SEMICONDUCTORS / PROCESSORS

FEATURE

Multicore CPUs: Processor Proliferation

From multicore to many-core to hard-to-describe-in-a-single-word core

By SAMUEL K. MOORE / JANUARY 2011

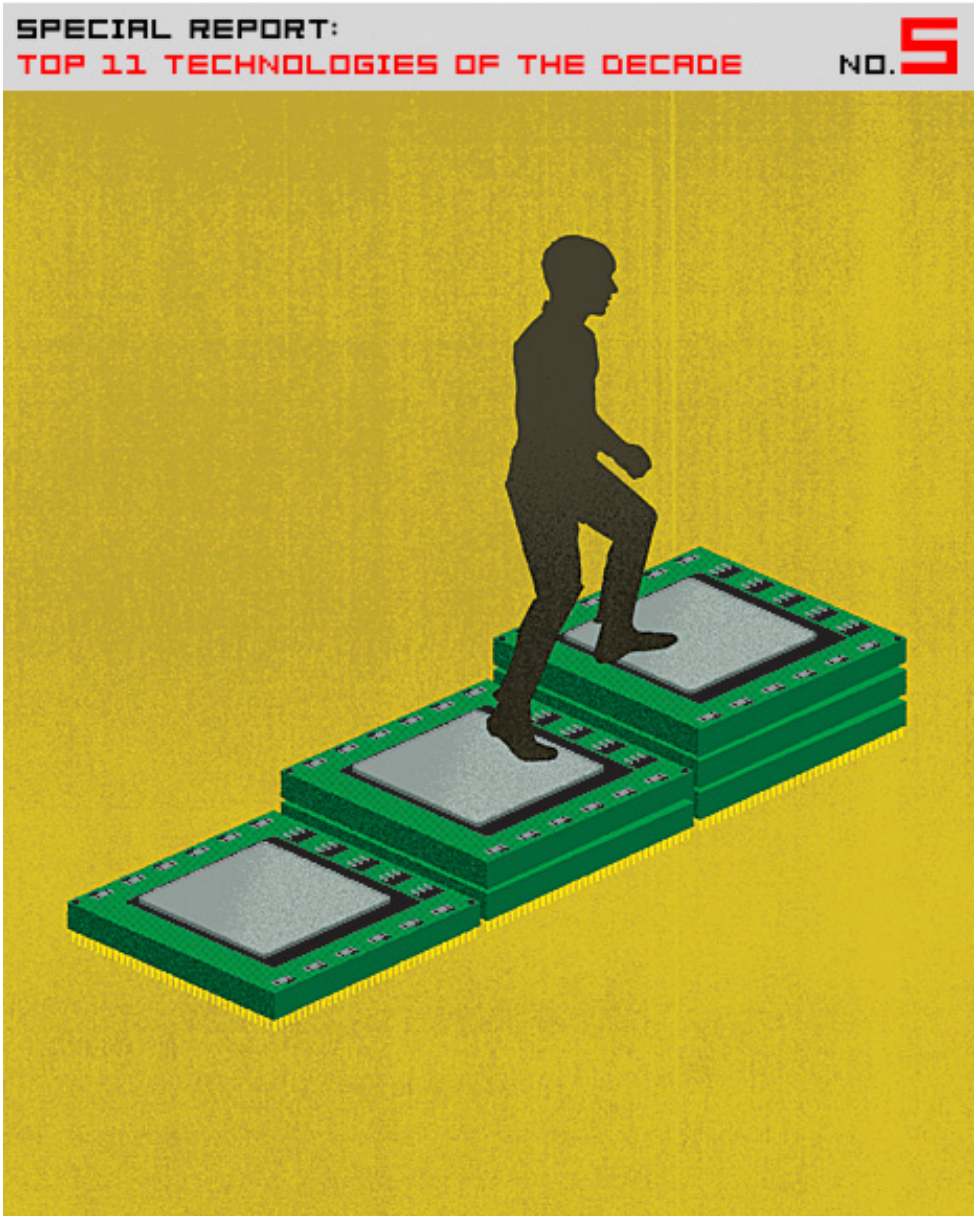

Illustration: Frank Chimero

This is part of IEEE Spectrum's special report: Top 11 Technologies of the Decade

Back in 1994, programmers figured that whatever code they wrote would run at least 50 percent faster on a 1995 machine and 50 percent faster still on a '96 system. Coding would continue as it always had, with instructions designed to be executed one after the other.

But Kunle Olukotun, then a newly minted professor of electrical engineering at Stanford, saw that the party couldn't go on forever. The microprocessors of the day couldn't scale up as efficiently as you'd expect through the mere addition of ever more and ever faster transistors, the two things that Moore's Law provided.

To solve that problem, Olukotun and his students designed the first general-purpose multicore CPU. This idea, more than any other in the past decade, is what has kept the semiconductor industry climbing the Moore's Law performance curve. Without multicore chips, the computing capability of everything from servers to netbooks would not be much better than it was a decade ago. Everyone's happy—except perhaps for the programmers, who must now write code with threads of instructions that must be executed together—in pairs, quartets, or even larger groupings.

It's not that old, single-core CPUs weren't already doing some parallel processing. When Olukotun began his work, most microprocessors had a "superscalar" architecture. In the superscalar scheme, the CPU contained many replicated

components, such as arithmetic units. Individual instructions would be parceled out to the waiting components. Scaling up such "instruction-level parallelism" meant building in more and more such components as the years rolled by.

Olukotun argued that within a few more generations, it wasn't going to be worth the effort. You needed to provide a quadratic increase in resources for a linear increase in performance, he said, because of the complexity of the logic involved in parceling out and keeping track of all the instructions. If you combined that with the delays inherent in the mess of interconnects among all those parts, it seemed a losing proposition. Doug Burger and Stephen Keckler, both computer scientists at the University of Texas, Austin, put a finer point on it later in the decade, calculating that instead of the 50 percent improvements everyone had gotten used to, the computing industry should start thinking 12.5 percent. And 12.5 percent isn't much of a reason to buy a new computer, is it?

Olukotun's answer was Hydra, a processor whose parallelism came not from redundant circuits within a single complex CPU but from building four copies of a simpler CPU core on one chip. That way, you save on interconnects and on the time lost casting instructions out and reeling answers back in. In Hydra, you got parallel processing without all the delay-inducing complexity. In 1998 "we wrapped up the hardware portion of the project and declared victory," says Olukotun.

It was a quiet victory. In the computing environment of the 1990s, Hydra seemed a little crazy, Olukotun says. Superscalar designs were still delivering 50 percent performance improvements every year. "It was by no means clear at the time that our view of the world was going to win," he recalls. And indeed it would be years before processor giants like Intel, Advanced Micro Devices, and IBM got the multicore religion Olukotun preached. And when they did, it would largely be for a reason he had hardly considered: power.

It turned out that the rising density of transistors on a chip intensified the hot spots in CPUs. This, even more than the resource-to-performance ratio that had bothered Olukotun, was the problem that seemed most likely to stop Moore's Law in its tracks. In presentations in 1999 and later, Intel engineers showed that if trends in microprocessors were to continue, by 2010 they'd burn as hot as the surface of the sun.

The answer was clear: Slow down the CPU's clock and add more cores. That way, you'd gain more from the extra parallelism than you lost from the slower processing. The chip would gobble less power and generate less heat.

It was a daunting engineering job, but the big processor makers were more prepared than you might expect, because they'd already redesigned the way that CPUs communicate with other chips. For Intel, the solution, called the front-side bus, debuted in 1996, in the Pentium Pro. According to Intel senior fellow Steve Pawlowski, the bus was, in large part, originally meant to save on testing and validation costs. It was a very convenient piece of luck, because when the time came to get two cores working together, the front-side bus was there, waiting to link them up. And in 2005 Intel released its first dual-core component, the Pentium D, which was really two single-core chips in the same package, tied together by the front-side bus.

Engineers at AMD—influenced by Olukotun, Burger, and Keckler—were more purposeful. They prepped the initial, single-core version of AMD's breakout server chip, the Opteron, with a redesigned communications component that would make a multicore version easy. That version came out in 2005. The component was the chip's "northbridge," a switchyard that acts as the chip's gateway to other chips in the computer.

IBM was, arguably, even more on top of the multicore revolution. Around the same time that Intel's Pentium Pro was released, the company began work on its Power4 processor. Looking for an advantage, IBM entertained a number of cutting-edge ways to enhance instruction-level parallelism in single cores, according to Jim Kahle, chief architect of that design. But, deciding to play it safe, his team rejected each. "Turned out to be a good idea," he says. The most conservative option was a dual-core processor. And so Power4, released in 2001, became the first mainstream computer processor with more than one core on a single die.

Olukotun himself wasn't absent from the revolution he predicted. In 2000, he took the lessons from Hydra and founded Afara Websystems. That start-up was acquired by Sun Microsystems in 2002, and its technology became Sun's powerful Web server CPU, the eight-core UltraSparc T1 (also known as Niagara), released in 2005.

"Cores are the new transistors"
Kunle Olukotun, Stanford University

Once the multicore revolution got going, it had a natural momentum. "As soon as we got to two cores, it became obvious we needed to start thinking about going to four," says AMD corporate fellow Chuck Moore. "And as soon as we got to four, we started thinking about going to six or eight."

So today programmers can again count on a solid 50 percent annual gain in effective processing power, driven not by raw speed but by increasing parallelism. Therein lies the rub. Back when Olukotun worked out Hydra, "it was unclear if you could take advantage of all the parallelism," he says. "It's still unclear today."

So where does it end? Sixty-four cores? Already there. Start-up Tilera Corp. is selling it (see "Company to Watch"). Two hundred? One thousand? "Cores are the new transistors," jokes Olukotun.

COMPANY TO WATCH:

Tilera Corp.,
San Jose, Calif.

In 2008, MIT professor Anant Agarwal transformed an academic project to efficiently make use of lots of simple cores connected in a mesh into Tilera, a company whose commercial processor has one of the highest core counts of all. It's selling a 64-core product now, the 100-core Tile-Gx starts sample shipments in mid-2011, and the company plans a 200-core product in 2013.

Just adding traditional cores isn't going to be enough, says AMD's Moore. The scheme may have saved the power-versus-performance curve for a time, but it won't do so forever. "These days, each core is only getting 8 or 10 watts," he says. "In some sense we're running back into that power wall." With its new Bulldozer architecture, AMD has managed to buy some breathing room by finding a set of components that the cores can share without seriously degrading their speed. But even so, Moore's best guess is that 16 cores might be the practical limit for mainstream chips.

Intel's Pawlowski won't put a number on it, but he will say that memory bandwidth between the cores is likely to be the big constraint on growth.

What will keep computing marching forward, according to Moore, is the integration of CPUs and graphics processing units (GPUs) into what AMD calls an accelerated processing unit, or APU. Say you want to brighten an image: Just add 1 to the number representing the brightness of every pixel. It'd be a waste of time to funnel all those bits single file through a CPU core, or even 16 of them, but GPUs have dedicated hardware that can transform all that data practically at once.

It turns out that many modern workloads have just that kind of data-level parallelism. Basically, you want to do the same thing to a whole lot of data.

That key insight drove AMD to acquire a leading GPU maker, ATI Technologies, and start work on jamming their two products together. So a future processor, from AMD at least, would probably contain multiple CPU cores connected to several GPU elements that would step in whenever the work is of a type that would gum up a CPU core.

With Cell, the processor released in 2006 to power the PlayStation 3, IBM has already gone in that direction. Instead of actual GPU functions, it developed a more flexible core that specializes in executing the same instruction on several pieces of data at once. IBM, with help from Toshiba and Sony, stuck eight of the new cores on the same chip with a more traditional processor core. But that's not quite where Kahle, who led the Cell project, sees things going in the future. Instead he expects to see a mix of general-purpose cores and cores specialized for one task—encryption, decryption, video encoding, decompression, anything with a well-defined standard.

Olukotun agrees that such a heterogeneous mix of cores is the way forward, but it's not going to be easy. "It's going to make the programming problem much worse than it is today," he says. "Just as things were getting bad for software developers, they have the potential to get worse." But don't worry. They're working on it.

For all of IEEE Spectrum's Top 11 Technologies of the Decade, visit the special report.