The computer revolution is most often graphically depicted by the increased density in integrated circuits. It is indeed spectacular that the number of transistors per chip has increased by an order of magnitude every 5 years and may reach 1,000,000 by 1985. This has brought the dramatic reduction in size and price. Computers have been applied to new activities and greater function has been added to computers. The so-called minicomputers have increased the typical main memory capacity more than an order of magnitude every 5 years for the past 10 years.

The trend is clear that, whether the computer is imbedded in another product or used as a computer, the amount of software is increasing rapidly. To be sure, electrical engineers will need to know much more about software, and testing/checkout/diagnostics for both hardware and software. But, the vast amounts of logical functions available to the electrical engineer make his/her world take on new enlarged dimensions. First, the engineer should be looking at a broader, total solution to problems. It is now possible to have a significant impact in improving the way people do things. Second, due to the size and complexity of tasks that can be built into computers, groups of people must coordinate their designs and simplify interconnections. Third, the interface to the user must be greatly improved. People should be able to solve their problems without facing the barriers of the computer system.

Hence, it is more important to learn about problem formulation and solutions than too much focus on current building blocks or technology.

E. DAVID CROCKETT
Hewlett-Packard Company
General Systems Division
Santa Clara, CA 95050

THE IMPACT OF VLSI ON COMPUTER SCIENCE EDUCATION

The new world of computation done in the LSI medium is characterized by some good news and some very bad news.

The good news is that now logic and memory are made of the same stuff; implemented in a uniform technology. Vast amounts of concurrent computation can be achieved by mixing computational elements in with memory elements, not separating them artificially as is done in Von Neumann machines. This approach to computation has the potential for orders of magnitude increase in performance and decrease in the energy, area, and time required for computation.

The bad news is that Computer Science is going to have to completely change its way of doing business to take advantage of the properties of the stuff out of which the brave new world of computers will be constructed. In the new medium, moving data around consumes most of the time, area and energy in a system. Logical operations when done locally are virtually free. This property assures that topology and topological properties of computations will never be hidden from the user of computation, even at the highest level. No longer is there a clear distinction between hardware and software. From now on, programming will consist of the act of mapping a computation with certain topological properties on to a computational structure with a different set of topological properties.

We are no longer blessed (or cursed) with artificial levels of packaging to provide de facto levels of organization and interconnect. Topological constraints are present at all levels of design. No memory is truly random access. Some parts are always closer, and therefore faster, and requiring less energy per access than others. The Von Neumann style no longer provides us a guide. We are on our own. While it may be difficult for machine designers to cope with the new world of VLSI, it will be much more difficult for conventional programmers to make this transition. The theory of analysis of algorithms and complexity of computation will require major revisions. Rather than a de facto mapping into a single computation stream, complexity theory is faced with a topological mapping between the properties of the algorithm and the properties of the computing structure. In this mapping, the preservation of geometric locality is richly rewarded. However, today we have no good mathematical measure of what locality means or how it can be expressed or measured. The enormous degree of concurrency which is there for the asking can only be used by algorithms of a fundamentally different sort than have been developed historically. Many algorithms for sequential machines destroy the locality which is inherent in the problem in order to save total computational steps. The "fast" Fourier transform and quicksort algorithms achieve a small number of operations at the expense of shipping data globally on a massive scale. Traditional computer languages not only have the awkward problems so eloquently described by Backus,[1] they are unable to represent computations which involve a very large number of things happening concurrently. The optimizing compiler of the future must be able to decompose problems into a vast number of independent processes and map these processes on to some computing structure with a given topological configuration.

Arising out of this brave new world are three new disciplines fundamentally different from those which are attacked in current computer science curricula.

(1) The design of machines, algorithms and notations for very large degrees of concurrency.
(2) A complexity theory for algorithms, computing structures, and the mapping between them.
(3) An amorphous area which I call the physics of computation.

The third area involves lower bounds on the time, area and energy required for any given computation. It includes primitives from information theory, computational complexity and thermodynamics. The combination of the last two areas should, in the not too distant future, be able to express a lower limit on the complexity of a computation by the unavoidable energy which is necessary for its switching and communications requirements.

All major academic disciplines are faced with periodic major revolutions in their content. Physics has been through several by now and there may be another in the wings. Computer Science has been able to exist for nearly half a century without a major revolution in its approach or the contents of its curricula. We should welcome the challenge of a major revolution in this area posed by the new technology.

REFERENCES

[1]Backus, J. Can programming be liberated from the von Neumann style? A functional style and its algebra of programs. *Communications of the ACM* (August 1978); Volume 21, Number 8.

CARVER A. MEAD
Department of Computer Science
California Institute of Technology
Pasadena, CA 91109