The year 1988 is coming to a close. It has been about 20 years since the introduction of the first microprocessor and roughly a decade since the appearance of the first digital signal processing (DSP) micro. Intel was the progenitor in both cases, but, as with most efforts, many diverse contributions were needed as precursors, and many companies followed with their own offerings. The first chips introduced (Intel's 4004 and 2920) seem, by today's standards, quite primitive. But each marked the beginning of an evolutionary cycle.

Growth, however, is exponential. As knowledge and experience grow, so does the complexity of the devices being built. And so grows the case with which end users can incorporate those devices into ever more complex systems.

The first special issue of IEEE Micro on DSPs appeared in December 1986. Readers received detailed architectural descriptions of what were then state-of-the-art DSPs: 16-, 24-, and 32-bit chips, typically having a dual-bus Harvard architecture and performing fixed-point arithmetic. Multiply-accumulate times were on the order of 100 nanoseconds. Most of the processors could effect finite-impulse-response digital filters at the rate of one tap per cycle. The execution times for 1,024-point, complex fast Fourier transforms, or FFTs, on second-generation fixed-point machines ranged from 3 to 8 milliseconds, approximately the times offered by array processors designed and marketed a decade earlier. However, a difference in cost, size, weight, and power consumption of three orders of magnitude separates the processors of these two eras.

The DSP micros described in this issue exhibit obvious improvements over their predecessors. Most important is the capability to perform floating-point arithmetic with multiply-accumulate times similar to those of earlier fixed-point DSPs but with an increased degree of parallelism. Thus, they provide a greater throughput for a given cycle time. Another factor is the availability, for these machines, of relatively powerful high-level language compilers.

While the floating-point format obviously allows an increase in dynamic range, and in most cases, of precision, it is floating-point capability in conjunction with advanced compiler availability that leads to a breakthrough not so immediately obvious. The nature of this advance follows from an examination of shortcomings in the traditional approach used in developing DSP systems. This methodology generally incorporates a high-level design phase and an implementation phase.
The design phase is often performed with the aid of a high-level language or a commercial, DSP-oriented “design system” that yields a nonreal-time, floating-point simulation on a general-purpose computer. Designers carry this step out to evaluate the efficacy of different algorithms as system components and to determine the various system parameters. Once this phase is completed, users can generally implement the system with programmable processors or special-purpose hardware.

The second, or implementation, phase has been distinct from the first for two reasons. General-purpose microcomputers are too slow, and their supporting components too expensive, to be regarded as targets for real-time implementations. On the other hand, it is most often not appropriate to use the 1986 generation of fixed-point DSPs for the simulation and parameter-setting phases. They are not usable primarily because of the programmer-intensive scaling considerations required to make the transition from the floating-point arithmetic implicit in a “programmer-efficient” simulation to the fixed-point programs hitherto associated with a cost-effective real-time implementation involving a DSP micro.

The new generation of floating-point digital signal processors, such as the AT&T DSP32C, Motorola DSP56002, and Texas Instruments TMS320C30, all described in this issue, obviate the need for such scaling. This attribute, coupled with the recent developments in personal-computer, or PC, technology and graphics processors, potentially offers significant help in overcoming the major barriers to speedy design and cost-effective implementation of complex DSP systems.

A new approach, which combines the two phases of the development process, is now possible. A typical development system could involve an iconic graphical interface (implemented in PC software), a compiler, and a PC plug-in board containing a floating-point DSP microchip and memory system. The designer could “implement” the DSP system by providing a block-diagram, “iconic” representation or description. This description would then be automatically transformed into a real-time (or near-real-time) implementation: a program for the floating-point DSP processor.

The program generated would consist of a combination of high-level-language code—easily compilable into DSP-micro machine language—and calls to efficient DSP-micro library procedures such as FFTs and digital filters. This software could then be tested in real time (or near-real time) with the aid of a PC. The PC can provide diagnostic aids via amenities, such as bulk storage and fast graphics, not normally found on the target board. When the software is functioning in real time, the actual implementation of the DSP system involves simply transporting the software to a minimal-configuration board that employs the same DSP CPU as the plug-in board in the host PC.

Note that the concept of a block-diagram DSP compiler is not new; it existed to provide nonreal-time simulation on a mainframe in the mid-1960s. Similarly, DSP “spreadsheet” packages are available for PCs, again providing nonreal-time simulation of complex systems. PC plug-in boards exist for the current generation of fixed-point DSP micros, but, again, users generally achieve the transition from floating-point simulation on the PC to fixed-point DSP micro software through arduous, manual programming. The fourth article in this issue documents such an example. It describes the phenomenon of high performance-to-cost ratios achieved in implementing multiple-copy DSP systems via fixed-point DSP micros that have made the manual-programming effort viable despite its undesirable shortcomings.

What would be novel and potentially of great value is the ability to move, without programmer intervention, from an iconic system-specification to real-time software for a cost-effective target system implementation. As noted, the combination of floating-point capability and efficient compilers—the former not really viable without the former—could make this step possible for the new generation of DSP micros.

Other scenarios arise.

The Next PC is the first to incorporate a DSP micro. The on-board Motorola fixed-point DSP56001 is complemented by numerous “canned” procedures. These procedures enable graphics and signal processing tasks to be carried out at rates two orders-of-magnitude faster than possible with the on-board MC68882 floating-point coprocessor. The inclusion of such resources in an off-the-shelf PC provides another demonstration that, as noted earlier, the relatively large cost of developing highly efficient programs for fixed-point DSPs can be amortized over many system copies.

The first IEEE Micro DSP issue noted that “the fastest general-purpose micros already approach the performance of the slowest DSP micros” and that further narrowing of the gap via “integration of an array multiplier and on-chip barrel shifter . . . cannot be far
Introduction

off. Now we read suggestions that the Intel 80486 will process instructions about four times faster than an equal-clock-rate 80386. PCs incorporating this chip would thus surpass the performance of first-generation DSP micros and also provide the environment (high-resolution graphics) requisite for implementation of some nontrivial DSP systems. The 80486 article also noted that, in addition to a low-end central processor incorporating on-chip 80387 functionality, a high-end version will be complemented with a superior performance, off-chip numeric processor capable of vector, integer, and floating-point operations. If true, such a device might equal or surpass the latest generation of DSP chips in performance for some floating-point applications!

Thus, the cycle of improvements in functionality and performance for both general-purpose and DSP micros continues. Speeds will continue to increase, costs will continue to decline, and ease of use will continue to grow. But these are incremental changes.

Is it the end of the evolutionary process for the DSP micro? The answer is a certain and emphatic no! We can offer conjecture of what is to come. Architectures incorporating such structures as systolic arrays and neural networks, not yet in commercial use, will replace those now considered conventional. Silicon will be supplanted by gallium arsenide or similar technologies in the short term. In the long haul, optical and biological processors will take the stead of today's semiconductor-based chips. One can imagine other sweeping changes.

It will be interesting to look back to this day a decade or two from now. Surely, much of what we regard today as being quite sophisticated will appear to be somewhat primitive then.

References


Stephen A. Dyer, an associate professor in the Department of Electrical and Computer Engineering at Kansas State University, Manhattan, also serves as associate head of that department. His interests include digital signal processing, computer-based instrumentation, computer-aided analysis and design, and communication theory.

Dyer received the BS in physics, MS in electrical engineering, and PhD in engineering from Kansas State. He is a member of the IEEE Computer Society and the editorial board of IEEE Micro.

L. Robert Morris' biography, picture, and address appears on p. 85 in this issue.

Questions concerning this introduction can be addressed to either author. Write to Dyer at Kansas State University, Dept. of EECE, Manhattan, KS 66506.

Reader Interest Survey

Indicate your interest in this article by circling the appropriate number on the Reader Interest Card.

Low 150 Medium 151 High 152