PORTABLE DESIGN™

HIGH-PERFORMANCE DSPs FOR PORTABLE APPLICATIONS

Architectural innovations ease the tradeoffs, complicate choices

John Donovan — Editor-in-Chief

Until fairly recently, if your application called for audio, video, or baseband signal processing, you simply looked around to determine which discrete fixed-point DSP best met your needs. Now you have a choice of fixed- and floating-point programmable DSPs with a wide range of architectures, both fixed and programmable; general-purpose processors (GPPs) with DSP capabilities; embedded cores with DSP "extensions"; fixed-function DSPs for specific applications (ASSP or ASIC); FPGAs running (massively parallel) DSP functions; and different permutations of almost all of these.

What's out there

The DSP market is huge, though the breakdown may not be what you'd expect (see Figure 1). According to Forward Concepts (Phoenix, AZ), worldwide shipments of all DSP-centric chips totaled an estimated $22 billion in 2005. Of this number, traditional general-purpose programmable DSPs—a market dominated by Texas Instruments, Freescale, Analog Devices, and Agere—amounted only to 34.8% of the market. The largest segment (48.2%) was nonprogrammable or fixed-function DSPs, which Forward Concepts calls "FASICs," or function- and algorithm-specific ICs that aren't usually touted as DSPs, even though they are. Leading "FASIC houses" include Conexant (modem chips), Broadcom (communications), Yamaha (sound), Philips (cell phone chips), STMicroelectronics (MPEG), Cirrus Logic (DVD and A/V receiver chips), and Zoran (DVD and DSC chips).

Adding to the confusion is the decision by leading DSP houses to reclassify some of their DSP offerings as something else. Agere, the largest maker of DSP-based HDD controllers, now calls its higher-performance versions SoCs. Texas Instruments no longer markets its DSC or DSL chips as DSPs but as ASSPs. Then there are "media processors," which are DSPs optimized for audio and/or video with

Reprinted with revisions to format, from the May 2006 edition of **PORTABLE DESIGN**

hard-wired accelerators, such as Philips' TriMedia and Texas Instruments' DaVinci.

The remainder of the DSP market consists of DSP ASICs (11.8%), reconfigurable DSPs (3.0%), and MPUs and other (2.2%). The latter category includes CPUs with DSP extensions, such as Intel's Pentium with MMX and SSE and Freescale's PowerPC (G4) with AltiVec.

The largest market for DSPs is wireless (39.9%), followed by consumer (27.0%), computing (15.6%), wireline (9.7%), instrumentation (3.7%), industrial (2.8%), and military/aerospace (1.4%). Programmable DSPs figure most heavily in communications markets, while consumer applications (MP3, AC-3, MPEG) see more FASICs. The biggest single market for DSPs, not surprisingly, is cell phones.

DSP architectures

In contrast to a GPP, a DSP is optimized for computationally intensive signal processing. DSP architectures are dictated by the algorithms that they process in hardware. Most signal processing involves simple, repetitive mathematical manipulations on high-speed data streams. GPPs have traditionally accomplished multiplication by a series of shift and add operations, each of which takes one or more clock cycles. DSPs—typically used for FIR filters, IIR filtering, convolution, and Fourier transforms—all have at least one single-cycle multiplier or multiply-accumulate (MAC) unit, often accompanied by an arithmetic-logic unit (ALU) and shifter. Higher-performance DSPs usually have several independent execution units that can operate in parallel. This parallelism enables DSPs to carry out multiple operations per instruction, whereas GPPs can generally implement only one operation per instruction.

To carry out all these calculations, DSPs need to make several memory accesses per instruction cycle, fetching new instructions and data while retrieving operands and storing the results of a previous operation. To do so, DSPs have evolved some specialized memory architectures. Lower-end GPPs—MCUs in particular—use a Von Neumann architecture, with only one data bus to handle both code and data; this renders them incapable of handling all but the least demanding signal processing. DSPs use a Harvard architecture, with separate data and address buses, as well as separate data and instruction caches (or at least multiported memories). This enables the MAC execution unit to fetch an instruction word and two data words every instruction cycle. The amount of on-chip memory varies with the application. Embedded applications generally have small caches, small external data buses, and address buses of 16 bits or less. Floating-point chips have large, fast external data buses and DMA capability.

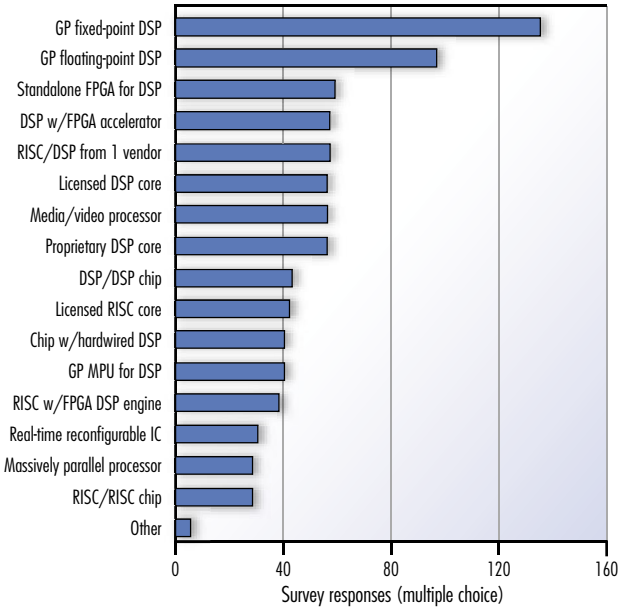

Figure 1. Chip types employed for DSPs. (Figure courtesy of Forward Concepts)

DSPs generally have dedicated address generation units that juggle the various registers that enable operand access while instructions are executing. These address units support a variety of addressing modes optimized for specialized processing chores (circular buffers, FFT, and single-cycle for-next loops).

With all this specialized hardware, DSP instruction sets tend to be complex, making them difficult to program in high-level languages such as C or C++, whose compilers kick out code that doesn't take full advantage of DSP at the hardware level. Despite considerable attention to DSP compilers, designers can still expect to put in some time hand-tuning DSP assembly code for time-critical functions.

Fixed vs. floating point

Most DSPs use fixed-point arithmetic, which makes for a low-power, low-cost silicon implementation. Designers must carefully determine the dynamic range and precision required by their application before settling on a fixed-point DSP. During operation, they must also take care to scale signals to maintain numeric precision and dynamic range. Fixed-point DSPs typically use 16-bit data words, though some go to 24 and even 32 bits for high-quality audio processing.

Floating-point processors offer a wider dynamic range because they can represent a much wider range of numbers. They're also much easier to program, since designers needn't worry about the constraints imposed by fixed-point processors. The downside is that they're more complicated and therefore use more silicon real estate. That translates to higher cost and power consumption, which has limited their use in portable designs. Floating-point DSPs typically use 32-bit data and instruction words.

VLIW vs. superscalar

DSPs really went "high performance" with the advent of multi-issue architectures in the mid-1990s. These DSPs issue and execute instructions in parallel groups rather than one at a time. They use very simple instructions and operate at much higher clock speeds than earlier DSPs.

There are two multi-issue architectures: VLIW and superscalar. Very long instruction word (VLIW) DSPs issue a number of instructions together in one long instruction word, each section of which is then broken out and worked on in parallel by dedicated hardware execution units. VLIW DSPs typically issue four to eight instructions per clock cycle.

In a VLIW architecture, the program determines which instructions are to be

executed in parallel. Superscalar processors, in contrast, dynamically schedule parallel operations based on data dependencies and resource contention, making it difficult to guarantee in advance that real-time processing needs will be met in every case. Superscalar DSPs are easier to program than their VLIW counterparts, but the lack of timing predictability when they're placed in the data path has limited their acceptance to date.

In addition to enabling more parallelism in execution, the use of long instruction words also makes possible larger, more uniform register sets and simpler instructions. The downside is that VLIW processors then require many more instructions than do simpler DSPs, which increases code size and memory usage—the price you pay for fast parallel processing. Memory bandwidth, bus widths, and clock rates all rise, which degrades the power profile. Still, progress is being made on the power front, with a few vendors targeting VLIW DSPs at the handset market.

SIMD

There are different ways to do parallel processing. You can perform entirely different instructions in parallel (all in the same clock cycle) using multiple MACs and ALUs. Or you can perform the same calculations on different operands in parallel. The latter is called "single-instruction, multiple-data" or SIMD for short. SIMD works extremely well in the sort of repetitive calculations used in multimedia signal processing.

SIMD isn't an architecture but an architectural technique that can be used in VLIW and superscalar DSPs, either by running parallel operations in matching pairs of MACs, ALUs, and registers, or by splitting execution units into smaller increments, breaking a 32-bit operand down, for example, into two 16-bit or four 8-bit ones. Intel took a SIMD approach to add DSP capabilities to the Pentium with its MMX and SSE extensions. It's straightforward to break down 64-bit registers and ALUs into four times as many 16-bit structures and run them all in parallel. The increase in speed for certain operations is dramatic.

CPUs as DSPs

Intel, AMD, and Freescale have all added DSP capabilities—primarily using SIMD—

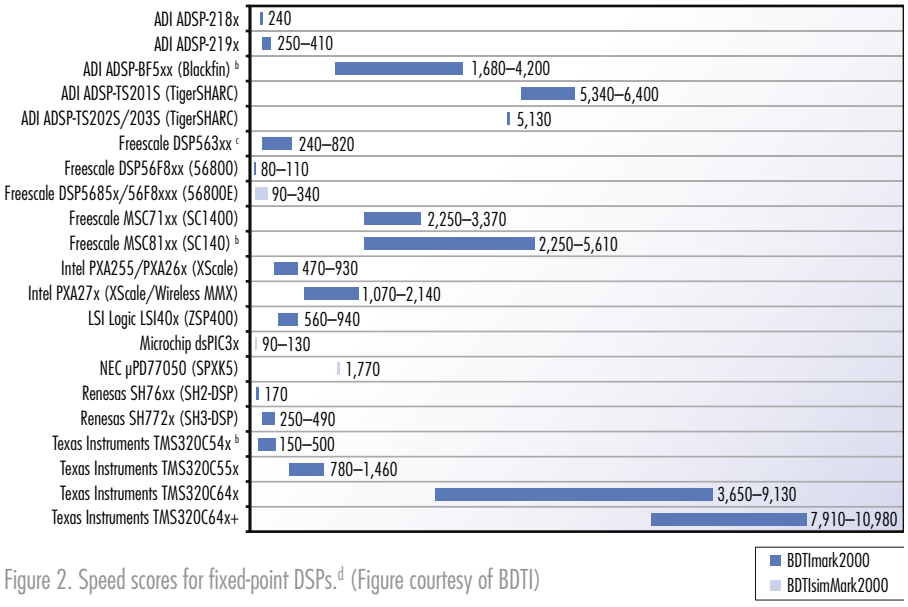

Figure 2. Speed scores for fixed-point DSPs.[d] (Figure courtesy of BDTI)

Table 1 – Speed scores for fixed-point packaged processors

Processor family	Clock rate (Min.-Max.; MHz)	BDTImark2000, *BDTIsimMark2000* (Min.-Max.)
ADI ADSP-218x	80	240
ADI ADSP-219x	100 to 160	250 to 410
ADI ADSP-BF5xx (Blackfin)[b]	300 to 750	1,680 to 4,200
ADI ADSP-TS201S (TigerSHARC)	500 to 600	5,340 to 6,400
ADI ADSP-TS202S/203S (TigerSHARC)	500	5,130
Freescale DSP563xx[c]	80 to 275	240 to 820
Freescale DSP56F8xx (56800)	60 to 80	80 to 110
Freescale DSP5685x/56F8xxx (56800E)	32 to 120	*90 to 340*
Freescale MSC71xx (SC1400)	200 to 300	2,250 to 3,370
Freescale MSC81xx (SC140)[b]	200 to 500	2,250 to 5,610
Intel PXA255/PXA26x (XScale)	200 to 400	470 to 930
Intel PXA27x (XScale/Wireless MMX)	312 to 624	1,070 to 2,140
LSI Logic LSI40x (ZSP400)	120 to 200	560 to 940
Microchip dsPIC3x	30 to 40	*90 to 130*
NEC µPD77050 (SPXK5)	250	*1,770*
Renesas SH76xx (SH2-DSP)	62.5	170
Renesas SH772x (SH3-DSP)	100 to 200	250 to 490
Texas Instruments TMS320C54x[b]	50 to 160	150 to 500
Texas Instruments TMS320C55x	160 to 300	780 to 1,460
Texas Instruments TMS320C64x	400 to 1,000	3,650 to 9,130
Texas Instruments TMS320C64x+	720 to 1,000	7,910 to 10,980

to their high-end CPUs. This approach works well, since CPUs run at high speeds and have wide signal paths and plenty of on-chip resources. High-end CPUs can sometimes outperform high-end DSPs in DSP functions.

The downsides are power and efficiency, both areas where dedicated DSPs come out ahead. High-end CPUs run much faster than even the fastest DSP, which makes them quite power hungry. Sure, you can scale back the frequency, but you can do that with DSPs, too, so where's the relative advantage? You're still robbing cycles from some other system function. Also, CPUs of this type are commonly superscalar, with less predictable timing than DSPs, making them a less reliable choice for real-time applications. DSPs still offer the best combination of performance and power efficiency.

MCU/DSP combos

While MCUs themselves are poor at signal processing, there's nothing that says you can't add DSP capabilities to your MCU.

Analog Devices' Blackfin combines a RISC MCU with a dual-MAC DSP engine on a single chip. Blackfin uses variable-length instruction words to minimize code size, a shortcoming of VLIW DSPs. It also includes memory management, critical to MCUs but not common in DSPs, which are stripped down for speed. The DMA channels are independent of the processor core, allowing it to operate deterministically and allowing the DSP data flow in any embedded application to be completely independent of the control processes.

Philips makes a combo processor, the PMX0102, which has an ARM7 core with a separate DSP engine for audio codec operations.

In contrast to dedicated DSPs, ARM's, ADI's, and Philips' chips can be programmed in C, easing system design.

The ARM7/9/11 processor cores are built around a general-purpose RISC architecture. The ARM9E adds DSP functionality that is adequate to handle moderately demanding DSP chores, in some cases (such as MP3 decoding) eliminating the need for a separate DSP. However, the inability to support parallel data moves—a hallmark of dedicated DSPs—means the DSP code must operate sequentially, increasing code size and thus memory requirements.

In addition to ARM, ARC, CEVA, LSI Logic, MIPS, StarCore, and Tensilica all make fixed-point, synthesizable RISC cores with notable DSP capabilities.

FPGAs as DSPs

If DSPs gain speed from running functions in parallel, what would happen if you ran them in massively parallel FPGAs? All the FPGA vendors encourage that sort of thinking and offer some DSP (mostly MAC) functions that can be incorporated in an FPGA fabric, enabling extremely fast operation. You can even add on-chip memory and drop in an MCU core.

You won't see an entire DSP core installed in an FPGA, for a number of reasons. DSPs tend to run much faster than FPGAs, and they need more granular clocking and power management than an FPGA can provide. On the other hand, FPGAs have a good working relationship with DSPs, for which they can act as coprocessors, offloading much of the number crunching as well as scaling the memory system up when the DSPs run out of horsepower.

FPGAs find their place in base station designs more than in portable applications. A single FPGA may be able to handle 10 to 20 OFDM channels, replacing an equal number of DSPs. However, in military and emergency services handsets with larger batteries and bandwidths than those found in cell phones, programmable, low-power FPGAs are increasingly replacing high-performance DSPs.

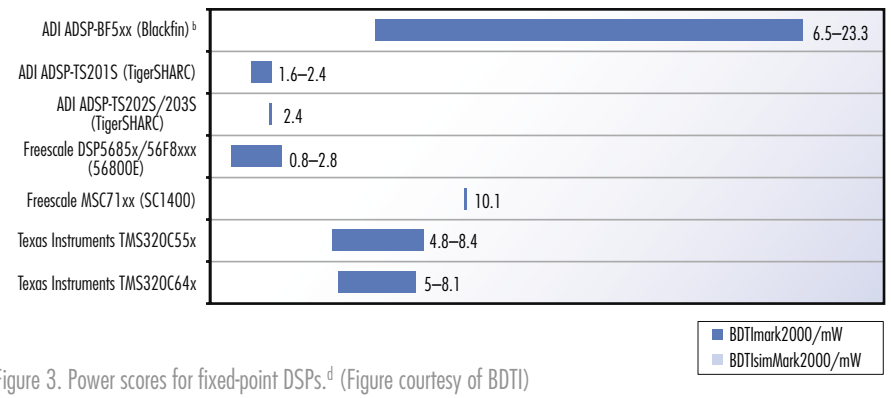

Figure 3. Power scores for fixed-point DSPs.[d] (Figure courtesy of BDTI)

Table 2 – Speed/mW ratios for fixed-point packaged processors

Processor family	Clock rate (Min.-Max.; MHz)	BDTImark2000, *BDTIsimMark2000* (Min.-Max.)	Power (Min.-Max.; mW)	BDTImark2000/mW, *BDTIsimMark2000/mW* (Min.-Max.)
ADI ADSP-BF5xx (Blackfin)[b]	300 to 750	1,680 to 4,200	24 to 644	6.5 to 23.3
ADI ADSP-TS201S (TigerSHARC)	500 to 600	5,340 to 6,400	2,180 to 3,930	1.6 to 2.4
ADI ADSP-TS202S/ 203S (TigerSHARC)	500	5,130	2,180	2.4
Freescale DSP5685x/56F8xxx (56800E)	32 to 120	*90 to 340*	120 to 213	*0.8 to 2.8*
Freescale MSC71xx (SC1400)	200 to 300	2,250 to 3,370	222 to 333	10.1
Texas Instruments TMS320C55x	160 to 300	780 to 1,460	62 to 204	4.8 to 8.4
Texas Instruments TMS320C64x	400 to 1,000	3,650 to 9,130	654 to 1,303	5 to 8.1

So just what does "high performance" mean?

The architectural complexity of DSPs, combined with the inevitable propensity of vendors to stress benchmarks that display their products in the best light, makes direct comparisons difficult. The amount of time required to complete a given task and the amount of power consumed in the process are the main things you need to determine.

Every vendor specifies how many millions of instructions per second (MIPS) its processor can perform. But the same instruction may accomplish varying degrees of work in different architectures. In the same vein, comparing how many millions of operations per second (MOPS) two processors can perform is complicated by the fact that the number of operations needed to accomplish a task varies between processors. A final favorite spec is millions of multiply-accumulate operations per second (MMACS), which may get your attention if you're designing filters, for example. But this spec fails to take into account the other things that a DSP may or may not be able to do in parallel that can greatly affect its performance. Comparisons in terms of MIPS, MOPS, and MMACS should only be made between DSPs with similar architectures, and even then they're of limited use.

Interpreting DSP power specifications is always a tricky undertaking. Power consumption varies widely with different instructions, data values, and memory requirements, not to mention different clock speeds. The ability of a processor to selectively power down unused portions of the chip can result in very different power profiles when using the same chip in different applications—or different chips in your application.

What really matters is how well a DSP will work in a given application. The Standard Performance Evaluation Corp., with its SPECmarks, evaluates CPUs but not DSPs, which make poor compiler targets for the C code in which their benchmarks are written. Realistic DSP benchmarks would need to have large portions written in assembly code, which would immediately make them product specific.

Berkeley Design Technology Inc. (BDTI) publishes DSP benchmarks using what it calls "algorithm kernel benchmarking." These benchmarks include a wide range of DSP functions, including FFTs, filters (FIR, IIR), Viterbi decoder, bit unpack, and a variety of vector processes. They specify the clock rates, memory libraries and other parameters to enable "apples-to-apples" comparisons between different processor families.

Figure 2 and Table 1 show the speed scores for a range of popular fixed-point DSPs. Figure 3 and Table 2 show the power scores for a number of these processors. The differences reflect the design tradeoffs dictated by their intended uses. For example, the TMS320C64x+ is the fastest of the lot, but it's also power hungry, drawing as much as 1.3 A when run flat out. That's because its designers pulled out all the stops to make it as fast as possible (though it can scale back the power considerably when called upon). The Blackfin, on the other hand, while fast, is much more power efficient—as befits a product targeting portable designs.

Which DSP?

Because of their architectural complexity, selecting a DSP is a lot more complicated than choosing an MCU. For starters, don't look for more precision than your application requires; more precision equates to more computations, higher bandwidth, greater memory requirements, more power, and higher cost. If you're looking to extend an existing design, check whether your MCU vendor offers a chip or core with DSP extension, then see if it can handle your requirements. For new designs, consider an MCU/DSP combo chip. If you still need a dedicated DSP, decide what type of architecture makes the most sense, then look for benchmarks comparing chips in that category.

In all cases, consider how much flexibility you need not only in this design but the one that follows. Choosing a programmable DSP can give you the flexibility to handle evolving communications protocols or codec formats. On the other hand, if you only need to handle specific functions, there is a wide range of fixed-function DSPs (by whatever name) that can fill the bill.

Choosing a DSP may be difficult, but that's mainly because there are so many choices. The upside is that there are a lot of ways to handle your signal processing tasks, both cheaply and well. **PD**

References

1. Will Strauss, *DSP Strategies: Embedded Chip Trend Continues*, Tempe, AZ: Forward Concepts, February 2006.
2. Steven W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*, San Diego, CA: California Technical Publishing, 2001.
3. Jennifer Eyre, Jeff Beier, *The Evolution of DSP Processors*, Berkeley, CA: Berkeley Design Technology Inc., 2000.
4. *Evaluating DSP Processor Performance*, Berkeley, CA: Berkeley Design Technology Inc., 2002.

Notes

[a] The BDTImark2000 and BDTIsimMark2000 provide a summary measure of signal processing speed. For more info and scores, see www.BDTI.com/benchmarks.html. Scores ©2006 BDTI.

[b] For one core.

[c] Benchmarked with 24-bit fixed-point data; all other processors benchmarked with 16-bit fixed-point data.

[d] BDTIsimMark2000 scores may be based on projected clock speeds. For information, see www.BDTI.com/benchmarks.html.

Blackfin is action packed

▶ V300 Portable Media Player

Open Source
Car Telematics
IPTV
Mobile TV
Telepresence
Biometrics
Driver Assistance
▶ **Streaming Media**
High Definition
Effects Processing
Triple Play
Image Processing
VoIP
Embedded Security
GSM/EDGE
Baseband Processing
Digital Radio
Global Positioning
Packet Processing
GCC/Linux
Cryptography
Rights Management

Low power, riveting performance

The V300 Portable Media Player needed a processor that could enable 8 hours of MPEG-4 or MJPEG video playback and 16 hours of MP3 music, and run its operating system and application set. After auditioning a long list of prospects, Blackfin® got the role. Why? Blackfin easily met the V300's performance goals while powering the system from a 600 mAh battery. Then Blackfin nailed the part with $5 to 1200 GMACS scalable performance for content and applications.

Get started using Blackfin now:
analog.com/blackfin-is-everywhere

ANALOG DEVICES