Thad Smith III was very observant in his reading of my article—he makes some valid points and some not so valid points.

First let me point out that my article is planned to be one of a series of four and does not explain all of the details of the complete near-optimal instruction set design. This was not explicitly stated in the article.

Smith states, “The article does not adequately address the issue of discarding instructions with low static frequency. Such instructions are essential to the processor. I suspect that the MC6800 RTI instruction (return from interrupt) falls into the author’s low static usage set . . . .” The purpose of the static instruction frequency usage is to guide us in the design of an instruction set. The RTI instruction does not fall into the low static frequency group; since it modifies the program counter it is in Group II (program modification instruction group, with a 25 percent instruction usage). The RTI instruction is provided by the MOVE source 1, destination 1, source 2, destination 2, CC instruction, in which the instruction takes the form of MOVE top of stack to processor status register, and next top of stack to program counter, on condition.

Smith also makes the point, and rightfully so, that “When deciding whether a particular instruction should be implemented or not, the designer considers not only the potential static and dynamic usage of the instruction, but also the cost of implementing it vs. the cost of not implementing it.” This is a very valid point, but I am not sure that most computer engineers apply this criterion. It is hard to believe that much is known about how microprocessors are used (or computers in general, for that matter), when 8.7 percent to 30.3 percent of the instructions provided are never used. This was pointed out in my article in Table 1. I believe the methods mentioned in my article, when used to include or exclude instructions, are valid and in time will prove to be a worthwhile design methodology.

Smith also points out that the DAA would be expensive to implement in macrocode. There is no evidence to support the requirement that the DAA should be implemented at all. Shustek (Reference 17 in my article) states that “it would probably be better to eliminate directly computing decimal arithmetic and do the conversions when necessary.” The research for my article shows that DAA instructions or their equivalent were used 1.82 percent of the time in the MOS6502, that BCD instructions were used less than 0.1 percent of the time in the MC68000, and that the TMS9900 doesn’t even have DAA or BCD arithmetic instructions.

“More critical to users are the useful instructions that are omitted, not because of lack of opcode space, but because of oversight or misjudged impact,” Smith notes. I think this point is exactly the problem that my article tried to address. We cannot solve a problem if we do not know what the problem is! My article, I hope, suggested ways to ensure that such “oversights” and “misjudged impact” will not become part of the design. My design did not omit the PUSH X instruction, as was the case in the MC6800. The PUSH X instruction may be implemented with a MOVE to top of stack instruction.

Quoting Smith again, “Although several minicomputers and some microprocessors initially had ad hoc instruction set designs, the latest offerings show careful consideration of

---

### Z-80® and 8086 FORTH

**FORTH Application Development Systems**

- including interper-compiler with virtual memory management, assembler, full screen editor, line editor, decompiler, demonstration programs, and utilities. Standard random access disk files used for screen storage. Extensions provided for access to all operating system functions. 120 page manual.
- Z-80 FORTH for CP/M-2.2 or MP/M ........................ $ 50.00
- 8086 FORTH for CP/M-86 ................................ $100.00
- PC/FORTH for IBM® Personal Computer .......................... $100.00

**Floating point extensions** for above systems. Specify software floating point, AMD 8511, AMD 9512, or Intel 8087 support ................................ additional $100.00

**Nautilus Cross Compiler** systems allow you to modify the FORTH nucleus, recompile for a different target computer, generate headerless and/or ROMable code. Supports forward referencing. Produces load map, list of unresolved symbols, and executable image in RAM or disk file. 100 page manual. (Prerequisite: one of the application development packages above for your host system)
- Z-80 host: 8080 or Z-90 target ............................. $200.00
- Z-80 host: 8080, Z-80 or 8086 target ........................ $300.00
- 8086 host: 8080, Z-80 or 8086 target ........................ $300.00
- PC/FORTH distributed on 5¼ inch soft sectored double density diskettes. All other software distributed on 8 inch soft sectored single density diskettes. Prices include shipping by UPS or first class mail within USA and Canada. California residents add sales tax. Z-80 is a trademark of Zilog, Inc. IBM is a trademark of International Business Machines Corp. CP/M is a trademark of Digital Research, Inc.

---

**Laboratory Microsystems**

4147 Beethoven Street
Los Angeles, CA 90068
(213) 306-7412

Reader Service Number 3