Western Region Technical Attachment
NO. 88-30
November 1, 1988

BENCHMARKING AFOS, OR IS THE IBM-PC JUST A REPACKAGED ECLIPSE S/230?

David Linder - WSFO San Francisco

One of the more interesting aspects of the AFOS Systems Manager (ASM) position at WSFOs is that the ASM frequently inherits the additional title and responsibilities of "resident computer expert." This "expert" is the National Weather Service's version of Shell's Answer Man. He (or she) is the one who people turn to when they have a computer question even when the topic is only remotely related to computer science. Thus, the questions cover a broad spectrum of topics -- from "what is binary arithmetic?" to "what is the best personal computer to buy?", or from "what are these funny switches on the front of the AFOS computer?" to "how many computer hackers does it take to change a light bulb?"

Often, the questions arise because of a lack of understanding about some basic computer science concept (it may be "basic" to a computer nerd but it is probably "advanced" to more normal folks). For example, I have heard many queries related to the computational capabilities, or lack thereof, of the Data General minicomputers (Eclipse S/230s) that form the backbone of the AFOS system. From the user's perspective, the AFOS system's response time is far from instantaneous and sometimes can be downright abominable. On the other hand, from an ASM's perspective, it is surprising how well the Eclipses do. Compared to the computers of 1988, the Eclipse architecture is ancient. Yet, the AFOS computers still do a reasonably good job driving as many as eight synchronous lines, a couple of dozen asynchronous lines, while simultaneously executing foreground and background programs.

The basic computer science concept at issue here is: "how do you measure the performance of a computer system?" Subjective evaluations, like those above, can lead to widely disparate results. In fact, just defining what you are trying to measure (computer performance) is an entire science by itself -- in many ways, the intricacies of computer benchmarking are analogous to the complexities encountered in weather forecast verification.

Still, even with the difficulties involved in designing adequate performance metrics, I wanted to produce some simple comparisons of Eclipse execution speed versus other commonly-used computer systems. The hope was that these measurements would give AFOS users a clearer depiction of their computer's performance compared to other machines used today.

To maintain simplicity, I designed benchmarks to measure just the most fundamental mathematical operations performed by the CPU component of each computer. In other words, I tried to exclude effects of overall I/O throughput, compiler efficiencies, and multitasking.

The benchmark programs were written in either FORTRAN (on the Eclipse) or C (on all other computers) with no compiler optimization. All of the programs followed this pseudo-coded algorithm (where is the operation to be benchmarked):

1) read clock;
2) loop thru 1 million iterations
a = b <op> c;
3) read clock and compute time used in step 2);
4) compute time used for 1 <op> by dividing result of step 3) by 1 million then subtracting time required for one cycle through a null loop;

I examined the machine code produced by each compiler to ensure that the benchmark was indeed measuring the mathematical operation and not some artifact of the compiler. The Eclipse benchmarks were executed several times during "quiet" AFOS periods to get a best-case measurement of Eclipse performance.

I had access to five different computers (listed in order of the age of the design): an Eclipse S/230, a plain-vanilla IBM PC (4.7 MHz), a Compaq-286 (12 MHz), a Macintosh II (16.5 MHz), and a Sun 4/280 (20-30 MHz). Although clock speed in an important discriminator, other significant differences in architecture among the systems listed also played a major role in the results. For each machine I measured the time required for: a null loop (the time it takes to cycle through a loop without any executable statements in it), a function call (where the function contained no executable statements), an add operation, multiply operation, and divide operation. The mathematical operations were examined for integers, single precision floating point numbers, and double precision floating point numbers. In general, the newer machines use higher precision 32-bit integers as opposed to 16-bit integers in the older ones.

The results are shown in Figures 1 and 2. The y-axis is shown in a logarithmic scale so that differences between the machines are understated. In general, AFOS and the IBM-PC perform similarly in all of the benchmarks and both badly trail the rest of pack. Their speed ranges from 20 to 500 times slower than the fast Sun workstation. The clear winner is the state-of-the-art Sun. Its results are even more exceptional when you consider that this is a multitasking UNIX machine which was usually running 5-10 other tasks when the benchmarks were run.

This experiment also shows how inefficient some mathematical operations are compared to others. Simple integer operations are usually 2-3 times slower than a null loop operation. Floating point operations are 2 (for the newer machines) to 10 times slower than their integer counterparts. Finally, double precision floating point operations are another 1.5 to 2 times slower than their single-precision counterparts. Therefore, it is no surprise that programmers are encouraged to use integer arithmetic.

Altogether these benchmarks required very little time to create and run on all of the machines involved. The results provide a realistic but somewhat naive look at the rapidly changing state-of-the-art in computer hardware design.

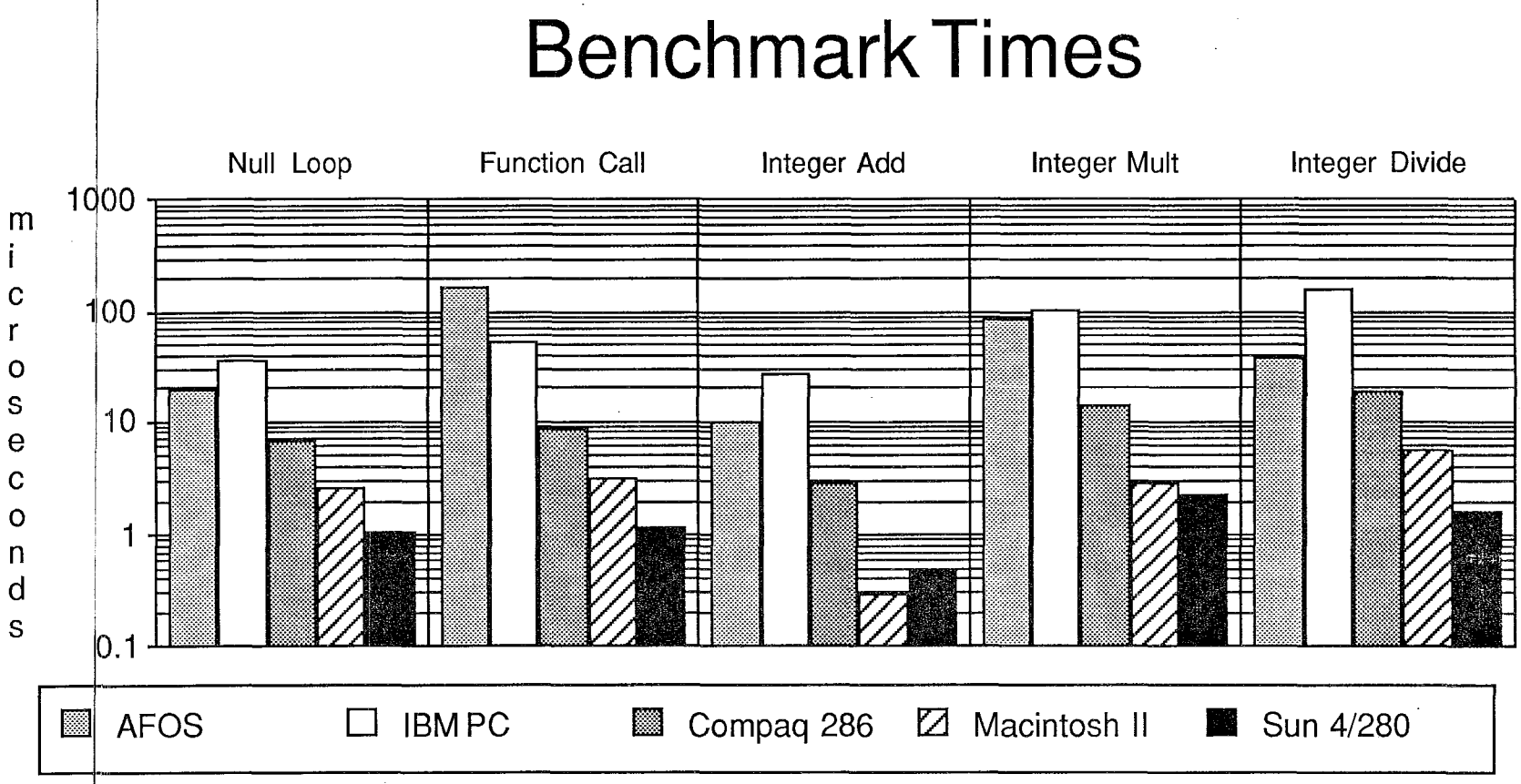

Figure 1: Benchmark execution time for each specified function on five different computers.

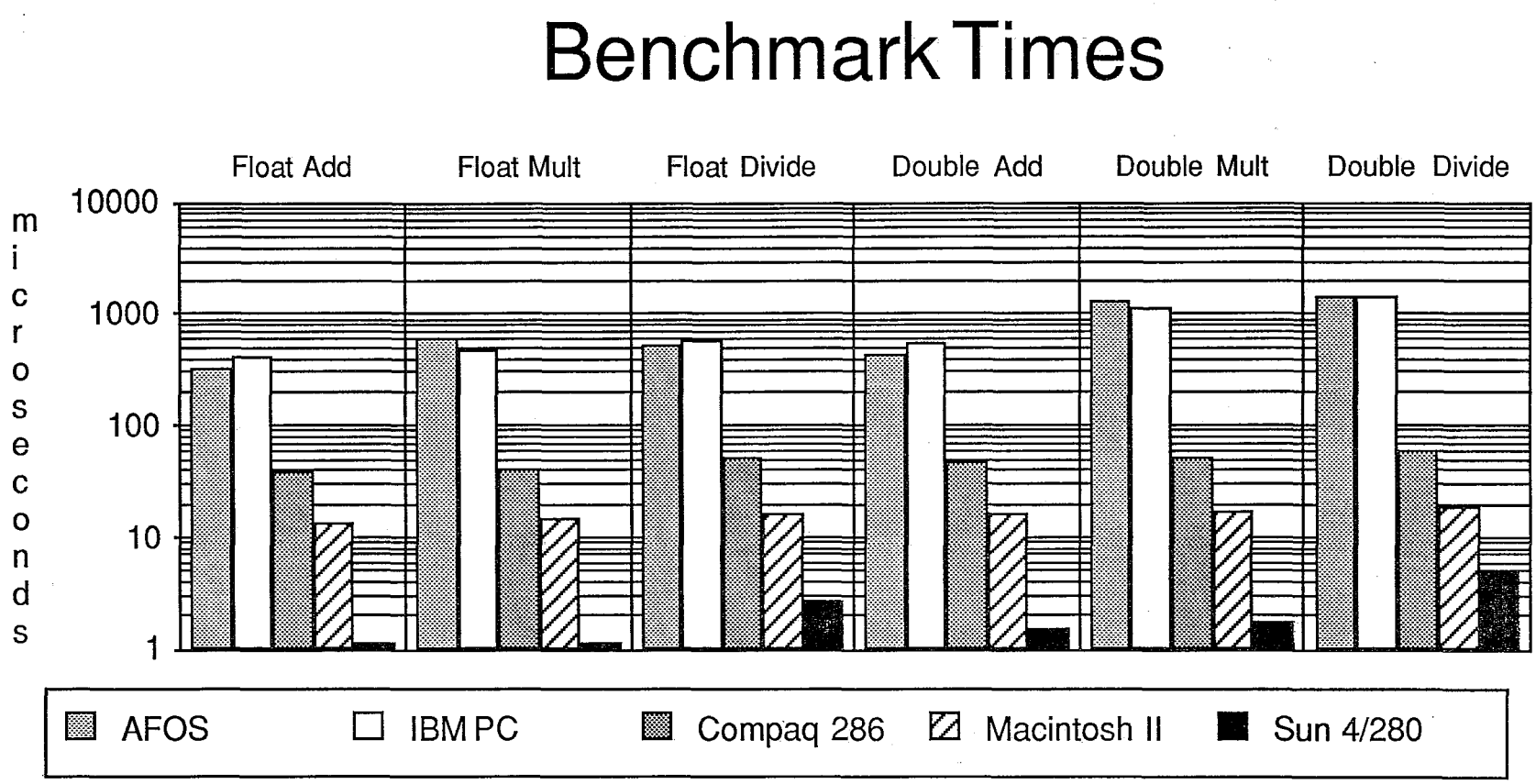

Figure 2: Benchmark execution time for each specified function on five different computers.