The Role of Processor Architecture in Computer Science

Abstract

Kevin Skadron
Dept. of Computer Science
University of Virginia
Charlottesville, VA 22904

1. Introduction

The purpose of this abstract is to briefly explain some of the issues facing processor architects, some of the techniques we use, and why these matters form an integral part of computer science. By *processor architecture*, I mean the study of the organization of resources and the representation of data within the computer processor. The larger specialty of *computer architecture* encompasses many other related issues in computer systems, like multi-processing and input/output, but for the sake of this short abstract there are enough exciting challenges in the specific area of processor architecture.

Clearly, an understanding of processor architecture is an integral part of a computer science education. The processor's physical implementation and the way it represents data combine to control how computer scientists and others use computers to manipulate information. Knowing what processors are capable of and how they operate in various circumstances guides research in diverse areas of computer science like algorithms, programming languages, fault tolerance, security, and computational science.

But continuing *research* in processor architecture is also an integral part of computer science. Because the architecture dictates computers' capability and how computers are used, improvements in architecture are a powerful enabling technology for other advances in computer science and other computational fields. While by no means an exhaustive list of interesting topics in the area, this abstract focuses on four major areas of challenge in processor architecture today: speed, power, workload characterization, and nanoscale computing.

2. Challenge Areas in Processor Architecture

Speed

Computer processors, or CPUs, have shrunk in size and improved in speed at a breathtaking rate in accordance with the oft-cited Moore's Law [Moore65]. Yet the trends observed by Moore merely provide more and faster transistors. Often neglected is the fact that much of the ongoing improvement in processing speed comes from architecture-level innovations that put the additional transistors to new uses. Recent examples include better instruction fetch via increasingly sophisticated branch prediction techniques, the ability to execute instructions out of program order, and more intelligent on-chip cache memories for frequently-used data—all of which boost execution throughput and are synergistic with increases in clock speed.

Computational speed remains one of the core concerns of computer science. Many applications in the fields of medicine, the physical sciences, the military, business, and of course computer and information science still cannot cope with desired problem sizes because the CPU is not fast enough. Examples include real-time image analysis for medical diagnosis and military target recognition, modeling of physical systems such as weather, real-time control of magnetic bearings for applications like energy-storage flywheels, and perhaps ironically, processor simulation. In addition to these recognized applications, further increases in speed will enable new applications not even anticipated today. Of course, new algorithms are also important in making new applications feasible, yet many challenging applications already use provably optimal algorithms. Research to continue making processors faster is therefore of paramount importance as an enabling technology for advances in so many other fields.

Multiprocessors are not a simple solution to these computational demands, because programming multiprocessor systems is difficult and error-prone, and many problems have such irregular structures or such fine-grained parallelism that they have not yet been shown capable of better performance with multiple processors. A

Prepared for the National Academy of Science's Computer Science and Telecommunications Board's
Committee on the Fundamentals of Computer Science, June-July 2001.

further problem is that the complex timing of events in multiprocessor systems makes their use especially difficult for real-time applications.

The example applications cited above represent the high-performance end of computational requirements. For many other applications, factors such as cost and power are more important design choices. Yet many such "embedded systems" applications are still constrained by performance, because meeting these cost and power requirements today comes at a severe cost in speed. For example, processing speed is one of the primary determinants of printing speed [Fisher98], and CPUs in today's palmtop computers are not capable of speed recognition, even though speech is one of the most promising user interfaces for such devices. Faster processors that can still meet cost and power requirements are therefore an enabling technology for better systems, new applications, and more intuitive user interfaces. Indeed, questions of processing speed pervade all areas of computer science, from algorithms to software engineering to user interface, by dictating what is feasible or practical in these areas.

Power

With the advent of mobile, battery-operated devices, the power consumption of CPUs becomes a major determinant of battery life. Battery life affects the choice of battery size and hence form factor and even feasibility, and so power considerations ultimately affect usability in several ways.

In addition to battery life, heat dissipation is an equally important consideration. Fred Pollack of Intel has pointed out that the power density of today's high-performance processors already exceeds that of a kitchen hot plate and is within an order of magnitude of the power density found in a nuclear reactor [Pollack99]. These processors must therefore be constructed with expensive thermal packages. Worse yet, these packages must also be over-designed, capable of withstanding long periods of peak execution that rarely occur in practice but cannot be ruled out.

Temperature is also important for embedded systems, where processors often dissipate less heat than do high-performance processors, but where cooling devices like fans and large heat sinks are not available. Embedded systems, like palmtop computing and "smart" objects and appliances, are of great interest to a variety of areas in computer science: networking, software engineering, human-computer interfaces, and even algorithms. Yet battery life and temperature are primary considerations in choosing whether a sufficiently powerful processor can be used in such devices. To be practical and cost-effective, these embedded processors must enjoy a long battery life (in many cases the device is not amenable to battery replacement) and operate at reasonable temperatures (especially for devices that are worn or held). Improvements in power consumption will improve battery life and heat dissipation, permitting a wider range of functionality in these smart devices and expanding the opportunities for research in other areas of computer science that use or study embedded systems.

All these issues motivate the development of techniques for reducing power consumption in the CPU. Unfortunately, reducing power consumption usually comes at the cost of reduced speed. Common techniques include cutting voltage or clock frequency and selectively turning off portions of the processor. This works well for applications that are not computationally demanding, but inhibits execution speed for applications that are. Questions of speed and power are therefore closely linked. By finding ways to link the manipulation of power and speed, architects can play an important role in reducing power dissipation while preserving performance. But the relationship between power and speed for various inter-related architectural features requires further understanding.

Workload Characterization

New techniques for controlling power and/or improving speed require the ability to simulate the behavior of hypothetical designs while executing realistic workloads, and the ability to characterize a multi-faceted parameter space to understand how the many design choices available interact, their tradeoffs, and optimal design points.

Unfortunately, these simulations attempt to model tomorrow's workloads and tomorrow's more sophisticated designs, yet these simulations must run on today's comparatively slower processors. This either requires immense computing resources or means that these systems cannot be fully simulated. This in turn requires the use of either much faster analytical models of processor behavior, or sampling techniques. Both are important areas of research in the architecture community. Yet analytical models have not yet gained acceptance for processor architecture because of the many simplifying assumptions these models must currently make. Sampling on the other hand is prevalent, because current simulators are simply too slow to do otherwise. Unfortunately, the sampling techniques currently in use are crude. Techniques are needed that more aggressively reduce simulation time but provide more rigorous bounds on possible error. Finally, in order to gain the quickest adoption, both improved analytical models and improved sampling techniques must be easy to integrate into existing processor models.

A further problem is that there is little consensus in the architecture community about exactly how much detail is required for modeling processor behavior, nor what actually constitutes a realistic workload, nor how aggressively

workloads may be sampled. All these choices dictate the simulations' computational requirements. The choice of detail also dictates development time for the simulator, and the choice of workload and sampling technique dictate how confident we can be that results are accurate and that important tradeoffs are actually being discovered.

Solutions to these problems will permit faster design of new processors, not only by reducing time spent on pre-silicon modeling, but also by minimizing time spent on post-silicon design errors.

Nanoscale Computing

To address the preceding themes, architects are pursuing a wide range of research, from ever more aggressive ways to exploit parallelism within a single chip, to new sampling techniques with more rigorous bounds on resulting error, to dynamic adaptability within the processor in order to rapidly respond to changing workload characteristics and provide the best possible power/performance characteristics.

These techniques can be immediately applied to contemporary processor designs. On a much longer time horizon, architects are beginning to consider the implications of new, nanoscale materials for electronics that have recently been proposed in lieu of silicon and offer the promise of much greater densities and hence greater capabilities and processing speeds. These materials may also require less power than silicon electronics.

Quantum cellular automata and molecular electronics are just two examples of new computing media that are generating great interest. Because these materials lead to different electronic structures, the conventional architectures used in silicon-based computing may no longer be the best organization of resources. Indeed, a recent paper by Niemier and Kogge [Niemier01] has explored the architecture of a processor built on quantum cellular automata and found a variety of important differences, like the ability to make wires serve double duty and also serve as computational elements. A further consideration is that recently-proposed materials are expected to exhibit various constraints that make their effective deployment difficult. For example, a recent paper by Ellenbogen and Love [Ellenbogen00] expresses concern that molecular electronics will not provide competitive switching times compared even to today's silicon processors, and will not scale in size and speed in the way we have come to expect with silicon. Another consideration is that, for manufacturing reasons, these and other new materials will likely show the greatest benefits for architectures with many replicated structures. The custom logic that makes up large portions of today's processors may perform relatively poorly by comparison. Considerations like these mean that the design of successful architectures that capture the benefits of new materials and mitigate their weaknesses is essential to the success of nanoscale computing, and is closely entwined with ongoing research in materials design and computer algorithms.

Overall, these new materials—and other novel computing systems like quantum computing with entangled states—fundamentally change the way data is represented and manipulated within the computer, and hence the way in which information is processed. This requires new architectures whose adoption as the basis for future CPUs will create changes that ripple though all areas of computer science, from algorithms to security to software engineering.

It is actually not clear whether any of the newly proposed materials will offer the same scaling that we enjoy with silicon. This means that if silicon-based electronics eventually stop scaling according to Moore's Law and no replacement is available, processor design will enter an era of "zero-sum" architecture in which resources are fixed and gains in performance or new functionality must come at the expense of larger and slower chips, higher power dissipation, or the sacrifice of some other on-chip capability—yet another challenge for architects.

3. Summary

It may seem that these questions are "just engineering" and do not constitute a core intellectual pursuit of computer science. Yet solutions to many of these questions require a thorough understanding of basic aspects of processor behavior, of the intrinsic properties of computational workloads, and of the interactions among all these factors. It may also seem that the most important of these questions is the long-term investigation of nanoscale materials. Yet nanoscale computing may prove an elusive goal. In the meantime, computer scientists and other users of information technology face immediate substantial problems that require continuing improvements in processing speed and continuing reductions in power and heat dissipation. All four challenge areas described in this abstract therefore constitute vibrant and vital areas of research in computer science.

Because the computer processor's implementation and the way it represents and manipulates data is so fundamental to computer science, processor architecture is a vital component of both computer science education and research. A solid foundation in architecture provides computer scientists with the tools necessary to most effectively solve their problems with the hardware available. And further improvements in characterizing workloads and designing fast but power-efficient processors are essential for continuing advances in fields ranging from computer security to medicine.

References

[Ellenbogen00] J. C. Ellenbogen and J. C. Love, "Architectures for Molecular Electronic Computers: 1. Logic Structures and an Adder Designed form Molecular Electronic Diodes." In *Proceedings of the IEEE*, pp. 386-426, Mar. 2000.

[Fisher98] J. Fisher, "Will Customization Become a Big Factor in Computer Architectures?" Keynote presentation at the 31st Annual International ACM/IEEE Symposium on Microarchitecture, Dec. 1998.

[Moore65] G. E. Moore, "Cramming More Components onto Integrated Circuits." In *Electronics*, pp. 114-17, Apr. 1965.

[Niemier01] M. Niemier and P. Kogge, "Exploring and Exploiting Wire-Level Pipelining in Emerging Technologies." To appear in *Proceedings of the 28th Annual International Symposium on Computer Architecture*, July 2001.

[Pollack99] F. Pollack, "New Microarchitecture Challenges in the Coming Generations of CMOS Process Technologies." Keynote presentation at the 32nd Annual International ACM/IEEE Symposium on Microarchitecture, Nov. 1999.