say, a large computer and download them to microprocessor-based systems over phone lines. CUFORM was able to perform the arithmetic calculations needed in the process of loading. Ian Willers of CERN visited UC Berkeley and joined the working group and helped develop the improvements and modifications that resulted in MUFOM. The load/relocate command was reworked considerably to improve usage of file space. A primitive library facility was provided. An iterate operator was added to allow a command to be repeated a number of times. This operator allows, for example, memory to be filled with an arbitrary number of floating-point operations, such as setting memory to Not-a-Number (NaN), which permits finding uninitialized variables. An error operator was added so a function can be calculated at link time, a feature which, if true, results in a link-time error being indicated. This is needed because language processors may be cognizant of error conditions that could arise but that are computable only at link time when linking with other modules is established.

The MUFOM standard is independent of the width of the word of the target computer. It deals with "minimum addressable units" of the target machine. This permits the linker to generate code for a variety of targets rather than for just one. It is not designed for achieving a maximum packing density. A subsequent binary packing transformation can be performed on the module if higher code density is desired. The linker does not need to know the width of the target machine to do, for example, library searching. The width of the target becomes significant only when the loader loads the code into memory.

Active participants in the 695 working group included Geoff Baldwin, Rick James, Greg Marshall (the final chairman), Jean Montuelle, Tom Pittman, Steve Savitzky, and Ian Willers.

754—Floating-point Arithmetic. Who would ever guess that floating-point arithmetic can raise the passions that it does?

When we started 754, our motivation was to try to define some formats so the chip houses would not give us the Big Endian/Little Endian byte ordering treatment all over again in the case of arithmetic. Dick Delp, then of AMD, chaired the working group at first. In the fall of 1977, John Palmer of Intel presented a proposal to the working group. This proposal was the original basis of the 8087 coprocessor chip. John kept mentioning a Professor Kahan of the University of California at Berkeley, whom Intel had retained as a consultant in preparing algorithms for the arithmetic processor. So in the spring of 1978 I called "Vel" and invited him to join the working group. And did he ever! With Professor Kahan came Jerry Coonen, a PhD candidate, and Professor Harold Stone, then visiting at Berkeley. The three fashioned a sophisticated proposal of their own that went well beyond the original Intel proposal. This became known to the world as the "KCS proposal."

The KCS proposal offered a procedure known as "gradual underflow" to maximize the accuracy obtainable in a given number of bits. This is particularly important when a limited field width, such as single-precision, 32-bit arithmetic, is employed. Other features introduced were user control of rounding direction, which makes it possible to do interval arithmetic efficiently, and the NaN (Not a Number). Interval arithmetic calculates exact upper and lower bounds for the true answer and thus accounts for any loss of precision due to round-off and truncation errors accumulating in the calculation. Despite user pressure, and the original raison d'être for the committee, the working group dropped any attempt to specify the absolute memory storage or the communication transfer format for binary floating-point numbers.

Now that there is an approved IEEE standard for floating-point arithmetic, David Hough, formerly of Apple and now of Sun Microsystems, tells me that he has a software implementation of the standard as it existed in 1980. This software provides a comparison for results and exceptions and thus is useful for debugging new implementations. This program is called IEEE Calculator, is written entirely in Pascal, and computes results a bit at a time. Hence, it is too slow for any use other than code debugging of implementations of the standard. This program has been placed in the public domain, and is unsupported. It is available on Usernet, the UNIX network. Questions should be directed to sun@dhg.

755—High-Level-Language Extensions for Microprocessors. The motivation for this effort was the fact that microprocessors are frequently used in direct hardware applications. But groups standardizing the high-level languages, particularly those controlled by CBEMA's X3 committee, had bent over backward to make it impossible to gain direct access to hardware in a high-level language. Verily it was joyous witnessing the passionate advocacies of some HLL gurus that bits are simply not made to be twiddled. And that is that. The result is that programmers are frequently forced to use assembly language when they perhaps would prefer an HLL. In my opinion, the upsurge of the C language in recent years has been at least partly the result of its recognition of assembly language capabilities that permit better modeling and use of actual microprocessor architecture.

Frequently the functionality needed by 755 can be provided by standard subroutines and functions rather than by changes to the language standards themselves. The principal extensions defined by 755, and their function or subroutine names, are

- memory access—PEEK and POKE,
- port and discrete I/O—INP and call OUT,
- interrupts—ARM and DISARM,
- calling of assembly language routines—CALLER, GETREG, and PUTREG,
- bit manipulation—IAND, IOR, IXOR, and INOT,
- subfield access—GETFLD and PUTFLD, and
- address binding for ROM vs. RAM—ROM and RAM.

Bruce Ravenel originally chaired 755. He resigned to cochair the 770 Pascal working group. Rick James then provided the leadership for 755 until its adoption. Tom Pittman made substantial contributions based on his continuing efforts in designing software for micro systems. Others involved included Wayne Fischer, Dick Karpinski, Dennis Paul, and yours truly, Smiling Bob.

885—Microprocessor Operating Systems Interface (MOSI). This standard was initiated in 1981 to provide a specification by which application programs can interface with operating systems, and by which operating systems can interface with the underlying hardware.

Reader Interest Survey

Indicate your interest in this department by circling the appropriate number on the Reader Interest Card.

High 182  Medium 183  Low 184