Reed Electronics Group

MICROPROCESSOR REPORT

www.MPRonline.com

THE INSIDER'S GUIDE TO MICROPROCESSOR HARDWARE

FLOATING POINT BUOYS CLEARSPEED

Massively Parallel Processor Delivers 25.6 Peak GFLOPS at 200MHz

By Tom R. Halfhill {11/17/03-01}

Once upon a time, there was a thriving market for floating-point math coprocessors—until Intel's 486 and other general-purpose processors integrated the FPU on chip, eventually sinking coprocessor companies like Weitek. Since then, the major CPU vendors have set the pace for floating-point performance. If you need more performance than integrated FPUs can deliver, the only alternatives are multiprocessor servers or supercomputers, custom logic in ASICs or FPGAs, or a few relatively unknown and exotic CPU architectures.

ClearSpeed fits the last category. In the Extreme Processors session at **Microprocessor Forum 2003**, the U.K.-based startup revealed a new massively parallel CPU architecture intended to revive the market for floating-point coprocessors. ClearSpeed's strategy is to offer much higher floating-point performance at much lower power levels than general-purpose CPUs do, enabling designers to build faster embedded systems and accelerator cards for PCs, workstations, and servers. Instead of bottom-trawling for the mass market, though, ClearSpeed is fishing for customers willing to spend $975 per chip for 25.6 billion floating-point operations per second (GFLOPS).

In price and floating-point performance, ClearSpeed's new CS301 chip appears to compete against Intel's Itanium-2, IBM's Power4, Sun's UltraSparc III, and high-end x86 processors from AMD and Intel. The CS301 is a coprocessor, however, not a standalone CPU, and it's far more suitable for embedded systems than any of the aforementioned chips, thanks to miserly power consumption in the 1.8–2.5W range. Nobody can beat 10 GFLOPS per watt.

Target applications include general scientific computing, medical imaging, genetic engineering, nanotechnology research, signal processing, simulation, financial modeling, weather forecasting, and defense-related image recognition. Indeed, ClearSpeed is the first company to pitch a processor to *MPR* by singling out applications like genome mapping,

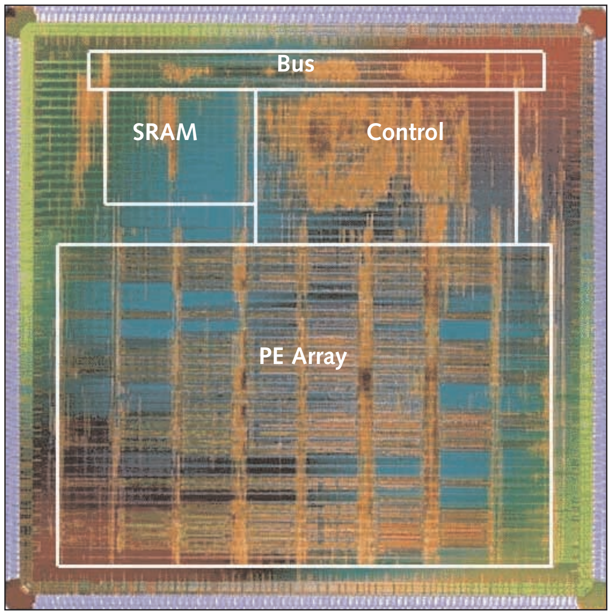

Figure 1. ClearSpeed CS301 die photo. The 64 parallel processing elements and their buses fill up most of the chip. At the upper right, notice the large function unit, which ClearSpeed calls the mono execution unit; it controls the processing elements. Global SRAM is at the upper left.

protein folding, drug discovery, climate analysis, and terrain recognition for guided weapons. The CS301 is small enough and cool enough to fit on a PCI plug-in card or even a PC Card, so it can boost the floating-point performance of virtually any kind of system.

ClearSpeed's Simon McIntosh-Smith, director of architecture, was showing off the first-pass silicon of the CS301 at Microprocessor Forum. Since then, ClearSpeed has determined that the first chips are working perfectly and will be qualified as production silicon. Production could begin as early as 1Q04, depending on demand.

IBM Microelectronics will manufacture the chip for the fabless semiconductor company, using a 0.13-micron eight-layer-metal copper CMOS process. Initial samples are hitting the target clock frequency of 200MHz at 1.2V, with 2.5V I/O (3.3V-tolerant). Power consumption is about 1.8W typical, 2.5W worst-case.

The die, shown in Figure 1, measures 72mm^2, contains 41 million transistors (32% logic, 68% memory), and is packaged in an 852-pin plastic BGA. Only 343 pins carry signals, because ClearSpeed adapted the package from a previous design. The company intends to shrink the package in a future revision. Except for the redundant pin count, the CS301's statistics are impressive for a massively parallel extreme processor based on a standard-cell, fully synthesized design.

Inside the Hive Mind

To squeeze 25.6 GFLOPS or 12,800 native mips out of a chip running at only 200MHz, ClearSpeed created a massively parallel architecture with 64 processing elements (including 128 FPUs); a high-speed on-chip bus that connects all the elements together; 384KB of local memory; hardware-controlled multithreading; and a few other interesting features.

At first glance, ClearSpeed's Multithreaded Array Processor (MTAP) architecture resembles two other extreme architectures recently announced by U.K. companies: the picoArray from PicoChip (see *MPR 10/14/03-03*, "PicoChip Makes a Big MAC," and *MPR 7/28/03-02*, "PicoChip Preaches Parallelism") and the D-Fabrix from Elixent (see *MPR 7/21/03-01*, "Elixent Expands SoCs"). All are massively parallel array architectures with dozens or hundreds of processing elements, each with its own local memory resources to reduce the need for off-chip I/O. Although the overall design of these chips is complex, their individual processing elements are fairly simple, so they don't follow the classic "Brainiac" philosophy of achieving high performance with intricate pipelines and complex control logic. Nor do they belong to the opposite class of "speed demons"—at only 200MHz, ClearSpeed's CS301 has the highest clock rate in this group.

Instead, these massively parallel processors more closely resemble a colony of bees with a large number of workers whose efforts for the common good are loosely coordinated by a central intelligence. For that reason, we have coined the term "hive mind" to describe these designs.

Despite the broad similarities to other hive-mind processors, ClearSpeed's MTAP architecture has some distinguishing characteristics, primarily that the processing elements are homogeneous. That attribute sets it apart from the picoArray and other parallel architectures that use different types of processing elements for different kinds of tasks.

In the CS301, each of the 64 identical processing elements has a 32-bit ALU; two 32-bit FPUs (a multiplier and an adder, with an associated division/square-root unit); a 16-bit multiply-accumulate (MAC) unit; and a load/store unit with 64-bit interfaces. All five arithmetic units can operate on every clock cycle. If every arithmetic unit in every processing element fired at once—an unlikely but entertaining scenario—the CS301's peak performance at 200MHz would exceed 100 billion native operations per second (GOPS). The FPUs can perform a 32-bit IEEE-754 floating-point multiply or add in four cycles, with single-cycle throughput. The processor also supports 8-, 16-, 24-, and 32-bit fixed-point arithmetic.

Unlike the function units in conventional processors—which usually share a common register file with other function units—each processing element in the CS301 has its own local register file, data memory, DMA, and address-generation logic. Missing is local instruction memory, another significant difference between the CS301 and some other massively parallel array processors, such as PicoChip's PC101 and PC102. Lacking local instruction memory, the 64 processing elements in the CS301 cannot execute different instructions at the same time. Instead, they act in concert as a very wide parallel processor that runs the same set of instructions against different parts of a data set. ClearSpeed refers to the 64 processing elements collectively as a "poly processor."

In keeping with the many data types this processor supports, the register file is uncommonly flexible. It can store 64 bytes and is byte addressable, so it's not limited to a specified number of register slots with fixed widths. Byte-size or larger operands can start at any boundary. The register file has two 32-bit read ports and one 32-bit write port for the processing element's function units, plus two 32-bit I/O ports for sharing registers with adjacent processing elements. Figure 2 is a block diagram of a single CS301 processing element.

To minimize off-chip I/O, each processing element has 4KB of SRAM for data. This allows a program to store a chunk of data in local memory while performing a series of operations. When I/O is necessary, each processing element has a load/store unit for programmed I/O or streaming I/O, each having its own pair of 64-bit interfaces to the I/O bus. The I/O units have independent DMA, so they can transfer data simultaneously.

Programmed I/O, in ClearSpeed nomenclature, means traditional load/store operations, with a few enhancements to take advantage of parallel processing. If two or more processing elements need data from the same address at the same time, the CS301 consolidates the requests, sends only one request over the bus, and "broadcasts" the returning data to all processing elements that need it. Programmed I/O can also fetch data by striding through memory in hops based on a constant offset from a base address, which is useful for fetching any data (such as video) that is striped in memory.

Streaming I/O is primarily for passing line-rate data through the parallel processing array. It assumes the data is arriving in a constant stream, instead of being pooled in memory at fixed addresses. The CS301 chops the datastream into pieces and distributes the data among the processing elements, transferring up to 128 bytes at a time to and from each element. Larger packets can be distributed across multiple processing elements. Different elements can receive different types of datastreams simultaneously, and the CS301 can simultaneously handle both programmed I/O and streaming I/O.

The CS301's I/O bus, called the ClearConnect bus (no relation to IBM's CoreConnect), is 64 bits wide and runs in full-duplex mode at 200MHz, providing 1.6GB/s of off-chip memory bandwidth in each direction. In addition, a pair of 800MB/s bridge ports allows system designers to daisy-chain as many as 16 CS301 chips together, so it's possible to build a multiprocessor system that delivers 409.6 peak GFLOPS and consumes only 40W, worst case, for the processors. In fact, ClearSpeed built a 200-GFLOPS demo system with four development boards—each with two CS301 chips—for the Supercomputing Conference in Phoenix, Arizona, this November.

Additional on-chip memory is available in the form of 128KB of global SRAM, which can store program instructions or data for all 64 processing elements. In multiprocessor configurations, this scratchpad memory is accessible to every other CS301 chip in the daisy chain, appearing as just another range of memory addresses.

Multithreading Improves I/O Efficiency

Three additional features distinguish the CS301: hardware-controlled multithreading; an unusual instruction pipeline that resembles the pipes of two different microprocessors bolted together; and an instruction set that's customizable at run time.

Hardware multithreading has been popping up in all kinds of places, from small packet processors like Ubicom's IP3023 (see *MPR 4/21/03-01*, "Ubicom's New NPU Stays Small") to powerful PC processors like Intel's Pentium 4 (see *MPR 12/2/02-01*, "Intel's Hyper-Threading Takes Off"). At last month's Microprocessor Forum, MIPS Technologies announced that some of its future embedded processors will use the technique. (See *MPR 11/10/03-01*, "Multithread Technologies Disclosed at MPF," and *MPR 11/17/03-03*, "Will Microprocessors Become Simpler?") The basic concept is to allow the microprocessor to manage multiple contexts of execution at the hardware level, transparently to the operating system.

ClearSpeed's implementation of hardware multithreading in the CS301 supports eight simultaneous threads—a respectable number. (Ubicom's more-specialized IP3023 packet processor also supports eight threads, whereas Intel's Hyper-Threading technology in the Pentium 4 currently supports only two threads.) ClearSpeed expects programmers to reserve one thread for system-level operations, dedicate another thread to compute tasks, and use one or two threads for asynchronous I/O. Hardware multithreading allows the processing elements in the CS301 to simultaneously fetch data with their dual I/O units while crunching data in another thread.

In other words, ClearSpeed uses hardware multithreading primarily to allocate the CS301's compute and I/O resources more efficiently. Other multithreaded processors tend to use a finer-grained version of hardware multithreading to reduce the incidence of pipeline bubbles caused by context switching. That version of the technology, called simultaneous multithreading, mixes instructions from different contexts of execution in the same pipeline at the same time, filling in the slots created by pipeline bubbles. Bubbles are a lesser problem in the CS301, because it's not superpipelined and probably won't switch contexts as often as a general-purpose processor does.

Mono Execution vs. Poly Execution

To manage hardware multithreading and other internal housekeeping chores, the front end of the CS301 pipeline consists of a function unit called the mono execution unit (MEU). It resembles a self-contained RISC processor, with its own ALU, load/store unit, 4KB instruction cache, 4KB data cache, and three-stage fetch-decode-issue pipeline.

Indeed, ClearSpeed refers to the MEU as the "mono processor" that feeds the massively parallel "poly processor" (the 64 processing elements). The mono processor's pipeline

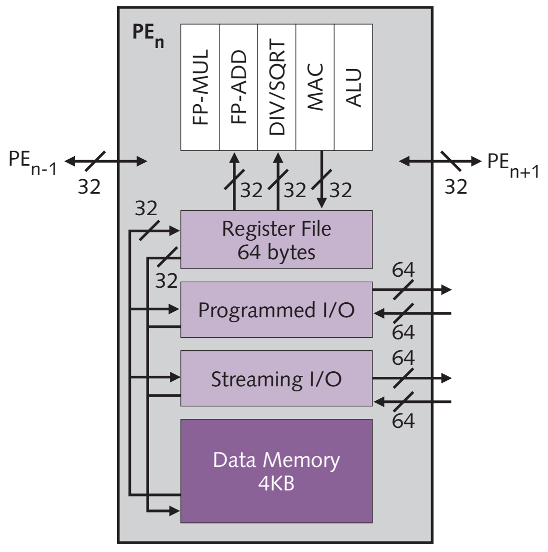

Figure 2. Each processing element in the CS301 resembles a miniature 32-bit microprocessor with its own function units, registers, local data memory, and I/O capabilities. All 64 processing elements on the chip are identical.

is actually the front end of a longer pipeline through the poly array. The mono processor locally executes scalar data operations (such as loop counters) and control-type instructions, but it passes other types of instructions into the pipelined function units of the processing elements. In a way, it's the queen bee of the hive mind. Figure 3 shows how the mono processor fits into a high-level block diagram of the CS301.

An unusual feature of the CS301's programming model is that the instruction opcodes don't distinguish between arithmetic operations destined for execution by the mono processor or the poly processor. Instead, programmers (or the C compiler) tag the *operands* to identify where execution should take place. Each operand has the prefix "m" or "p," to specify mono or poly, followed by an integer that specifies the operand's size in bytes, followed by a suffix that defines the operand's data type. Here's an example of a poly add instruction that the CS301 will execute with its parallel processing elements:

```
add 0:p4u, 4:p4u, 8:p4u;
```

All the values in this example are 32-bit unsigned integers, as indicated by the "4u" suffix. The first operand specifies the destination register, and the other two operands are

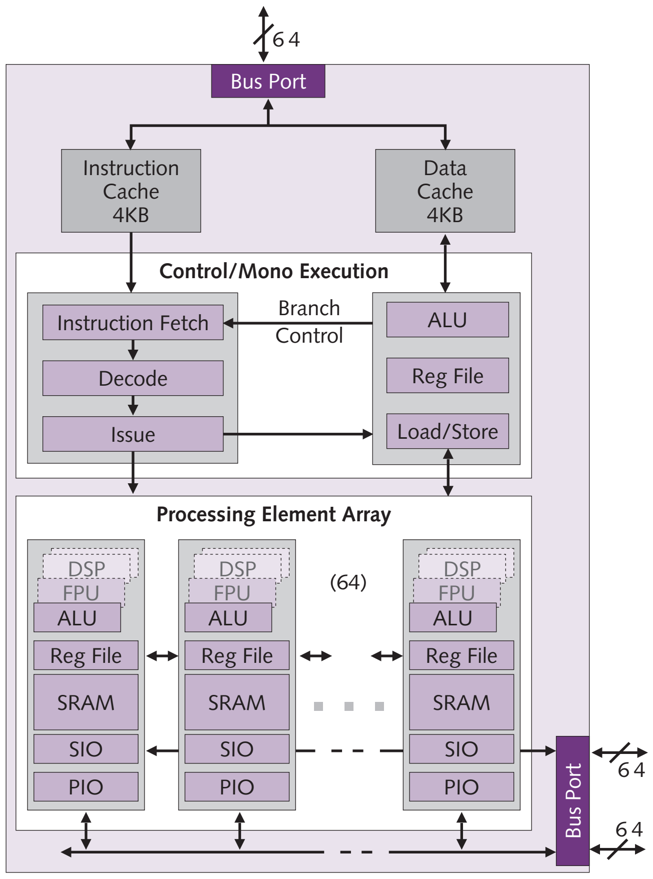

Figure 3. CS301 block diagram. At the top is the mono execution unit or "mono processor," a processor-within-a-processor that executes some instructions itself and passes other instructions along to the array of parallel processing elements, or "poly processor."

sources. This unorthodox instruction format allows a single instruction to perform an arithmetic operation using both the mono and poly processors, as in this example:

```
mul 0:p4f, 4:p4f, 0:m4f;
```

In this case, the mono processor will hold one of the source operands (0:m4f, a 32-bit floating-point value) in a local register and "broadcast" it to all the processing elements in the poly array. The processing elements will multiply the mono operand by another 32-bit floating-point value in their own registers (`4:p4f`). Finally, the processing elements will store the 32-bit results in their own destination registers (`0:p4f`).

Despite the complexity of this architecture, ClearSpeed makes the same claim as many other extreme-processor vendors: it's programmable in C, not just in assembly language. ClearSpeed makes a strong case: a program written in ANSI-standard C will run on the CS301 without any modifications at all. The only catch is that the mono processor will execute everything. To exploit data parallelism with the poly processor, programmers will have to modify the C source code.

ClearSpeed says existing C code is easy to modify, especially if it already uses the vector-processing extensions of another instruction set, such as the x86's SSE or the PowerPC's AltiVec. Mainly, it involves adding the keyword poly to the operands. Here's a before-and-after example of a short C subroutine that performs vector additions on two 64-element arrays of 32-bit floating-point values:

```
/* BEFORE */
void vec_add_64_elements(float *sum, float
*a, float *b)
{
 unsigned int i;
 for (i=0; i<64; i++)
 sum[i] = a[i] + b[i];
}

/* AFTER */
void vec_add_64_elements(poly float *sum,
poly float *a, poly float *b)
{
 *sum = *a + *b;
}
```

When the second version of the routine is compiled, the result will be object code that has the prefixes and suffixes seen in the assembly-code examples. ClearSpeed says the CS301 supports all features of ANSI C, including pointer arithmetic by both the mono processor and poly processor units.

ClearSpeed's software-development kit runs on Windows, Linux, and Solaris. It includes a C compiler, assembler, visual debugger, reference source code, optimized code libraries, instruction-set simulator, and cycle-accurate simulator. More optimized code and support is available from ClearSpeed and its development partners.

Instruction Set Is Customizable

The CS301's instruction-set architecture is an alphabet soup of RISC, CISC, VLIW, and SIMD, with a dash of FPGA thrown in for extra flavor. It's RISC-like in the sense that most instructions follow a load/store model and the familiar three-operand format (two input operands plus a destination register). In addition, the instructions are fixed length at 32 bits. Like CISC instructions, however, the CS301's instructions translate into microcode-type instructions that may require multiple cycles to execute. This feature is what allows programmers to modify the instruction set at run time.

Microinstructions are encoded in on-chip SRAM. In effect, it's an instruction-set lookup table, like the microcoded instruction sets in many CISC processors. A regular instruction acts like a pointer into the table, which issues 100-bit-long microinstructions to the processing elements. There's enough memory in the lookup table to encode a maximum of 256 instructions, but the base instruction set has fewer than 100. Programmers can use the extra memory in the table to define their own instructions for specific applications, a powerful feature. Moreover, the instruction set can change at run time, so the CS301 can adapt to different applications on the fly.

Configurable processors like ARC International's ARCtangent-A4 and Tensilica's Xtensa V have customizable instruction sets, but their configurations are fixed at design time. The instruction sets don't change at run time, as the CS301's can. However, developers can customize an ARC or Tensilica processor more extensively by modifying the I/O buses, adding new registers, and designing entirely new function units with application-specific logic. The I/O buses and register files in the CS301 are not configurable, and custom instructions must use the processor's existing function units.

The number of custom instructions programmers can create for the CS301 depends on the instructions' complexity. To reach the theoretical maximum of 256 total instructions, each instruction would have to execute in an average of four clock cycles. A complex instruction that requires 64 cycles would occupy the space of 16 four-cycle instructions. Such a complex instruction is not unimaginable, because the CS301 allows programmers to encapsulate an entire subroutine in a single instruction.

For instance, one custom instruction could perform the butterfly operation of a fast Fourier transform (FFT). The group of microinstructions that forms this complex instruction would resemble the bundle of operations in a VLIW instruction, or perhaps a SIMD instruction that operates on multiple operands. And like a complex CISC instruction, the custom instruction would reduce the demand for instruction-fetch bandwidth, because one instruction would do the work of a whole routine.

Programmers can use their custom instructions with ClearSpeed's software-development tools, including the C compiler, assembler, debugger, and simulators. The C compiler won't automatically use custom instructions during compilation, but programmers can invoke the instructions with inline assembly language or intrinsic functions. This is similar to the way custom instructions work with the tools for configurable processors.

EEMBC benchmarks for the ARCtangent-A4 and Xtensa show that custom extensions can boost the processor's baseline performance by an order of magnitude or more. Without the ability to incorporate application-specific logic, the CS301 almost certainly can't match those results, but the performance improvement afforded by custom instructions should still be significant.

Is the Memory Bus a Bottleneck?

Unfortunately, ClearSpeed lacks any SPEC or EEMBC scores to tout—the company hasn't yet joined either benchmarking consortium. There are, however, some data points available for estimating the CS301's performance: peak GFLOPS and some informal benchmarking by Lockheed Martin, a potential customer.

Lockheed used a cycle-accurate simulation of the CS301 to run two programs optimized by WorldScape Defense, a U.S. company that specializes in software for sensor arrays, immersive imaging, telepresence, and other visualization technologies. One program executed eight FFTs in parallel, each a 1,024-point, complex floating-point FFT. The other program executed eight pulse-compression routines in parallel, each consisting of an FFT, an inverse FFT (iFFT), and a complex multiply by a stored reference FFT. As Table 1 shows, the CS301 outperformed a Motorola PowerPC MPC7410 and delivered much greater power efficiency in this limited test.

Calculating peak GFLOPS is a straightforward way to compare microprocessors, but it has limitations. The CS301 has two FPUs in each of its 64 processing elements, and they can work in parallel, so they can execute a theoretical maximum of 128 32-bit floating-point operations per clock cycle. At the target clock frequency of 200MHz, this adds up to a peak 25.6 GFLOPS—an impressive number that easily beats the peak GFLOPS for general-purpose processors in PCs, workstations, servers, and even the world's most powerful supercomputer.

For instance, an Intel Pentium 4 processor can execute four 32-bit floating-point operations per cycle, using SSE

	Frequency	Power	FFT/sec	Pulse/sec
ClearSpeed CS301	200MHz	2.0W	113,740.0	49,960.0
Motorola MPC7410	400MHz	8.3W	25,331.6	6,490.6
Perf. Gain	—	—	4.5x	7.7x
Pwr Efficiency	—	—	18.6x	31.9x

Table 1. In these informal benchmarks by Lockheed Martin, ClearSpeed's CS301 easily beats a Motorola PowerPC processor that runs at twice the clock rate and consumes four times as much power. Equally important for many embedded systems, the CS301 has an even larger advantage in power efficiency. ClearSpeed says it has since verified these simulated results on actual silicon: performance is slightly better, and power consumption slightly lower.

instructions and 128-bit registers. At 3.4GHz—the fastest clock frequency likely to be available in 1Q04, when the CS301 may ship—the Pentium 4 can deliver a peak 13.6 GFLOPS, or slightly more than half the performance of the CS301 at 17 times the clock speed. Of course, for the price of one CS301 chip, a customer might be able to buy two 3.4GHz Pentium 4 chips, which would provide an aggregate peak of 27.2 GFLOPS. But the pair of Pentium 4 chips would burn nearly 200W, whereas the CS301 doesn't even simmer at 1.8–2.5W.

Workstation/server processors also fall short of the CS301's peak 25.6 GFLOPS. Intel's Itanium-2 can execute a peak 6 GFLOPS at 1.5GHz. IBM's PowerPC 970, known to Apple users as the PowerPC G5, can execute a peak 16 GFLOPS at 2.0GHz when using its vector FPU for SIMD operations (four 32-bit multiply-adds per cycle). Even the custom ES microprocessors in the world's fastest supercomputer—the Earth Simulator at the Japan Marine Science and Technology Center—cannot match the CS301's peak performance. The ES processors are based on NEC's SX-6 microprocessor, which can execute a peak 8 GFLOPS at 500MHz. (See *MPR 3/17/03-01*, "Scalable MicroSupercomputers.")

But there's a catch. Peak GFLOPS represent the maximum possible performance under ideal conditions at a specific moment in time. It's a far cry from sustained performance under real-world conditions. For instance, on paper, the Pentium 4's peak floating-point performance easily beats the Itanium-2. The SPEC benchmarks, which simulate actual conditions with a real workload, tell a very different story: the 1.5GHz Itanium-2 scores 2,119 SPECfp_base2000 compared with 1,092 for a 3.06GHz Intel Xeon. The Itanium-2 delivers twice the sustained performance at half the clock speed of the Pentium 4 because it has a much larger register file and an architecture that allows more-efficient instruction scheduling for compute-intensive applications.

The CS301's potential for sustaining its superlative peak performance is suspect. The bottleneck will likely be I/O bandwidth to main memory, at least in certain kinds of applications. With a 64-bit bidirectional bus that runs at 200MHz, the CS301 provides only 1.6GB/s of off-chip memory bandwidth in either direction, or 3.2GB/s of aggregate memory bandwidth. (The bridge ports provide another 1.6GB/s of aggregate I/O bandwidth, but only with other CS301 chips in multiprocessor designs.) For most floating-point applications, 3.2GB/s seems like insufficient plumbing to supply a massively parallel processor that has 128 FPUs.

In comparison, the Itanium-2 has a 128-bit bus that runs at 400MHz, providing 3.2GB/s of bandwidth in each direction, or 6.4GB/s of aggregate I/O bandwidth. The Pentium 4 has the same amount of I/O bandwidth as the Itanium-2, albeit with a bus half as wide (64 bits) and twice as fast (800MHz). The PowerPC 970/G5 can provide more than 7.0GB/s of aggregate data bandwidth, using a pair of 32-bit unidirectional buses that can run at 1.0GHz (one-half the 2.0GHz core clock frequency).

Several factors may work in the CS301's favor. General-purpose processors tend to run applications with a higher instructions-to-data ratio than scientific applications have, so they need more I/O bandwidth for instruction fetching. The ability to customize the CS301's instruction set at run time can greatly reduce the number of instructions it needs to fetch. The CS301's processing elements are more self-contained than ordinary function units, thanks to their local register files and data memories, so they need not access off-chip memory as often. And hardware multithreading will allow the CS301 to use its I/O bandwidth and compute resources more efficiently.

Even so, the CS301 will fare better in compute-intensive applications rather than data-intensive ones. In other words, it's more suitable for applications in which the processor carries out many operations on the same data instead of applications requiring relatively few operations on streams of data. When the 64 processing elements are fetching data from their local memories over their internal 32-bit buses, the on-chip data bandwidth is 51.2 GB/s (64 × 32 bits × 200MHz).

Therefore, as ClearSpeed suggests, the CS301 may indeed excel at testing many different drug molecules against a protein. It could store the protein data in the local memories of the processing elements and run 64 different models of the drug molecule simultaneously. However, the CS301 may be less impressive at applying a signal-processing algorithm to a continuous datastream from a broadband communications channel. Only careful benchmarking will reveal these characteristics.

A Unique Extreme Processor

By definition, all the microprocessors we call extreme processors have exotic architectures, unusual features, and potentially outstanding performance in at least some applications. Yet even in this exclusive club, ClearSpeed's CS301 is unique. Its parallelism is becoming more commonplace, but the additions of eight-level hardware multithreading and a programmable instruction set differentiate the CS301 from other extreme processors in significant ways. Certainly, its 25.6 peak GFLOPS and 10 GFLOPS per watt put the CS301 in a class by itself.

That said, the CS301 isn't quite the revolution that some press reports have implied. The CS301's ability to sustain its high floating-point performance is yet to be demonstrated under real-world (or even industry-standard synthetic benchmark) conditions.

ClearSpeed has been suggesting that the low-wattage CS301 can bring world-class floating-point capability to any PC or laptop by adding a PCI card or PC Card, but that would impose an even tighter I/O bottleneck on data throughput. The PCI buses in most PCs and the PCMCIA CardBus interfaces in most laptops are only 32 bits wide and run at 33MHz, offering a mere 133MB/s of bandwidth—about 4% as much bandwidth as the CS301's I/O bus. In some applications, a plug-in accelerator with a great deal of onboard memory to buffer the data might be useful, but it clearly isn't the ideal way to take advantage of the CS301's capabilities.

The ideal system would be designed around the CS301, much as Japan's Earth Simulator is designed around NEC's custom ES microprocessors. The cool-running CS301 lends itself to a massively parallel system architecture that packs dozens or hundreds of chips into a small space. Server blades that plug into a high-bandwidth backplane are another possibility. Of course, those kinds of systems would require a bigger design win than a PCI card or a PC Card—but an extreme processor deserves an extreme system architecture. ◇

Price & Availability

ClearSpeed has production samples of the CS301 processor and says it can ramp up production in 1Q04, if warranted by customer demand. Simulators, development boards, and software-development tools are available now. Production chips will cost $975. For more information, visit *www.clearspeed.com*.