8086 LIMITATIONS WORKING GROUP REPORT

Chairman, Tony Hart

Attendees:

D. Howarth, D. Lee, T. Almy, M. Mistry, P. Bigelow, L. Szasz, D. Wood, D. Angel, D. Briggs, Y. Krol, J. Lundin

It was agreed that the 8086 was quite adequate for applications fitting within a 16-bit address space. Problems arise, however, as code size approaches the 64K limit.

One suggestion was to use a 32-bit version of Forth, but this would result in increased execution time. The possibility of a direct-threaded implementation was raised; this would use the intersegment subroutine instructions built into the 8086.

While still working within a 64K model, several ideas were considered such as:

1. Using RAM disks and overlaying code
2. Using memory management hardware to perform bank-switching
3. Separating code, data, and headers and placing them in separate segments
4. Using separate assembler-coded modules and accessing them via long calls or software interrupts

Another possibility was to have two or more independent Forth kernels executing in separate segments. In addition to allowing more code, this could be used as a method of multitasking.

The problem of addressing large areas of data was seen to be less severe. It is easy to implement extended address range operators. There is, however, currently no standard for such operators. Some expect a 32-bit linear pointer; others use a 16-bit segment and 16-bit offset (for which the order may vary). It was suggested that by using special defining words, it would be possible to hide much of this inconsistency.

It was also pointed out that it might be profitable to align more complex data structures, such as records, on segment boundaries. They would then be accessed with special operators which would require only a 16-bit segment pointer (the offset pointer defaulting to 0).

Although all of these ideas are interesting, most have been presented before and actually implemented in commercially-available products. No new breakthroughs came out of the working session.