Gordon Bell, assistant director for computer and information science and engineering at the National Science Foundation since July 1986, was a founder of the Dana Group, in Sunnyvale, Calif., before he joined the NSF. He was also a founder and vice chairman of technology for Encore Computer Corp., in Marlboro Mass.

From 1960 to 1983, Bell was with Digital Equipment Corp., where he rose to the position of vice president of engineering and was chief technical officer for 11 years. At DEC, he led the development of the highly successful VAX lines of superminicomputers.

From 1966 to 1972, while serving DEC as a consultant, Bell was on the faculty of Carnegie-Mellon University, where he was one of the architects of its pioneering systems in parallel processing.

Bell, a Fulbright scholar with a BSEE and MSEE from MIT, is a member of the National Academy of Engineering and a Fellow of the IEEE, the American Association for the Advancement of Science and the Association for Computing Machinery. He has been honored with the Mellon Award, the McDowell Award and the Eckert-Mauchly Award for contributions to computer design.

Computer Architecture

Gordon Bell, Assistant Director,
Computer and Information Science and Engineering
National Science Foundation

The microprocessor has assimilated most of the ideas in computer architecture that were developed in mainframes, superminis and minis. Most microprocessor designers were not schooled in computer architectures. They had to learn about computing, and that process took a long time. Today, however, microprocessors have evolved to the point where there are a number of them that I would call "almost-good" micros. From the viewpoint of architecture and other important issues, some of them are actually much better than minis and mainframes.

If I were to ask myself, "What are the significant ideas in architecture over the past 15 years?" I'd answer that the most important one that comes up, overall, is the adoption of the vector as a data type. The TI ASC, of more than 10 years ago, is one example. Another is the CDC Star and—most significant—the Cray 1. This is the cornerstone architecture on which others have built.

Next is the idea of the multiprocessor. I'm particularly interested in the idea of taking multiple microprocessors and making either high-performance multiprocessor machines or high-availability machines, like the Stratus machine and other parallel machines.

My big complaint, though, is that microprocessors haven't really addressed multiprocessing correctly. We've known about multiprocessing for some time and we should have dealt with it.

I claim that the emergence of the microprocessor as the heart of a general-purpose computer occurred in the 1981-1982 timeframe. That's when, with the 68000 chip, you had a big enough address space—a 32-bit address space. I don't regard earlier micros as suitable for computing. To me, the Apple Macintosh is the first significant machine to use it.

Intel's microprocessors spawned a lot of machines on the market. But Intel forced all users to retrace history. The 8080, for example, was a low-performance microprocessor whose main contribution was allowing people to learn to build PCs and such. Mostly, it was a controller.

By architecting the operating system the way it was set up and using the architecture of the 8080 and its derivatives, Intel and Microsoft forced users to relive all of computing history.

The good news was that everybody got a chance to relive history. Everybody got a chance to live through all the dumb things that were done through history: small address spaces that forced people to write small programs; the lack of memory management; the necessity for overlays and all the techniques that you use when you don't have a big enough memory.

And then, though the machines got better, you had to contend with MS-DOS, a '70s-style operating system, which didn't have a big enough memory, either.

I think the whole thing was an interesting experience, but a painful one for the poor users. The 8086 did make an impact, for it was the basis of the PC family. I put its successor, the 80286, into the same category. The 80386, however, stands a better chance.

So, from a computational standpoint, Intel's microprocessors sent people down the wrong path. They may all recover, someday, if MS-DOS can adopt the Mac operating system.

All the Unix machines, the workstations and the Macintosh stayed away from Intel's microprocessors. They all went with the 68000 just because of its bigger address space. There is nothing that is near the Mac yet. The 80386 is important because it has a very large address space—and the potential to be a real computer.

Everything is going to go the way of the Mac. The IBM PS/2 will end up looking like the Mac with windows and graphics capability. But it's going to have a hard time getting there because there is so much of an investment in software in the old models.

Not that the Intel microprocessors were not important. They were, but for a different reason. The Intel micros led to an enormous popularizing of computers, but they set us back technologically. The Motorola microprocessors enabled new kinds of machines to be built.

So the 68000 led to workstations, to the Mac, to Unix computers and also to parallel machines.

All this leads to the class of machine I call the "multi," which is a multiple microprocessor machine. This is where it gets interesting, for now we take off to go after the mainframes. I don't think Motorola will continue to play such a pivotal role in the future because there are now lots of 32-bit micros, and the modern ones are based on the reduced-instruction-set-computer (RISC) concept.

I should mention here that the C programming language has been and will be very important. We will all program in C. People will abandon the 68000, because everybody will be weaned from assembly language and will program in C. C will be the base machine. That will be another significant event.

But back to microprocessors. For reasons I mentioned before, the 80386 may become as significant as the 68000. But I think the MIPS chip is the most significant chip in the last few years. It is a RISC computer, and it is a factor of two to three times faster than any of the other chips. It is also smaller, and outperforms the fastest superminicomputers, such as DEC's 8800. Sun has also introduced its chip architecture.

Motorola and Intel have followed the traditional computing line established by the DEC VAX and the IBM 360. IBM discovered RISC in the late '70s, but it's taken 10 years for RISC chips to make an appearance. I think all computers will eventually switch to RISC. It's an idea whose time has come because memory and logic now operate with the same speed.

The next big idea is massive parallelism. Micros facilitate it because, essentially, you can have virtually zero-cost processing nodes. You can put them together just as you put memory chips together, and in doing that you can get enormous processing power.

Parallelism has been attempted since the early days of computers but nobody really made it work. The Burroughs 5000, for example, was a dual processor. It could run in a multiprocessing mode in which you ran independent jobs. What we're talking about today is running a single job—a very large job—on a lot of processors.

I see three distinct architectural styles made possible by the almost-zero cost of microprocessors. There's the multiprocessor form, which is a number of processors sharing memory. There's a multicomputer form, which is a number of independent computers that distribute a problem throughout the computers and the computers talk to one another in a message-passing fashion. The transputer was designed to operate this way, but I don't see it as important simply because it mixes the computer and a particular communications method too closely. And then there is the workstation-cluster form, in which the workstations are essentially multicomputers that talk to one another over a local-area network.

The microprocessor (or one-chip processor) has had an incredible impact—and that impact will increase, partly because of its cost and now because of its prodigious power. Micros have crossed the line to the point where they will take over the niche occupied by many ECL machines. Finally, we see the one-chip ECL processor by 1990. It will simply blow away the largest traditional architectures at DEC, IBM, Intel and Motorola . . . and allow very low-cost, but 50- to 100-Mips, performance. Of course, these architectures will continue to be implemented as "code museums" to run the billions of lines of user code. They won't be that significant in the new applications where performance is important. The future really looks exciting. **EET**