WINNER
INTEGRATED CIRCUITS

Sun's Big Splash

THE NIAGARA MICROPROCESSOR CHIP IS SUN'S BEST HOPE FOR A COMEBACK

BY LINDA GEPPERT

The Sunnyvale, Calif., campus of Sun Microsystems Inc. is a quiet and peaceful place with six low-rise buildings connected by tree-lined walkways. But the tranquility masks a frightening reality—Sun is in serious economic trouble. The company was badly splattered by the burst of the dot-com bubble of 2000. Revenues for this once towering colossus of the server industry went south, and its stock plunged from more than US $60 in 2000 to less than $3 in 2002. Recently, the stock has been slowly but steadily climbing, and at press time it was selling at more than $5—a sign that the worst may be over for the company.

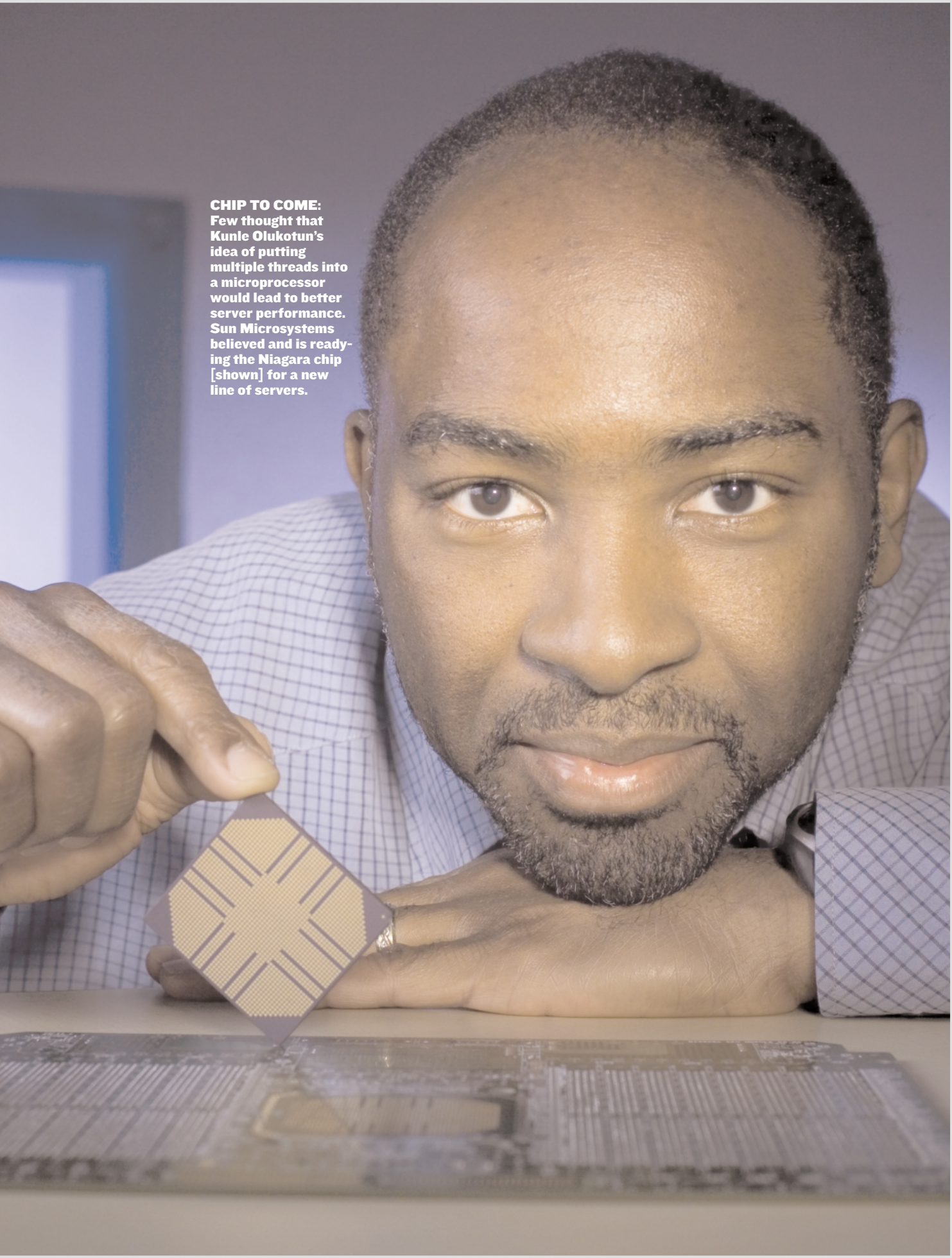
But Sun, headquartered in Santa Clara, Calif., is still far from its glory days of the last decade. It could use a small miracle to get back solidly on its feet, and at last the company may have one: a new microprocessor chip intended for the volume servers that are the heart of data centers running the information and Web processing for businesses, universities, hospitals, factories, and the like. Sun's engineers have had working chips since last spring and are now heavily into testing and debugging them and making design changes for the next fabrication run in early 2005.

The server business generates $50 billion a year, according to Jessica Yang, a research analyst at IDC, Framingham, Mass., and Sun's share recently is about 12 percent—down from 17 percent just four years ago.

Sun's new chip, called Niagara for the torrent of data and instructions that flow between the chip and its memory, was designed from the ground up to do away with the impact of latency—the idle time a microprocessor spends waiting for data or instructions to arrive from memory. This latency is one of the biggest impediments to the microprocessor's ability to do real work.

Niagara was not conceived at Sun. It started life in classic Silicon Valley fashion, as the brainchild of a Sunnyvale start-up called Afara Websystems Inc.

ROBERT HOUSER



CHIP TO COME: Few thought that Kunle Olukotun's idea of putting multiple threads into a microprocessor would lead to better server performance. Sun Microsystems believed and is readying the Niagara chip [shown] for a new line of servers.

"'Afara' means 'bridge' in the west African language Yoruba," explains Stanford University professor Kunle Olukotun, one of the company's founders [see photo, "Chip to Come"]. Completing the founding trio are Les Kohn, a microprocessor guru who has designed microprocessors for both Sun and Intel Corp., and industry insider Fermi Wang. The company did not plan to sell microprocessors but instead to sell complete servers built around its approach to microprocessor design.

When funding all but vanished after the terrorist attacks of 9/11 and the recession that began in 2001, Afara began negotiations with Sun, and in 2002, the server giant acquired the start-up.

NIAGARA DIVERGES FROM OTHER MICROPROCESSORS in the way it processes instructions—the individual commands that come from software applications like databases and spreadsheets. These applications enter the microprocessor as a stream of instructions, which the microprocessor executes. The instructions tell the microprocessor what data to operate on, what operation to perform on them, and what to do with the result. The instructions travel through a series of circuits called a pipeline, which resembles an automobile assembly line.

Each stage of the pipeline performs one step of the instruction execution every clock cycle. In the first stage of the Niagara design, the pipeline gets an instruction from memory, an ADD instruction, for example. The second stage selects that instruction for execution. The third stage determines what kind of instruction it is—in this case, an ADD instruction. The fourth stage executes the instruction. The fifth stage is used for getting data from memory. ADD instructions, however, do not access memory, but get their data from registers. So the instruction passes through the memory stage and on to the final stage, during which the pipeline writes the results of the operation back into a register.

Every clock cycle, a new instruction enters the pipeline. If all goes well, the instructions march through the pipeline in lock step, and one instruction is completed with each tick of the microprocessor's clock. So a six-stage pipeline can have six instructions at different stages of execution at one time.

The rate at which a microprocessor executes instructions is the most important measure of its performance. It is simply the product of its clock frequency and the number of instructions it executes in each clock cycle. Ever since the first microprocessor was invented in the early 1970s, architects have been on a relentless scramble to improve performance.

Typically, the designers have pursued their goal on two fronts: increasing the clock frequency and increasing the number of instructions the processor can execute in one clock cycle. Thanks largely to the semiconductor industry's ability to produce ever smaller transistors, clock frequencies have soared over the past 30-plus years by five orders of magnitude—from tens of kilohertz to more than 4 gigahertz.

To increase the number of instructions per clock cycle, architects have added more pipelines. Some microprocessors today, such as Intel's Pentium 4, have eight pipelines, allowing these chips, in principle, to complete eight instructions in parallel during a single clock cycle.

In the current generation of microprocessors, some designs have gone a step further and put the essential elements of two microprocessors on one piece of silicon. Such a dual-core microprocessor, with eight pipelines in each core running at 4 GHz, could pump out an astounding 64 billion instructions per second if it could complete one instruction per pipeline per cycle.

But unfortunately, such microprocessors are rarely so efficient. If an instruction in a pipeline needs data from memory, it has to wait until the data arrives before it can actually execute the instruction. So pipelines spend a lot of time stuck in neutral—as much as 85 percent of the time, in fact.

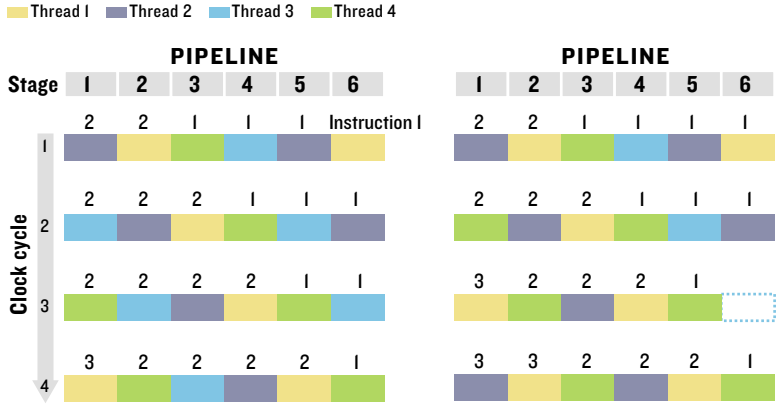
The length of the individual delays depends on where the sought-after data is. If it's in the high-speed on-chip cache memory, the wait could be only a few clock cycles. But if the data is not in the cache, its retrieval from off-chip main memory may take hundreds or even thousands of clock cycles. It's no wonder that minimizing this crippling latency is perhaps the most important challenge facing microprocessor designers.

To improve throughput, designers have created ever more complex mechanisms, such as executing some instructions in a different order from the way they occur in the instruction stream (out-of-order execution) or beginning the execution of instructions that may never be needed (speculative execution). "But no matter what you do," says Fred DeSantis, vice president of engineering for Horizontal Systems at Sun, "eventually you still have to go to memory." And wait.

Niagara's architects adopted a different approach—one that had been kicking around in the corridors of university engineering departments for decades. The idea, called multithreading, consists of dividing the instruction stream into several smaller streams, known as threads. The concept was first developed for Control Data Corp.'s CDC 6600 supercomputer in the 1960s.

In the Niagara design, each pipeline can handle four threads. In each cycle, the pipeline begins the execution of an instruction from a different thread. So, for example, when an instruction from thread one is at stage three in the pipeline, an instruction from thread two will be at stage two, and one from yet a different thread will be at stage one.

If the pipeline cannot continue to execute the thread-one instruction because it needs data from memory, it



THREADS ON THE MARCH: The Niagara microprocessor's six-stage pipeline operates on six instructions at a time from four different instruction threads [color-coded]. At each clock cycle, Niagara is working on at least one and sometimes two instructions from each thread in different pipeline stages. For example, in clock cycle 1 [left, top], instruction 1 of the yellow thread is in the final stage [No. 6] of the pipeline. Each clock cycle, the instructions move one stage forward in the pipeline [to the right in the diagram].

All processing need not stop and wait when information must be fetched from main memory—as in a microprocessor with a single thread. See, for example, instruction 1 in the blue thread [right]. Instead, Niagara does not execute any more instructions from the blue thread until the needed information has arrived from memory. But, in the meantime, it continues to execute instructions from the three other threads.

stores the information about the stalled instruction in a special type of on-chip memory called a register file. At the same time, it continues with the execution of the thread-two instruction, rotating among the three threads that are available [see diagram, "Threads on the March"]. Then, when the needed data for the other thread becomes available, the pipeline jumps back to that thread, using the register file to pick up exactly where it left off.

In conventional microprocessors, architects obtain multigigahertz speeds by increasing the number of stages in the pipeline. Basically, a processing step that could be completed in one clock cycle in a slower pipeline now needs two or more clock cycles to do the same job in a faster chip. But because of Niagara's ability to keep its pipelines running almost all of the time, the architects do not have to run the microprocessor at multigigahertz speeds in order to get good performance. And the slower speeds translate to a simpler pipeline. The simpler pipeline and lower frequency let Niagara run at much lower power than comparable microprocessors.

The set of instructions that a microprocessor executes defines its architecture. Designs based on that architecture may differ in many ways, such as the size of their memories or the number of pipelines, but if they execute the same set of instructions, they are of the same architecture. Afara chose to use the Sparc architecture, developed by Sun, as the starting point of its design because of the many applications that run on it, explains Olukotun. Sparc is the basis for most of the microprocessors in Sun products.

"Afara's approach was to build a very simple, straightforward pipeline, maximize the number of threads the pipeline can efficiently handle, put it into a microprocessor core, and then maximize the number of cores you can put on a chip," says DeSantis. Each Niagara chip has eight cores, which use the Sparc instruction set, and the pipeline in each core can switch among four threads. So a single Niagara chip can handle 32 threads and execute eight instructions per clock cycle [see figure, "Built for Speed"]. The chip rarely wastes time waiting for data to come back from memory.

SUN HAS LONG SEEN THE MERITS of multicore, multithreaded microprocessors. The company's first such design was called Majc, developed for network communications in the late 1990s [see "Microprocessors: The Off-beat Generation," *IEEE Spectrum*, July 2000]. At the time of the Afara acquisition, Sun engineers were at work on another such design, code-named Honeybee. Clearly it made little sense for Sun engineers to pursue two similar designs slated for the same applications.

Both the Afara design and Honeybee were the responsibility of DeSantis. Within weeks of the acquisition, upper management and key technical developers decided to pursue the Afara design, now code-named Niagara.

But Sun is not the only company to pursue a multithreaded approach. One example is a recent version of Pentium, from Intel Corp., in Santa Clara, Calif., that has two-way multithreading, which Intel calls hyper-threading. "The difference," says Poonacha Kongetira, a senior engineering manager at Sun, "is that Intel's chips are first designed for a single thread, because that is consistent with desktop microprocessors, their main business. Then they extend the design to add another thread."

Niagara, on the other hand, was optimized for multithreading from the very beginning. "It's not meant for desktop applications," says Kongetira, "but for commercial workloads like databases and Web servers—applications with a lot of independent queries that can be easily broken up into threads."

To further reduce latency, Niagara's architects gave the chip a fat, fast interface between the processor cores and its two different kinds of on-chip memory. Each chip has two levels of memory. Every core has its own level-one memory, and the whole group of cores share a larger, level-two memory. The level-two memory can send data to the cores at an aggregate rate of 100 gigabytes per second.

The idea for what became Niagara came to Olukotun in 2000, when he began talking with people who were running data centers. Servers were being equipped with ever more power-hungry microprocessors, and data-center managers "were complaining that they were running out of power and space," he says.

Olukotun realized that multiple processors on a chip, multithreading, and data centers were made for each other. A data center contains row upon row of racks, each about the size of a home refrigerator. Each rack holds anywhere from one server to hundreds, depending on their type, size, and power consumption. Data centers need to do many things efficiently at the same time, which is exactly the point of multithreading.

In endless rounds of discussions with venture capitalists, Olukotun swore that his approach would improve server performance by a factor of 10 over what Intel could do at the time.

Not surprisingly, few believed him, Olukotun told *Spectrum*. They argued that microprocessor design is mature and that it would be difficult to compete with the incumbents. But he stubbornly stayed on message: "Here's a market that needs a new approach, and it's an approach that none of the incumbents is taking," he remembers saying. And the approach fitted nicely with the capabilities of a start-up, because it could vastly improve performance while at the same time making the design simpler than that of conventional microprocessors.

Early in the process of establishing the start-up, the venture capitalists introduced Olukotun to Les Kohn. "That was critical," Olukotun recalls, "because he was a master microprocessor architect and we meshed really well." Kohn told him that he was done designing microprocessors. "But I managed to convince him that he really needed to do just one more."

WITH NIAGARA, KONGETIRA CAME FULL CIRCLE. He had left Sun in 2000 to go to Afara and came back to Sun after the acquisition. He remembers how closely the Afara team worked together: "If a problem came up, it was like a red flag to a bull," he told *Spectrum*. "Five guys would be sitting in their cubicles thinking about the problem, and one of them would pop his head up over his cubicle wall with a possible solution. Then there would be a discussion, and if it didn't work, the heads would go back down again."

No microprocessor had ever pushed multithreading as far as Niagara would, so no designers had ever faced some of the problems that Olukotun and his team did. One of their biggest challenges was making sure there were no bottlenecks in the flow of data

NIAGARA

GOAL: Design and build a microprocessor that works 10 times as efficiently as existing devices for processing applications and at half the power consumption of comparable devices

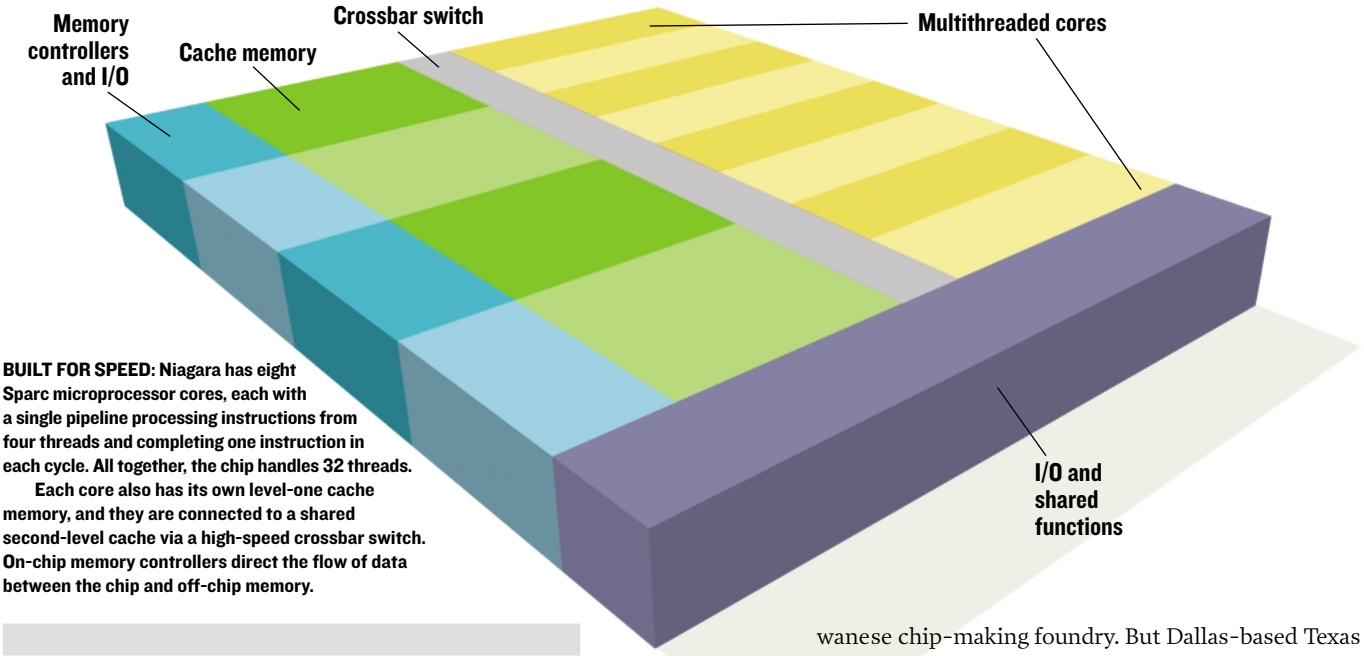
WHY IT'S A WINNER: Data center operators, with their thousands of power-hungry servers, should find Niagara's power and processing efficiency to be just what the doctor ordered

ORGANIZATION: Sun Microsystems Inc.

CENTER OF ACTIVITY: Sunnyvale, Calif.

NUMBER OF PEOPLE ON THE PROJECT: 100

BUDGET: Confidential



BUILT FOR SPEED: Niagara has eight Sparc microprocessor cores, each with a single pipeline processing instructions from four threads and completing one instruction in each cycle. All together, the chip handles 32 threads.

Each core also has its own level-one cache memory, and they are connected to a shared second-level cache via a high-speed crossbar switch. On-chip memory controllers direct the flow of data between the chip and off-chip memory.

between memory and the pipelines. "It's a bandwidth game," Olukotun explains. "It follows the dictum that you can always buy bandwidth, but latency is fixed by the speed of light, and you can't bribe God."

Early on at Afara, progress on the microprocessor was swift. It took less than three months from the time they started designing the chip until they were able to execute instructions through a software model of the pipeline. But then in 2001, funding started to dry up. Olukotun remembers ruefully how he, Kohn, and Wang were supposed to meet with potential backers in the World Trade Center in New York City in the late summer of 2001. But on the day of the meeting—11 September—they learned from a taxi driver that a plane had hit the World Trade Center. "We could see the smoke coming out of the WTC," recalls Olukotun. After being stuck in a motel for two days, they finally got to meet with their potential funders. "And in the end, I think those guys signed on."

Afara's plan to sell servers based on the new chip meant that the start-up needed a lot of money to get to market. But after 9/11, the funding climate turned foul. Venture capitalists were steering clear of huge projects. That's when Afara began negotiations with Sun.

They found an early champion in Rick Hetherington, currently Sun's chief architect for Horizontal Systems, which develops and sells servers for data centers and Web systems. Like Sun's engineering manager Kongetira, Hetherington had left Sun to join a start-up. His start-up was funded by Sequoia Capital, the same venture capital firm that was funding Afara. So when Sequoia invited Hetherington to meet with Olukotun, he did so and was impressed with Afara's technology.

After Hetherington's start-up foundered, he returned to Sun, which was then considering acquiring Afara. He started writing memos and sending e-mails to Mike Splain, chief technology officer for the processor group, and other senior managers, urging the acquisition. "I told them, 'We've got to get this technology in and build it,' " he recalls.

After the acquisition, Hetherington put together a road map for the development of subsequent microprocessors based on Niagara. He would not elaborate on the details, but apparently the development of Niagara 2, the second generation of the design, is already staffed and under way.

"Acquisitions are tough to do," admits Hetherington. Although Afara was the first Sun acquisition he was directly involved with, he has noticed that relatively few such acquisitions succeed. "Only about one in four make it from beginning to end and turn into successful products," he notes.

Any number of things might have derailed the Afara acquisition. First, Afara was planning to have its chips made at TSMC, the huge Taiwanese chip-making foundry. But Dallas-based Texas Instruments builds Sun's microprocessors. It wasn't just a problem of loyalties: every foundry has its own circuit libraries and design rules, and a change in foundries would mean a change in the design.

Second, Afara was planning to use Linux as the operating system, while Sun's microprocessors run Solaris, the company's proprietary operating system. In many cases, too, Afara was using computer-aided-design software products that were different from Sun's, forcing decisions over which software to use. And then there was the merging of two different corporate cultures.

Sun got its way on the foundry and the operating system issues. The two groups compromised on the design software, using some of Sun's software and some that the Afara designers had used.

"Once we got past those issues, things started to click," adds Allan Strong, senior engineering director for Niagara, who came to Sun with the Afara acquisition. For Strong, the high point in getting the new chip to market was when the first packaged chips came back from the foundry last spring. Engineers quickly fired up the chips and were able to run the operating system on them. "Until that happens, you never really know," Strong says. "You've done the simulations, but all it takes is one missing rectangle" out of untold millions on one of the masks to make the chip fail.

When the chip is rolled out, Niagara will go into a new line of Sun's Web and application servers. The company plans to sell them as single servers and as two servers combined into a single chassis, says Hetherington. They will run in large systems that are built up by connecting many servers together. The difference between this type of system, which Sun calls "horizontal," and large multiprocessor systems, which the company calls "vertical," is that in a horizontal system, each server runs its own copy of the operating system and has its own memory. In a vertical system, on the other hand, servers run a single operating system and share memory.

The company expects Niagara systems to be available in 2006. And if the chips hold up to expectations, we could see Sun rising once again. ■