Microprocessor Architecture

Guest Editor's Introduction
Frederick W. Clegg
Hewlett-Packard Corporation

This marks the second issue of Computer devoted to the Asilomar Workshop on Microprocessors. As with the first such issue (January 1976), no attempt has been made to cover the workshop in toto (a summary report appeared in the July 1976 Computer). Rather, the goal has been to spotlight a few of the more interesting developments and philosophies that were explored during the April 1976 workshop.

Many of the salient points touched on in the first two sessions are covered in the first paper, "Advances in Microcomputer Development Systems," by Larry Krummel and Gaymond Schultz of Ramtek. This paper draws a distinction between "microcomputer systems" as direct descendants of minicomputers, and highly dedicated "user-packaged microprocessor systems." Different constraints inherent in each are examined, and the highly hardware-oriented, lower-language-level coding often characterizing the latter types of systems is stressed. (Its authors are among the pioneer developers of the technique of in-circuit emulation, so forgive them their bias, if any, in favor of that approach.)

Some of the constraining effects imposed on microprocessor architectures by the state of semiconductor technology, on the one hand, and the state of maturity of the software art, on the other, are examined by Bernard Peuto (Zilog, Inc.) and Leonard Shustek (Stanford Linear Accelerator Center) in "Current Issues in the Architecture of Microprocessors." One of the novel contributions of this work is its inclusion of experimental data on how microprocessors are used. Some startling parallels with much larger computer systems are observed (perhaps suggesting that microcomputers are not so unique as we workshop steering committee members would have led you to believe!).

"Our third paper is somewhat tutorial in nature. In "Microprocessor Input/Output Architecture," John Wakerly surveys a number of the different approaches to microprocessor I/O and the various types of specialized LSI I/O support chips now available. This is a subject area presently confused by each manufacturer's differing nomenclature for the various common functions to be performed and the parts supplied to perform them. John's paper should do much to provide an orderly perspective.

In the concluding article, consultant Dennis Allison proposes "A Design Philosophy for Microcomputer Architectures" which is significantly different from any philosophy apparent in today's microprocessor marketplace. (Allison would surely argue that his philosophy is a saner one as well—although many readers will doubtless challenge some of his positions.)

Those readers of this issue who shared our good fortune to attend the 1976 Asilomar Workshop on Microprocessors may opine that both the workshop and this issue of Computer have raised more questions than they have answered. Even if this be the case, a worthwhile function has still been filled, for there can be little question that the impact of LSI microprocessors on modern technology will ultimately prove as great as that of the transistor and the monolithic integrated circuit. There is reason to suspect, however, that the microprocessor revolu-
tion may end up being one of the most painful and difficult technological revolutions in recent history. This is because many of those who will ultimately cause the potentials of microprocessors to be fulfilled come from backgrounds which only poorly prepare them to do their jobs effectively.

When the first transistors became available, circuit designers invented numerous analogs between these new "active devices" and their vacuum tube forerunners. It was through a gradual evolutionary process that configurations unique to transistors and having no vacuum tube counterparts (such as amplifiers employing complementary pairs of transistors) came about. Similarly, the first integrated logic circuits caused little consternation. The engineers called upon to use them had already developed a host of tools and techniques applicable to them in the contexts of earlier, discrete component logic circuits. Only gradually did techniques emerge that were unique to integrated circuits (such as the biasing of inverters into odd regions so that they could be used as amplifiers or oscillators).

We should not be surprised by the fact that learning to use microprocessors effectively is proving more difficult than learning to use transistors or integrated circuits. After all, microprocessors are functionally much more complex than these earlier innovations. This is only a small part of the story, however. I (and many others) believe there is a much more fundamental element complicating the microprocessor learning curve: software. Even if most microprocessor users were veterans at developing, maintaining, and managing software, there is ample evidence to convince us that the major step in learning to use any programmable processor effectively consists in our learning to program it effectively. The situation is much worse than that, however. In an appallingly large number of instances, those creating software for microprocessors are not veterans but rather complete neophytes at the software game. This is not too surprising. As has been observed by several authors, more often than not it is necessary for the user of microprocessors to possess at least a modicum of competence in both the disciplines of electrical engineering and programming, and it appears easier to make an electrical engineer at least minimally competent at programming than it is to make a programmer minimally competent in electrical engineering. This may serve as at least a partial explanation of why we seem to be witnessing a second re-run of all the old arguments over high-level versus low-level programming, tightness of code, efficiency, and the like, which first took place in the 1950's. The first re-run, of course, was occasioned by the arrival of minicomputers in the mid 1960's. The scenario was similar to that of today: a whole new class of users was "discovering" the stored-program computer and how to use it. Perhaps the participants felt that the drastic differences in cost and size between the minis and the big machines meant that experiences gained with the maxis had minimal relevance in the mini world . . . or perhaps they were too lazy or too hurried to ask what wisdom earlier experiences with maxisystems might contain. It is reasonable to expect, of course, that the present second re-run of this spectacle will be far grander than the first simply because, owing to the phenomenal cheapness of the microprocessor, the cast of characters involved will be vastly greater.

These thorny issues were among the most central ones discussed at the 1976 Asilomar Workshop. The success of this workshop and its predecessor in 1975 has led to hopes that this series may become a regular annual feature of the activities of the IEEE Computer Society's Western Area Committee, in much the same way that the Lake Arrowhead Workshops, also sponsored by the committee, have become a popular yearly event. The 1977 Asilomar Workshop, tentatively entitled "Microprocessors in Perspective," is scheduled for 20-22 April 1977. It will attempt to assess the most current architecture and technology of, and software and design techniques for, microprocessors. Additionally, new applications and educational and societal impacts will be explored. Further details on the planned program may be found in the announcement contained in the Update section of this issue of Computer.

Frederick W. Clegg is a project manager for Hewlett-Packard in Cupertino, California. Earlier he was a member of the electrical engineering faculty at Stanford University and, subsequently, Santa Clara University. At the latter institution he led the development of a comprehensive curriculum in digital systems and computer science.

His major professional interests lie in computer reliability, digital systems development and test equipment, and microprocessor system hardware and software. He has published several papers in these areas.

Clegg received the BSEE (magna cum laude) from Oakland University in Rochester, Michigan, and the MSEE and PhD in electrical engineering from Stanford University. He is a member of Eta Kappa Nu, Tau Beta Pi, Sigma Xi, the IEEE, and the IEEE Computer Society.