Visionaries See Beyond Superscalar

But They Disagree Over What Will Follow—MP on a Chip?

by Linley Gwennap

Gathering at this year's Microprocessor Forum, a group of computer visionaries looked beyond the next few generations of superscalar processors to see what lies ahead. Will superscalar techniques reach a limit after four-to-six-issue designs? Will new techniques continue to deliver ever-increasing performance, or will computer designers be out of jobs as circuit design becomes the driving factor?

John Hennessy, Stanford professor and co-founder of MIPS, took the first swing at these questions by pointing out some of the difficulties of extending today's superscalar designs much further. "I'm sure that the people who have built caches to support two load/stores per clock are quaking at the thought of doing four or five or six load/stores per clock," he said. "Similarly, the issues of dispatching instructions become much simpler if we move toward a true multiprocessor."

The multiprocessor that Hennessy advocates is not necessarily the type that is used today. If we agree that an MP design is needed, he asked, "Should we just take two processors and slap them down, maybe with a single secondary cache, or should we integrate those two processors at a more detailed level? …I wouldn't be surprised to see machines that are a balance between two individual processors and a superscalar machine."

Perhaps such a design could fetch instructions from multiple streams simultaneously, maintaining a copy of the register file for each context, but draw on a common pool of functional units to execute the instructions. This could be what Digital's Dick Sites meant when, at last year's Forum, he mentioned a future processor with multiple program counters.

This virtual multiprocessor would require a single-chip implementation for such tight integration. As Hennessy pointed out, future chips with 20 million transistors (or more) will have plenty of room for multiple processors.

Tom Blank of Maspar agrees with this point; after all, his company has already put 32 RISC processors on a single chip. He believes that even current superscalar processor designs have gone too far down the path of complexity. "The concept of a RISC processor is basically dead at this point," he asserted. "How much more complexity do we want to put into these supposedly RISC processors before we just start increasing the number of processors?"

Josh Fisher, a researcher at Hewlett-Packard Labs and co-founder of Multiflow, provided a variation on this theme. He agrees with Blank, saying that, "Out-of-order execution and scoreboarding, all that stuff is the exact opposite of what RISC is all about."

Very Long Instruction Words

Fisher's dream machine, however, is different than either Blank's or Hennessy's. Harking back to the comparison of Brainiacs and Speed Demons *(see **0703ED.PDF**)*, he said, "What we really need are Speed Demon/Brainiacs. If we take the functional units available in the Brainiacs and get rid of all of that run-time control, we will be able to build machines that have incredibly fast cycle times, because they are just functional units without the complex control."

Fisher alluded to the very long instruction word (VLIW) machines that Multiflow developed in the mid-eighties. The basic principle of VLIW is to provide a large number of functional units (adders, multipliers, load/store, etc.) in the hardware but give the compiler the burden of scheduling instructions for these units. As indicated by the name, a VLIW processor fetches a single gigantic instruction per cycle; the instruction has been carefully encoded by the compiler so operands and commands flow directly to the functional units with minimal decoding.

The downfall of the traditional VLIW machine is the requirement that code be recompiled specifically for each processor, taking into account the particular configuration and number of functional units in that CPU. The concept of software portability via binary compatibility, so important today, is eliminated by VLIW machines.

Fisher pointed out a possible solution to this conundrum: If the final portion of compilation is done when software is installed on a particular system, the final binary could be targeted specifically for that system. Today, this technology is untested, but Fisher is optimistic; quoting his colleague Bob Rau, he hopes that, "In ten years, all high-performance microprocessors will be VLIW but will be called superscalar."

Read My Lips—No New Instruction Sets

Dennis Allison, former Chief Scientist at HaL Computer, sees a similar problem for future highly superscalar processors. "The biggest bottleneck, even for the RISC architectures, is the size of the installed base," he said. "This means backwards compatibility forever and ever, and means that we can't do some of the things that we would like to do to improve performance."

His solution is much like Fisher's install-time compiler: "I think you'll see a new kind of system program that

(left to right) John Hennessy, Josh Fisher, Dennis Allison, Don Alpert, Tom Blank, and Michael Slater (moderator) trade visions of the future.

takes an old binary, expends an enormous amount of effort to unpack it and deduce what it's doing, and then readjust it so that it will execute quickly on a particular machine."

Allison takes a more conservative stance, however, regarding the more exotic ideas of Fisher and Hennessy. He sees future increases in performance coming from faster clock speeds, improved circuit design, wider superscalar dispatch, and improved instruction scheduling in both hardware and software.

This conservatism extends to instruction sets as well; despite the "warts" of existing architectures, Allison sees no room for new ones, due to the installed base of software for current instruction sets. Hennessy agreed that "none of the RISC machines is much worse than any of the others. One good circuit designer is worth more than the differences among the RISC architectures."

Don Alpert, senior architect at Intel, admits that his company faces a bigger challenge due to its CISC-style instruction set, but feels that it can be met. "Some people are skeptical that we can build six- or eight-issue x86 machines, but anyone who believes that is underestimating his competition," he said. Even RISC-guru Hennessy admitted that such a design "is doable. It's a question of whether Intel has more money than anyone else."

In Search of More Parallelism

All the panelists agreed that compilers will continue to play a key role in improving performance. Hennessy pointed out that this approach is necessary but no easier than beefing up the hardware; as he noted, "The only thing more complex than a state-of-the-art processor is the guts of a state-of-the-art compiler."

To take full advantage of the MP systems proposed by Hennessy, compilers must take a single application and break it into multiple tasks, thereby increasing the parallelism. This technique, known as multithreading, is in its rudimentary stages today, and even Hennessy admitted that "the problems with languages and compilers are very tough and need a lot of work."

Blank agrees that the multithreaded approach is good for many applications, but for others he proposes a "super vector" machine. In supercomputer terminology, multithreaded MP is called multiple-instruction, multiple-data (MIMD) while the vector approach is known as single-instruction, multiple-data (SIMD). Blank sees the latter technique as more applicable to data-intensive applications such as sifting through large databases or processing video signals.

Fisher stood alone in his belief that superscalar techniques have plenty of headroom for single-processor implementations. He believes that the academic studies in this area are misleading, pointing out that "these studies show that, for some applications, you can't get as much parallelism as we already get [in current implementations]. They can't take into consideration things like what algorithms the compiler will use to rearrange code, things that are comparable to the wild things that vectorizers do now…The truth is, only a handful of people have worked at the really hard problems in compiling for instruction-level parallelism."

Alpert also sees hope for increases in instruction-level parallelism, but believes it will come from new applications, not new compilers. "We always optimize our compilers for the previous-generation workload," he stated. "Future workloads could have significantly more parallelism [in areas such as] audio and video compression, speech recognition, and image processing."

Do We Really Need More Performance?

Perhaps, in the end, the need for increased performance will expire before our ability to deliver it. Blank proposes that, once we have enough power to create the ultimate user interface—perhaps virtual reality, or a direct connection to the brain—only remote servers will need better performance. But this ultimate interface will require at least an order of magnitude more performance than we have today. Hennessy isn't concerned about such a limit. "I'm an optimist," he declared, "and I hope that the thoughts of our programmers will never be so limited that they can't find the next thing to do." ♦