cover story

It took Transmeta engineers $100 million, five years of secret toil, and a little magic to create fast low-power chips that turn into x86s in a microsecond

LINDA GEPPERT
&
TEKLA S. PERRY
Senior Editors

TRANSMETA CORP.'S CRUSOE CHIPS, due to ship in May or June, look nothing like Intel Corp.'s Pentium processors. In fact, they do not even have a logic gate in common. They are smaller, consume between one-third and one-30th the power (depending on the application), and implement none of the same instructions in hardware.

But the Crusoe microprocessors [Fig. 1] can run the same software that runs on IBM PC-compatible personal computers with Pentium chips—for instance, Microsoft Windows or versions of Unix, along with their software applications.

That's the magic trick. And it took a bunch of engineering magicians—and over $100 million of venture capital—to pull it off.

Transmeta's magic show started more than five years ago. David Ditzel, then the chief technical officer of Sun Microsystems Inc.'s Sparc business, headquartered in Palo Alto, Calif., had studied ways to assist Sparc processors in running x86 software by emulation. He hired Colin Hunter as a short-term contractor on a project to determine what new instructions might be added to Sparc to help make emulation run faster. They completed the project and produced an internal report. But it appeared unlikely that merely adding a few new instructions to Sparc would significantly enhance the processor's ability to run x86 software.

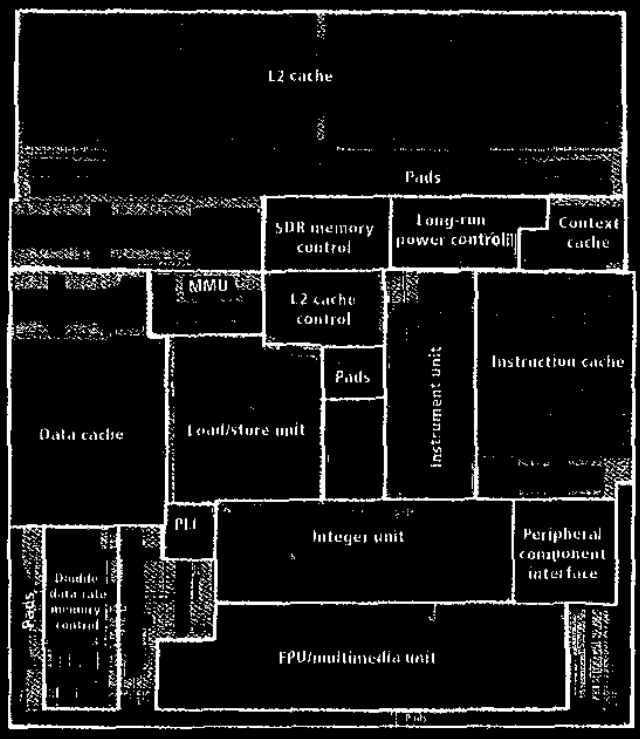
Ditzel had also become concerned about the ever-growing complexity of microprocessor design. He had long been a champion of simple microprocessors: with a professor from the University of California at Berkeley, David Patterson, he had coauthored the pioneering 1980 paper "The Case for the Reduced Instruction Set Computer." But as time went on, he told *IEEE Spectrum*, more and more functions got piled into RISC chips.

This complexity meant that RISC chips were getting bigger and hotter and were taking much longer to design and debug, and improvements in performance were limited. Some chip designs were so complex, in fact, that hundreds of engineers were needed for one design team. Looking out 10 years into the future, Ditzel thought things would only get worse.

So, in early March 1995, he quit his job at Sun. Within a few weeks, he had an idea worked out for a new type of microprocessor. The new device would be fast and simple, and although it would bear no resemblance to an x86 processor, it would be surrounded by a layer of software that could transform, on the fly, an x86 program into code that the simple microprocessor could understand. The technique, called dynamic binary translation, gives programs the impression that they are running on an x86 machine.

Ditzel called on Colin Hunter again and the two prepared to file papers to incorporate as a company. But first they needed a name, one that would not give away what they were doing and one not already taken by any of the other numerous technology companies in California. After running various combinations of high-tech sounding syllables past the California Secretary of State's office, they found one that was available—Trans-

0018-9235/00/$10.00©2000 IEEE



[1] The highest-performance Crusoe chip, the TM5400, is 73 mm² in area. It contains a 128KB level one (L1) cache and a 256KB level two (L2) cache, on-chip LongRun power control, an integrated peripheral-component interface bus and double- and standard-data-rate dynamic RAM controllers.

The chips were fabricated by IBM Corp. at its foundry in Burlington, VT. They feature five levels of copper interconnect and a minimum feature size of 0.18 μm.

(ILLUSTRATIONS: TRANSMETA CORP.)

meta. "We thought we'd change it later," Ditzel said, "but now that it has so much recognition, we'll keep it."

Ditzel and Hunter started making the rounds at various venture capital companies. Meanwhile, the team grew.

The two were joined first by Steve Goldstein, a former vice president of sales and marketing at Ross Technology Inc. (which closed in 1998). Also signing on was a group of Sun engineers who had also been struggling with the problem of how to create a fast emulator: Doug Laird, now senior vice president of product development; Greg Zyner, a very large-scale integrated chip designer; Malcolm Wing, a chip architect; Ed Kelly, a systems engineer; and Bob Cmelik, a software engineer.

The company set up shop in Ditzel's Los Altos Hills house, taking over the living room and two spare bedrooms and equipping them with Sparcstation computers, PCs, printers, an overhead projector, a fax machine, a copier, whiteboards, and obligatory munchies. The team met there several afternoons every week.

Meanwhile, Laird and Zyner set up camp in the living room of Laird's Los Gatos ranch house to sketch out the chip design on a whiteboard commandeered from Laird's five-year-old daughter. Laird and Zyner would visit Wing's Menlo Park apartment, where Wing was working on the overall chip architecture and working with Cmelik on hardware and software tradeoffs.

By summer 1995, funding was getting to be a big concern. Transmeta learned that the money supposedly promised by a venture capital firm was to come from a new fund that had not actually been financed. To get their hands on some money quickly, Laird and Ditzel took a contract from the Advanced Research Projects Agency (ARPA) (now DARPA), Arlington, Va., to write several white papers about high-speed CMOS design techniques. They received $250 000 for this work. The proceeds were used to pay salaries to several members of the group, and to rent a real office building in Redwood Shores, Calif.

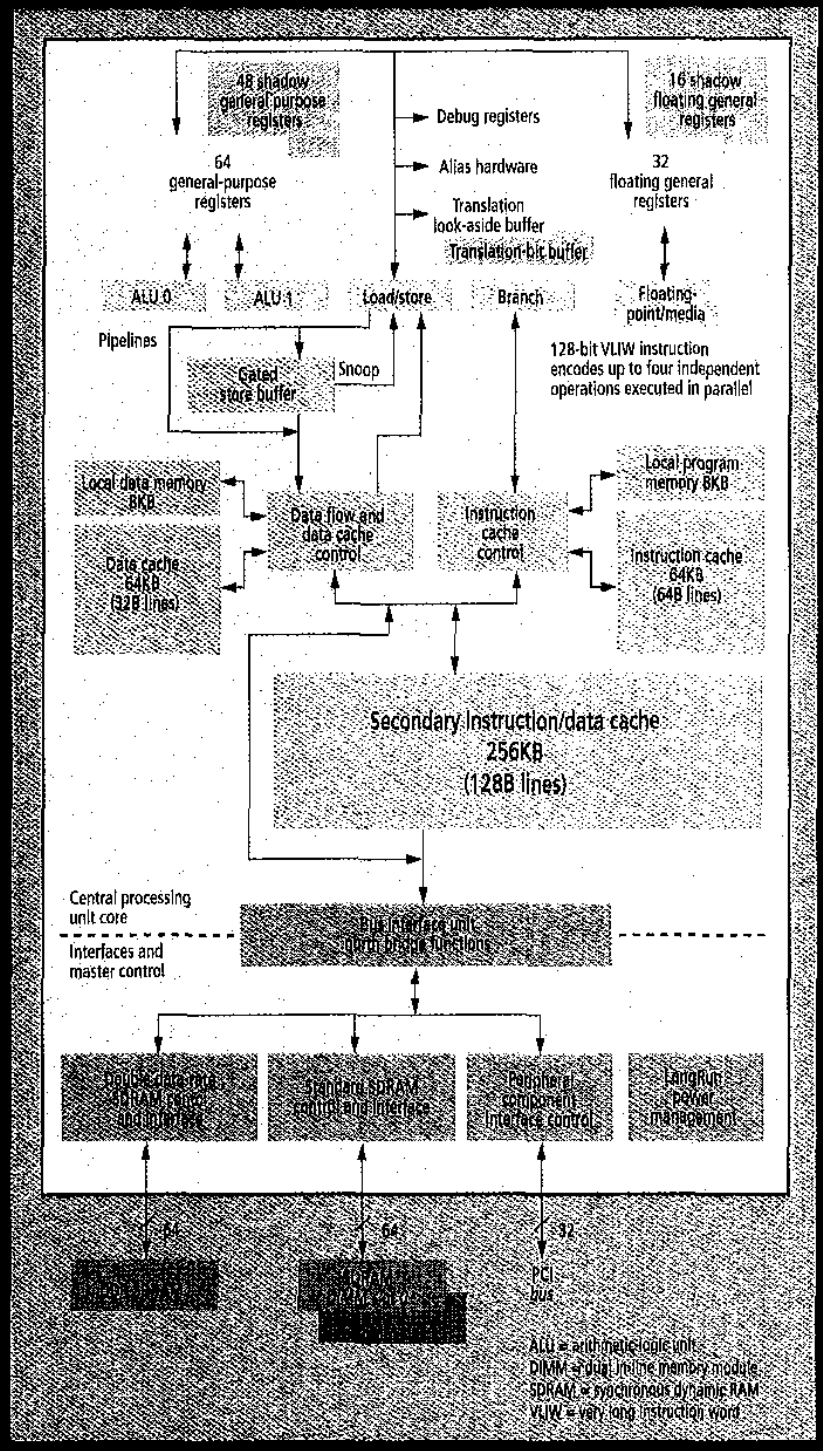
MAKING PROGRESS

Despite the engineers' worries over financing, the technical work on the new microprocessor was proceeding, and some key breakthroughs had been made.

For one, Ditzel chose to base the chip's design on a well-known technique called very long instruction word (VLIW) [Fig. 2]. The attraction of a VLIW microprocessor was the simplicity of its design and its high performance.

A growing difficulty with other commercial architectures, both RISC and x86, stemmed from a common method of improving performance, namely issuing multiple instructions per clock, a technique called superscalar execution. In RISC and x86 superscalar designs, scheduling the instruction order and determining which instructions can be executed at the same time is left to the microprocessor hardware. This setup greatly complicates the design of these systems, slowing them down, adding cost, and burning power. As designers add more and more execution units to the chip in their search for better performance, a point of diminishing returns is reached when gains are largely eroded by the added complexity.

VLIW processors also execute many instructions in parallel (the Transmeta chips can execute four), but it is the job of a compiler (read software) to schedule the instructions. This also fits in with Transmeta's scheme of assigning more work to the software. In Transmeta-ese, individual instructions are called atoms and the VLIW instruction groups are called molecules. The company designed its final chip so that the atoms arrive at the processor bundled by the compiler into molecules composed of two or four



[2] Very-long-instruction word (VLIW) architecture forms the basis of the Crusoe chips. It consists primarily of working general-purpose and floating-point registers and their shadow registers, the executions units, memory, and memory control circuits. The chips have a built-in north bridge, and the TM5400 features LongRun power-management circuitry.

atoms that can be processed together, and the processor executes them.

Another early breakthrough was understanding the factors that traditionally made emulation slow and developing alternatives to eliminate these obstacles. A key reason for the sluggish performance was the extra instructions that an emulator has to run to match the exact state of a processor in a different architecture. "In traditional emulation," Laird told *Spectrum*, "you are taking a program written for a processor with one architecture and getting it to run on a processor with a different one, and the states of the two processors are not the same."

For instance, an x86 program may expect a processor to set a condition code, and the program performs a branch operation based on the value of that condition code. But when the program is run on a PowerPC, say, the condition code is not generated in the same way that an x86 processor would have generated it. So the emulator has to go through a number of PowerPC instructions to set the condition code in the same way as the x86.

"What we discovered," said Laird, "was that if you can facilitate implementing the state of the first processor in the second one by designing certain registers to hold that state, the emulation software doesn't have as big an overhead."

Another difficulty about emulation has to do with so-called exceptions, which are caused by processor faults, errors, traps, or other exceptional events. Since exceptions halt the execution of a program, the operating system must find the cause of the exception and re-execute the instructions that faulted in a way that isolates the fault. The question of how to deal with exceptions was brought up early in the design process. It was Cmelik who identified the seriousness of the problem—not solving it would mean a dead-end for the technological approach being taken.

The problem arises, explained Laird, because the VLIW program they created reorders the x86 instructions. So if the x86 program creates a fault, such as a divide-by-zero—although it may happen infrequently, it still may happen—the processor has to be able to create the exact same state as any other x86 processor would, and hand it off to the operating system to deal with the fault.

The solution came several weeks later with a novel hardware/software combination called commit and rollback, which, according to Wing, "is

really the fundamentally different thing about our machine."

Commit and rollback was implemented by creating an extra set of registers, called shadow registers, in addition to the working registers. With the execution of a software commit instruction, the shadow registers duplicate the data in those working registers. As the operation progresses, the working registers are updated by each computational operation. But the shadow registers are not updated until the processor receives an all-clear signal in the form of another commit instruction, indicating that no exception occurred.

When the processor hits a fault, Transmeta's software issues a rollback instruction, and the information in the shadow registers is copied back into the working registers. "So we can reverse the execution," said Laird. "You come to a state, say, 'Oops, I did a bad thing,' go back in time instantly in one cycle, and start again." The next time around, the software schedules the operations more conservatively, say, by executing the instructions in precisely the same order as the original x86 program.

The team realized that, in the case of a rollback, data to be stored in memory would also have to be rolled back. They came up with a circuit called a gated store buffer to keep track of the stores between commit points. If an exception occurs in this period, the system can instantly roll back to the previous state and discard those stores.

The gated store buffer has a committed and an uncommitted side with a "gate" in the middle. After some compilation creates the data to be stored, the data goes to the uncommitted side of the buffer. After a commit instruction, the gate opens and the data on the uncommitted side moves to the committed side and is then stored in memory.

This process may involve a substantial amount of data. A single x86 instruction, for example, can modify 130 bytes of memory. Other superscalar microprocessors also need store buffers, but nothing quite so big.

IT'S CODE MORPHING!

While development of the chip architecture was progressing, it was beginning to look as if the group might never get funded. Group members kept explaining to venture capitalists that with their revolutionary software-based microprocessor, they could attack markets previously owned by x86 chips, but no one bit. By the end of the summer of 1995, Ditzel and Hunter had pitched nearly 30 venture capitalists; Laird often went along as an observer.

"They just didn't get it," Laird said. "Dave [Ditzel] would start talking about dynamic binary translation, and their eyes would just glaze over. We were pumped up, saying this is a great idea, it is a new microprocessor, and nobody has ever done it this way, but we could've been from Mars for all they cared. We were just getting too technical."

"It was a hard sell," Ditzel told *Spectrum*. "We were saying we wanted to do hard core R&D and develop this big new idea and it would take four years. And the venture capitalists would say, 'Couldn't you just have a simple idea you could do in six months?' "

So in midsummer the entire team sat down at their new offices in Redwood Shores to figure out another way to pitch their ideas. They concluded that they needed to sum up the essence of what they were doing in a word or two, a simple, catchy name that the venture capitalists would understand. After tossing around several ideas, Cmelik threw out the term "Code Morphing" and they knew they had it.

They also discarded some of their more technical PowerPoint slides and came up with a simple sketch of their concept, which they called the amoeba [Fig 3]. The amoeba explained how a traditional microprocessor was, in their design, to be divided up into hardware and software.

Ditzel went back to the venture capital community with the new pitch. Laird sat on the sidelines with his watch. "I timed how long it took, from the first time Dave said Code Morphing, to the time the venture capitalists started using the word themselves," Laird said. "It was less than 5 minutes."

Within a few weeks, several venture capital firms were competing to fund the group. By October they had commitments from Institutional Venture Partners, Menlo Park, Calif., and Walden Group, San Francisco. The check for $3.5 million arrived in December 1995.

"We hadn't changed the principles, we hadn't changed who we are, we hadn't changed anything except how we presented it," Laird said. "We said 'Code Morphing software' and snap, we got funding."

Since trademarked, the buzzword aptly describes what the software does: it takes x86 instructions and recompiles them on the fly into VLIW instructions. As it recompiles them, it optimizes them, making them run, in many cases, more efficiently than the original x86 code. What happens with x86 applications is that, in the rush to market faced by software writers, often applications are compiled without the highest levels of compiler optimization to facilitate debugging. Once the software works, it is shipped; there is no time in the schedule to go back and recompile and re-test, meaning that many software applications have room for improvement.

On a typical software application program, such as Microsoft Word, Code Morphing works like this: it starts with the x86 binary code for a program section to, for example, edit text. In real time, the code goes into Transmeta's software and comes out the other side transformed into VLIW code. In the software's sequence of operations, the x86 instructions are first translated into a sequence of VLIW atoms. Then an optimizer, using some new and some well-known compiler techniques, checks to see if the code can be improved—for instance, by the elimination of redundant atoms.

Finally, a scheduler reorders the atoms and groups them into molecules [Fig. 4]. Once translated, the VLIW code is stored in a special part of memory, accessible only by the Code Morphing software, so that particular program need not be translated again.

But that is not the end. The new software continues to monitor how an application is being used. If it finds that a user is spending a lot of time changing the font, for instance, it turns on more levels of optimization to make that part of the program run faster. "We only optimize that portion of the code [being used]," explained Laird. "For the things that are executed infrequently, there is no reason to put in that overhead."

One of the challenges of creating the Code Morphing software was to make the Crusoe processor, in many cases, bug-compatible with the x86 so that it would generate the so-called Blue Screen of Death at many of the same times an x86 processor would.

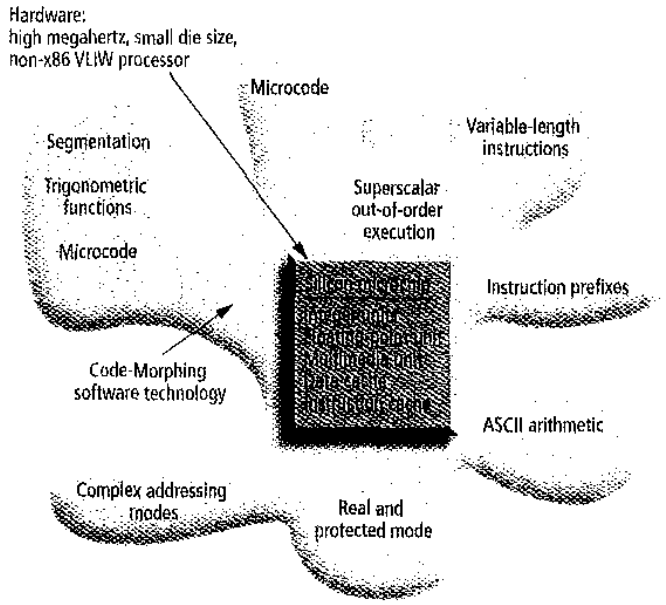
A REAL COMPANY

Now that the funding was in place, it was time in late 1995 to build this small team of engineers into a real company and actually implement the new microprocessor architecture on a chip.

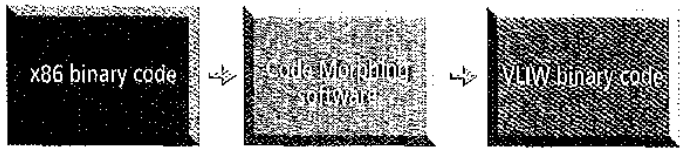
The design the team came up with contained only about half the logic transistors of an x86 processor. It included five execution units—two arithmetic-logic, a load/store, a branch, and a floating-point—and it could execute four instructions in a cycle. Sixty-four general-purpose and 32 floating-point working registers were shadowed by 48 general-purpose and 16 floating-point registers. Memory, memory management, and the so-called north bridge (usually a separate IC) rounded out the design.

Even more important was what the design did not include. It had no superscalar decode, grouping, or issue logic. It had no register renaming or segmentation hardware. And it had no floating-point stack hardware. Nor did it have memory management in the front end of the machine. It also had less interlock and bypassing logic than a traditional central processing unit. This structure contributed to a simpler design with far fewer transistors, which was the key to low power.

In late 1995, Transmeta started hiring engineers to join the eight founders and begin mapping out details of the architecture. The first few hires were people whom



[3] The Transmeta founders credit a simplified sketch of their proposed architecture, which they called the "amoeba," with convincing the financial community that their idea could work. In this concept, the x86 architecture is an ill-defined amoeba containing such features as segmentation, ASCII arithmetic, and variable-length instructions; the square inside the blob is the proposed VLIW processor and its functions.



x86 instructions	Morphed VLIW instructions			
1. movl %ecx,$0x3	1. addi %r39,%ebp,0x2fc;			
2. jmp #bl1	2. addi %r38,%ebp,0x304;			
#bl1:	3. ld %edx,[%r39];	add %r27,%r38,4;	add %r26,%r38,-4;	
3. movl %edx,0x2fc(%ebp)	4. ld %r31,[%r38];	add %r35,0,1;	add %r36,%esp,0x40;	
4. movl %eax,0x304(%ebp)	5. ldp %esi,[%r27];	add %r33,%esp,0x6c;	sub.c %null,%edx,%r31	
5. movl %esi,$0x0	6. ldp %edi,[%r26];	sel #le %r32,0,%r35;		
6. cmpl %edx,%eax	7. stam 0,[%r36];	sel #l %r24,%r35,0;	add %r25,%esp,0x7c;	
7. movl 0x40(%esp,1),$0x0	8. stam %r32,[%r33];	add %ecx,0,3;	sub.c %null,%esi,%edi;	
8. jle skip1	9. st %r24,[%r25];	or %eax,0,0;	brcc #l1,<exit2.>;	
9. movl %esi,$0x1	10. bi <exit1.>			
skip1:				
10. movl 0x6c(%esp,1),%esi				
11. cmpl %edx,%eax				
12. movl %eax,$0x1				
13. jl skip2				
14. xorl %eax,%eax				
skip2:				
15. movl %esi,0x308(%ebp)				
16. movl %edi,0x300(%ebp)				
17. movl 0x7c(%esp,1),%eax				
18. cmpl %esi,%edi				
19. movl %eax,$0x0				
20. jnl exit1				
exit2:				

VLIW = vey long instruction word

[4] Code Morphing software transforms x86 binary code into individual instructions, called atoms, for the Crusoe processors. The compiler within the new software then looks for atoms that can be executed together and groups them into very long instruction words called molecules. Molecules may contain two or four atoms. In cases where an atom is to be executed alone, or only three atoms are to be executed together, the second or fourth position is filled by a no-operation instruction.

Laird, Hunter, or Ditzel had known for years, starting with Godfrey D'Souza, a Sun engineer who would have been in the founding group had he been in a financial position to work without a salary. In 1996, some 80 more engineers were added, mostly mid-career engineers who had years of experience in the jobs they were to take on for Transmeta.

Signing on so many experienced engineers so fast in Silicon Valley's tight job market turned out to be surprisingly easy.

"My being old helped," Laird said. (He is 44.) "I've been around a long time; I know a lot of people."

Ditzel also had a lot of contacts. "I had worked at Bell Labs," he told *Spectrum*, "and when you work there, you tend to get invited to lots of places to see their secret projects. I had been doing a lot of work for IEEE and ACM [Association for Computing Machinery] on conferences, and I had gone to school with people who had gone on to be professors at universities. So I was able to just pick up the phone and call the right people."

When Ditzel and Laird made such calls, they provided little information to their prospective hires—just that they had a new company and were doing something really cool and new in computer architecture. After they were sure the person was interested—and was the right fit—they brought out a nondisclosure agreement. Only after it was signed did they reveal any details about their plans.

The experience of Guillermo Rozas was typical. Rozas, a software engineer and now Transmeta's director of product development, was at Hewlett-Packard Laboratories, in Palo Alto, in 1997 when he heard from a close friend who had signed on with Transmeta. As Rozas explained, "He was a really smart guy, and he told me there were really smart people here that would be fun to work with. I didn't know all that much more when I came in, other than a lot of people I had known had mysteriously disappeared inside Transmeta."

Also recruited was Stephen Herrod, now director of software productization, who was at Stanford University, California, before joining Transmeta. He had done his Ph.D. dissertation on runtime code generation, citing a number of papers and researchers in the field. "When I searched out where all those people were now, it turned out that all of them were at Transmeta," he told *Spectrum*. "I did know someone here from conferences, so I called him up and asked if I could come in. I was about the 15th software person hired, and the other 14 were largely the people whose work I had been studying."

In late 1996, after some hundred people were on board, Laird decided it was time to hire a few engineers right out of college.

"You need a good distribution of experience," he said. "If you have all senior level people, and there are a lot of details that need to be taken care of, they are not going to want to do that." He and Ditzel called their professor friends and asked for their best students, eventually hiring around 30 graduates. A number of these students were interviewed without even knowing what Transmeta did, only that their advisor had told them that Transmeta was a hot start-up.

Despite the large numbers of engineers that were being hired from Silicon Valley's top companies–Hewlett-Packard, MIPS Technologies, Silicon Graphics (but not Intel)—little information about Transmeta's work was being leaked.

"Our approach was simple: to use software as a key piece of the microprocessor," Ditzel said. "So if that one simple idea leaked out, our competitors could get a project going. If it didn't, then they couldn't have a competitive product out in five weeks—it would take them five years."

They kept the secret virtually leak-free by what Ditzel calls rifle-shooting. "Leaks come from people you interview and don't hire. But if you rifle-shoot the exact people you want, all you have to do is impress them about what you're doing and hire them. Then once they've joined your company, they won't leak." He says some 90 percent of engineers offered jobs by Transmeta accepted.

"People were excited about this project because it was one of the first really different types of computer systems that had been designed in the past several years," Ditzel told *Spectrum*. "The hardware guys loved it because they could start with a blank sheet of paper, they didn't have to be compatible with an old instruction set. The software guys liked it because they could ask the hardware guys for special features."

Because the company was hiring so many senior people, the decision was made in the beginning that, even though funds were tight, every engineer would have a private office (as soon as they were available—some employees did double up temporarily). Other amenities include a well-stocked kitchen with drinks, sandwich makings, and snacks. Dinner is ordered in four nights a week.

The atmosphere is as open as a college campus (complete with a busy foosball table)—perhaps even more so. Said Keith Klayman, a member of the technical staff: "Like at a university, we can go to anyone here if we have a question. But at the university, the professor was in maybe once a week. Here, the high-level people are always around and accessible."

Every engineer also has at home a company-provided computer that connects to the Internet through a high-speed digital subscriber line (DSL). With this equipment, people with families can go home for dinner, get back to their engineering work around 10 p.m., and then sleep late in the morning. One winter the company even rented a cabin in the Lake Tahoe ski area and equipped it with computers and DSL capability, so engineers could get their winter skiing in without losing time from their projects.

The lack of borders between hardware and software engineers at Transmeta is, employees report, unique in their experience. Whenever a technical problem is discussed, both hardware and software engineers team up to address it. Sometimes a problem faced by the software engineers is made solvable by a change in the hardware; sometimes it goes the other way. As a result, the company's fleet of rattletrap bicycles, used by the engineers to travel between the buildings housing the two teams, get a lot of use [Fig. 5].

HOUSTON, WE HAVE A PROBLEM...

After three years of work, in August 1998, the first chips came back from IBM Corp., which had signed on as manufacturer. To check out the performance of the chips, the Transmeta engineers ran several benchmarks, both for Unix and Windows. The chips ran Unix benchmarks as fast as had been expected; the first magic trick had worked.

But when the engineers assigned to performance analysis started testing Windows benchmarks, they had a nasty surprise. The Windows benchmarks reported scores far lower than expected. Transmeta had reached into its magic hat to pull out a rabbit and had instead come up with a turtle.

"It was like in the *Apollo 13* movie," Laird said, "We wanted to say, 'Whoops, Houston, we've got a problem here.' "

Laird was philosophical about the situation. "We're engineers," he told *Spectrum*. "We didn't need to panic. We needed to understand what was going on. And so we analyzed it, moved teams of hardware and software people onto it, and started fixing it."

But not all the engineers at Transmeta were so sanguine.

"We had been riding high, blindly expecting the chips to do everything that we had promised," recalled Klayman. "When they didn't, it was a real morale killer." Some of them felt it was never going to work, and since nobody was motivated, no work was getting done. Then Doug Laird told them to drop everything else they were doing, as there was still a chance to right the ship.

The company held an all-hands meeting, in which Laird told everyone the truth—that they had run into a wall running Windows benchmarks. But he reassured them that, by working together, they could fix the problem. Murray Goldman, a member of the board of directors, pledged that the board would stand by their efforts, implying that more money would be raised, should it be needed.

Looking back, Laird said a problem might have been expected with Windows95 applications. "Most of us came from a Unix background, we knew how Unix applications behaved. But we didn't really understand Windows95," he said.

Apparently Windows95 still had a lot of old 16-bit code in it, whereas Unix (as well as Windows NT) used a flat memory model with pure 32-bit code. Supporting 16-bit code was something that Transmeta had decided to offload into software.

Once they realized this, they redesigned the hardware to give better support to Windows95 applications. They also increased the size of the caches because Windows95 applications tend to use more memory than Unix applications.

The redesign process added about a year to Transmeta's development time. In fact, getting products to market took longer than any of the founders had anticipated. "If we had had a better idea of how long it would have taken, we probably would not have done it, I suspect," said D'Souza.

TO MARKET, TO MARKET

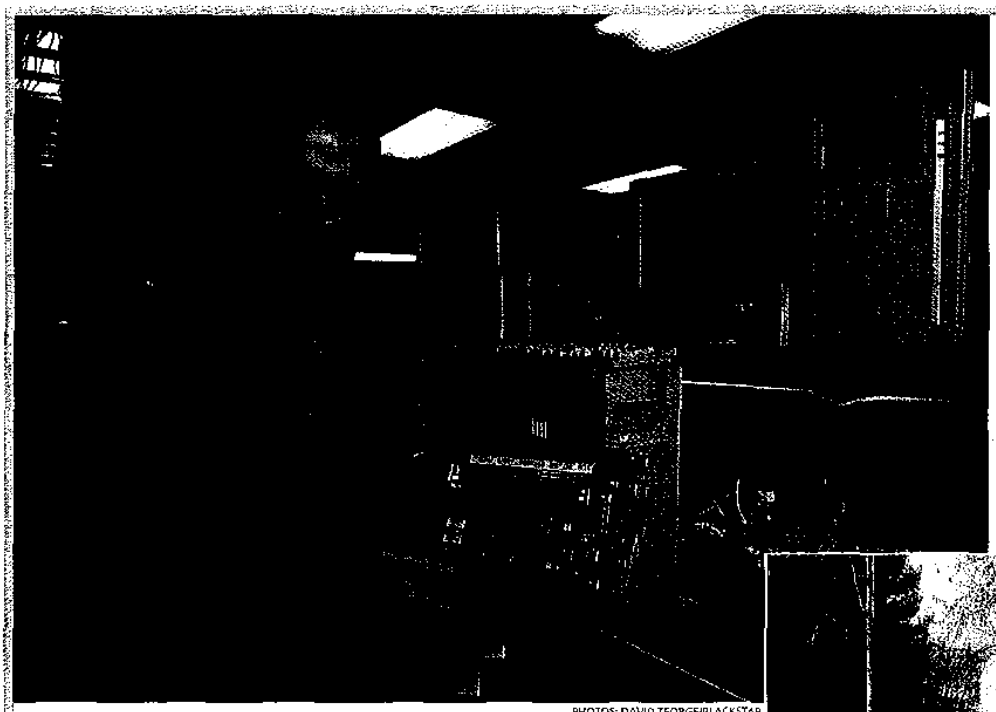
While the engineers were struggling to redesign the chip to run Windows applications at a reasonable speed, a marketing team was taking the show on the road, showing off their concept to OEMs, and asking them if Transmeta was making chips that would sell, and, if so, into what market.

The feedback from the OEMs was almost unanimous, Ditzel said. While they had been presenting their product as appropriate for both the desktop and mobile markets, customers disliked the split focus. They wanted chips optimized for mobile computing.

"Customers told us consistently," Ditzel said, "that they had pretty good chips for desktops and servers, but the road ahead for mobile chips looked horrible; there was nothing coming out that was usable. So, they told us, if you are going to build us a chip, go build us a mobile chip."

The most important parameter for the mobile market is a chip's power consumption. Ditzel said he and Laird had always thought that the hardware/software architecture had a lot of potential for reducing a chip's power consumption, and in general the team designed the chip's circuits with low power in mind. They had not pitched this feature to venture capitalists, because, Ditzel said, it was impossible to know how significant the drop in power was going to be.

By late 1998, with the initial market research complete and prototype chips on which to measure power consumption in



PHOTOS: DAVID TEORGE/BLACKSTAR

[5] Transmeta engineers built a PC-compatible board so they could run Windows applications on their chips. The board is held at left by Doug Laird, vice president of product development.

Transmeta's hardware and software teams are housed in two buildings, about 1 km apart in an office park. Designing chips with so many software functions requires interdisciplinary teams, so Transmeta's fleet of second-hand bicycles gets a lot of use in cross-campus commutes—even by chief executive officer David Ditzel [below].

hand, the decision to focus on mobile computing was made, and power consumption issues came to the forefront.

POWERING DOWN

"A number of people have said that designing lower-power chips means doing a lot of little things—a little bit here, a little bit there," Laird told *Spectrum*. "And if you do a lot of it, the sum of it is good."

One of the biggest little things that the Transmeta team did was to offload a good bit of the microprocessor function onto the software, which allowed them to design simple streamlined hardware with about half the number of transistors of an x86 chip. "Obviously," continued Laird, "if you have fewer transistors, you burn less power."

The team also used virtual devices to cut down on the amount of hardware. A virtual device is one that is not exactly the same as the device expected by the program, but produces the same result. It works by using the Code Morphing software to monitor the input and output instructions to the device, then to send those instructions to the virtual device instead. For example, Crusoe incorporates, on-chip, a separate IC called the north bridge, which couples the processor to the peripheral component interface bus and to external memory.

The north bridge features architecturally defined registers, to which the program sends input and output instructions. To be compatible with the architecture for which the instructions are written, those registers must be constructed so that any application, or the operating system, can manipulate them correctly.

But rather than implementing those registers exactly as in a conventional north bridge, Transmeta engineers employed the Code Morphing software to intercept the instruction to the north bridge registers and send it instead to the registers defined in the Crusoe architecture. Ditzel predicts that the team will be virtualizing more circuits as time goes on.

Another technique is to turn on only those functional units that are absolutely needed to execute an instruction. The process requires a separate clock for each combination of functional units that is turned on during the execution of an instruction. This approach was carried out so thoroughly that a vendor supplying a computer-aided design simulation tool complained that the Transmeta design "broke his tool" because the processor had over 10 000 clocks to control which units get turned on, and when.

But the biggest breakthrough in low-power design came with the development of the so-called LongRun technology, which uses the Code Morphing software to monitor applications as they are running. Then LongRun hardware adjusts both the supply voltage and the clock frequency so that each application runs only as fast as it must to get the job done. Since the processor is running at maximum efficiency, it is maximizing battery life.

Traditional power-management systems also adjust power, but are much less refined. They often try to extend battery life by varying the duty cycle, repeatedly turning the central processing unit on for a fraction of a second, then off for a fraction of a second. "Imagine that you wanted to make the light in a room half as bright," explained Marc Fleischmann, manager of the LongRun power management team. "It would seem silly to do that by flipping the light switch on and off rapidly. But that's exactly how power management works on traditional notebook computers."

Rather than a light switch, Fleischmann compares LongRun to a dimmer control. While applications are running, Transmeta's software observes the traditional power management states and the time spent in the sleep mode; then on-chip LongRun circuitry reduces the frequency and the voltage to precisely match just what the user needs.

"If you spend 40 percent of your time in sleep mode, that means you only need to run at 60 percent of the performance level. So we reduce the frequency from 700 MHz to about 400 MHz, say. And we ramp down the voltage correspondingly. Adjusting both frequency and voltage is a far more efficient way to extend battery life," Fleischmann told *Spectrum*.

"The major point," added Laird, "is that

LongRun is an extension of power management, not a substitution for it."

All told, the efforts to reduce power consumption on the Crusoe chips can reduce power by a factor between three and 30, depending on the application, compared with a typical x86 processor, according to Fleischmann.

SOFTWARE'S EDGE

As the design of the microprocessor evolved, other advantages of moving functions into software became apparent. "Having software involved gave us more opportunities than we initially thought," Ditzel said.

Processor upgrades are simplified because the layer of software between the applications and the chip frees the designers to change the chip architecture without causing x86 software developers to have to recompile their code. Code Morphing software can be updated independently of hardware by loading a software upgrade into Flash memory.

The software also helps the debugging process. When the hardware design team got the very first silicon, they found plenty of bugs. They knew that the software layer would help them debug the chip, but no one appreciated ahead of time just how powerful that help would be, according to D'Souza. They were able to work around a lot of the bugs, he said, by performing operations in a different way.

The engineers were always able to boot Windows, even on buggy silicon. As each bug was found (and fixed with software), it was added to the list of revisions for the next design.

What's more, the software layer was also used to increase performance by improving the timing of critical paths. For instance, engineers found that when two particular atoms were paired together in a molecule, the processor ran sluggishly. Otherwise, the chip could run at a much faster clip. So the hardware designers asked the software designers to modify the scheduler so that these two atoms would not appear in the same molecule. "All of a sudden," said D'Souza, "we were running at 600 MHz instead of 466 MHz."

CRUSOE LIVES

By August 1999, the first of the redesigned chips came back from the IBM fab. This time, it ran Windows applications just fine. This chip, for the mobile computing market, became the TM5400. The original design, which was intended for running Linux for the Internet appliance market, became the TM3120.

The TM5400 is similar to the TM3120, but has added the LongRun feature to conserve power. This chip also has more on-chip cache memory than the TM3120 to support x86 applications for Windows-based notebook computers. The TM3120 runs at 400 MHz, while the TM5400 runs at up to 700 MHz.

Transmeta engineers intentionally designed Crusoe to be simpler than conventional x86s slated for mobile applications, but to achieve comparable performance by running at a higher frequency. The fastest mobile Pentium III clocks in at 650 MHz.

Of course, the performance of the Crusoe chips depends on the application. "I think it's fair to say that Crusoe is faster on some applications and not as fast on others," said Ditzel.

For most mobile applications, all of the TM5400's processing power is often not even needed. The effectiveness of LongRun lies in making the processor run at just the right frequency to deliver the performance demanded by the application while conserving power.

The microprocessor family was formally branded Crusoe, after the fictional adventurer and traveler, Robinson Crusoe. "It was friendly, short, and easy to remember," Ditzel said. "So you'll remember it's a mobile chip."

Finally, on 19 January 2000, after nearly five years of effort and over $100 million invested, Transmeta pulled back its curtain at a large press conference at Villa Montalvo, a grand old estate in the hills of Saratoga, Calif.

Meanwhile, engineers at one of Transmeta's unmarked buildings raised a huge black flag with the yellow Crusoe logo from the roof of their building. The flag could be seen by Intel engineers driving to and from their nearby offices.

Bennett Smith, a consultant in microarchitecture, computing platforms, and related intellectual property, is impressed by Transmeta's technology. "They have a sophisticated approach to power consumption that looks pretty amazing," he told *Spectrum*. On the negative side, he has heard concerns that the company's chips are just too expensive. "Companies designing for the portable market may have difficulty justifying the intellectual property premiums built into Transmeta's business plan," he said. Smith and Bruce Shriver are co-authors of *The Anatomy of a High-Performance Microprocessor: A Systems Perspective* (IEEE Computer Society Press, Los Alamitos, Calif., 1998).

Writing in *Cahners Microprocessor Report*, 14 February 2000, Tom R. Halfhill also expressed cautious praise: "Revolutionary may be an overstatement, but they are definitely different....The TM5400's LongRun feature is one of the most innovative technologies introduced by Transmeta. To our knowledge, no other microprocessors can conserve power by scaling its voltage and clock frequency in response to the variable demands of software."

Indeed, the chips still continue to amaze their creators.

Referring to the prototype system that the Transmeta team used to test the Crusoe chips, Rozas said, "I've been seeing these things run now for a year and a half. I know them inside out. Yet, I am still amazed every time I start it up and [a Crusoe-chip computer] looks like a normal PC.

"Considering the complexity of the project, it is amazing how well it works, how fast it works, and how low-power it is," Fleischmann commented. "For the end-user, this is just a normal PC, but under the hood, it is a technological marvel. I am in a state of wonder, too—and I am proud."

THE NEXT GENERATION

Variations of the current generation (both low-cost versions and higher-performance versions) are also being designed. (The part numbers were purposely picked to be in the middle of the range, leaving room for both new versions.)

Transmeta's next generation may have a fundamentally different architecture, even a different instruction set—whatever it takes to make it better, because use of Code Morphing software obviates the need for legacy hardware.

The design will most likely include the latest submicron CMOS technology, including shielded clock lines. The computer-aided design tools will need to make accurate models of inductive coupling between the interconnect structures on the chip. To the engineers, this is a chance, once again, to start with a blank sheet of paper and to rethink the first generation's tradeoffs between hardware and software. To the user, though, the next Crusoe will still appear as an x86.

"Usually you say the next generation will be bigger and better," Ditzel said. "But in this case, I'll say it will be smaller and require even less power." ◆

TO PROBE FURTHER

Information on Transmeta Corp., white papers describing in detail its Code Morphing technology, videos of the Crusoe product launch event, recent news articles, and employment opportunities at the company are available at www.transmeta.com.

Detailed analysis of the Crusoe processor architecture is to be found in the article "Transmeta breaks x86 low power barrier," by Tom R. Halfhill, *Microprocessor Report*, 14 February 2000, p. 1 and pp. 9–18.

Transmeta will be making presentations at the Embedded Processor Forum in June (see www.mdronline.com) and at the IEEE's Hot Chips meeting in August in California (see www.hotchips.org).