Guest Editors' Introduction

Billion-Transistor Architectures

Advances in semiconductor manufacturing will permit an unprecedented number of transistors on a single processor die. But what architecture will make the best use of these riches?

Doug Burger
James R. Goodman
University of Wisconsin-Madison

The circumstances in which computer architects will find themselves in the next 15 years are truly daunting. By the end of this period, microprocessors will have more than a billion logic transistors on a single chip. Tiny transistors and wires will have feature sizes less than a tenth of a micron. The time required to send a signal along an on-chip wire will become proportionately much greater than that needed for a transistor to switch. Off-chip communication will become relatively slower. Minimizing power dissipation and the resultant heat will be paramount, despite reduced voltage levels. Although these predictions are not controversial, their implications for microprocessor architectures certainly are.

THE DEBATE

During an informal discussion at the 1996 International Symposium on Computer Architecture (ISCA-23), several architects had a boisterous discussion (read argument) over the direction that future architectures will take. Our community generally understands the evolving possibilities and the underlying constraints. The rate of progress is so great, however, that radical models easily dismissed only a few years ago are now feasible, and there is little agreement on which models are likely to achieve dominance. We organized this special issue because our discussion at ISCA was sufficiently controversial and interesting to appeal to a wide audience. Our goals for this issue are to explore both the trends that will affect future architectures and the space of these architectures.

ADDRESSING THE DEBATE

The articles in this issue fall into two categories. The first category contains three articles, which appear in Cybersquare. Each describes one trend that will affect future microprocessor architectures. In the second category, each article makes the case for a different billion-transistor architecture. Although these articles represent the state of the art and the authors' best guesses, the future is notoriously hard to predict in our breakneck-paced field. Technology trends are generally easier to predict than their effects, but trend estimates can be wildly inaccurate. Intel's 1989 prediction for 1996 processors underestimated performance by a factor of four.[1] Forecasting the effects of technology is even harder, as illustrated by several well-known quotes:

- "Everything that can be invented has been invented." US Commissioner of Patents, 1899.
- "I think there is a world market for about five computers." Thomas J. Watson Sr., IBM founder, 1943.
- "There is no reason for any individuals to have a computer in their home." Ken Olsen, CEO of Digital Equipment Corp., 1977.
- "The current rate of progress can't continue much longer. We're facing fundamental problems that we didn't have to deal with before." Various computer technologists, 1955-1997.

Given the wide scope of these articles and the credibility of the authors, however, it is certain that many of the ideas discussed in this issue will be incorporated into the first billion-transistor processor.

FUTURE TRENDS

Before we discuss the possible alternatives for microprocessors' evolution, it is important to understand the driving factors, five of which we discuss next.

Hardware trends and physical limits

In its 1994 road map,[2] the Semiconductor Industry Association predicted the course of semiconductor technology over the next 15 years. The SIA predicted that by 2010, industry would be manufacturing 800-million-transistor processors with thousands of pins, a 1,000-bit bus, and clock speeds over 2 GHz. Such chips would produce a predicted maximum of 180 W (allowing the computer of 2010 to serve also as a barbecue grill or space heater).

The most important physical trend, however, is the fact that on-chip wires are becoming much slower relative to logic gates as the on-chip devices shrink. It will soon be impossible to maintain one global clock over the entire chip. Sending signals across a billion-transistor processor may require as many as 20 cycles. In the first of the short trend articles, Doug Matzke of Texas Instruments describes these effects in detail.

System software

Much of the performance gain in recent years has come from exploitation of parallelism, as processors overlap multiple instructions (pipelining) and simultaneously execute multiple instructions (superscalar execution). It is probable that future processors will harvest significantly more parallelism; the question is

0018-9162/97/$10.00 © 1997 IEEE

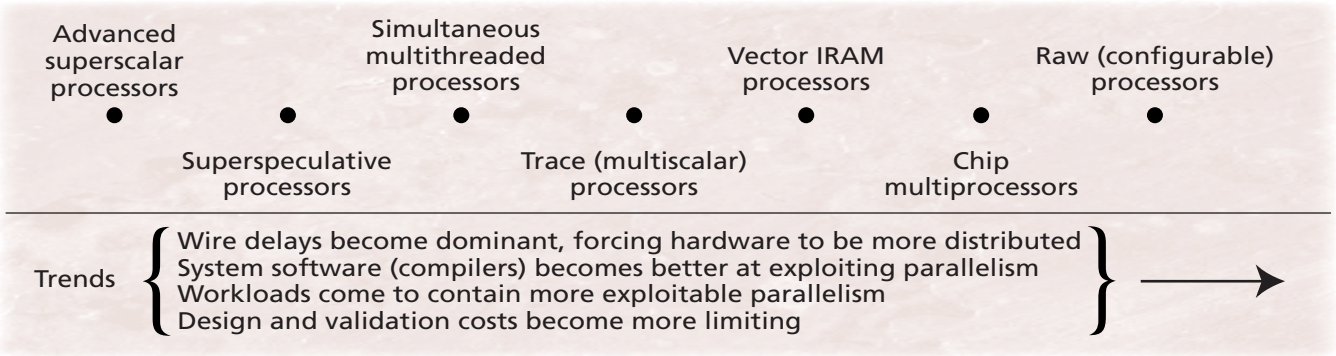

Figure 1. Surveyed processor architectures.

whether hardware alone will continue to extract that parallelism—the norm today—or whether the compiler and runtime system software will also play a key role. A quantum leap in compilers' ability to automatically extract parallelism from code would have enormous ramifications for future architectures. The issue of compatibility with legacy software also hinders architectural innovation. The second of the trend articles, by Joseph A. Fisher of Hewlett-Packard Labs, addresses these issues.

Future workloads

Architectural design is driven by the dominant anticipated workload. The most important workloads will certainly change over the next two decades and are perhaps the most difficult trend (of those discussed here) to predict. While future markets will undoubtedly support more customized, application-specific processors, this issue focuses on high-performance, general-purpose processors. For such chips, there is a consensus that the user interface will consume a greater proportion of processors' power, and that multimedia workloads will continue to grow in importance. The third trend article, by Keith Diefendorff of Apple and Pradeep Dubey of IBM, therefore describes the architectural implications of multimedia workloads.

Design, verification, and testing

Modern, high-end microprocessors are so complex that their design teams now consist of hundreds of engineers. In addition to the difficulty of managing the design, both verifying that the design works correctly and testing each finished chip have become a major component of the design cycle. Validation and testing now account for 40 to 50 percent of an Intel chip's design cost, and 6 percent of the transistors (for built-in self-test) on the Pentium Pro.[1] If these trends persist, architectures that simplify the interaction among on-chip components and/or reduce the number of interacting components (thus lending themselves to faster design and validation) will have a greater advantage over architectures that do not.

Economies of scale

Fabrication plants now cost about $2 billion, a factor of ten more than a decade ago.[1] Manufacturers can only sustain such development costs if larger markets with greater economies of scale emerge. These larger markets may drive substantially different workloads, thus affecting the architecture. For instance, the primary beneficiaries of the microprocessor revolution so far have been science and business. To open larger markets, microprocessor-based systems must offer the average consumer more than spreadsheets and Web browsing. Large markets imply the mass marketing of computer chips (recently evidenced by Intel's chromatic, disco-dancing MMX designers). Whether advertising considerations will eventually affect architectures is an open question.

We do not address the last two factors in the trend articles, but they are important nonetheless. We next describes several specific billion-transistor architectures and how they might evolve in light of these trends.

FUTURE ARCHITECTURES

Figure 1 depicts the processor architectures that the articles cover, organized in a loose order along the horizontal axis. The direction that future architectures take will be partially determined by the trends discussed in the previous section, four of which are listed in Figure 1. As these effects become more significant, they will drive architectures toward the right of Figure 1.

These trends, however, are counterbalanced by the importance of maintaining software compatibility and retaining the current programming model. If the market continues to insist on compatibility with legacy code, the market will prevent architectures from evolving toward those on the right of the figure. The farther to the right the architectures are, the more they depart from current programming models and practices.

Although this ordered model does not apply perfectly to every trend, it is a useful illustration. With this framework in mind, the surveyed architectures are as follows:

- *Advanced superscalar processors* will scale up from current designs to issue 16 or 32 instructions per cycle.
- *Superspeculative processors* enhance wide-issue superscalar performance by speculating aggressively at every point in the processor pipeline.
- *Simultaneous multithreaded processors* share an aggressive pipeline among multiple tasks when there is insufficient instruction-level parallelism (ILP) in any one task to fully use the pipeline. (Due to space limitations, this important part of the spectrum of architectures will appear in the

September-October issue of *IEEE Micro*.)

- *Trace processors* facilitate high ILP and a fast clock by breaking up the processor into multiple distinct cores, and breaking up the program into traces (dynamic sequences of instructions). One core executes the current trace while the other cores execute future traces speculatively.
- *Vector IRAM processors* couple vector processor execution with large, high-bandwidth, on-chip DRAM banks, which provide the vector units with sufficient bandwidth at a reasonable cost.
- *Chip multiprocessors* (CMPs) place a small number of distinct processors (four to 16) on a single chip and run parallel programs and/or multiple independent tasks on these processors.
- *Raw processors* implement highly parallel architectures with hundreds of tiles—very simple processors, each with some reconfigurable logic—on a single chip, controlling execution and communication almost entirely in software.

The article summaries on pp. 26-27 contain detailed descriptions of the articles supporting each architecture.

COMMONALITIES

The three uniprocessor articles (advanced superscalar, superspeculative, and trace processors) have significant similarities: each maintains compatibility with old binaries, and each argues for trace caches, better branch prediction, and data value speculation. These three articles have different foci, however. Unlike the advanced superscalar proposal, the trace processor is almost completely distributed and focuses on a coarser grained parallel execution of separate traces. The superspeculative proposal, conversely, proposes aggressive, fine-grained speculation at every point in a unified pipeline.

The multiprocessor articles all argue that increasing design complexity and clock speed limitations will force designers to replicate a number of small fast processors on a single chip. The articles differ in the actual size and number of the processor cores. These articles all agree on one other issue: Compilers, no matter how good, will never be able to effectively parallelize all tasks. The solution described in both the CMP and Raw articles is the same as that first proposed for multiscalar processors: Treat consecutive portions of the dynamic instruction stream as speculative threads. (See the sidebar "Multiscalar: Another Fourth-Generation Processor," p. 72.) Although other solutions have been proposed, the concept of speculative, temporal threads is likely to grow in importance as a means of finding parallelism.

All the articles agree that future processors will have large on-chip memory capacities, and all but two assume that large, multimegabyte, level-two caches will be the norm. The vector IRAM article argues that much or all of the system main memory will exist on-chip, due to the greater density that on-chip DRAM banks provide. The Raw article assumes that, instead of a large, centralized on-chip cache, the on-chip memory will be finely distributed among the tiles.

This is an exciting time to be an architect. On-chip transistor budgets will soon allow virtually anything to be implemented—designers' imaginations will likely be one of the prime limitations. It's possible that the pace of semiconductor technology advances will slow due to cost or market constraints, or the obstacles posed by quantum effects (thus spurring growth in the customized processor and/or parallel computer markets). It's also possible (and we believe more likely) that semiconductor technology will not reach any fundamental limits for decades. In either event, the road to billion-transistor processors and beyond will continue to be a wild ride. ❖

Acknowledgments

We gratefully acknowledge the large number of people who submitted outstanding papers, over one hundred referees, and especially our contributors, who were forced to deal with one micromanaging and one macromanaging guest editor.

References

1. A. Yu, "The Future of Microprocessors," *IEEE Micro*, Dec. 1996, pp. 46-53.
2. *The National Technology Roadmap for Semiconductors*, Semiconductor Industry Assoc., San Jose, Calif., 1994.

Doug Burger *is a PhD candidate at the University of Wisconsin-Madison. His dissertation research concerns memory hierarchies for advanced microprocessors. Burger received an MS from the University of Wisconsin-Madison and a BS from Yale University, both in computer science. He is an Intel Foundation Graduate Fellow, and is also a student member of the IEEE, the Computer Society, and the ACM.*

James R. Goodman *is a professor of computer sciences at the University of Wisconsin-Madison. His current research focuses on high-performance memory systems and computer systems of the future. Goodman received a PhD from the University of California at Berkeley. An early contributor to multiprocessor cache-snooping literature, Goodman has actively participated in the development of IEEE Std 896 (Futurebus) and 1596 (Scalable Coherent Interface). He has published papers in the area of cache-coherence algorithms, shared-memory multiprocessor architectures, database systems, interconnection networks, virtual memory, memory-register organization, and memory systems design.*

Contact the authors at the Computer Sciences Dept., Univ. of Wisconsin-Madison, 1210 West Dayton St., Madison, WI 53706; {dburger,goodman}@cs.wisc. edu.

Simultaneous Multithreading: A Platform for Next-Generation Processors

Susan J. Eggers, Joel S. Emer, Henry M. Levy, Jack L. Lo, Rebecca L. Stamm, and Dean M. Tullsen

This article presents *simultaneous multithreading* (SMT), a processor design that improves performance by increasing utilization of all processor resources, both memories and functional units. An SMT processor combines hardware features seen in two other types of processors: From wide-issue superscalars it inherits the ability to issue multiple instructions per cycle. Like multithreaded processors it holds the hardware state (such as registers, PC, and so on) for several programs and/or threads at once. The result is a processor that can issue multiple instructions from multiple threads *each cycle*.

Simultaneous multithreading differs significantly from other architectures in its ability to effectively exploit *all* types of parallelism. When a program has only a single thread, all of an SMT processor's resources can be dedicated to that thread; however, when thread-level parallelism (TLP) exists, this parallelism can compensate for a lack of per-thread instruction-level parallelism (ILP), which causes substantial underutilization of processor resources on current superscalars. By using all types of parallelism (both ILP and TLP) together, an SMT processor utilizes the functional units more effectively. It achieves the twin goals of greater instruction throughput and significant program speedup on multithreaded and multiprogramming workloads, while maintaining the same level of performance for single-threaded programs.

The article's simulation results demonstrate that an SMT processor can achieve significant gains in utilization relative to a wide-issue (single-threaded) superscalar, as well as significant speedups for parallel programs compared to small-scale on-chip multiprocessors. It also shows that SMT adds minimal hardware complexity to today's advanced, dynamically scheduled microprocessors. Given the performance potential and the straightforward path from existing machines, we expect to see SMT processors in the near future.

This article will appear in the September-October issue of *IEEE Micro*.