Cahners
MICROPROCESSOR REPORT
VOLUME 13, NUMBER 8
JUNE 21, 1999
THE INSIDERS' GUIDE TO MICROPROCESSOR HARDWARE

Embedded Benchmarks Grow Up

EEMBC Offers a Better Alternative to Dhrystone MIPS

by Tom R. Halfhill

What's faster than a speeding bullet and more powerful than a locomotive? Without reliable benchmarks, we have to depend on Superman's marketing department for the answer.

Embedded developers won't have to surrender their faith to the marcom gods much longer. After years of searching for an alternative to Dhrystone and other marginally useful benchmarks, the industry finally has a way to compare the performance of microcontrollers, microprocessors, compilers, and other system components. It's a series of benchmarking suites from EEMBC (pronounced "embassy"), the *EDN* Embedded Benchmark Consortium. (*EDN*—the trade publication that was the driving force behind the initiative—is published by Cahners, which also publishes *Microprocessor Report*.) The nonprofit consortium currently has 29 members, including every important vendor of embedded processors (see sidebar below).

EMBEDDED PROCESSOR FORUM

EEMBC has been working on its benchmarking methods for almost three years (see MPR 4/20/98, p. 13, and MPR 3/8/99, p. 3). At last month's Embedded Processor Forum, EEMBC went public with preliminary results based on version 0.9 of the benchmark source code. The first official results based on the 1.0 version of the benchmarks aren't due until 3Q99. In the meantime, EEMBC wants the industry to start thinking about how to interpret the data in ways that make sense.

Benchmark Results Tricky to Interpret

Interpreting EEMBC's data isn't a straightforward task. Unlike SPEC, a similar consortium whose benchmarks measure the performance of desktop PCs and servers, EEMBC isn't summarizing the results as composite scores. There are no aggregate "EEMBCmark" numbers similar to SPECint95 or SPECfp95. Instead, EEMBC is releasing the raw results of every test in its various suites and is leaving it to embedded developers, marketing departments, analysts, and journalists to figure out what the numbers mean.

Of course, that won't stop anyone who knows how to use Excel from conjuring composite numbers from the raw data. Purely as an exercise, MPR produced the chart in Figure 1, which compares the performance of five processors that ran EEMBC's automotive/industrial test suite. The chart is based on the preliminary benchmark data shown in Table 1. (All of the preliminary results are available on the EEMBC Web site at *www.eembc.org*.)

To create a composite score, we normalized the results to one processor (the Infineon SAB C167CS), then computed the geometric mean of the individual test results and plotted the chart on a logarithmic scale. Why not a linear scale? Because most of the bars would have been invisible next to the bar of the 450-MHz AMD K6-2, a desktop PC processor that's 88 times faster than the baseline 25-MHz Infineon.

And therein lies the danger. Obviously, it would be a mistake to conclude that a high-end PC processor like the K6-2 is the best choice for automotive engine control (unless you're designing the Batmobile). The EEMBC benchmarks

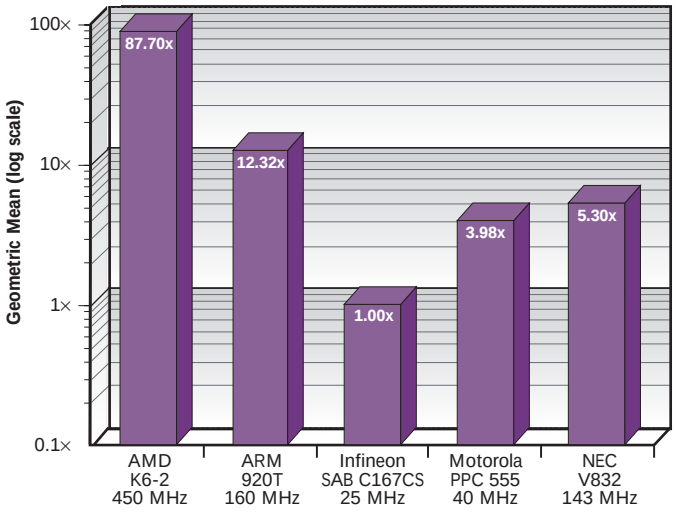

Figure 1. Geometric means are one way to derive composite scores from the EEMBC data, but the results may be misleading.

are orders of magnitude more sophisticated than Dhrystone loops. Boiling the results down to a single magic number defeats what EEMBC has labored for years to achieve.

EEMBC's Suite Spots

Any meaningful interpretation of the scores must start with an understanding of how EEMBC measures performance. There's still plenty of room for controversy within the bounds of EEMBC's rules.

Currently, EEMBC has five benchmarking suites that cover various applications for embedded processors: automotive/industrial systems, consumer electronics, networking, office automation, and telecommunications. Technical subcommittees of EEMBC members define the benchmark tests in each suite. A sixth subcommittee recently formed to explore how EEMBC should benchmark cores from IP vendors.

After extensive discussion, the subcommittees came up with a series of tests based on commonly used algorithms in each category. Table 2 lists all of the tests in each suite. More tests will be added, and some tests will be changed for the 1.0 version of the benchmarks, which EEMBC hopes to firm up this month.

MICHAEL MUSTACCHI

EEMBC president Markus Levy describes the new benchmarks at the Embedded Processor Forum.

Right away, it's apparent that the EEMBC suites are a cross between purely synthetic benchmarks, such as Dhrystone 2.1, and true application benchmarks, such as the ZD WinBench suite for PCs. The EEMBC programs are synthetic in the sense that they are not real applications, but they consist of algorithms typically found in real applications. This is similar to SPEC's approach for PC/server benchmarks, and for the same reason: easier portability across different microprocessor architectures.

EEMBC has faced a much greater challenge than SPEC, however, because there are more embedded processor architectures to test and the architectures are more diverse. EEMBC's goal is to make the benchmarks portable across all embedded CPUs, from 8-bit Vespas to 64-bit Ferraris.

To that end, EEMBC wrote all of the programs in ANSI C and tried to make allowances for the inevitable differences between widely divergent architectures. For instance, to handle different native word lengths, the source code uses abstract type definitions for common data types. The EEMBC code supports 8-, 16-, 24-, 32-, 48-, and 64-bit integers. If an algorithm uses floating-point math, there are corresponding abstract floating-point types. Processors that support longer data types can therefore take advantage of their native word lengths without any alterations to the source code.

EEMBC is documenting the benchmark tests in a databook that's available on the Web site. (The version currently posted is a work in progress.) EEMBC's policy is full disclosure—those who aren't satisfied with composite scores and bar charts can dig into the benchmark data, the algorithms, and the detailed reports that vendors must file with their results. The only thing that isn't public is the source code—that must be licensed. Still, there's enough data available for intelligent developers to draw their own conclusions without relying on spin doctors.

For example, the automotive/industrial suite includes a test program based on an inverse discrete cosine transform

Processor	AMD K6-2		ARM 920T		Infineon SAB C167CS		Motorola PPC 555		NEC V832	
Core Frequency	450 MHz		160 MHz		25 MHz		40 MHz		143 MHz	
Compiler	MS Visual C/C++		ARM Dev Kit 2.50		Keil C166 3.12		Diab 4.3p Rev 6		Green Hills 1.8.1	
	Raw	Normal	Raw	Normal	Raw	Normal	Raw	Normal	Raw	Normal
Table Lookup	319,264	168.9x	43,478	23.0x	1,890	1.0x	14,767	7.8x	31,559	16.7x
Angle-to-Time Calc	715,564	74.1x	73,529	7.6x	9,653	1.0x	36,273	3.8x	17,115	1.8x
Pulse-Width Modulation	1,347,709	41.2x	344,828	10.6x	32,680	1.0x	59,740	1.8x	147,146	4.5x
CAN Response	2,812,149	61.3x	438,597	9.6x	45,872	1.0x	104,861	2.3x	204,335	4.5x
Tooth-to-Spark Calc	125,552	165.0x	15,152	19.9x	761	1.0x	4,861	6.4x	10,483	13.8x
Road Speed Calc	2,180,445	49.3x	454,545	10.3x	44,248	1.0x	66,344	1.5x	199,602	4.5x
Infinite Impulse Response	96,374	38.8x	9,709	3.9x	2,481	1.0x	8,380	3.4x	8,703	3.5x
Bit Manipulation	11,860	86.6x	2,703	19.7x	137	1.0x	341	2.5x	877	6.4x
Basic Integer and FP	384,615	445.8x	6,154	7.1x	863	1.0x	13,295	15.4x	2,180	2.5x
Pointer Chasing	23,738	82.5x	3,361	11.7x	288	1.0x	594	2.1x	1,619	5.6x
Matrix Math	1,375	382.1x	n/a	n/a	4	1.0x	66	18.3x	8	2.1x
Cache Buster	982,318	34.4x	468,750	16.4x	28,571	1.0x	78,390	2.7x	175,682	6.1x
Inverse DCT	15,221	263.8x	2,083	36.1x	58	1.0x	471	8.2x	976	16.9x
Fast Fourier Transform	627	60.9x	125	12.1x	10	1.0x	35	3.3x	47	4.5x
Inverse FFT Filter	683	75.0x	n/a	n/a	9	1.0x	37	4.1x	50	5.5x
Finite Impulse Response	46,581	24.3x	21,277	11.1x	1,919	1.0x	4,348	2.3x	11,606	6.0x

Table 1. EEMBC currently reports raw scores as iterations per second, so higher numbers are better. We normalized these preliminary scores in the automotive/industrial suite to the Infineon processor. The chart on page 1 is distilled from this data.

(iDCT). The preliminary EEMBC databook says the test "simulates an embedded automotive/industrial application performing digital video and graphics applications such as image recognition." There's no reason that someone who's interested in the graphics performance of an embedded CPU for a digital set-top box couldn't use the results of that test too, along with the JPEG compression and decompression tests in the consumer suite. Likewise, a CPU's scores in the fast Fourier transform (FFT) and finite impulse response (FIR) tests will be useful for developers working on audio applications and many other things besides engine control.

Indeed, a creative approach to the data will be almost imperative while EEMBC labors to flesh out the suites with more tests. The automotive/industrial suite is currently the most complete, with 18 tests. That's as many as the other four suites put together.

EEMBC's technical subcommittees are concentrating on the most important algorithms first. The networking suite, for instance, has only two tests in the 0.9 version of the benchmarks, but those two include the Dijkstra and Patricia algorithms. Dijkstra is part of the Open Shortest Path First protocol that's the most common method for routing packets on the Internet. Patricia (a rather contrived acronym that means "practical algorithm to retrieve information coded in alphanumeric") is a tree-search routine that's also used for packet routing. New tests under development for the networking suite will benchmark operations required for virtual private networks (including authentication, encryption, and data compression); routines that set up and tear down point-to-point protocol sessions; and additional packet-processing functions performed by LANs and WANs.

Despite EEMBC's efforts to make the suites universally portable, some of the programs and associated data files simply won't fit within the limited address space of 8-bit processors, or would take too long to run. Two examples are the JPEG tests in the consumer suite and the Patricia test in the networking suite. In those cases, EEMBC uses smaller data sets. But that means comparisons across different CPU architectures are subject to debate. All of the programs currently report test results in the form of iterations per second, so higher raw numbers are better. Yet an 8-bit processor won't be running exactly the same code or manipulating the same data as a 32-bit processor.

In fairness, no benchmarking can produce an apples-to-apples comparison in such cases. Presumably, embedded-system designers who are evaluating CPUs will narrow their choices by other means and won't need to compare the raw performance of an 8051 to that of an UltraSparc 2e.

Rather than cripple the benchmarks in ways that could make the results less valid on larger processors, EEMBC is considering a two-tiered approach: one collection of suites for 8- and 16-bit chips and another for 32- and 64-bit chips. That would also allow the suites to include different algorithms that more accurately reflect how embedded designers use different architectures in real-world applications.

Two Ways to Test

When EEMBC releases the first official results, there will be two sets of numbers. One set will be similar to the preliminary results already made public. Called "out-of-the-box" scores, they represent a level of performance that any embedded developer could achieve merely by building the EEMBC source code with the same compiler, using the same optimizations. EEMBC requires vendors to build the code with a publicly available compiler—not an in-house supercharged compiler, which some CPU vendors have used to soup up their SPEC numbers—and to report the compiler optimizations they used. All of this information will be posted on the EEMBC Web site.

EEMBC refers to the second set of results as the "optimized" or "full-fury" score. Early on, EEMBC recognized that embedded developers do a lot of fine-tuning in high-level languages and assembly language. To mirror that real-world experience, vendors can optimize the EEMBC programs in the same ways they would optimize real applications. Depending on the rules each technical subcommittee defines for

Algorithm	Test Suite
Table Lookup and Interpolation	Auto/industrial
Angle-to-Time Conversion	Auto/industrial
Pulse-Width Modulation	Auto/industrial
CAN Response to Remote Request	Auto/industrial
Tooth-to-Spark Calculation	Auto/industrial
Road Speed Calculation	Auto/industrial
Infinite Impulse Response (IIR) Filter	Auto/industrial
Bit Manipulation	Auto/industrial
Basic RTOS Context Switching	Auto/industrial
Basic Integer and Floating Point	Auto/industrial
Pointer Chasing	Auto/industrial
Matrix Math	Auto/industrial
Cache Buster	Auto/industrial
Inverse Discrete Cosine Transform (IDCT)	Auto/industrial
Reserved (Future Test)	Auto/industrial
Fast Fourier Transform (FFT) Filter	Auto/industrial
Inverse FFT Filter	Auto/industrial
Finite Impulse Response (FIR) Filter	Auto/industrial
JPEG Compress	Consumer
JPEG Decompress	Consumer
IrDA Transmit	Consumer
IrDA Receive	Consumer
Image Enhancement	Consumer
Dijkstra Routing	Networking
Patricia Table Lookup	Networking
Bitmap Rotation	Office automation
Bezier Curves	Office automation
Text Processing	Office automation
Dithering	Office automation
Autocorrelation	Telecommunications
Bit Allocation	Telecommunications
Inverse FFT Filter	Telecommunications
FFT Filter	Telecommunications
Cascaded Biquad IIR Filter	Telecommunications
Viterbi Decoder	Telecommunications
Convolutional Decoder	Telecommunications

Table 2. EEMBC has five benchmark suites covering different categories of embedded applications. More tests will be added, and some tests will be changed in the 1.0 version of the benchmarks.

The Usual Suspects

EEMBC (*www.eembc.org*) has 29 full-fledged board members. Membership costs $30,000 a year, which includes a seat on the board and on all of the technical subcommittees that define the benchmark suites.

Companies that don't want to join the board or that don't qualify (only semiconductor companies are allowed to sit on the board) can participate in the benchmark definitions by paying $7,500 a year to join a subcommittee. For $15,000 a year, a company can join all of the subcommittees. EEMBC will also license the benchmark source code to nonmembers (such as tool vendors) for an initial fee of $30,000 plus $5,000 for each additional year.

The board members are ARM, AMD, Analog Devices, ARC Cores, BOPS, Conexant Systems, DSP Group, Fujitsu, Hitachi, IBM, Infineon, IDT, Intel, LSI Logic, Lucent, MIPS Technologies, Mitsubishi Electric, Motorola, National Semiconductor, NEC, Panasonic, Philips, QED, SandCraft, STMicroelectronics, Sun Microelectronics, Tensilica, Texas Instruments, and Toshiba.

each test, those optimizations may include rewriting the EEMBC source code, calling special function libraries, hand-tuning critical loops in assembly language, and taking advantage of special hardware in the processor. For example, Motorola could use the PowerPC 555's built-in timers to speed up the pulse-width modulation test in the automotive/industrial suite.

This is one of the more controversial aspects of the EEMBC benchmarks. Will the optimized scores reflect the abilities of the processors—or the skills of the programmers and the resources of the vendors? Critics say it'll be nothing more than a coding contest.

EEMBC defends the optimized benchmarks, pointing out that vendors will be able to demonstrate the unique features of their CPUs, compilers, and libraries. Tinkering with a few lines of code to ensure that a program takes advantage of a branch-delay slot or the latency of a critical operation can make a huge difference in real-world embedded performance. And real-world performance that's achieved by using real-world techniques is what EEMBC wants to measure.

But even if all's fair in love and war, it's not in benchmarking. EEMBC has laid down some rules to prevent the worst offenses. When rewriting the EEMBC source code or substituting assembly-language code, vendors must preserve the intent of the original algorithms. They can't use undocumented compiler switches or libraries that aren't available to developers. Nor can they pull the kinds of tricks that have subverted other benchmarks in the past, such as hard-coding the results directly into the programs, or even into the compilers. And vendors must report all of the modifications they've made and post the information on EEMBC's Web site.

ECL: The Benchmark Cops

EEMBC isn't naive enough to believe that vendors will follow the rules just to play fair. To police the benchmarks, two founders of EEMBC have formed an organization called EEMBC Certification Laboratories (ECL). Markus Levy, a technical editor at *EDN*, is also the president of EEMBC and the CEO of ECL. Alan R. Weiss, former manager of software technologies at Motorola, is the chairman and CTO of ECL.

For a fee ranging from about $3,200 to $5,000, ECL will verify the benchmark results submitted by vendors and certify that they were produced according to the rules. EEMBC members can use uncertified benchmark results for internal purposes and can share them with customers under nondisclosure agreements, but they can publish only the ECL-certified scores.

And ECL serves another purpose: as an independent organization that can sign nondisclosure agreements, it isolates EEMBC members from obtaining proprietary information about other companies' unreleased products.

ECL conducts the benchmark testing within a "test harness"—a control program that runs on a DOS or Unix host. Vendors must port a test harness adaptation layer (THAL) to their processor boards. The THAL is really a simple API that recognizes a few basic commands, such as start benchmark, stop benchmark, send data, and receive data. To certify a vendor's benchmarks, ECL follows this procedure:

- Rerun the vendor's compiled binaries to see if the results are reproducible. This prevents a vendor from submitting sanitized binaries that weren't used to produce the claimed results.
- Recompile the EEMBC source code with the same compiler and switches used by the vendor, then rerun the new binaries and compare the results with the vendor's claims. This prevents vendors from using secret compiler optimizations, from modifying the EEMBC source code for the out-of-the-box scores, or from altering the compiled binaries.
- Rerun the vendor's binaries against other data sets that EEMBC holds secret to see if the results are still valid. This prevents vendors from using data-specific optimizations that wouldn't work with real-world data.
- Rerun the test suite using a private version of the benchmark code that was compiled with the same tools used by the vendor. Among other things, the private code uses different variable names than the regular EEMBC source code. This prevents compiler vendors from implementing EEMBC-specific optimizations that identify the source code by its unique characteristics. (Some compilers are known to have "Dhrystone" switches or special compilation routines that short-circuit the loops and play other tricks with benchmark source code.)
- Examine the vendor's binaries with special tools to verify that the code follows EEMBC's rules. To ensure that optimized binaries don't take illegal shortcuts with the original algorithms, ECL's programmers will even inspect the vendor's source code line by line and verify the output. For example, they will compare a compressed JPEG file with a precompressed version of the same file to make sure the test machine

doesn't just return a bunch of data and claim it's a compressed image.

SALT verification seems simple by comparison, but ECL says it can finish the certification process in about two weeks.

Life After Dhrystone

Nobody claims EEMBC will put an end to the fine art of "benchmarketing." Indeed, the new wealth of data will probably inspire marketing departments to invent some more creative ways to make a pokey processor look like a speed demon. But even those who criticize some aspects of EEMBC (such as the optimized tests) agree it's a significant step beyond Dhrystone—a nearly worthless metric that survives only in lieu of viable alternatives. There is a good reason why critics deride MIPS as "meaningless indicator of processor speed" or "marketing's idea of processor speed."

In addition to providing a much more accurate way to evaluate the performance of embedded processors, the EEMBC benchmarks can also measure the relative performance of other system components and tool chains. By running the benchmarks on a test system with the same processor but different peripheral parts, vendors can isolate bottlenecks in subsystems. By compiling the benchmarks with different tools, they can find out which ones produce the best code for a particular application. Already, some EEMBC members are reporting differences of 40–50% in performance when building the preliminary suites with different compilers.

Even when using the same compiler, results can vary greatly, depending on the compiler flags. Modern compilers may have hundreds of flags, so the EEMBC benchmarks can help programmers figure out which optimizations work best for the algorithms that really matter in their applications.

Only two things are missing from the EEMBC picture: an easily quoted composite score and a common baseline for comparing relative performance. Yes, a composite score like SPECint95 is an inadequate summary of the data. But people will insist on one anyway, so EEMBC might as well standardize on a weighting method that prevents the most abominable distortions. Similarly, a standardized baseline could be a useful reference point. Dhrystone MIPS were originally based on the performance of a VAX 11/780, a popular machine at the time. (The standard conversion factor: 1,757 Dhrystone iterations equal one VAX MIPS.) Perhaps EEMBC could normalize its scores to the performance of a Motorola 68000, the first chip in the world's most popular embedded processor architecture. Without such a composite score, the EEMBC benchmarks probably won't replace Dhrystone MIPS entirely, because marketing departments insist on a single figure of merit.

Unless a serious disagreement fractures the consortium—unlikely at this point—EEMBC still appears destined to become the most respected and useful benchmark in the embedded industry. M