How Many VAXes Fit in the Palms of Your Hands?  
Exploring an old benchmark on a new CPU chip  

John L. Henning, Oracle, Nashua, NH, USA

**Background: Benchmarking in the 1970s and 1980s**

During the early history of microprocessors, benchmarks were of interest to customers and important to marketing, but results and methods were not comparable. For example, a 1985 Performance Summary [1] from a vendor of popular minicomputers contains results from a variety of benchmarks, with a variety of weaknesses:

*Instruction timing:* A 10-page table provides instruction timing for 7 models, ranging from 0.096 for a bit clear to 9007 for a CISC polynomial. (Although not stated, the unit is presumably microseconds.) Concerns: (1) The assembly-language benchmark program is not provided. (2) Customers would not know whether their own applications primarily use fast or slow instructions. (3) Comparisons are provided among the single vendor’s systems, but no comparisons are provided to other systems. (4) Even if such comparisons were available, they would not be very meaningful given architectural differences.

*Fortran benchmarks such as GAUSS, HANOI, HUGHES, PRIME:* Times are provided for 50 Fortran benchmarks on 5 computer models. Because these are written in a higher-level language, they may be more meaningful than the assembly-language benchmark of individual instructions. Concerns: (1) Although the benchmarks are claimed to be "industry standard", it would have been useful to include a reference to where they may be found. (2) The text says that multiple benchmarks were modified "to reduce variability" but does not define what that means. (3) It is not clear whether times can be compared to times seen on systems from other vendors, especially given that there were modifications. (4) It is noted that some vendors may have "omitted operating system overhead in their quotes" of the benchmark results. (5) Some of the benchmarks had "dead code" which was optimized away when the compiler recognized that it served no useful function. For two of the benchmarks, the result was a run time of zero seconds.

*Transaction processing:* A 40-page chapter provides information about several transaction processing workloads, comparing 6 computer models while varying load. Concerns: (1) The benchmarks were not available to customers. (2) They exercise the vendor software environment, and are not portable. Therefore, it is impossible to do comparisons.

*Whetstone and Dhrystone:* For these benchmarks, references are provided to versions of source code. Concerns: (1) As noted by Reinhold Weicker, the author of Dhrystone [2], both of these benchmarks are "synthetic": they collect and measure program fragments. As such, they may miss important characteristics of real applications. (2) Both had several popular versions, and it was not always clear which version a vendor quoted. (3) Dhrystone - unlike many previous benchmarks - includes a series of run rules, written by Weicker [3]. However, Dhrystone did not have a mechanism for enforcement of the rules or for peer review of results.

**SPECmark**

At the time that Weicker was publishing his Overview of Common Benchmarks [2], the Standard Performance Evaluation Corporation was just coming into existence and publishing its first results. Weicker noted that "SPEC's goal is to collect, standardize, and distribute large application programs".

The initial benchmark suite was termed "SPECmark" (later known as SPECmark89 or SPEC CPU 89). SPECmark improved comparability because: (1) Application programs provide more meaningful data than synthetic kernels and instruction timings. (2) SPEC controls the source code, thereby reducing ambiguity as to what is measured. (3) The benchmarks were ported to multiple environments. (4) Run rules constrain practices that are allowed. (5) Reporting rules require that sufficient information is provided so that results can be reproduced. (6) For results published by SPEC, testers are required to submit their results for peer review.

Perhaps most importantly, (7) SPECmark checks whether or not the program obtained acceptable answers.
benchmarks omit this step, and without it, results may be meaningless. As a witty computer scientist observed: "I can make it run as fast as you like if you remove the constraint of getting correct answers." [4]

SPECmark performance was calculated relative to the performance of the "reference system", the then well-known VAX 11/780, which defined performance of 1.0. Figure 1 shows a model of a VAX 11/780. The actual size of the CPU cabinet on the left side of the model is about 150 x 120 x 75 cm (60 x 45 x 30 in.), weighing 500 kg (1100 lb.), with a power requirement of 6225 W [12]. For example, the result disclosure page for the SPARCstation 330, excerpted in Figure 2, shows that this 1989 desktop system was already over 10x as fast as the 1978 dual-refrigerator-sized VAX.

![Figure 2](image-url) A 1989 result using the original SPECmark. In 1989, a single chip provided >10x the performance of the VAX 11/780

**SPECmark Rating for a Modern System**

An attempt was made to discover the SPECmark rating of a contemporary system using a contemporary compiler. Results for a single core of an Oracle Cloud system with Intel Xeon Gold 6354 [5] using GCC 10.2 are shown in Figure 3.

![Figure 3](image-url) SPECmark® Release 1.2b Summary

**Unofficial SPECrate89 Throughput**

Although Figure 3 provides an estimated SPECmark value for the Oracle system, the test used only 1 core on a 36-core system. It would be interesting to have a measure of full-system CPU performance. Over the years, SPEC CPU defined several throughput-oriented SPECrate metrics for multiple processors. [9] [10] Although the definitions have varied in the major releases of SPEC CPU, all of them include:

- Multiple identical copies are started.
- The observed time is from the start of the first copy to completion of the last copy.
- The SPECrate metric is inversely proportional to the observed time.

The system achieved an estimated SPECmark rating of 17,826. This result is termed an "estimate" because of changes to SPECmark that have not been approved by SPEC, including:

- Modify timing to use perl `Time::HiRes` instead of `/bin/time`.
- Include appropriate header files, such as `stddef.h`, `string.h`, `errno.h`.
- Use `stdargs.h` instead of `varargs.h`.
- Resolve symbol clashes.
- Adjust `static` and `extern`.
- Fix argument types where these led to incorrect answers.
- Attempt to fix compiler warnings which may be relevant to compiling in 64-bit mode. Due to time constraints, this attempt was cut short, and as can be seen in the notes section of Figure 3, some benchmarks were compiled in 32-bit mode.

The attempt to use this long-retired benchmark demonstrated additional ways in which SPEC has improved comparability over the years. (8) Starting with SPEC CPU 2000, all benchmark tuning is placed in a single config file which is published with the result. (9) The original SPECmark had several benchmarks which read no input files. This is dangerous because if too much is known at compile time, ultimately, a benchmark may be reduced to a print statement. (10) Although SPEC CPU prefers benchmarks that are derived from real applications, several SPECmark benchmarks are sufficiently small [6] that they appear to be kernels. Later SPEC CPU releases refreshed the suites with new applications and new versions of old applications, leading to much larger source code, as shown in Figure 4 and in the description page for SPEC CPU 2017 [7].

![Figure 4](image-url) SPEC CPU Suite Growth Updated 4/2017

Later SPEC CPU releases refreshed the suites with new applications and new versions of old applications, leading to much larger source code, as shown in Figure 4 and in the description page for SPEC CPU 2017 [7].
The SPECrate metric is proportional to the number of copies. (Exception: SPECmark89 v1.2b reported the number of copies, but did not multiply by them.) Beyond the above list, the definitions have varied, usually by including additional constant factors that we intended to cause the reported results to fall within a desired range.

In the interest of providing some measure of full system SPECmark performance, the year 2017 method of calculating throughput was employed [1], which is simply:

\[ \text{ncopies} \times \text{reftime} / \text{observed time} \]

where the reftime is the time for a single copy on the reference system – in this case, the VAX 11/780. The results are shown in Figure 5, which is marked "Unofficial" because it not only uses the unapproved changes of Figure 3, it also uses an anachronistic method of calculating the throughput.

Summary

How Many VAXes Fit in the Palms of Your Hands? If you hold one contemporary Xeon Gold 6354 in each palm, you hold the processing power of over 400,000 VAX 11/780s.

[1] The 1985 Performance Summary is the third edition of a glossy, typeset, well-organized document with 164 pages and many tables and graphs. It includes work by multiple performance groups at a now-defunct computer manufacturer. The full title is not provided here because it is labeled "For Internal Use Only", although one suspects that customers may have routinely seen copies or excerpts.