shifts and adds are not, coefficients which can be implemented by mere shifts and adds may be more attractive. I thank you for your interest in this matter and hope that I am able to substantiate your findings soon.

Michael Andrews
Colorado State University

More on instruction sets
Editor:

Let me see if I’ve got this straight: Dennis Fairclough, in his article on microprocessor instruction sets (May 1982) and in his subsequent reply to Thad Smith III, would replace the 6800 instruction RTI with “MOVE source 1, destination 1, source 2, destination 2, condition code.” And this is supposed to be a simplification?

When I first saw Fairclough’s article, I felt he was definitely on the right track — perhaps because his data confirms a long-held suspicion of my own, that computer manufacturers tend to give us too cotton-pickin’ many instructions. The familiar “embarrassment of riches” syndrome. The efforts to instrument actual code and use the results to synthesize a simpler instruction set are to be greatly applauded. Where Fairclough and I part company is where he begins to assign mnemonics to the set.

Perhaps we should ask ourselves what the goal is we seek. Is it to make the instruction set of a real microprocessor simpler, easier to use, more regular, and more efficient to mechanize? (All worthy goals, and definitely in the spirit of the May article.) Or is it simply to try to express an instruction set with a minimum number of mnemonics? I submit that the latter goal is a mere parlor game and has nothing to do with Fairclough’s original direction. If that is the goal, though, we can probably think of many even smaller sets. I’m rather partial to the universal “TEST bit 1, set bit 2, clear bit 3, branch on condition,” although traditionalists may prefer a Turing machine.

When Zilog introduced the Z-80 as an extension of the 8080, they also changed the instruction mnemonics. Their set was applauded by many because of its regularity and its rational extension to additional addressing modes. In using it, however, I and many others have found it not to be so superior. The reason is simple: Of the 158 Z-80 mnemonics, 39 of them begin with “I.D.” The information content of these mnemonics is practically nil. In practice, one has to look at the form of the arguments to determine what instruction is really being called for. I have no doubt that Dennis Fairclough’s MOVE instruction suffers from the same malady.

So please, Dennis, continue your work to define a simple and regular instruction set. Let the op codes for MOVE, CALL, RETURN, PUSH, etc., be similar if you like. But don’t try to extend this regularity to the point of hiding the true function of the instruction from the programmer. There is no place in the real world for an instruction set that doesn’t include CALL, RET, PUSH, and POP.

While we’re on the subject, I’d like to suggest mnemonics that are even more meaningful to the programmer, even at risk of having a larger set of mnemonics. These include GOTO for an unconditional branch, IF for a conditional branch, LOAD for a memory-to-register move, STORE for register-to-memory, and MOVE (or COPY) for a register-to-register move. Form follows function. I’ll bet I can map them onto the Fairclough set of operations.

Jack W. Crenshaw
Tampa, Florida

Errors detected in CAPS programmer’s card
Editor:

The ASCII code table in the lower right corner of page 23 of the August 1982 issue contains several errors which should not have been passed on, even if the originator of the table had them wrong:

5E is a circumflex, not an up-arrow;
5F is an underline, not a left-arrow;
60 is a grave accent (reverse quote);
2E is a period, not a bullet; and
1F is US (unit separator) not VS (the legend, however, is correct).

It would be nice if those quoting ANSI standards would do so correctly.

Michael D. Shapiro
San Diego, California

The material in question was reprinted from a copyrighted programmer’s reference card. IEEE Micro cannot alter such previously published material. However, we thank the writer for bringing this to our attention.

—Ed.

IEEE MICRO