Different strategies boost DSPs' abilities
Blalock, Garrick
Electronic Engineering Times; Feb 10, 1997; 940; ABI/INFORM Trade & Industry
pg. 68

Different strategies boost DSPs' abilities

By Garrick Blalock
DSP Engineer
Berkeley Design Technology Inc.
Fremont, Calif.

Spurred by the voracious number-crunching requirements of such popular DSP applications as digital wireless communication, the demand for high-performance DSP processors has never been greater. According to DSP market-research firm Forward Concepts, annual sales of programmable DSP processors will exceed $3 billion in 1997. DSP processor vendors have raced to meet this demand with new processors promising state-of-the-art performance. Not wanting to miss the boat, vendors of general-purpose processors and microcontrollers have added DSP capabilities to their products and advocate using a single chip for both DSP and general processing. System designers have never faced as many options for implementing high-performance DSP.

Today's processors use a variety of strategies to increase their DSP performance, including single-instruction, multiple-data (SIMD) instructions, multi-issue architectures, control and signal-processing architecture integration, and refinements on traditional DSP architectures.

SIMD instructions partition registers and ALUs so that multiple items of data are present in one register or memory location and can be processed in parallel by one instruction. For example, a SIMD processor might have a 64-bit register that can be partitioned into eight 8-bit data elements, four 16-bit data elements, two 32-bit data elements or one 64-bit data element. Typically, a SIMD processor performs the same operation, such as addition or multiplication, on multiple pairs of data elements using just one instruction. SIMD instructions are commonly used to add DSP capabilities to 32- or 64-bit RISC/CISC architectures, since these architectures often already contain the wide buses and registers needed. However, SIMD techniques also can be used on dedicated DSP processors–Texas Instruments' TMS320C8X, for example.

One highly publicized general-purpose processor to use SIMD instructions for DSP is the Intel P55C, the first Pentium processor to include the MMX multimedia extensions. One MMX instruction operating on two 64-bit registers partitioned into 16-bit data elements can multiply four pairs of multiplicands, add the 32-bit products in pairs and save the two 32-bit sums in one 64-bit register with throughput of one instruction cycle. These types of operations are very useful for calculating vector dot products and complex number multiplications.

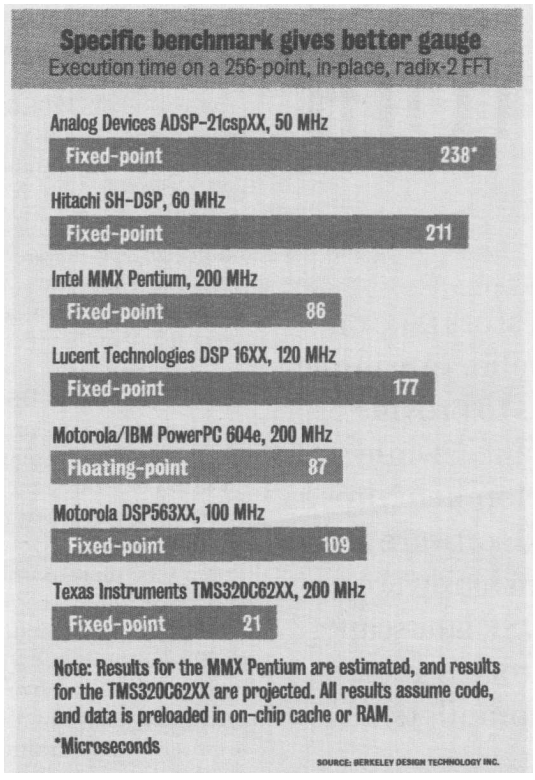
Sun Microelectronics has also developed SIMD multimedia extensions for its Ultrasparc architecture. Like the MMX extensions, Sun's Visual Instruction Set (VIS) allows 64-bit registers to be partitioned into 8-, 16- or 32-bit data elements that can be manipulated in parallel.

One of the attractions of SIMD instructions is the ability to select an appropriate data word length. If less precision is needed, programmers can use 16-bit data elements and operate on four elements in parallel, for example. Alternatively, if more precision is needed, the programmer can choose 32-bit data elements at the price of performing fewer operations in parallel.

If SIMD instructions use fixed-point arithmetic, parallel processing can be accomplished by simply partitioning an existing data path. For example, if a processor contains a 32-by-32 64-bit multiplier, the multiplier could be dissected into four 8-by-8 16-bit multipliers that operate in parallel. Unfortunately, realizing SIMD instructions' performance potential often requires restructuring algorithms so that elements can be processed simultaneously. This requirement can make optimizing code for SIMD instructions very difficult and some applications may see little improvement over non-SIMD instructions. For example, applications with sequential data dependencies, such as adaptive filtering, may be limited in how many calculations can be done in parallel. In many DSP applications, however, SIMD instructions are very effective.

Multi-issue architectures can execute several different instructions in parallel. This approach is most commonly used in high-performance general-purpose processors not specifically intended for DSP. For example, the Motorola/IBM PowerPC 604e features a four-issue dynamic superscalar architecture that issues up to four instructions in parallel. The processor examines the serial instruction stream and dynamically determines which instructions can be executed in parallel. Because the architecture restricts which instructions can execute in parallel, typically fewer than four instructions are actually issued in a clock cycle. Additionally, most 604e instructions do less work than instructions on dedicated DSP processors. Still, the 604e's parallelism, coupled with clock speeds of over 200 MHz, yields very strong DSP performance.

SIMD instructions and multi-issue architectures are not mutually exclusive. For example, the MMX Pentium can issue up to two SIMD instructions in parallel. With both SIMD instructions and multi-issue architectures, taking full advantage of the parallel-execution abilities of processors typically requires careful programming and advanced tools that are often unavailable. This concern is especially relevant to general-purpose processors, which often lack cycle-accurate simulators and other tools needed to develop efficient DSP code.



System designers never had so many options for implementing the high-performance digital signal processor.

Texas Instruments has announced the TMS320C62XX family of DSP processors. The C62XX family comprises the first dedicated DSP processors to take advantage of a VLIW-like architecture, and thus promises excellent performance. Unlike the dynamic instruction grouping used by superscalar general-purpose processors, grouping on the C62XX is done by the programmer or the compiler. The C62XX fetches eight instructions at a time and, based on a bit in each instruction, executes between one and eight of the instructions in parallel. This type of static instruction scheduling greatly simplifies predicting code-execution time, but shifts the burden of optimizing instruction grouping to the programmer and code-development tools. In the best-case scenario, the C62XX can perform four arithmetic operations, two multiplications and two address calculations in one instruction cycle.

Vendors of processors with traditional DSP architectures have not stood still. For example, Motorola has recently refined its 24-bit fixed-point DSP5600X family of processors to develop the new DSP563XX. The most notable change is the addition of a deeper pipeline that increases the instruction-execution throughput to about twice that of the DSP5600X. Motorola also added special instructions, such as bit-manipulation instructions and conditional execution instructions, that speed the processing of many DSP-intensive applications. The DSP563XX also supports a coprocessor interface that has allowed Motorola to add hardware optimized for key DSP algorithms, such as Viterbi decoding, which is heavily used in wireless communications. Like many DSPs, the DSP563XX features peripherals designed to increase the processor's efficiency in DSP applications. For example, to reduce the number of instructions needed for high-bandwidth I/O, a six-channel direct-memory-access (DMA) controller allows transfers of data between any combination of internal or external memory locations without the CPU's intervention.

Other vendors also are improving their DSP processors. Analog Devices has introduced the ADSP-21cspXX, a new family of fixed-point 16-bit processors based on the ADSP-21XX family. The ADSP-21cspXX extends the ADSP-21XX's pipeline from two to three stages, allowing the processor to run at faster clock speeds. To simplify programming, the ADSP-21cspXX instruction set is more orthogonal than the ADSP-21XX instruction set. Furthermore, performing algorithms on two unrelated data streams has been facilitated by the extensive use of context-switched shadow registers. These shadow registers help reduce the code complexity and Mips-usage associated with switching between interrupts, core processing tasks and multiple data sets.

Lucent Technologies has refined its 16-bit fixed-point DSP16XX-family architecture to run at 120 Mips, the fastest instruction-execution rate of any programmable DSP currently in mass production. Lucent has also pushed the 16XX-family operating voltage down to a 1.8-V nominal on some family members. In contrast to many general-purpose processors, which consume far too much power for portable applications like cellular telephones, members of the DSP16xx family can consume as little as 40 mW at 60 Mips.

Traditional dedicated DSPs offer many advantages. Because DSP-processor instruction sets and peripherals are targeted specifically at signal processing, they typically offer very competitive price/performance ratios, especially in applications that can use fixed-point arithmetic. Unlike general-purpose processors, DSPs typically have signal-

►CONTINUED ON PAGE 91

Different strategies can increase DSP performance

►CONTINUED FROM PAGE 68

processing-oriented software-development tools, such as cycle-accurate software simulators, that ease the development of critical real-time applications. Furthermore, because of their specialized instruction sets, DSP processors can typically achieve good DSP performance at lower clock speeds than general-purpose processors. These lower clock speeds help DSPs achieve power consumption lower than most general-purpose processors with equivalent performance. Unfortunately, most DSPs lack strong microcontroller features, such as instructions well-suited for C compilers and flexible I/O. So, designers using dedicated DSPs in an embedded system must often include a system microcontroller as well. For example, in a cellular telephone, the DSP is likely to handle speech coding and baseband functions while a microcontroller handles the user/machine interface and other control processing. The incentive to merge these two processors is great: lower parts count, smaller board size, lower power consumption, fewer programming models to learn, etc.

Dual controls

Integrating a dedicated DSP data path with a microcontroller architecture creates processors well-suited for both signal processing and system control. For example, the Hitachi SH-DSP adds a complete fixed-point DSP data path and instruction set to Hitachi's successful SH-2 microcontroller architecture. This unusual hybrid approach allows programmers to add DSP functionality while protecting their investment in SH-2 code, which runs unaltered on the SH-DSP. Programmers can access the SH-DSP's DSP data path by simply adding DSP instructions to a SH-2 program. The SH-DSP fetches instructions sequentially and issues DSP and microcontroller instructions to the appropriate execution unit.

Because hybrid architectures like the SH-DSP are partitioned into a dedicated DSP data path and a microcontroller data path, both system control and signal processing can be implemented efficiently. However, the presence of two separate instruction sets can complicate programming, and interactions between the two data paths can sometimes hinder performance.

An interesting counterpoint to microcontrollers with DSP enhancements such as Hitachi's SH-DSP is Motorola's recently announced DSP568XX family. The DSP568XX provides architectural and I/O features to address the needs of microcontroller applications.

The execution time for some of the processors discussed above can be found in Berkeley Design Technology's (BDT) FFT benchmark, one of 11 benchmarks in the BDT Benchmark suite research reports, *DSP on General-Purpose Processors* and *Buyer's Guide to DSP Processors*. The reports also contain results for a variety of DSP functions, cost/performance and memory usage, as well as more detailed analysis of each processor.

The best approach to high-performance signal processing depends strongly on the application. In PC applications, for ex-

►CONTINUED ON PAGE 92

New schemes build better DSPs

▶CONTINUED FROM PAGE 91

ample, which already contain high-performance host processors, implementing DSP on the host processor–perhaps with SIMD extensions or by using the processor's multi-issue capability–may be an attractive way to add DSP capabilities with little additional hardware cost. On the other hand, in portable consumer applications such as pagers, the need to minimize system cost, parts count and power consumption is likely to lead engineers to lower priced traditional DSPs or hybrid architectures.

Designers can look forward to several positive trends in processors for DSP. Manufacturers will continue to add specialized peripherals, to increase performance and allow further integration of system components. Both SIMD and multi-issue processors will evolve to allow further parallelism in instruction execution. To allow designers to run larger real-time applications, processor vendors are increasing the sizes of on-chip memory and caches or adding faster external-memory interfaces. Processors intended for portable applications will consume less power with the help of better power-management features and lower operating voltages.

DSP-device designers will continue to improve their architectures' suitability for compiled code, reducing the amount of code that must be tediously written in assembly language. Likewise, processor vendors and third-party companies will improve their software design tools, although tool improvements often lag the increasing complexity of processors. Finally, as semiconductor-fabrication technologies progress, processor operating speeds will undoubtedly increase.

Landscape shifting for DSP design

▶CONTINUED FROM PAGE 86

internal dedicated hardware. The DSP monitors loop conditions and terminations to decide, in parallel with all other operations, whether to increment the program counter or branch without cycle-time penalty to the top of the loop. A RISC processor, on the other hand, has to do a test-and-branch at the end of every loop, costing at least an additional cycle every loop and every pass.

DSP looping hardware typically supports at least four levels of nested loops; RISC processors have yet to evolve to those basic signal-processing needs.

Language levels

RISC processors have traditionally been programmed in high-level languages such as C; the performance-driven DSPs traditionally were handled in assembly language. Though still generally true for fixed-point DSPs, which now can handle algebraic assembly language to ameliorate the ease-of-use issue, floating-point DSPs are now more efficiently programmed in high-level languages.

In each of these areas, RISC and DSP architectures remain wide apart. As prices and functionality become similar in the future, designers will be forced to consider whether their products will be used in the real-time environments where DSPs will continue to excel.

As DSP performance reaches 1,500 Mflops in the years ahead, the performance rift between RISC and DSP will widen. DSPs are more efficient for real-time signal processing and will continue to be so.