supply chain? Surely there must be some way to continue to squeeze out more performance and higher efficiency from an integrated circuit? How about new-age transistors that use different materials (GaAs, anyone?) or physical processes (spin torque?), letting us build circuits that have orders-of-magnitude higher speed with correspondingly lower operational energy than Si? There have been many interesting proposals along these lines, but at least so far, a detailed analysis by Intel's Dmitri Nikonov and Ian Young[2] indicates that nothing proposed to date will exceed the performance of scaled CMOS by enough to make a significant difference at the circuit level.

Can't we just find a better way to use the CMOS process steps we already have? Perhaps the answer is 3D integration or stacking to attain enormous parallelism. However, it seems unlikely to me that the semiconductor industry will be able to continually double the number of layers in a stack every two years or so up to 10^6 to achieve effective areal transistor densities of $10^{18}/cm^2$ (but see above for the success of my previous predictions). Even if they could be built, such "bricks" would require adiabatic or nearly reversible operation to dramatically lower the energy per operation to prevent circuits from vaporizing when powered up. This is possible, but the problem is that it requires very slow clock speeds[3] that squander the struggle to achieve the scaled 3D integration. Certainly, there will be more applications of multilayer integration (that is, 2.5 D) and stacked chips, but this approach is improbable to provide exponential improvements over decades.

Can't we just supplement CMOS with more exotic computation, such as quantum or optical? I see these being important niches someday, likely in a few decades, but not mainstream. Another alternative is, "Can't we just get along within the limits of terminal 5-nm CMOS?" The problem with this approach is that we face exponentially increasing demands for computing, but the power available on the grid will be fixed for some time. By the end of the decade, there will be 8 billion people on Earth, carrying 20 billion mobile devices and interacting with 100 billion intelligent agents, both fixed and autonomously mobile. Providing food, water, education, and employment opportunities that befit the individual dignity of each person is an unprecedented economic and technological challenge. The Internet of Things and big data analysis provide the promise of solving some of the severe problems we face, but the data generated is growing exponentially faster than our ability to turn it into information. Any linear response to an exponential challenge will fail.

Time for Change

As I said, no easy fixes: we'll wind up using every trick we've thought of and continually search for more to scale our computing exponentially in the future. How much further can we go? In principle, there are still many orders-of-magnitude improvement in terms of energy cost per logic, transport, or erasure operation available before computing runs into any fundamental physical limit, if indeed there is such a thing.[3]

A coherent and interdependent set of significant changes to machines and algorithms will be required to maintain extended, exponential scaling in computation. These can't be created in isolation; we don't have the luxury to pursue independent, parallel development isolated from existing infrastructure. In addition, these changes will have to be carefully staged because even if revolutionary changes in computer hardware do occur, they will need to look like evolutionary changes to programmers and application developers. Those of us working to build new computers have to understand that end users can't just stop everything they're doing to rewrite all their legacy code, nor can we wait until an entirely new generation of computer scientists is trained on a transformed curriculum. Thus, new systems must be capable of running existing applications, and do so out of the crate significantly better than current systems to justify the investment in new hardware. The goal of a technology replacement or substitution needs to be that the new system is capable of dramatically better performance than the old technology as programmers learn how to take advantage of additional features, which should be as easy to use as possible, and at the same time enable new applications that were previously

impossible. This isn't a new insight: hardware and software have always had to co-evolve or play leap frog so that both could improve together.

However, there are also fundamental changes occurring in the nature of the computations that people want to perform. Instead of traditional high-performance computing that requires extended precision arithmetic to calculate an answer to 18 or more significant figures, or enterprise computing devoted to performing as systems of record for transactional business logic, today's dominant workloads often involve searching through terabytes or more of stored and streaming data to find something of significance. The metric to classify a highly performing computer is changing from FLOPS (floating-point operations per second) to TEPS (traversed edges per second) or GUPS (giga-updates per second). Probably the only way to significantly improve computational throughput and efficiency in the future will be to design machines that are optimized for a particular task; post Moore's law, there's little hope that improvements in general purpose processors will eventually yield improvements to all existing software.

Instead of general-purpose machines containing many identical processing units, we'll need to automatically and economically design and build bespoke systems that contain heterogeneous computing elements that can work together on a plug-and-play interconnect in response to a customer/application need. Thus, I predict that we'll see the opposite trend in chip design of the past several decades. Dogged pursuit of Moore's law led to increased capture and integration of functions onto general-purpose chips because the increase in performance from the next generation of CMOS meant that the nonrecurring investments involved in designing and fabricating special-purpose chips were quickly lost. Conversely, with the end of transistor scaling, the efficiency and price advantages will go to new types of ASICs as accelerators that are hyperoptimized for a specific computing function and manufacturing cost. For this approach to become reality, there must be an open interface as well as industry standards that enable the heterogeneous technologies to communicate with each other at high bandwidth and low latency, such as the Gen-Z fabric (http://genzconsortium.org). Excess industrial capacity at the penultimate or even older process scaling steps can be reutilized to democratize innovation, allowing a wider range of competitors to contribute to the advancement of computing, and drive a new economics of optimization.

Thus, to launch a new era in computing will require an innovative and flexible platform. One example is a research program that began in Hewlett Packard Labs called The Machine. In what follows, I describe some of this platform's attributes: memory-driven computing utilizing mainly nonvolatile memory instead of DRAM and hard disks, a photonic fabric that connects at the chip level, and a heterogeneous processing environment in which the program is sent to the data. Once such a base platform exists, it can be continually improved by attaching special-purpose accelerators or ASICs to the existing Gen-Z like fabric that perform certain tasks orders of magnitude faster and more efficiently than standard CMOS, which will provide significant net advances for particular applications. One place to find inspiration for such accelerators is in the types of computation performed in brains. Finally, I speculate that the fastest and most energy-efficient computers may be those that don't calculate an answer to a problem—rather, we may teach computers how to first guess an answer and confirm it or to look up answers if they already exist when needed.

Memory-Driven Computing

Nearly a decade ago, I participated in a DARPA-sponsored exercise led by Peter Kogge to determine what would be required to build an exascale computer (one that could in principle perform 10^{18} FLOPS in a scaled-up version) using 2016 technology.[4] The issue that really surprised me was that the actual computation wasn't the limiting factor in terms of time and energy consumption, but rather it was moving data back and forth among processors, cache levels, main memory, and storage, and the energy required to hold state in DRAM chips.

Memory-driven computing[5] addresses these issues by collapsing the memory storage hierarchy and replacing, as much as possible, DRAM, solid-state storage, and hard disks with inexpensive, high-density, low-power nonvolatile memory (see Figure 1). This will open up the von Neumann bottleneck that has plagued computing since the advent of the first stored program computers. The basic idea is that there's a very large core (potentially many petabytes or more) of shared memory holding the entire dataset of interest, surrounded by processing of appropriate types. The advances in photonics described below allow these memory pools to scale economically from gigabits to petabytes, from milliwatts to megawatts, and from tens to hundreds of thousands of computational devices. Memory becomes central and nonvolatile, while computation becomes

peripheral and ephemeral, marshalled to advance the state of memory, and then just as quickly deactivated.

Originally, I/O operations were used for just that, getting data into and out of the computer from peripherals that were actually located on the system's periphery. But over time, we overloaded I/O operations with memory-related functions: interprocess communication, memory sharing, and clustering. Memory-driven computing uses the memory fabric for these functions and returns I/O back to its original role. All other uses are replaced by pointer passing in the shared memory region. For this concept to be viable, the nonvolatile memory must be significantly less expensive per bit to be affordable and much less power hungry than DRAM, but it doesn't have to be quite as fast as long as the data are byte addressable. The simplifications to the hardware and the operating system can still make the system-level performance superior to a conventional hierarchical architecture, in which all data are collected from where they're stored and sent to a central location for processing, at least for many types of data-intensive computing.

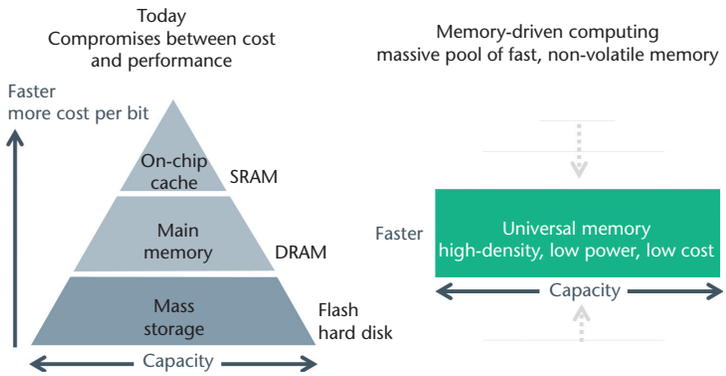

Figure 1. Schematic illustrations of the traditional memory and storage hierarchy that defines computers today (left) and the flattened structure possible with the introduction of inexpensive, high-density, low-power nonvolatile memory (right). Computers today can have a hierarchy of memory and storage types up to 10 layers deep defined by economics; users purchase as much fast memory as they can afford and back that up with cheap storage to hold what doesn't fit in memory. However, the result is that up to 90 percent of a computer's work can be shuffling data between tiers, the von Neumann bottleneck, rather than performing the computations that users require. Nonvolatile memory can replace many of the layers of the hierarchy with a single technology, holding the entire dataset for a problem and eliminating the time and energy wasted on data shuffling.

Photonic Interconnect/Fabric

Once the memory bottleneck is opened up, we're immediately confronted by the need for increased speed and reduced latency in the interconnect. James Meindl predicted that the "tyranny of interconnectors" would eventually become the primary limiting factor for electronics.[6] Photonic interconnect wasn't considered to be viable for computers, primarily because of the expense of hand-built communications-grade gear that was available. However, the technological advantages of photons over electrons in terms of energy dissipation, signal integrity, bandwidth per channel from wavelength division multiplexing breaking through chip in/out limits, and decreased latency because of both intrinsic speed and elimination of repeaters spurred renewed interest and development.[7]

The primary problems to be solved were the energy efficiency of electronic to photonic conversion (and the opposite), coupling laser light onto and off the die, and most importantly the cost of manufacture and assembly. All these issues have been addressed in standard CMOS foundries to produce "silicon photonics," which include microfabricated silicon waveguides, grating couplers, modulators, and detectors integrated directly with analog/digital control electronics. The photonic layer and the electronic control plane can be fabricated on separate dies and then flip-chip bonded to processors, accelerators, or other types of chips and packages to provide the connectivity within the fabric of the computer.[8] Photonic interconnect is now a rapidly growing and advancing engineering discipline that's in the advanced development and early commercialization stages. The potential for dramatic improvements in performance over electronic interconnect were always obvious, but the primary obstacle to adoption was always cost, which is finally being addressed through integration.[9]

Accelerators

The introduction of a memory-driven computing platform enabled by photonic interconnect is a major change to computer architecture, but it's a one-time event that might not be replicated for decades. Once the advantages of a new architecture have been realized, what can then be done to continually improve the performance and efficiency of computer hardware? A possible scenario is that specialty accelerators will be built to address classes of computation or even specific applications, so that computers can be more easily customized for particular problems. Generally, we now think of accelerators for computers as GPUs or FPGAs, and certainly these types of processors offer significant advantages over general-purpose CPUs for certain types of computation. However, they're still conventional CMOS circuits that need to be programmed, which limits their speed and energy efficiency.

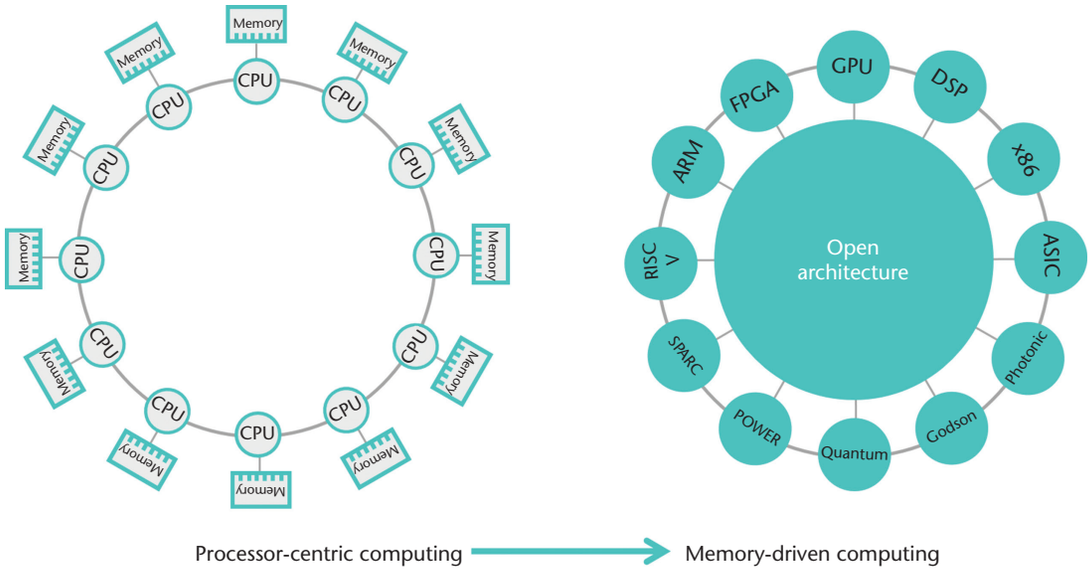

Figure 2. An alternate view of the comparison between processor-centric (left) and memory-driven (right) computing. An open photonic fabric should be able to connect and support a heterogeneous environment with every type of processor, accelerator, or memory module built by any vendor. Using a common platform, computers can be customized to a user's needs by selecting an appropriate set of technologies, whether they're standard components for high-performance computing or specialized neuromorphic chips for search engines. The open nature of the fabric can also ignite innovation by allowing any academic, government or commercial entity that can design a chip to have it fabricated in a foundry and participate in the computing ecosystem.

In many electronic systems, ASICs are designed and built to perform a particular task that can replace an FPGA with a higher speed and efficiency and lower total cost solution. This is only economically feasible if the market for such chips is large enough that the nonrecurring engineering costs associated with designing and manufacturing the ASICs can be amortized sufficiently. Alternately, in the past couple of decades, many of the tasks previously performed by ASICs in a computer have been integrated into CPUs because they had increasingly large numbers of transistors to perform more functions. However, with the end of Moore's law, highly repetitive tasks dominating certain applications could be performed using ASICs that can communicate over a photonic fabric as a peer with a set of heterogeneous processors and other accelerators (see Figure 2). Even significant blocks of code can be translated into hardware using an automated chip designer like PICO (program in, chip out).[10] Thus, end users can obtain significant and continuous performance improvements for their applications through increased customization and availability of new accelerators on top of a memory-driven computing platform with an open fabric.

Once the potential of using ASICs as accelerators has been recognized, we can now think in even broader terms about what new devices and circuits we could build. An example of this is an analog vector-matrix multiplication "dot product engine" based on a crossbar array of memristors, or nonvolatile resistive switches with a continuous range of resistance states,[11] which would be the same basic devices used in the nonvolatile memory described earlier. This is an example of utilizing the physics of a system, in this case Kirchoff's laws, to perform an important computation. The elements in a real-valued positive matrix can be represented by corresponding cell conductances in a crossbar. If a set of voltages that correspond to an input vector are applied to the rows of the crossbar, the output currents on the columns represent the dot product of the vector with the matrix. The time required to perform the computation is essentially a single time step independent of matrix size, instead of $O(n^2)$, where n is the vector length, required for performing the dot product mathematically. This provides orders-of-magnitude improvements in time and energy expenditure for this particular operation even when compared with a pure CMOS ASIC, as long as the application is matched to the bit accuracy of the analog computation.

Accelerating vector-matrix multiplication benefits a range of applications, notably signal processing

(discrete Fourier transforms), image processing (convolutions, filtering), and neural network training and inference. The performance benefits in such cases arise from doing repetitive computations as close to the site of the stored memory as possible.[12] Such new circuits and architectures for computation can further take advantage of low-power, emerging devices,[13,14] and the combined benefits may lead to over five orders-of-magnitude improvement in processing throughput divided by power (ops/sec/Watt) for applications such as convolutional neural networks.[15,16] Although the announcements of the birth of artificial intelligence have been nearly as numerous (and incorrect) as the predictions for the death of Moore's law, multiple breakthrough performances recently achieved with deep learning networks foreshadow a diverse range of applications tackled no longer by software subroutines written by programmers, but by artificial neural networks that learn directly through exposure to data.[17]

Inspiration

Beyond a relatively straightforward accelerator such as the dot product engine, are there other functions that could be expressed in a mixed technology ASIC (such as analog/digital CMOS + devices that perform computation via a physical process)? We can look for further inspiration by better understanding the types of computational primitives and algorithms within brains. Although our understanding of brains today is limited, we know enough now to design and build circuits that can accelerate certain computational tasks. As we learn more about how brains communicate and process information, we should be able to harness that understanding to create a new and hopefully exponential growth path for computing. Even relatively simple neural nets are capable of complex problem solving if they've been trained with a large enough dataset. Will we be able to implement relatively sophisticated tasks, such as recognizing a potential threat from something that has never been seen before to protect computers and datacenters from a first instance of a virus or other attack? How about being able to generalize from experience to recognize when the solution to a problem that has been solved previously can be applied to a new query, or forward a particular problem to a different machine or person that probably already knows the answer? At this stage, these are just interesting speculations, but they could become legitimate subjects for study within computer science as neurophysiologists learn more about the circuitry and psychologists learn more about algorithms in brains.

The end of Moore's law could be the best thing that has happened to computing in decades. It will force those who build computers to be more creative, to explore new realms in computer science and try out different architectures, circuits, device concepts, physics, and materials, and to look for new computational paradigms from biology and elsewhere. The goal isn't to replicate the brain or exactly duplicate any of its functions, but rather to find inspiration for new ways of solving problems with a computer and utilizing them in optimized systems for the applications that need them. Science and engineering may finally be ready to tackle the aspirations that Turing and von Neumann had for computers in the 1940s, or the flowering of creativity could take us in completely new and unanticipated directions. Will there be a new technology that enables exponential performance improvements for decades to come? We don't know for certain, and that's what makes the future of computing so exciting. ■

Acknowledgments

I thank my many colleagues at Hewlett Packard Labs with whom I've collaborated over two decades for educating me about how computers are actually used and what technologies can make them more useful. In particular, R.G. Beausoleil, K.M. Bresniker, R. Lewington, and J.P. Strachan provided important feedback and assistance on this manuscript.

References

1. J.M. Shalf and R. Leland, "Computing Beyond Moore's Law," *Computer*, vol. 48, no. 12, 2015, pp. 14–23.
2. 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.
3. R.P. Feynman, *Feynman Lectures on Computation*, T. Hey and R.W. Allen, eds., Addison-Wesley, 1996, pp. 137–184.
4. K. Bergman et al., "Exascale Computing Study: Technology Challenges in Achieving Exascale Systems," tech. report, vol. 15, Defense Advanced Research Projects Agency Information Processing Techniques Office (DARPA IPTO), 2008.
5. K.M. Bresniker, S. Singhal, and R.S. Williams, "Adapting to Thrive in a New Economy of Memory Abundance," *Computer*, vol. 48, no. 12, 2015, pp. 44–53.
6. J.D. Meindl, "Beyond Moore's Law: The Interconnect Era," *Computing in Science & Eng.*, Jan./Feb. 2003, pp. 20–24.

7. R.G. Beausoleil et al., "Nano-Electronic and –Photonic Interconnect," *Proc. IEEE*, vol. 96, no. 2, 2008, pp. 230–247.
8. R.G. Beausoleil, "Large-Scale Integrated Photonics for High-Performance Interconnects," A*CM J. Emerging Technologies in Computing Systems*, vol. 7, no. 2, 2011, article no. 6.
9. S. Rumley et al., "Silicon Photonics for Exascale Systems [Invited Tutorial]," *J. Lightwave Technology*, vol. 33, no. 4, 2015, pp. 547–562.
10. V. Kathail et al., "PICO: Automatically Designing Custom Computers," *Computer*, vol. 35, no. 9, 2002, pp. 39–47.
11. M. Hu et al., "Dot-Product Engine for Neuromorphic Computing: Programming 1T1M Crossbar to Accelerate Matrix-Vector Multiplication," *Proc. 53rd Annual Design Automation Conf.*, 2016; doi 10.1145/2897937.2898010.
12. Y. Chen et al., "DaDianNao: A Machine-Learning Supercomputer," *Proc. 47th Annual IEEE/ACM Int'l Symp. Microarchitecture*, 2014, pp. 609–622.
13. M. Suri et al., "Phase Change Memory as Synapse for Ultra-Dense Neuromorphic Systems: Application to Complex Visual Pattern Extraction," *IEDM Technical Digest*, 2011, pp. 4.4.1–4.4.4
14. G.W. Burr et al., "Experimental Demonstration and Tolerancing of a Large-Scale Neural Network (165,000 Synapses) Using Phase-Change Memory as the Synaptic Weight Element," *IEEE Trans. Electron Devices*, Nov. 2015, pp. 3498–3507.
15. T. Gokmen and Y. Vlasov, "Acceleration of Deep Neural Network Training with Resistive Cross-Point Devices," arXiv:1603.07341 [cs.LG], 2016.
16. 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.
17. E.P. DeBenedictis, "Rebooting Computers as Learning Machines," *Computer*, vol. 49, no. 6, 2016, pp. 84–87.

R. Stanley Williams is a Senior Fellow in the Systems Architecture Lab at Hewlett Packard Labs. His research interests are in nano and nonlinear electronics, chaos, computation, and cognition. Williams received a PhD in physical chemistry from the University of California Berkeley. He's a Senior Member of IEEE. Contact him at stan.williams@hpe.com.