Foreword

by Maurice V. Wilkes
Cambridge, England

This is an excellent time for a new book on computer design. During the last ten years the subject has undergone a marked renaissance. It has become less dependent on intuition and personal opinion, and more on measurement and rational analysis. The subtitle of the author's earlier book *Computer Architecture: A Quantitative Approach* makes this point.

Patterson and Hennessy played their part in these developments. Indeed, widespread public discussion of the new ideas may be said to have begun with the publication in 1980 of a paper by Patterson and Ditzel entitled "The Case for the Reduced Instruction Set Computer." The acronym RISC derives from this paper. Patterson proceeded to put RISC ideas into practice by developing the Berkeley RISC with the aid of a group of students. This design became the basis of the SPARC workstations. Hennessy applied his energies to the MIPS design project at Stanford and became one of the founders of the MIPS Computer company.

Like many seminal ideas, RISC is deceptively simple. The emphasis is not on the size of the instruction set, but on its nature; RISC instruction sets have made it possible to apply, in a single chip processor, subtle techniques for instruction level concurrency that were previously to be found only in large computers.

There is a lot more to computer design than is comprised in the RISC philosophy, as will be shown by a glance at the diagram entitled "The Five Classic Components of a Computer" which appears, with differing highlighting, at the head of various chapters of this book. But RISC has been a unifying influence. Another major unifying influence has been the need to work within the boundaries of a silicon chip, where there is never enough space and, if one thing goes in, another must go out. Performance depends critically on decisions taken at the chip level. No longer can system design be a subject divorced from computer implementation, a change which may have left some people high and dry, but which is nevertheless wholly to the good. Nor can system design be divorced from consideration of the software interface, a point which the authors bring out both in their subtitle and in their text.

The computer field, particularly on the software side, abounds with examples of new ideas that have found their way into industrial practice by being taught in universities and have thereby become part of the professional tool kit

which graduating students have carried with them into industry. In hardware, it is often the other way round; teaching follows practice. This is another reason for welcoming a book by two engineers who can write of current practice with authority.

As I followed the authors through their unhurried chapters, I was conscious of the all-seeing eye of the evaluator pictured at the chapter heads. Ever watchful, she—I think it is a female eye, but I cannot be sure—seemed to be saying that every argument has another side, and that every insight gains by being put into perspective.

I think students will enjoy learning from this book; at least, I hope so and I give them my good wishes.