INTRODUCTION

The introduction of the digital computer into the academic environment created a situation that would lead to impact on how teaching and research are conducted at virtually all levels, and in virtually all disciplines. The further combined evolution of semiconductor technology with software practice and methodology has resulted in a new field which has developed at an astounding rate. The impact of this rapid change in the academic environment has forced universities to modify their classic attitude of slow change on curricula and research matters to yield to external demands for education and research in computer related topics.

Though the implication of the computer has been present in all areas of the academic institution, this paper will be directed to education about the computer; that is, computer science and computer engineering. Within these areas the changes that have occurred can in some way be observed by the changes that have occurred in pedagogical hardware. Initially, most programs in these areas relied on a "large" central computer resource of the institution, and perhaps, a modest logic design laboratory. With their introduction, minicomputers were acquired in increasing numbers and it became common for a major department to have its own system. More recently, low priced microprocessor systems are placing significant computer power in the hands of almost any department.

An equally significant development went on in the classroom as in the laboratory. Initially, programming classes were presented often by the computer center staff. These quickly moved into more academic settings within mathematics and electrical engineering departments. From this point options and minors developed. These, in turn, developed into degree offering departments, which included in their offerings the tool courses but also moved up to more abstract courses, such as switching and automata theory, computability and formal languages.

* On leave from the University of South Florida, Tampa, Fla.

In the sections that follow, the development of the academic concepts of this area of education will be considered, with primary emphasis on the more recent developments.

THE DEVELOPMENT OF CURRICULUM IN COMPUTER SCIENCE

The single most significant document in computer science education was the report of the Curriculum Committee on Computer Science (C3S) of the Association for Computing Machinery, most commonly known as "Curriculum '68." The report gives detailed specifications for twenty-two courses fundamental to the study of computer science, as well as putting these courses into a prerequisite structure that also involves mathematics.

While the questions of definition and justification of computer science are not addressed in the report, the subject matter is classified into three areas of computer science and two related areas. A core program is defined, and based on this core area the undergraduate program is detailed involving computer science courses, mathematics courses, and technical electives.

Graduate degree programs are mentioned but not detailed to the extent of the undergraduate program. The master's program is outlined in terms of areas while doctoral programs are only mentioned in general terms, with ideas put forth for further work. Additional topics which received some attention in the report included service courses and minors, continuing education programs, program implementation problems, and facilities.

"Curriculum '68" may be seen as an expansion and refinement of the 1965 report of C3S. This earlier report spent a great deal more time on the definition of computer science as a discipline, and its relationship to other disciplines, especially mathematics.

The 1965 report of C3S was, in turn, an outgrowth of a panel presentation at the 1963 ACM national meeting that was reported in 1964 in The Communications of the ACM.
by Perlis, Arden, Forsythe, Korfhage, Gorn, Muller, Keenan, and Atchison and Hamblen. In these reports one finds a definition and specification of computer science; courses and program description emphasizing the first course, numerical analysis, logic and logic design; and descriptions on programs then existing in the United States. Within the courses and program descriptions emphasis was placed on the role of programming projects and exercises as an integral portion of the programs, and the close ties of computer science education to mathematics, at that time, was noted.

COMPUTER ENGINEERING

While the more theoretical aspects of computing machinery was being addressed by C3S, the improvement in economics in the computer industry opened new areas in logic design. Not only was computer design the main objective of studying combinatorial and sequential circuits, but also new areas in process control, data communications, systems analysis and simulation required this knowledge. A trend was initiated in colleges of engineering and particularly among electrical engineering departments to create the discipline of Computer Engineering.

In September 1967 the COSINE Committee of the Commission on Engineering Education published recommendations for an undergraduate course program. Though originally designed as an enhancement of an electrical engineering curriculum this report was, in effect, the initial development of the area of computer engineering.

The report presents a series of subject areas in a computer science program in electrical engineering consisting of basic subject areas including programming principles, computation structures, discrete mathematics, and machines, languages and algorithms; and recommended elective areas including digital devices and circuits, switching theory and logical design, programming systems, numerical methods, optimization techniques, circuit and system theory, information theory and coding, functional analysis, combinatorics and applications, probability and statistics and symbol manipulation and heuristic programming.

The COSINE Committee did not regard its mission as the same as C3S and did not equate electrical engineering and computer science. Nonetheless, it is of note that much of the material recommended parallels that of "Curriculum '68." The main differences appear in the advanced areas, and in a lack of a course or area corresponding directly to compiler construction. The core of the two programs are virtually the same, however, the work reported on by the COSINE Committee does not place the area of data structures in the fundamental position given by C3S.

The COSINE Committee issued eight other reports dealing with the first course in electrical engineering, computer organization, digital subsystems, industries' reaction, a computer engineering option, digital system laboratory courses, operating systems, and the minicomputer in laboratory programs. Perhaps the lack of a stable operating organizational base such as a professional society prevented the COSINE reports from having the broad impact of the more integrated report of C3S. An effort following in the steps of the COSINE Committee has been the DISE Committee, funded by the National Science Foundation, and with some of the same problems. It should be noted, however, that the efforts of the DISE project are more directed at dissemination of materials than at the definition of programs.

CURRICULUM DEVELOPMENT SINCE "CURRICULUM '68" AND THE COSINE REPORT

Subsequent to the publication of "Curriculum '68" and the COSINE Report, considerable work has gone on relating to computer science education. A significant development in this period was the formation of the ACM Special Interest Group on Computer Science Education (SIGCSE), which provides a continuing organization for the presentation and exchange of ideas in the field.

C3S has continued its activities following the publication of "Curriculum '68." Under the sponsorship of this Committee, a series on doctoral programs was published, a summer institute program for smaller colleges was conducted, course recommendations for smaller schools were prepared, and guidelines for masters programs were prepared.

Additional curriculum work has been conducted within ACM by the Curriculum Committee on Computer Education for Management (C3EM). This Committee has prepared guidelines for both graduate programs and undergraduate programs in information systems which integrates materials associated with computer science with materials associated with business.

As was indicated in the previous section, the COSINE Committee was active in this period, and in addition to the publication cited, the Committee also conducted a series of workshops to assist in the interpretation of their work.

The Committee on the Undergraduate Program in Mathematics (CUPM), also published recommendations in the area of computational mathematics during the period. This work represented an extension of earlier work describing a mathematics program with work in computing.

In addition to these efforts, the literature of computer science education contains numerous references to specific courses and topics providing an expansion of the course material given in the curriculum studies, and presenting something of a dynamic updating of the recommendations. The more recent work of this type emphasizes the area of software engineering and has been addressed in a recent volume. A comprehensive survey of the post "Curriculum '68" literature in computer science education has been published, and thus will not be further addressed at this time.

NEW CURRICULUM RECOMMENDATIONS IN COMPUTER SCIENCE AND ENGINEERING

The Education Committee of the IEEE Computer Society has been meeting regularly for over two years to design a
curriculum which may serve as a model for both the computer science and computer engineering approaches. This work led to the publication of a report of the Model Curricula Subcommittee of the Education Committee in February 1977, although a preliminary report on the work had appeared earlier.

Two aspects of this report should be taken into account for its correct interpretation:

1. The total curriculum as presented probably exceeds the resources or time allocations of most undergraduate programs and, as a result, needs to be adapted to regional needs, local resources, and departmental orientation.

2. A central “core” of subject matter is outlined that is considered the kernel of an undergraduate program in computer science and engineering.

There is no pretention that the report is either exhaustive, complete or definitive. It is claimed that it does represent the views of representative segments of the educational community acting in a committee mode with suitable compromises.

The general distribution of the recommended course modules is divided into five areas as follows:

(a) Digital Logic—5 courses
(b) Computer Organization and Architecture—5 courses
(c) Software Engineering—8 courses
(d) Theory of Computing—4 courses
(e) Laboratories—7 courses

The titles of these courses are given in Figure 1. For catalog descriptions and more detailed information, the interested reader is referred to the full report referenced above.

It should be noted that from the number of courses, the software engineering area is the predominant area in the curriculum, thus, giving the curriculum a pragmatic and applied flavor. In the theory of computing area, a modest amount of course work is recommended which presents a number of very abstract principles in an introductory fashion.

Of particular note is the laboratory sequence. Most laboratories in present computer science and engineering programs, whether they are a part of a given course or a separate course, deal with a specific subset of hardware or

---

**Figure 1**—The courses of the IEEE/Computer Society curriculum recommendations
software problems. Therefore, a laboratory sequence should allow a student to become familiar with current technology to apply theory and to solve problems. Thus, the main objective is to expose the student to methods applicable to the "real world," to enhance creativity, and to develop problem solving skills. In the process, the student needs to learn a variety of techniques for designing, implementing, debugging, and maintaining a project.

The approach suggested here has a two-dimension organization.

1. **Vertically**, the labs are organized in a series of graded courses in which the student advances gradually from well-defined experiments to more sophisticated projects.

2. **Horizontally**, the labs are used as media to apply hardware and software principles in an integrated form.

The value of such experience becomes clear when the student realizes that the solution of a hardware design problem requires considering its software implications and vice versa.

The prerequisite structure for the program is given in Figure 2 which indicates the positioning of both the courses and the laboratories.

**THE NEW RECOMMENDATIONS OF THE CURRICULUM COMMITTEE ON COMPUTER SCIENCE (C3S) OF THE ACM**

During the same period that the Computer Society was developing recommendations, C3S was preparing an update and modification to "Curriculum '68." This appeared in draft form in June, 1977. Although it will not be covered in this paper, it should also be noted that a subcommittee of C3S prepared draft recommendations for a program at the two year level at the same time. These reports were published in this way as working papers to obtain maximum input regarding their content prior to publication of the final committee reports which is anticipated in 1978.

The C3S report is broken down into the general areas of the core curriculum, Computer Science electives, the undergraduate program, service courses and other considerations including facilities, staff and articulation.

The core material is considered to be at both the elementary and intermediate level. The material at the elementary level is first presented as a list of topics in the areas of programming, software organization, hardware organization and data structures and file processing. It is anticipated that many implementations are possible from this listing of topics, and it is emphasized that programming projects should be stressed throughout the course sequence. As a guide to implementation, but certainly not the only implementation, a five course sequence is proposed at this level: CS1-Computer Programming I, CS2-Computer Programming II, CS3-Assembly Language Programming, CS4-Introduction to Computer Organization, and CS5-Introduction to File Processing. Three intermediate level courses are then proposed to complete the core or that material required to provide minimum requirements common to all computer science undergraduate programs. They are CS6-Operating Systems and Computer Architecture I, CS7-Data Structures and Algorithm Analysis, and CS8-Organization of Programming Languages.

Following the core material illustrated in Figure 3, two intermediate level electives are proposed, as are nine advanced courses, and provisions for special topics courses. The course prerequisite structure in the report is given in Figure 3, and the interested reader is referred to the full report for more detailed information.

It should be particularly noted that the intermediate level core courses are recommended to contain a strong emphasis of fundamental concepts exemplified by various types of programming languages, architecture and operating systems, and data structures. Neither theoretical treatments nor case study approaches in and of themselves are adequate or appropriate at this level. Advanced level courses may be used for such treatments. Special topics courses are suggested and it should also be noted that which of the special topics are offered will be dependent on the resources of the department, however, it certainly is the case that in time, some of the material listed under this category, should be integrated into the courses more fully specified, or replace entire courses in the curriculum. Monitoring this phase of the program will be a continuing activity of C3S.

Tied into the prerequisite structure of the program are six mathematics courses:

- MA 1 Introductory Calculus
- MA 2 Mathematical Analysis I
- MA 3 Linear Algebra
- MA 4 Discrete Structures
- MA 5 Mathematical Analysis II
- MA 6 Probability and Statistics

Descriptions of the courses MA1, MA2, MA3, MA5 and MA6 may be found in the 1965 CUPM recommendations. MA4 represents a more advanced course in discrete structures than that given in "Curriculum '68," and builds on concepts developed in the study of calculus and linear algebra, emphasizing applications of discrete structures to computer science. The complete prerequisite structure is given in Figure 4 which includes all the courses mentioned, except CS 10, Computer and Society, and the Special Topics courses whose prerequisites will vary depending on circumstances.

The major will normally consist of the eight courses of the core material plus four additional courses selected from the recommended computer science intermediate and advanced electives with no more than two from any one specific subfield of the discipline. It is further recommended that the student take mathematics courses MA1, MA2, MA3, and MA4, and depending on computer science electives selected, MA5 and MA6 may be required.

As has been indicated, further details are available to the reader in the report, and in fact, this paper has only addressed those portions of the report dealing with the under-
Figure 2—The prerequisite structure of the IEEE/Computer Society laboratory sequence
CONCLUSIONS, OUTLOOK AND PROJECTIONS

As has been shown in this paper, computer science and computer engineering have grown considerably in the past ten years. It would appear significant that it took ten years from the publication of the COSINE recommendations in 1967 to the publication of the model curricula of the Computer Society, while it also took ten years from the publication of “Curriculum ’68” to the final version of the report updating that document.

What is perhaps most significant, however, is the convergence that can be seen occurring between the groups. Though working almost in total independence of each other, the two committees prepared guidelines remarkably similar in philosophy and content.

This similarity may be best seen in considering the core material of the two programs; CS1 Computer Programming I is similar to SE-1 Introduction to Computing, CS2 Computer Programming II is similar to SE-2 Data Structures I, CS3 Assembly Language Programming and CS4 Introduction to Computer Organization cover much the same material as CO-1 Introduction to Computer Organizations, CS5 Introduction to File Processing is similar to SE-3 Data Structures II, CS6 Operating Systems and Computer Architecture I is similar to SE-6 Operating Systems and Computer Architecture I, CS7 Data Structures and Algorithm Analysis covers much the same material as TC-2 Design and Analysis of Algorithms and SE-5 Data Base Systems, and CS8 Organization of Programming Languages is similar to SE-4 Programming Languages.

As one would expect, there are differences in the recommendations. The recommended core of the Computer Society program more strongly emphasizes the hardware and engineering areas, while those of CS more strongly emphasize software and theoretical areas. But while these differences exist, the similarities strongly point to the fact that the discipline of computer science and engineering is approaching a well defined state.

It is fortunate that the two committees produced their reports at essentially the same time. This gives the educator the opportunity to study both documents, and select those parts that best fit his institutional needs in constructing a
program, for it must be emphasized that these are only guidelines.

Coursework may be outlined on paper, however, more critical questions of feasibility of implementation in a variety of settings exists. While it is not anticipated that there will be complete agreement on courses and curriculum design on the part of all interested parties, by continued interchange of ideas, at least a consensus of the critical issues to be covered may take place. To foster this interchange, both committees welcome input from all interested parties.

REFERENCES

24. Committee on the Undergraduate Program in Mathematics (CUPM), Recommendations for an Undergraduate Program in Computational Mathematics, CUPM, Berkeley, Calif. 1971.
32. Committee on the Undergraduate Program in Mathematics (CUPM), A General Curriculum in Mathematics for Colleges, CUPM, Berkeley, Calif. 1965.