Two Questions for Drs. Payne and Bhandarkar concerning their paper "VAX FLOATING POINT: A Solid Foundation for Numerical Computation" presented at Electro/80

W. Kahan
May 9, 1980

The first question is not designed to make the VAX look bad; in fact your proposal for enhanced VAX floating point is rather better than has been provided on many another machine which we have had to live with for a long time. This question is provoked by your several mistakes concerning the comparative merits of the KCS proposal for an IEEE/CS floating point standard:

1. How do you reconcile your condemnation of the KCS scheme with the ample evidence that, roughly speaking,

A: Numerical programs run noticeably better on KCS arithmetic than on enhanced VAX;

B: Numerical programs that run equally well on both systems are significantly easier to write for KCS arithmetic than for enhanced VAX.

These statements will be amplified and clarified later.

The second question responds to your concern for cost/benefit trade-offs, taking into account that those trade-offs must be perceived differently by hardware vendors, by software suppliers and by consumers of numerical results. Assuming assertions A and B to be substantially correct, the only trade-off that could weigh against KCS would be its implementation cost if that cost were too high:

2. Is it fair to say that the principal costs to DEC of implementing KCS in the VAX line are perceived by you to consist mostly of the costs of changing your previous commitments?

Now to amplify and clarify assertions 1A and B and question 2.

1A: When from given data a given numerical program produces results different using KCS arithmetic than using the enhanced VAX arithmetic, the difference must be caused by roundoff or by exceptions. On those occasions when a significant difference is due solely to roundoff, KCS's results must almost surely be the more accurate since KCS specifies unbiased rounding and tighter limits (e.g. in $\sqrt{x}$) than VAX does; even so, results differing significantly only because of roundoff are frequently both wrong. Not so for results differing significantly because of some exception like underflow; then KCS's gradual underflow loses so much less information than VAX's flush-to-zero that the KCS result is almost surely the correct one unless the program was designed specifically to falsify this statement. Of course, significant underflows are rare, so rare that hardware vendors tend to overlook them, but not so rare that conscientious programmers can ignore them. Overflows are very likely to be significant when they occur; that is why KCS's overflow threshold was designed to be twice as big as VAX's. All told, those rare exceptions that cause a program to malfunction do so significantly less often when they are all handled by default as KCS specifies than when exceptions are handled as VAX specifies, some by default (underflow to zero) and others left to local option; at least this is so in my experience with a wide range of users, programs and machines (Ferranti Mk1, ILLIAC I, IBM 650, EDSAC II, IBM 7090-7094, IBM 360-370, B5500, CDC 6400, PDP-10, PDP-11, HP9825, VAX-11/780, ...).

1B: Ideally, if a numerical problem's data and its solution both lie within the computer's range, then a numerical program intended to solve that problem should do so to within practically unavoidable uncertainties imposed by roundoff. Departures from this ideal instigate acrimony like

"Your program has a bug," says the user;
"Your data is unreasonable," says the programmer; and both say
"This hardware doesn't work as well as ... on these problems."

Conscientious programmers attempt to match the ideal in so far as they can do so without unduly sacrificing performance, or using capabilities (like double or quad precisions) not available on all of the computers for which the programs are intended, or incurring other costs like complicated tests prone to error and peculiar to just one computer.

No instance has been found where a conscientious applications-programmer's task would be harder with KCS arithmetic than with VAX's, but numerous instances including some of the most important numerical computations have been shown to be significantly harder to program conscientiously on VAX than with KCS arithmetic of the same precision:

Matrix multiplication, inversion, eigenvalues, eigenvectors.
Complex multiply, divide, absolute value, ...
Acceleration of slowly convergent sequences.
Numerical quadrature and ordinary differential equations.
Evaluating continued fractions and polynomials, and finding their zeros.
Precision-doubling algorithms like T. J. Dekker's.
Interval Arithmetic.

2: Among the ways in which KCS could cost more to implement than VAX's arithmetic are these:

i) More hardware, though this might be traded off with (ii) or (iii).
ii) More microcode or support software.
iii) Slower execution speed.
iv) More engineering time and delay to market, hence loss of market share.
v) The cost of *any* non-evolutionary change from prior commitments.

Implementers of KCS including J. Palmer at INTEL, G. Taylor at Berkeley, J. Coonen at Berkeley and Zilog, D. Hough at Tektronix, G. Stafford at Beckman Instruments, and others, all agree that the first three costs are inconsequential for any but the fastest machines provided the designer is aware of all that needs doing. (For the fastest parallel or pipelined array-oriented designs, competing with the CRAY-1, the KCS proposal recommends its optional extended format for accumulating registers. At the cost of some extra hardware, that format augments both range and precision to defend against both over/underflow and roundoff's depredations when handling very large arrays. Even the VAX architecture sneaks a kind of extended format into its EMOD and POLY instructions despite the architects' proclaimed distaste for the concept. Otherwise quad does for VAX, but slower, what extended does for KCS.)

The fourth cost, delay to market, must weigh heavily upon any pioneer. At first this cost depends upon how clever are a company's engineers. Later "what one fool can do, another can." I believe DEC's engineers are clever. Therefore I do not underestimate this cost when I doubt that it alone could decide the issue -- unless those engineers were already too far committed to something else. Which brings us to the fifth cost, the cost of change.

About two years ago, roughly when the KCS proposal was being developed on the west coast, the G format was introduced (in a VAX at Harvard's Astrophysical Lab) to remedy shortcomings in the exponent range inherited by the VAX from the PDP-11. Since then the G format has been adopted into an enhanced architecture for a line of VAX-like machines of varying levels of performance. When the KCS proposal appeared, with its single and double formats almost the same as the respective single and G formats of the enhanced VAX, the reasons for the differences were at first not appreciated at DEC. I imagine that whenever advantages of KCS began to be perceived they appeared inadequate to justify a change in VAX's format. Few computer purchasers know how to assess the quality of a machine's arithmetic until they have lived with it awhile, and fewer care, but all who own a PDP-11 can see whether a new machine's arithmetic matches the old or not. Therefore a change to KCS must have appeared more likely to lose VAX sales than to gain them ... except if KCS became widely adopted as a standard.

DEC's dilemma is familiar; "the better is the enemy of the good." That dilemma was brushed aside by the IEEE/CS Floating Point Standard subcommittee two years ago when it decided that to choose a format already most widespread in use must imply the choice of IBM's hexadecimal format, a horrible thought. So the committee sought the *best* format for future microprocessor designs unencumbered by the past's mistakes. We all knew that the PDP-11 format for single and G format for double were "safe" choices from a marketing viewpoint, but we chose to try to find something better. The KCS proposal serves a machine supplying only single precision (32 bit) floating point rather better than would the corresponding subset of VAX's architecture; and on the other hand a full implementation of KCS including single, double and either extended or quad as described in Coonen's paper in *Computer* (Jan. 1980) defines a better standardized environment for numerical software than does the enhanced VAX.