THE ART OF PROCESSOR BENCHMARKING: WHAT MAKES A GOOD BENCHMARK, AND WHY YOU SHOULD CARE

A WHITE PAPER

BY

BERKELEY DESIGN TECHNOLOGY, INC.

WWW.BDTI.COM

2006

Good benchmarks are essential for evaluating and comparing processors, for making informed business decisions, and for developing credible marketing programs. But not all benchmarks are created equal and using bad benchmarks can result in poor decisions, ineffective marketing, and failed products.

With benchmarks available from many sources, it can be difficult to determine whether benchmark results are reliable or whether they are comparable to one another. BDTI has been developing and implementing digital signal processing benchmarks for over a decade, and in this white paper we'll explain the key factors that determine a benchmark's quality and relevance. We'll explore common approaches to processor benchmarking for signal processing-based applications, and explain their trade-offs and potential pitfalls.

Understanding what makes a good benchmark—and perhaps as importantly, what makes a bad one—will enable you to know when to trust benchmark results, when to ask questions, and how to use benchmarks wisely.

Who Cares About Benchmarks?

Nearly everyone who works with processors uses benchmark results in one way or another. Executives use them to help make critical business and technical decisions. Engineers and managers use them to evaluate in-house processors or to help choose a processor for a new product. Marketers use benchmark results for competitive analysis and to add credibility to their marketing programs— which can be particularly effective if the benchmarks are from a trusted, independent source.

Good benchmarks, used properly, are an invaluable tool for all of these purposes. Unfortunately, there are many ways to create bad benchmarks. Relying on bad benchmarks can result in poor decisions, ineffective marketing, and in some cases, products that don't work.

We at BDTI have spent over a decade developing and implementing processor benchmarks and analyzing the results, so we're keenly aware of the potential pitfalls. If you're going to be using benchmark data to help make critical decisions or to sell your product to customers, then you need to understand the key factors that determine a benchmark's credibility, relevance, and applicability.

Benchmark Complexity Trade-Offs

There are many approaches to benchmarking a processor, each with different benefits and drawbacks. One of the key factors that determines a benchmark's usefulness and relevance is its complexity. If a benchmark is too simple, it won't do a good job of representing real-world performance. Too complex, and it won't be practical to implement—which means that few processors will have results available on that benchmark. In addition, the more complex the benchmark, the less widely applicable the results. If the benchmark is a simple algorithm kernel like an FIR filter, for example, the results will be useful for helping to predict performance across a wide variety of DSP-oriented applications. If the benchmark is an H.264 video codec, it will only be useful for a small class of applications—but for those applications, it will be far more accurate than algorithm kernel benchmarks.

The key advantage of more complex benchmarks is their increased accuracy for the applications they represent—at the cost of considerations such as ease of implementation and widespread applicability. Benchmark designers must weigh this benefit against the penalties and choose a complexity level that will provide the most meaningful results without becoming completely impractical.

For the purposes of discussing benchmarking for signal processing applications, we have defined four levels of benchmark complexity, as shown in Figure 1.

The simplest type of benchmark is based on basic operations, such as MACs or adds. These metrics are easy to measure, but are much too simple to provide useful information about how a processor will perform in a real application. For example, although many signal processing applications make heavy use of MACs, many others do not. And even the most MAC-intensive application performs many other types of processing. Furthermore, a processor may not be able to reach its peak MAC rate because of limiting factors such as memory bandwidth, pipeline latencies, or algorithm feedback requirements. None of

these performance issues are captured by operation-level benchmarks, which makes them nearly useless for evaluating a processor's suitability for an application.

At the other end of the complexity spectrum are full-application benchmarks, such as the complete functionality found in wireless handsets and portable audio players. Like operation-level benchmarks, full-application benchmarks have several significant drawbacks. Because of their complexity, they are unlikely to be implemented on many different processors (particularly if benchmark implementers will be optimizing key portions of the implementation, as is often done in real-world applications). Thus, there are likely to be few results available to users—which can make it impossible to do meaningful competitive analyses. In addition, results for full-application benchmarks have very narrow utility. That is, they reveal little about a processor's performance in applications other than the one used for benchmarking—even if those applications are seemingly similar.

Between these extremes lie algorithm kernel benchmarks and application task benchmarks. Collectively, these two types of benchmarks are, in BDTI's view, a "sweet spot" for benchmarking; all of BDTI's benchmark suites fall within these two categories. The BDTI Benchmark Suites and their complexity classifications and trade-offs are listed in Table 1 on page 3.

Benchmarks Should Be Tied to Applications

Whether the benchmark is an algorithm kernel or an application task, the benchmark workload must be closely tied to the application it represents. Benchmarks designed to measure processors' performance in embedded signal processing applications should be different from benchmarks designed to measure processor performance on other types of embedded applications—because the applications themselves are different.

There are two distinct areas in which benchmarks must be representative of the applications for which they are used. The first is most obvious—regardless of the benchmark's

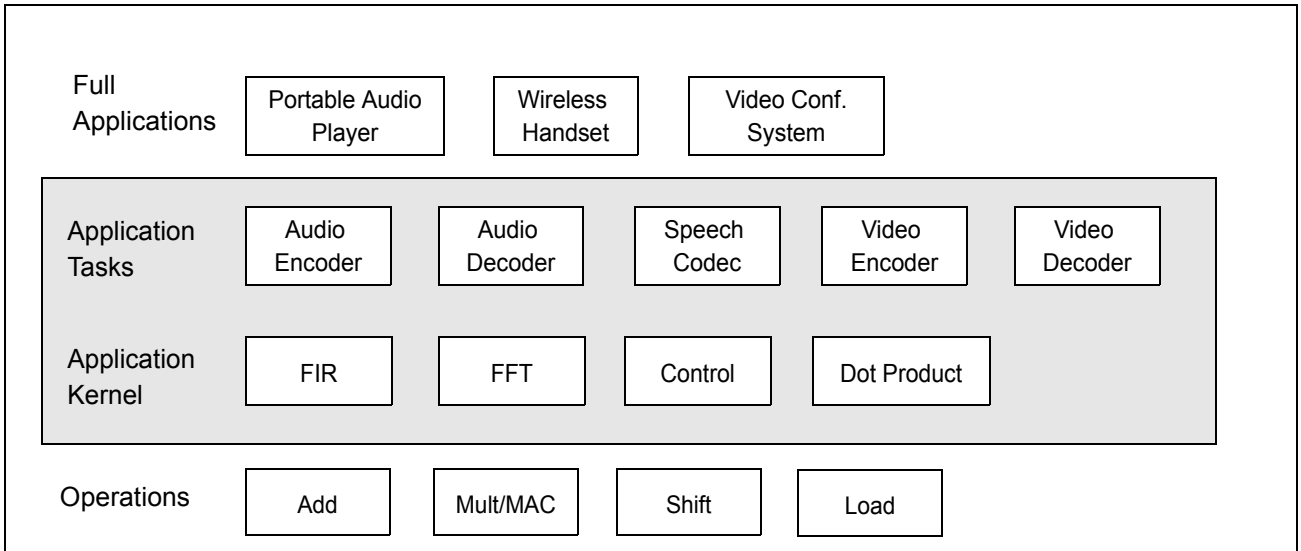


Figure 1: Types of signal processing benchmarks, from simple to complex. Shaded area indicates the types of benchmarks included in BDTI's benchmark suites.

BDTI Kernel Benchmark Suites™ Measure processor core performance on relatively simple algorithm kernels.		
BDTI DSP Kernel Benchmarks™ BDTImark2000™	This suite includes 12 key DSP algorithm kernels, such as FIR filters, a Viterbi decoder, and an FFT. The BDTI DSP Kernel Benchmarks are used to generate BDTImark2000 scores (The BDTImark2000 is BDTI's composite DSP speed metric.)	Advantages: These benchmarks are widely applicable, relatively simple to implement and optimize, and quick to simulate. Results are available for many processors. Disadvantages: These benchmarks do not consider system-level performance and are less accurate than application task benchmarks.
BDTI Video Kernel Benchmarks™	This suite includes 7 building-block functions critical to many video applications, such as motion compensation and image resizing.	
BDTI Application Task Benchmark Suites™ Measure system-level performance on simplified application tasks.		
BDTI Communications Benchmark (OFDM)™	The BDTI Communications Benchmark (OFDM)™ is an application-oriented benchmark based on an orthogonal frequency division multiplexing (OFDM) receiver. It is representative of the processing increasingly found in communications equipment for applications such as DSL, cable modems, and wireless systems.	Advantages: These benchmarks are more accurate for applications specified, and they consider system-level performance. They are simpler to implement than full standard-based solutions. Disadvantages: These benchmarks are applicable to fewer applications, are available for fewer processors, and are more complex to implement than the BDTI Kernel Benchmarks. They may also be slow to simulate.
BDTI Video Encoder and Decoder Benchmarks™	These benchmarks are representative of the video encoding and decoding workloads found in a wide variety of mobile, home, and surveillance applications. They are loosely based on the H.264 standard, but are simplified in order to reduce implementation and optimization effort.	
BDTI Solution Certification Benchmarks™ Measure system-level performance on multimedia HW/SW solutions.		
BDTI H.264 Solution Certification Benchmark™	This benchmark verifies and certifies the performance of a full hardware/software H.264 solution. Results are measured using consistent parameters (e.g., frame size and bit rate).	Advantages: This benchmark verifies performance of existing H.264 solutions using consistent parameters, enabling apples-to-apples comparisons. Disadvantages: This benchmark is primarily relevant for H.264 solution providers and users; it does not effectively predict performance on most other applications.

Table 1: The BDTI Benchmark Suites.

complexity level, it must perform the kinds of work that a processor will be expected to handle in real-world applications.

The second area is more subtle. Not only must the benchmark perform relevant work, it must be implemented in a way that is similar to how the corresponding real-world applications are implemented.

Implementation Approaches

There are many approaches for implementing processor benchmarks. For example, benchmarks can be implemented using unmodified high-level reference code or can be optimized by hand at various levels. Benchmarks can be optimized to achieve maximum speed, to achieve minimum cost, or to achieve some other application-specific goal. Benchmarks can be required to achieve a particular level of numeric precision, or the precision can be left unspecified. When using benchmark results, it is critical to understand the implementation approach and assess whether it is relevant to your application. If the benchmark approach isn't a good match to your application approach, the benchmark results probably aren't relevant for your application.

Embedded signal processing software, for example, is typically hand-optimized—at least the performance-critical sections (which are often the sections that benchmarks attempt to model). This is because these applications often have stringent constraints on speed, energy consumption, memory efficiency, and/or cost, and unmodified compiled code is typically not efficient enough to meet these constraints. Therefore, a benchmark that is implemented using unmodified reference C or C++ code is unlikely to yield results that are representative of how the processor will perform when used in a real application, running optimized software.

The implementation approach should be defined based on what's commonly used in relevant end-products. For example, the BDTI DSP Kernel benchmarks (which include common DSP algorithms like FIR filters and FFTs) are carefully hand-optimized, because this is how DSP kernels are most often implemented in products. The BDTI Application Task Benchmarks, on the other hand, are typically implemented using some combination of high- and low-level coding and optimization, the specific ratio of which is guided by trade-offs in memory, speed, and programming effort. This approach is used because the BDTI Application Task Benchmarks are similar to full applications, which are typically *not* completely hand-optimized; performance-critical sections are well-optimized, but other sections may be left to the compiler.

In some cases, the processor itself places constraints on the coding approach. Some processors do not support assembly language programming, for example, and instead require the programmer to use a combination of C and intrinsics. Some processors are so difficult to program in assembly language that they are unlikely to ever be programmed that way. In both cases, the processors should be benchmarked using whatever approach is most likely to be used in a product, given the processors' specific attributes and constraints.

Allowable Optimizations

Once the benchmark workload and general implementation approach is defined, careful attention must be given to the types of optimizations that are permitted. One approach is

to allow every possible optimization. This approach to optimization is unlikely to yield realistic benchmark results because it tends to encourage implementations that ignore performance on one metric (such as memory efficiency) in exchange for optimal performance on another metric (such as speed). It's an approach that's rarely taken in real applications, which tend to have constraints on multiple metrics. Benchmark implementers must be given clear guidelines on how to make reasonable trade-offs between all of the relevant metrics. These may include speed, memory use, energy efficiency, cost efficiency, or (for some technologies) die area. Other optimization guidelines should include:

- Whether the benchmark must be implemented in a way that is general (for example, whether an FIR filter must be written with the number of samples and taps defined as variables or whether they can be hard-coded)
- Whether the implementation can use different data types in certain parts of the algorithm (e.g., use 8-bit data for one section and 16-bit data for the rest)

Sometimes there is no single "right" approach, though it is usually possible to determine reasonable guidelines based on surveying what is typically done in real systems. In any case, it is essential that whatever guidelines are chosen are applied consistently across all benchmark implementations. It is also vital to have an impartial arbiter who evaluates each implementation to determine whether its optimizations fall within the acceptable range.

It's Not All About Speed

Benchmarks can be used to evaluate processors and other technologies (such as FPGAs) on a number of different metrics. The first metric people typically focus on is speed, but memory use, energy efficiency, and cost-effectiveness also exert a significant effect on a product's suitability for an application. In fact, in embedded systems, these latter metrics are often more important than speed—assuming that the processor has enough speed to meet the needs of the application. One common misuse of benchmarks is to use results that were optimized for maximum performance on one metric (say, speed) to show the processor's performance on another metric (say, memory use) without noting the optimization priority.

Evaluating Benchmark Credibility

Assuming that a benchmark has been thoughtfully designed and fairly implemented, it is tempting to assume that the benchmark results are trustworthy. Unfortunately, even when the benchmarks are sound, there are a number of ways in which benchmark results can be misleading. Here, we list a few key questions to ask when evaluating the quality and credibility of benchmark data.

- **Are benchmark results comparable?** Processor vendors often implement their own benchmarks and compare their results to benchmark results published by other vendors or by third parties. While vendors' benchmarks can be useful and informative, you should keep in mind that they are often designed to highlight the specific strengths of one particular processor, not to make fair and accurate comparisons across many processors. More generally, benchmark results from one vendor may not

be comparable to results from other vendors, even if the algorithm is apparently the same. For example, some vendors include descrambling in their FFT benchmarks, others don't—and details such as these are rarely disclosed.

- **Are benchmark results presented for products that are currently available or for future products?** Processor vendors sometimes present benchmark results for chips or licensable IP that haven't yet been fabricated and compare these results to those for their competitors' existing chips. Projections can be useful, but should be evaluated with the knowledge that the chip may not actually reach the target clock speed in the time period projected—and even if it does, the competitors' products may have improved in the interim. In addition, it is common for the behavior (in terms of cycle counts) of actual silicon to differ—sometimes significantly—from that of the initial simulations.

- **Are all benchmark parameters shown?** Benchmark data is often presented without key parameters being defined. For example, in video codec benchmarks, it may not be clear what the bitrate is or the quality of the video output. Without these parameters it is impossible to know whether the benchmark results are comparable to results from other sources. And if you're planning to extrapolate performance for an application with different parameters, be extremely careful. Changing a parameter—even slightly—can result in non-linear and hard-to-predict changes in performance. For example, if a video benchmark is reported for a small frame size that fits entirely in on-chip cache, it's unwise to use this data to make predictions about the processor's performance on a larger frame size. The larger frame may not fit in on-chip cache, in which case performance may be significantly degraded by limited off-chip memory bandwidth, cache miss penalties, etc.

- **What are the system-level assumptions?** Benchmark implementers typically have to make some assumptions about the system under consideration, particularly with respect to the on- and off-chip memory size and bandwidth. If such assumptions are not representative of real systems, or are twisted to favor one processor over another, then the benchmark results are not useful.

For licensable cores, there are additional considerations. Here, the system-level assumptions can be difficult to pin down, but reasonable assumptions must be made in order to obtain accurate and useful benchmark results. In addition, the clock speed, power consumption, and size of the core will vary significantly depending on the fabrication process used. Even for cores fabbed using the same process, there will be process variations. A key question to ask is whether the benchmark results are obtained using typical fabrication process variations—or worst-case. Core vendors often choose to report benchmark results using the typical clock speed, but in most cases chip designers are more interested in the worst-case number. It's important to know what the fabrication assumptions are when comparing benchmark data for licensable cores.

Benchmarking Other Technologies

As any system engineer knows, programmable single-core processors are only one of several classes of processing solutions for digital signal processing workloads. It's also possible to use FPGAs, hard-wired engines, and multi-core solutions, among other choices. It can be difficult to compare the performance potential of different classes of processing solutions, and it is unusual to find apples-to-apples benchmark data spanning multiple classes of processors.

While some benchmarks are primarily appropriate for programmable processors, it is BDTI's view that, whenever possible, benchmarks should be defined in a way that is *independent* of any specific processing technology. To enable fair comparisons across technologies, benchmark implementers must be free to leverage each technology's capabilities in a way that is representative of how they will be used in products. The BDTI Communications Benchmark (OFDM), for example, is specified in a way that allows it to be implemented on nearly any type of processing technology. As a result, BDTI has been able to use this benchmark to compare the performance of high-performance DSPs to that of FPGAs, with some interesting and surprising results.

Even when signal processing benchmarks are only intended to be used on programmable processors, it's essential not to constrain benchmark implementations in ways that make them impossible to implement efficiently on processors other than DSPs. Other classes of processors (such as general-purpose processors) should not be unnecessarily penalized for using different architectural features to accomplish the same tasks.

Beyond Benchmarks

Good benchmark data is useful by itself, but is more useful when accompanied by a thoughtful analysis. Understanding *why* a processor performs as it does on specific benchmarks is often as important as the benchmark results themselves. It's useful to know which features helped or hurt a processor on a specific benchmark, because this helps you understand more about how well the benchmarks will predict the processor's performance in your application.

Of course, benchmarks don't capture all of a processor's strengths and weaknesses. Making well-informed decisions about processors requires a thorough investigation of not only the processor's quantitative characteristics, but also a variety of qualitative issues. The quality of software development tools, the vendor's roadmap, the availability of relevant off-the-shelf software—all of these issues can be more important than the processor's raw performance. Like with benchmarks, having a consistent framework in which to evaluate qualitative factors across processors and vendors is extremely useful.

For these reasons, BDTI typically performs three phases of analysis: benchmarking, analysis of benchmark results, and qualitative analysis. This allows us to arrive at a holistic assessment of a processor's strengths and weaknesses.

Choose Your Benchmarks Wisely

For benchmarks to be accurate and credible, they need to be well-designed, consistently implemented, and thoughtfully used. If they fail in any of these areas the repercussions can be disastrous. We've seen companies select a processor for a new product, spend

months implementing and optimizing their software, and then realize that the processor they selected doesn't have enough performance for the application—all because the benchmarks they used for their processor evaluation neglected to capture key processing requirements. We've seen companies spend months developing and implementing their own benchmarks to assess how their processor stacks up against the competition—only to discover that their results aren't comparable to anyone else's and are therefore useless for competitive analysis. Worse yet, companies sometimes don't realize that their benchmark results aren't comparable to others until they've already used them to make key decisions.

There's not much to be done about these problems after the fact; it's essential to make sure from the outset that you're using the right benchmarks for the task at hand.

BDTI has been designing and implementing signal processing benchmarks for over a decade, and we've become experts in the field. Every time we design a new benchmark, we spend months making sure that all of the issues we've raised in this paper are addressed, and that the benchmark results will provide trustworthy, meaningful information for the people who use them. Our business depends upon it.

Whether you're using benchmarks to help make technical or business decisions or to sell your product, your outcome will depend on the quality of the benchmarks you're using—so make sure that they're good ones.

About Berkeley Design Technology, Inc.

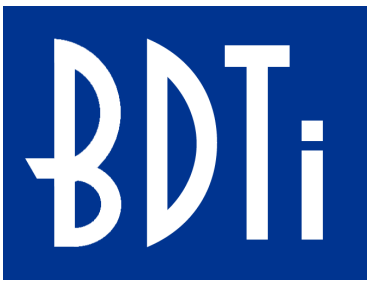
BDTI provides analysis and advice that help companies develop, market, and use signal processing technology.

BDTI is a trusted industry resource for:

- Independent benchmarking and competitive analysis
- Advice and analysis that enable credible, compelling marketing
- Guidance for confident technology and business decisions
- Expert product development advice
- Industry and technology seminars and reports

BDTI customers include

3Com
Agere Systems
Altera
AMD
Analog Devices
Apple Computer
ARC
Arithmatica
ARM
Ascom Timeplex
Avid
Bang & Olufsen
Cadence Design Systems
Canon
Catalytic
CEVA
Cirrus Logic
Cisco
Conexant
Creative Technologies
CSF Thomson
Dow Chemical
E.M. Warburg, Pincus
Ericsson
Euphonix
Faraday
Freescale
Fujitsu
General Dynamics
Glenayre
Harris Corporation
Hewlett-Packard
Hughes Aircraft
Hynix
IBM Microelectronics
IDT
Infineon Technologies
Intel
Intersil
ITT
Jet Propulsion Laboratory
LG
Lockheed Martin
LSI Logic
Mentor Graphics
Mercury Computer
Microchip
Microsoft
Minolta
MIPS
Mitel
Morpho Technologies
Motorola
National Semiconductor
NEC Electronics
Nokia
Northern Telecom
Philips Semiconductors
Plantronics
Qualcomm
RealNetworks
Renesas Technology
ROHM
Samsung
Scientific Learning
Siemens
Silicon Graphics
Silicon Labs
Skyworks
Sony
SoundID
StarCore LLC
STMicroelectronics
Stretch
Sun Microsystems
Synopsys
Tensilica
Texas Instruments
Thales Group
Thomson Consumer Electronics
Toshiba
U.S. Navy
Westinghouse
Wind River Systems
Xilinx
Zilog
Zoran



Berkeley Design Technology, Inc.
2107 Dwight Way, Second Floor
Berkeley, California 94704 USA
Phone: +1 (510) 665 1600
Fax: +1 (510) 665 1680
Email: info@BDTI.com
Web: www.BDTI.com