COMPUTING CURRICULA GUIDELINES
FOR
ASSOCIATE-DEGREE PROGRAMS

IN

COMPUTING AND ENGINEERING TECHNOLOGY

TWO-YEAR COLLEGE
COMPUTING CURRICULA TASK FORCE

Computing and Engineering Technology Committee

David Hata (Chair)	Portland Community College
Thomas Bingham	St. Louis Community College
John Debo	University of Central Florida
Warren Hill	Weber State University
John Impagliazzo	Hofstra University

THE ASSOCIATION FOR COMPUTING MACHINERY

1993

ACM Two-Year College Education Committee
and
Task Force Steering Committee

Karl Klee (Chair)	Jamestown Community College
Richard Austing	University of Maryland
Helene Chlopan	University of Kentucky
John Impagliazzo	Hofstra University
Joyce Currie Little	Towson State University

Association for Computing Machinery, Inc.
1515 Broadway, 17th Floor
New York, New York 10036
(212) 869-7440

GUIDELINES FOR
COMPUTING AND ENGINEERING TECHNOLOGY

TABLE OF CONTENTS

PART I
CURRICULA STRUCTURE

PART II
CURRICULA KNOWLEDGE UNIT DETAILS

PART III
CURRICULA IMPLEMENTATION SAMPLES

PART IV
SUPPORTIVE INFORMATION

LIST OF FIGURES

LIST OF TABLES

PART I

CURRICULA STRUCTURE

1.0 INTRODUCTION AND CHARTER

The development of curriculum guidelines for computing in engineering technology education by professional organizations dates back to the early 1960's. Prior to that time, the major source of computer education and training was the computer manufacturer. In 1968 and 1978 the Association for Computing Machinery (ACM) published recommendations for an undergraduate program in computer science [1, 2]. Since then, several sets of curriculum guidelines have been published by the Computer Society of the Institute of Electrical and Electronic Engineers (IEEE/CS), the ACM, and other professional organizations.

Curriculum studies have generally focused on four-year programs. However, the IEEE Computer Society did support a project to study two-year programs in computer systems technology. The 1982 report entitled *Recommendations and Guidelines for Associate Degree Programs in Computer Systems Technology* [3] examined entry-level requirements for computer systems technicians and proposed a two-year curriculum. The following report represents an extension of the 1982 IEEE Computer Society report. The scope of the report has been expanded to encompass several engineering technology curricula related to computing. The report also includes elements of required knowledge and sample curricula for each program type.

In 1990 the Association for Computing Machinery (ACM) awarded the ACM Two-Year College Computing Curricula Task Force preliminary funding to develop curricular guidelines for two-year computing programs. The Task Force identified five curricula areas:

Computer Support Services (CSS)
Computing and Engineering Technology (CET)
Computing for Information Processing (CIP)
Computing Sciences (CS)
Computing for Other Disciplines (COD)

The curricula area of computing and engineering technology is the subject of this

report.

The intent of these guidelines is to serve the needs of academic institutions (two-year colleges, four-year colleges and universities) offering two-year programs leading to an associate degree. Programs offered under the rubric of computing and engineering technology should find these guidelines particularly helpful in developing competent graduates to enter the industrial community, or to continue their academic pursuits toward a four-year, baccalaureate degree. The guidelines are based in part upon *Computing Curricula 1991* [4], a publication sponsored jointly by ACM and the Computer Society of the IEEE, which presents curricular guidelines for four-year college programs in computer science and computer engineering. Computing technology curricula for two-year programs are addressed in this publication. For this reason, the guidelines presented herein are both timely and unique.

2.0 GOALS AND OVERVIEW OF THIS REPORT

This report has the following objective: to provide curricular guidance for implementing two-year, associate degree programs in the areas of computing as related to engineering technology.

Curricula, based on the guidelines which follow, are intended to prepare graduates for entry-level positions that require specific education in the technology of computing and its applications. These positions span a spectrum of work assignments which include system development and implementation, manufacturing, installation, management, and maintenance. A blend of hardware and software, systems, and interpersonal communication skills for these positions is required.

The present recommendations contain the following parts:

I. A set of curricular and pedagogical considerations that govern the mapping of the requirements into two-year programs of study. These include the roles of laboratories, mathematics and science, a notion of underlying themes, and other educational experiences that combine to make a complete two-year program.

II. A collection of subject matter modules called *knowledge units*, that

comprise the common requirements essential for a two-year curriculum.

III. A curricular mapping of the knowledge units into sample courses and curricula. Each knowledge unit within a course will have a measure of its emphasis and depth of coverage in that course.

Two-year programs must respond to local conditions and the curricula offered must consider these industrial needs. Consequently, this report does not contain a single prescription of courses for all associate degree level programs in computing and engineering technology. However, they do provide sufficient guidelines to allow institutions to style their programs to fit within local institutional constraints, local business needs and current job market opportunities, and to meet accreditation guidelines.

2.0 CURRICULA GOALS AND PROFILES OF GRADUATES

Computing and engineering technology offers students and graduates opportunities to study and work in a variety of areas of two merging technologies: computing and engineering. The following identifies three programs of study, as well as the expectations which may be attained by the graduates of these programs.

3.1 Curricula Goals

The first goal of a two-year technical program is to prepare graduates to enter the work force as technicians, possessing skills that match local industry needs. Hence, the two-year program must provide a coherent and broad-based coverage of the disciplines of computing, electronics, computer technology, mathematics, science and liberal studies. Graduates should develop a reasonable level of understanding in the subject areas that define the discipline, as well as possess an appreciation for the relationships among them. Graduates should be able to apply their knowledge to specific problems and produce appropriate solutions.

Secondly, the two-year program should prepare students for advanced, formal education leading to a baccalaureate degree and life-long learning. To accomplish this, the curriculum should provide transfer courses, the proper mix of academic and technical courses, and an environment for studying the ethical

and societal issues that are associated with the computer field. This is especially important for cooperative programs between area two-year and four-year schools.

3.2 Three Curricula Areas

Computing and engineering technology may be divided into three distinct curricula areas that share certain commonalities:

Electronic Engineering Technology (EET)
Software Engineering Technology (SET)
Computer Engineering Technology (CET)

Electronic engineering technology is a hardware-oriented curriculum with an emphasis on analog and digital devices, computer architecture and related hardware. Software engineering technology focuses upon the use and development of computer applications software to design, build, modify and test electronic (digital) devices and software systems; this area is not to be confused with software engineering which is commonly reserved for the study of large-scale software systems. Computer engineering technology is a *mid-road* curriculum sharing elements of both electronic and software engineering technology areas, and focuses on the implementation, maintenance and operation of computer systems and network.

The disciplines of computing and engineering technology have two essential components:

- A basis in what is traditionally considered *engineering technology* in two-year programs.
- The study of computers as an integral part of the curricula.

Engineering technology programs, such as electronic engineering technology, are those programs which are recognized by the Technology Accreditation Commission of the Accreditation Board for Engineering and Technology (TAC/ABET). Although computers are used in the vast majority of curricula, including physics, biology, music, and psychology, computers are not an integral part of the curricular definitions. In the disciplines defined by computing and engineering technology, subject areas such as computer architecture, programming and design are salient components of the curricula.

Each of the three two-year programs in the field of computing and engineering technology share common attributes, values and curricula. The following discussion describes these attributes and the values they share, and it forms a basis for the curricula that follow.

3.3 Profiles of Graduates

The primary goal of two-year computing and engineering technology programs is to provide coherent and broad-based coverage of the discipline so that graduates of the program will possess current technical and non-technical skills necessary for entry in the work force at the technician level. The skill sets possessed by program graduates will vary in breadth and depth from institution to institution, reflecting differences in the mission of each institution and the constituency that is served. It follows, therefore, that graduates will receive different levels of coverage in the subject areas that define the curriculum, as well as a difference in balance of emphasis.

To meet this goal, computing and engineering technology programs should prepare students to apply their knowledge to specific problems and to provide solutions to them. This includes the ability to define a problem clearly, to evaluate and choose an appropriate solution strategy, to specify, implement, test and modify that solution, and to interpret, analyze and document the results of their findings. In addition, graduates must be able to function effectively in a team environment, possess good oral and written communication skills, and be conscious of the importance of human relations, especially as they relate to computing and technology.

Another goal of two-year computing and engineering technology programs is to provide an interface to upper-division programs that lead to a baccalaureate degree. This continued upward mobility in a chosen career path prepares students for the more general challenges of their professional and personal life. Students enrolled in any two-year program should consult with colleges and universities offering baccalaureate programs to determine the transferability of specific courses, and should be provided with proper advisement and accurate information regarding this process.

Finally, computing and engineering technology programs should provide an environment in which students are exposed to the ethical, societal and cultural

issues that are associated with the field. This includes maintaining currency with recent technological advances, upholding professional standards, understanding the global marketplace, and developing an awareness of roles and responsibilities of the discipline itself.

3.4 Career Opportunities

Three different types of technical specialties are described in this report: computer engineering technology, electronic engineering technology, and software engineering technology. The technical specialists that graduate from these two-year programs perform different types of tasks in industry and require a different mix of software and hardware skills to execute these tasks.

The graduate of a two-year computer engineering technology program may work in companies that manufacture, install and maintain, and use computer systems, including computer networks. Technical skills within these companies will require hardware skills in digital and computer circuitry and system/network architectures. Software skills will require the ability to install and use current computer operating systems and resident software tools. Finally, these positions will require strong interpersonal communication skills to help users of computer systems understand and apply computer-based tools to a wide variety of settings and applications.

The graduate of a two-year electronic engineering technology program may work in companies that manufacture, install, and maintain a wide variety of electronic instruments, devices and systems. Technical positions within these electronics manufacturing companies, and companies that are users of electronic systems, will require stronger skills in both digital and analog electronics than that provided by the computer engineering technology curriculum. Computer related skills will focus on embedded applications of microprocessors that provide data acquisition, signal processing, and control functions. Employers will also look for strong communication skills, the ability to work in team environments, and a fundamental knowledge of quality concerns.

The graduate of a two-year software engineering technology program may work in companies that produce a growing array of software tools and products, or in companies that produce products that incorporate resident and applications software. Technical positions suited to the software engineering technology graduate will require the knowledge to generate, test, validate, update and

maintain a variety of software products. Employers will look for a working knowledge of current operating systems, programming languages, and industry standard methods and practices. Hardware knowledge will be limited to an introduction to digital circuits and microprocessor systems.

The following table illustrates the appropriate mix of hardware and software skills required in these three careers.

Career	Hardware	Software
Electronic Engineering Technology	75%	25%
Computer Engineering Technology	50%	50%
Software Engineering Technology	25%	75%

4.0 COMPONENTS OF CURRICULA DESIGN

4.1 Subject Areas and Knowledge Units

For each curriculum under computing and engineering technology, (CET, EET and SET) a set of subject areas is defined, the details of which are discussed in Section 5.1. In turn, each subject area is divided into a set of knowledge units (KU) which represents the core of the body of knowledge for each discipline.

Knowledge units are presented by subject area and are not necessarily in the order of presentation. The details of the knowledge units are found in Part II of this report. It is intended that these units would be mapped into courses as determined by individual institutions. Sample courses are presented in Part III.

4.2 Underlying Themes

Certain fundamental concepts recur throughout the discipline and play an important role in the design of individual courses and whole curricula. These fundamental concepts are referred to as *underlying themes* in this document. The underlying concepts represent a kind of *glue* that can be used to provide

cohesiveness for this discipline. They represent fundamental analysis, principles, technical concerns, and pedagogical issues that the student completing this program is expected to encounter over a variety of circumstances. Details of these themes are addressed in Part III of this report.

5.0 BUILDING THE CURRICULA

This report presents three separate curricula in the area of computing and engineering technology. However, because of the mix of hardware and software skills, there exists certain commonality among the programs. These curricula with their commonalities are illustrated in Part III of this report.

The sections that follow give subject area and knowledge unit breakdowns for all curricula in computing and engineering technology with a brief interpretation for each subject area. Exit competencies are discussed for each curriculum in Parts III and IV. Each curriculum is founded upon a course structure composed of knowledge units taken from respective subject areas for each curriculum, together with exit competencies. Curricula electives and professional responsibility are also discussed.

For each of the three curricula (CET, EET and SET) a set of core subject areas is prescribed in which each subject area is divided into knowledge units. The knowledge unit content for each subject area is to be interpreted as the minimum required in order to produce a technically competent graduate in the field.

5.1 Subject Area Identification

The recommended curricula in computer engineering technology, electronic engineering technology, and software engineering technology are characterized by subject areas for each discipline. They are included here in their entirety because of the broad area of computing and engineering technology, and of the certain commonalities endogenous to the three curricula. The knowledge units described under each subject area are identified by their tag code followed by the name of the unit. Each of the subject areas is distinguished by a specific application domain of theory, techniques and tools.

The subject areas for the three disciplines of computing and engineering technology are identified and briefly summarized as follows. The subject titles also indicate the two-letter tag code of the knowledge unit format discussed in Section 4.1.

Circuits (CR)
Deals with methods of analyzing passive electronic circuits. Fundamental questions include: How are passive circuits designed to control currents and voltages? How do passive circuits provide signal processing functions? What tools are available to analyze passive circuit behavior?

Computer Architecture (CA)
Deals with methods of organizing hardware into efficient, reliable systems. Fundamental questions include: What are the various methods of implementing processors, memory and data transfer in a machine, and what are the advantages and disadvantages of each?

Computing Systems (CS)
Deals with the design, installation and maintenance of computing systems. Fundamental questions include: How are computing systems implemented? What strategies are best for maintaining and updating computing systems and networks?

Data Communications (DC)
Deals with the transmission of information between computer systems. Fundamental questions include: What are current techniques for data transmission? How can one install, test and maintain computer networks and data communication systems?

Devices (DV)
Deals with the application of discrete devices and linear integrated circuits. Fundamental questions include: What are the operating characteristics of circuits containing discrete semiconductor devices and linear integrated circuits? What signal processing and control functions are provided by these active devices?

Digital Circuits and Systems (DG)
Deals with digital logic devices from small-scale integration to complex logic functions and programmable logic devices.

Fundamental questions include: What methods should be used in implementing logic circuits? How are logic circuits analyzed and tested?

Linear Integrated Circuits (LC)
Deals with linear integrated circuits and their application. Fundamental questions include: What are some basic types of linear integrated circuits? How do these devices operate and what are their operational characteristics? How can correct operation be determined?

Microprocessor Systems (MS)
Deals with the hardware and software of microprocessor systems. Fundamental questions include: How does the hardware in a microprocessor system perform fundamental operations such as fetch data and operation codes from memory, execute interrupts, and interface with input and output devices? What role does software play in a microprocessor system? How does one analyze the interaction of hardware and software?

Operating Systems (OS)
Deals with operating system modes, process and resource management, and memory and file system management. Fundamental questions include: How do operating systems differ in structure and operation? How are memory and file systems managed? What attributes of operating systems must the user know to effectively use an operating system?

Program Design Techniques (PD)
Deals with the program development, implementation, validation, and documentation practices. Fundamental questions include: How are algorithms developed? How is software tested and validated? What are some documentation techniques that will effectively communicate the intent and operation of computer programs?

Programming Methodology (PM)
Deals with the design and implementation of programs and software systems that meet safe, reliable, and dependable specifications. Fundamental questions include: What are the principles behind the

development of programs and software packages? How does one determine that a program or system meets specifications?

Quality Control (QC)
Deals with statistical methods that can be used to measure, improve and maintain performance quality in a hardware/software system. Fundamental questions include: What parameters need to be controlled? What data should be collected and how? How should the data be analyzed, interpreted, and used to control a process?

Software Testing and Validation (ST)
Deals with methods that determine the correctness of programs and systems of programs. Fundamental questions include: What strategies are used for software testing and validation? How can failures be analyzed and corrected? What is the interrelationship of software development, implementation, and testing?

Details of the knowledge units for each subject area may be found in Part II of this report.

5.1.1 Computer Engineering Technology

The following constitute the nine technical components of a curriculum in computer engineering technology (CET).

Circuits
Computer architecture
Computing systems
Data communications
Devices
Digital circuits and systems
Microprocessor systems
Programming methodology
Quality control

5.1.2 Electronic Engineering Technology

The following constitutes the ten technical components of a curriculum in electronic engineering technology (EET).

Circuits
Computer architecture
Computing systems
Data communications
Devices
Digital circuits and systems
Linear integrated circuits
Microprocessor systems
Programming methodology
Quality control

5.1.3 Software Engineering Technology

The following constitutes the eleven technical components of a curriculum in software engineering technology (SET).

Circuits
Computer architecture
Computing systems
Devices
Digital circuits and systems
Microprocessor systems
Operating systems
Program design techniques
Programming methodology
Quality control
Software testing and validation

5.2 Emphasis of Knowledge

Emphasis of knowledge is divided into three paradigms: *theory*, *analysis* and *design*. *Theory* corresponds to the fundamental knowledge base of a particular concept that is generally supported by fact or accepted understanding. *Analysis* corresponds to the scientific method used in verifying or testing a theoretical phenomenon, or in discovering a certain principle in the subject. *Design* corresponds to building an entity (hardware and/or software) that demonstrates the general theme of a concept. Emphasis of knowledge is applied to courses which are appended with any combination of letters of T, A or D, representing

theory, analysis and design, respectively. This is further discussed in Part III of this report.

5.3 Depth of Knowledge

Courses also indicate an expected depth of knowledge called a *depth indicator*. This indicator is identified by an integer 1 through 5 which is appended to the course description. Depth of knowledge is considered a minimal understanding of a topic within its course setting. Details of this are identified in Part III of this report.

5.4 Course Structure

The subject areas and knowledge units (listed in Section 5.1 and detailed in Part II) specify the scope of topics all students should study in a particular discipline of computing and engineering technology. The lecture hours associated with each knowledge unit gives an approximate indication of the depth to which a topic should be covered. The prerequisite structure suggests that some sequencing is required in the formation of courses from knowledge units.

5.5 Curricula Electives

Elective courses are constructed in the same manner as regular courses. There is, however, the possibility of expansion of knowledge units which are additional to those shown in Section 5.1 and Part II of this report. Elective courses are particularly important because they allow the possibility of a greater depth of knowledge units within courses, and *pull together* material learned in more than one course. Elective courses also allow students flexibility in tailoring their degree program to their career goals.

The following suggestions describe possible topics that could be the basis for one or more elective courses for any of the curricula presented.

Automated Test Systems
Student will test devices, circuits, and systems using automated test stations. Course content will include configuration, calibration, and operation of test stations as well as data collection and analysis

strategies.

Design Project
Students are expected to research, design, build, test, document, and report on a team project. The project could be hardware, software or a combination of the two, and is intended to be a capstone project.

Microcontrollers
Microcontroller concepts. Comparison of various microcontrollers. Application of a specific microcontroller in a defined environment.

Satellite Communications
Use of satellites for data communications, characteristics of satellite links, and economics of satellite systems.

Technical Sales
Elements of technical sales and service form the manufacturing to the retail level. Includes the selling of both products and services to businesses, industries, professionals, and public and private institutions.

Technical Writing
Techniques of gathering, organizing, and presenting technical information. Technical communications derived from realistic situations found in industry.

Programming
Use of a current and appropriate computer programming language in problem solving. Emphasis is placed on real-world problems.

Project Management
Students will investigate software development approaches and team management techniques. Group assignments will provide experience in structuring and managing teams.

Institutions are encouraged to enrich the curricula offered by developing elective courses which will enhance the marketability of their students.

5.6 Social and Professional Context

Undergraduates need to understand the basic cultural, social, legal and ethical issues inherent in the discipline of computing. They should understand the historical background of the discipline, the current situation, and future trends. They should also understand their individual roles in this process, as well as appreciate the philosophical questions, technical problems, and aesthetic values that have played an important part in the development of the discipline.

Students also need to develop the ability to ask serious questions about the social impact of computing and to evaluate proposed answers to those questions. Future practitioners must be able to anticipate the impact of introducing a given product into a given environment. Will that product enhance or degrade the quality of life? What will the impact be upon individuals, groups and institutions?

Finally, students need to be aware of the basic legal rights of software and hardware vendors and users, and they also need to appreciate the ethical values which are the basis for those rights. Future practitioners must understand the responsibility they will bear for what they do, and the possible consequences of failing to maintain appropriate professional standards.

The graduate of this program may be expected to work without close supervision, to be responsible for tangible and intangible assets, to represent the firm before suppliers and customers, to have access to the firm's confidential information, and to deal with problems which arise in the modern work place. To prepare the graduate for these aspects of the work place, the program should introduce the student to elements of professional responsibility and ethical behavior, to problem solving and decision making, and to the fundamental rights of intellectual property.

Professional responsibility and ethical behavior may be presented as case studies, illustrating the potential conflicts in loyalties to the firm on the one hand and the customer on the other, with actions benefitting the firm on the one hand and the individual on the other, and so on. Attention should be paid to the issues of dealing with confidential information of the firm as well as that of the customer.

6.0 REQUIREMENTS FROM OTHER DISCIPLINES

The curricula of computing and engineering technology require studies from other disciplines. These include mathematics, science, oral and written communication, and general education courses. These areas are essential for communication both in mathematical, oral and written form, as well as for understanding the relation of technology to the world around us, its people and cultures.

6.1 Mathematics

A strong basis in college algebra and trigonometry (precalculus mathematics) is necessary to be able to use mathematics in engineering technology programs. Curricula should also include selected topics from statistics, logic and discrete mathematics. At least one course in integral and differential calculus is required.

6.2 Science

Basic science courses are an important part of an engineering technology program. These courses enable students to acquire a fundamental knowledge about nature and its phenomena. Courses should emphasize the understanding, measurement, and quantitative expression of physical processes. Laboratory work, including experimentation, observation, measurement and report writing, should be a required part of the study of science. For programs in computer, electronic, and software engineering technology, general physics is the preferred science area because it illustrates phenomena in nature that have a direct relationship to computing and technology.

6.3 Oral and Written Communication

Good oral and written communication skills are a necessary achievement of a college graduate. Technically trained individuals are considered educated when they are able to communicate, both orally and in writing, their technical findings, thoughts and philosophies. Associate degree programs should include course work in English composition, including both written and oral presentation, literature, and technical writing. Furthermore, writing must be integrated into

technical courses. Student reports must be of professional quality and effectively communicate to the intended audience. Students should also be given the opportunity to make oral presentations and to critique presentations made by their peers. Furthermore, the ability to read and assimilate technical literature is an important skill that will enable students to keep current in their field. Students should be encouraged to read technical publications and to use manufacturer's literature and manuals in support of their laboratory assignments.

6.4 General Education

With the internationalization of markets, it is important that students become familiar with the heritage and culture of people other than those of the United States. The study of social cultures or foreign languages enhance both the professional and personal achievement of all individuals. Familiarity with the arts and literature, awareness of environmental and ecological concerns, and the understanding of local and global issues in politics and economics, are necessary elements to a well-educated and competent professional.

7.0 RESOURCES

The following ingredients are requisite for developing, implementing, and maintaining viable programs. Laboratory resources provide the practical training needed of engineering technologists. Faculty and staff must be qualified to deliver the instruction at the appropriate level and in a manner conducive to learning, and the institution must support the instructional program.

7.1 Laboratories

It is important that instruction in engineering technology be conducted in an atmosphere of realism. Adequate laboratory facilities are central to effective achievement of this goal. Laboratory equipment and computers should be of the type that may be encountered in industrial practice. Furthermore, each student should have the opportunity to become thoroughly familiar with the use and

operation of equipment and computers commonly used in their major field of study. Manuals, equipment catalogs, professional magazines, and journals should be readily accessible and used by students.

Laboratories must be accessible to students. Laboratories should not only be scheduled for closed laboratory sessions, but should be scheduled for open sessions which allow students free access to facilities to complete their assignments in a timely fashion or for independent study.

Provision for updating equipment is important. A plan for laboratory improvement should be implemented. This plan should include the purchase or lease of equipment and software, licenses and contracts, and regular maintenance procedures. This plan should also provide adequate technical support for hardware and software.

7.2 Faculty and Staff

Programs that are recommended in this report can only be well-implemented with an appropriate complement of full-time faculty whose primary field of expertise is in disciplines of engineering, engineering technology, and computing. Basic credentials for faculty should include some industrial experience and a master's degree in engineering, engineering technology, or in a closely related field, if the degree is primarily analytical and the subject clearly appropriate.

The number of faculty members needed in a program depends on the number of students in the program, the portion of evening and co-op students, and other duties assigned to the technical faculty. The number of faculty must be great enough to provide a breadth of perspective, program continuity, and timely course offerings.

Due to the rapidly changing field of technology, faculty members must maintain current knowledge of their field including an understanding of the tasks industry expects of graduates. Faculty members can remain current by active participation in professional societies by attending conferences and meetings, continuing education, reading the literature, and periodic returns to industry. The institution should have a planned, adequately funded program for professional development of its faculty.

7.3 Institutional Support

The institution must provide adequate support for the instructional program including secretarial and technician support, office space, travel funds and release time. Satisfactory procedures and qualified support personnel are required to keep laboratory and instructional equipment in good repair and working order. Faculty office space should be adequate with privacy for helping, counseling and advising students. Institutional and department budgets should provide travel funds and release time for faculty to attend conferences and professional meetings.

8.0 PROGRAM ISSUES

There are several external considerations which influence program quality. For computing and engineering technology, accreditation is an important factor since it provides the benchmarks by which the overall program can be periodically reviewed. Other considerations are the industrial and academic communities and the articulation developed among them.

8.1 Accreditation

Accreditation of engineering technology programs helps students, parents, counselors, potential employers, public bodies, and officials to identify educational programs which meet the minimum criteria established by accrediting agencies. The Technology Accreditation Commission of the Accreditation Board for Engineering and Technology (TAC/ABET) is recognized by the U.S. Department of Education and the Council on Post-Secondary Accreditation as the sole agency responsible for accreditation of education programs leading to associate degrees in engineering technology [5]. Institutions developing programs within computing and engineering technology should consider eligibility for accreditation by the TAC/ABET, as this is an important criterion when implementing any of the three programs of computing, electronic or software engineering technology.

8.2 Articulation

Articulation, the process by which a student moves from one educational level to another or from one program to another, must be considered when designing a curriculum. Educational institutions and programs should work together to provide the student with a continuous experience as the student moves through the educational system. Critical interfaces include the connection between high school and the community college, and between the associate-degree program and the bachelor's degree program

8.3 Tech Prep/Associate Degree Programs

Two-Plus-Two (2+2) Tech Prep/Associate Degree programs strive to produce this seamless interface between high school programs and associate degree programs [6]. The goal of the 2+2 Tech Prep/Associate Degree effort is to create a new culture within high schools that combines attributes from the traditional college preparatory and the vocational cultures. That is, the Tech Prep culture would have the forward-looking attribute of the college preparatory culture that encourages students to take academic courses that prepare them for post-secondary education. In addition, the Tech Prep culture includes the technical emphasis of the vocational culture, encouraging students to begin making career choices.

9.0 SERVICE COURSES

A role of the two-year college is to help serve the needs of local industry and the community. Hence, the faculty must be especially sensitive and receptive to meeting these needs by cooperating with faculty from other disciplines and professionals from industry.

Programs in the area of computing and engineering technology should be flexible enough to engage in educational activities that support community needs. These needs include the education of individuals through a non-traditional format. The non-traditional student is usually over thirty years of age who is seeking entry to the job market, career retraining or career enhancement. Courses for such students should be tailored to meet their needs and should be offered at convenient times. Service courses in the area of computing and engineering technology include computer modelling, computer-aided design, computer-aided manufacturing, introduction to robotics, integrated circuit development, and

computer graphics.

10.0 CONCLUSION

This report has presented curricular recommendations for associate degree programs in the discipline of computing under the rubric of computing and engineering technology. Specifically, the report includes recommendations for the curricula of computer engineering technology, electronic engineering technology, and software engineering technology. Recommendations prescribe knowledge units rather than courses, allowing individual institutions to tailor their own curriculum to their specific program requirements. Knowledge units indicate not only topics, but emphasis and depth of intensity.

In addition to the technical subject areas, related educational topics are discussed. Curriculum design should strive for parallel strands which have a relationship to each other. For example, trigonometry in the mathematics strand is related to alternating current in the circuit analysis strand. Each subject area is a strand and is related to other subject areas.

Course development, therefore, is not merely the expansion of a subject area, but rather the creative selection of various knowledge units from different subject areas. The manner of selection of knowledge units is determined by each institution, and should result in a modern presentation of courses offered by these colleges. For each discipline (CET, EET and SET) the subject areas and the corresponding knowledge units are to be interpreted as the minimum requirements for each program.

This report was prepared by the Computing and Engineering Technology Committee of the Two-Year College Computing Curricula Task Force of the Association for Computing Machinery.
The Steering Committee of the Task Force is appreciative of the many individuals and organizations that supported this curriculum project. Their names appear in the Executive Report.

PART II

CURRICULA KNOWLEDGE UNIT DETAILS

1.0 OVERVIEW OF KNOWLEDGE UNITS

The fundamental elements upon which courses in a curriculum are formulated are knowledge units (KU). Knowledge units may be considered as content clusters within a curriculum. Within these units are imbedded the following elements: subject name, knowledge unit tag and name, description, topics, prerequisites, requisite for, and activities.

1.1 Knowledge Unit Format

Each knowledge unit has a format as outlined in Figure II-1. The *Subject Name* identifies the topic category and is followed by a two-letter code in parenthesis. For example, the subject name

CIRCUITS (CR)

identifies the subject category with its tag code. This tag code is then used to identify the particular knowledge unit tag for each subject.

The *Knowledge Unit Tag and Name* (KU Tag and Name) identifies the particular knowledge unit, and has the format

Letter-1 Letter-2 Integer : KU Name

Letter-1 and *Letter-2* form the subject tag code that identifies the subject area. *Integer* identifies a particular knowledge unit under a given subject area. Code tags are identified by a knowledge unit name. For example:

CR1: ELECTRICAL PARAMETERS

indicates that the knowledge unit entitled *Electrical Parameters* is in the subject area of *Circuits* and it is the first in the set of knowledge units for that subject area.

The *Description* of a knowledge unit is a brief statement that describes the highlights of the particular unit. The *Topics* section identifies the particular concepts to be addressed. Knowledge units of subject areas are related to other knowledge units within the curriculum. *Prerequisites* refer to knowledge units that must be completed prior to completing the current knowledge unit. *Requisite For* indicates knowledge units for which the current knowledge unit is a prerequisite. *Activities* take the primary form of a laboratory experience in a setting that is open or closed.

SUBJECT NAME	**CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**CR1: ELECTRICAL PARAMETERS**
DESCRIPTION	Introduction to basic electrical parameters, definitions and units including voltage, resistance, current and power.
TOPICS	Minimum number of hours: 3 1. Current 2. Resistance and color coding 3. Voltage 4. Power
PREREQUISITES	None
REQUISITE FOR	CR2
ACTIVITIES	Closed laboratory Use of multimeter to measure continuity, voltage, resistance and current as well as interpretation of instrument specifications.

Figure II-1 **Model of Knowledge Unit**

1.2 Activities

Because of the unique field of computing and engineering technology, activity (laboratory) recommendations are also included within each knowledge unit. The activity portion of a knowledge unit consists of two parts, an indication as to whether or not the laboratory experience should be opened or closed, and a brief statement of the recommended experience. *Open laboratories* are those in which individuals are to complete assignments outside of the classroom environment and with little or no supervision. *Closed laboratories* are those in which individuals are to perform or complete required assignments within an environment under direct supervision.

2.0 SUBJECT AREAS AND RELATED KNOWLEDGE UNITS

The following identifies the subject areas in computing and engineering technology with their related knowledge units.

Circuits (CR)

- CR1: Electrical Parameters
- CR2: Electrical Relationships
- CR3: Electrical Signals
- CR4: Circuit Configurations
- CR5: Circuit Reduction
- CR6: Circuit Simulation and Analysis
- CR7: Capacitance and Inductance
- CR8: Transient Analysis
- CR9: Frequency Response
- CR10: Spectral Analysis

Computer Architecture (CA)

- CA1: History of Computing
- CA2: Machine Organization
- CA3: Assembly Language Programming
- CA4: Memory Systems
- CA5: Interfacing
- CA6: Alternative Architectures

Computing Systems (CS)

- CS1: System Organization
- CS2: Hardware Specifications
- CS3: System Software
- CS4: Computer Peripherals
- CS5: Computing Environments
- CS6: Applications Software
- CS7: Network Implementation
- CS8: Network Management

Data Communications (DC)

- DC1: Data Transmission Basics
- DC2: The Telephone System
- DC3: Data Transmission via Analog Signals
- DC4: Data Transmission via Digital Signals
- DC5: Protocols
- DC6: Optical Data Transmission
- DC7: Local Area Networks

Devices (DV)

- DV1: Amplifier Concepts and Characteristics
- DV2: Operational Amplifiers
- DV3: Linear Integrated Circuits
- DV4: Diodes
- DV5: Transistors
- DV6: Opto-Electronic Devices
- DV7: Device Modeling and Systems Simulation

Digital Circuits and Systems (DG)

- DG1: Number Systems
- DG2: Boolean Algebra
- DG3: Logic Functions and Devices
- DG4: Logic Simplification Methods
- DG5: Complex Combinational Circuits
- DG6: Sequential Logic Devices and Their Applications

DG7: Data Conversion
DG8: State Machines
DG9: Logic Analysis Techniques

Linear Integrated Circuits (LC)

LC1: Voltage and Current References
LC2: Filter Concepts and Circuits
LC3: Phase Locked Loops
LC4: Oscillators
LC5: Grounding and Shielding
LC6: Modeling and Simulation

Microprocessor Systems (MS)

MS1: Machine Organization
MS2: Assembly Language Programming
MS3: Hardware Specifications
MS4: Interfacing
MS5: System Level Implementation
MS6: Alternative Architectures

Operating Systems (OS)

OS1: Operating System Models
OS2: Processor Management
OS3: Device Control
OS4: Memory Management
OS5: File System Management
OS6: Operating System Case Study

Program Design Techniques (PD)

PD1: Software Engineering Concepts
PD2: Requirements, Specifications and Software Development
PD3: Algorithms and Data Structures
PD4: Software Design, Documentation and Implementation
PD5: Software Integration, Testing and Validation
PD6: Team Programming Management

Programming Methodology (PM)

PM1: History of Programming Languages
PM2: Introduction to a Programming Language
PM3: Data Types
PM4: Algorithms and Program Design
PM5: Program Documentation, Testing and Verification

Quality Control (QC)

QC1: Reliability and Maintainability
QC2: Statistical Methods
QC3: Reliability Evaluation
QC4: Failure Mode Effects Analysis

Software Testing and Validation (ST)

ST1: Concepts of Software Testing and Validation
ST2: Module Testing
ST3: Integration and System Tests
ST4: Other types of Tests and Validation

3.0 DETAILS OF KNOWLEDGE UNITS

The details of the knowledge units for computing and engineering technology are presented below, alphabetically by subject area as in Part I, Section 5.1 of this report.

SUBJECT NAME	**CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**CR1: ELECTRICAL PARAMETERS**
DESCRIPTION	Introduction to basic electrical parameters, definitions and units including voltage, resistance, current and power.
TOPICS	Minimum number of hours: 3 1. Current 2. Resistance and color coding 3. Voltage 4. Power
PREREQUISITES	None
REQUISITE FOR	CR2
ACTIVITIES	Closed Laboratory Use of multimeter to measure continuity, voltage, resistance and current as well as interpretation of instrument specifications.

SUBJECT NAME	**CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**CR2: ELECTRICAL RELATIONSHIPS**
DESCRIPTION	Electrical relationships among voltage, current, resistance, and power. Includes Ohm's Law and Kirchhoff's Laws.
TOPICS	Minimum number of hours: 6 1. Ohm's Law 2. Kirchhoff's Voltage Law 3. Kirchhoff's Current Law 4. Power computations
PREREQUISITES	CR1
REQUISITE FOR	CR3
ACTIVITIES	Closed Laboratory

Use of experimental techniques to verify electrical relationships and laws.

SUBJECT NAME	**CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**CR3: ELECTRICAL SIGNALS**
DESCRIPTION	Electrical signals and their parameters. Includes dc, sinusoidal, and pulse waveforms. Introduction to the use of meters and the oscilloscope to measure signal parameters and signal generators to produce waveforms.
TOPICS	Minimum number of hours: 3 1. DC signals 2. Sinusoidal signals - RMS and peak-to-peak 3. Pulse signals 4. Signal generation 5. Measurement techniques
PREREQUISITES	CR2
REQUISITE FOR	CR4
ACTIVITIES	Closed Laboratory Use of signal generators to produce wave forms, and use of meters and the oscilloscope to make signal parameter measurements.

SUBJECT NAME	**CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**CR4: CIRCUIT CONFIGURATIONS**
DESCRIPTION	Circuit configurations, characteristics, and applications. Includes series connections, parallel connections, and their combination.
TOPICS	Minimum number of hours: 6 1. Series connections 2. Parallel connections 3. Series-parallel combinations 4. Applications of series and parallel connections
PREREQUISITES	CR3
REQUISITE FOR	CR5, CR6, CR7
ACTIVITIES	Closed Laboratory Construct series and parallel circuits and experimentally verify proper

circuit operations.

SUBJECT NAME	**CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**CR5: CIRCUIT REDUCTION**
DESCRIPTION	Introduces the concept of equivalent circuits. Includes Thevenin's and Norton's Theorems and their application in circuit reduction and minimization.
TOPICS	Minimum number of hours: 3 1. Equivalent circuits 2. Voltage and current sources 3. Thevenin's Theorem 4. Norton's Theorem
PREREQUISITES	CR4
REQUISITE FOR	CR8
ACTIVITIES	Closed Laboratory Use equivalent circuit techniques to analyze electric circuits and networks.

SUBJECT NAME	**CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**CR6: CIRCUIT SIMULATION AND ANALYSIS**
DESCRIPTION	Introduces the concept of circuit simulation, device modeling, and computer simulation software.
TOPICS	Minimum number of hours: 3 1. Concept of simulation 2. Device models 3. Circuit simulation software 4. Application of simulation
PREREQUISITES	CR4
REQUISITE FOR	CR8, CR9, DV4
ACTIVITIES	Closed Laboratory

Comparison of circuit simulation results to actual experimental data.

SUBJECT NAME	**CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**CR7: CAPACITANCE AND INDUCTANCE**
DESCRIPTION	Introduction to the properties and characteristics of capacitance and inductance. Includes the concepts of lumped and distributed parameters or components.
TOPICS	Minimum number of hours: 3 1. Capacitance 2. Inductance 3. Electric and magnetic fields 4. Lumped vs. distributed parameters
PREREQUISITES	CR4
REQUISITE FOR	CR8, CR9
ACTIVITIES	None

SUBJECT NAME	**CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**CR8: TRANSIENT ANALYSIS**
DESCRIPTION	Transient analysis of R-C and R-L circuits. Includes an introduction to exponential functions.
TOPICS	Minimum number of hours: 4 1. Exponential functions 2. R-C circuits 3. R-L circuits 4. Simulation techniques
PREREQUISITES	CR5, CR6, CR7
REQUISITE FOR	DV1
ACTIVITIES	Closed Laboratory

Construct and experimentally determine the operational characteristics of R-C and R-L circuits.

SUBJECT NAME	**CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**CR9: FREQUENCY RESPONSE**
DESCRIPTION	Concepts of impedance and transfer functions and their application to the frequency analysis of electric circuits.
TOPICS	Minimum number of hours: 6 1. Impedance (resistance and reactance) 2. Transfer functions 3. Single--frequency analysis and phaser representations 4. Bode Plots (magnitude and phase) 5. Simulation
PREREQUISITES	CR6, CR7
REQUISITE FOR	DV1, CR10
ACTIVITIES	Closed Laboratory Design, predict and experimentally verify the response of a circuit.

SUBJECT NAME	**CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**CR10: SPECTRAL ANALYSIS**
DESCRIPTION	Introduces the concept of signals as a composite of dc, fundamental, and harmonic components. Applies superposition to the analysis of electric circuits.
TOPICS	Minimum number of hours: 3 1. Superposition 2. Decomposition of signals into components 3. Spectral analysis 4. Instrumentation (Fourier analyzer, spectrum analyzer)
PREREQUISITES	CR9
REQUISITE FOR	LC2
ACTIVITIES	Closed Laboratory

Experimentally determine the spectral content of signals.

SUBJECT NAME	**COMPUTER ARCHITECTURE**
KNOWLEDGE UNIT TAG AND NAME	**CA1: HISTORY OF COMPUTING**
DESCRIPTION	A brief introduction to the history of digital computing with an emphasis on the problem or problems to be solved through the invention of the computing machine.
TOPICS	Minimum number of hours: 6 1. Mechanical Computers (Pascal, Hollerith, Babbage) 2. Electronic Computers (von Neumann machines, ENIAC) 3. Large Scale Computers (main frame, mini, super) 4. Microcomputers (IBM-PC, Apple Macintosh) 5. Workstations (Sun, Apollo, IRIS)
PREREQUISITES	None
REQUISITE FOR	CA2
ACTIVITIES	None

SUBJECT NAME	**COMPUTER ARCHITECTURE**
KNOWLEDGE UNIT TAG AND NAME	**CA2: MACHINE ORGANIZATION**
DESCRIPTION	Introduction to the architecture of a CPU and timing, control, data, and address signals. Includes memory organization and timing.
TOPICS	Minimum number of hours: 4 1. Block diagram system overview 2. Internal architecture of CPU 3. Bus interface and timing 4. Memory organization
PREREQUISITES	CA1
REQUISITE FOR	CA3, CA4

ACTIVITIES	Closed Laboratory Use of a single-board microcomputer trainer. Use of oscilloscopes and logic analyzers to view system signals.

SUBJECT NAME	**COMPUTER ARCHITECTURE**
KNOWLEDGE UNIT TAG AND NAME	**CA3: ASSEMBLY LANGUAGE PROGRAMMING**
DESCRIPTION	Assembly language programming procedures and processes, use of an instruction set, and software documentation practices.
TOPICS	Minimum number of hours: 12 1. Instruction set 2. Addressing modes 3. Program development and implementation 4. Documentation
PREREQUISITES	CA2
REQUISITE FOR	CA5
ACTIVITIES	Open Laboratory Exercises include programming at the assembly language level and the use of an assembler and debugging software/hardware.

SUBJECT NAME	**COMPUTER ARCHITECTURE**
KNOWLEDGE UNIT TAG AND NAME	**CA4: MEMORY SYSTEMS**
DESCRIPTION and addressing	Physical implementation of memory systems, specifications, organization,
TOPICS	Minimum number of hours: 4 1. Memory device types 2. Memory specifications 3. Memory organization and addressing
PREREQUISITES	CA2
REQUISITE FOR	CA5, CS1
ACTIVITIES	Closed Laboratory Design and implementation of additional memory modules to an existing microcomputer system.

SUBJECT NAME	**COMPUTER ARCHITECTURE**
KNOWLEDGE UNIT TAG AND NAME	**CA5: INTERFACING**
DESCRIPTION	Input/output control and interfacing. Techniques for interrupt handling, DMA, and parallel and serial I/O.
TOPICS	Minimum number of hours: 10 1. Input/output control methods 2. Interrupts 3. Direct Memory Access (DMA) 4. Parallel and serial I/O 5. I/O programming
PREREQUISITES	CA3, CA4
REQUISITE FOR	None
ACTIVITIES	Closed Laboratory Design the interface, connect and program peripheral devices and write interrupt driven I/O service routines.

SUBJECT NAME	**COMPUTER ARCHITECTURE**
KNOWLEDGE UNIT TAG AND NAME	**CA6: ALTERNATIVE ARCHITECTURES**
DESCRIPTION	Comparison of CISC, RISC, microcontroller, and special function processors
TOPICS	Minimum number of hours: 4 1. Comparisons 2. CISC vs RISC 3. Microcontrollers 4. Special function processors (e.g., co-processors, math)
PREREQUISITES	CA5
REQUISITE FOR	None
ACTIVITIES	None

SUBJECT NAME	**COMPUTING SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**CS1: SYSTEM ORGANIZATION**
DESCRIPTION	The basic building blocks of memory, central processing unit, clock, and input/output devices used to create a computing system
TOPICS	Minimum number of hours: 3 1. Basic building blocks and their interrelation 2. System integration
PREREQUISITES	CA4
REQUISITE FOR	CS2
ACTIVITIES	None

SUBJECT NAME	**COMPUTING SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**CS2: HARDWARE SPECIFICATIONS**
DESCRIPTION	The specifications for the hardware components of a computer system. Includes a discussion of the operation of each system component.
TOPICS	Minimum number of hours: 9 1. CPU board specifications 2. Memory specifications (RAM, ROM, cache) 3. Disk storage specifications (floppy and hard disk) 4. Display specifications (monitor) 5. Input device specifications (keyboard)
PREREQUISITES	CS1
REQUISITE FOR	CS3

ACTIVITIES	Closed Laboratory Assemble a microcomputer-based computer system.

SUBJECT NAME	**COMPUTING SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**CS3: SYSTEM SOFTWARE**
DESCRIPTION	The concept of software as the interface between the user and the computer. Includes resident, diagnostic, operating system, and user software, its installation and organization on disk.
TOPICS	Minimum number of hours: 9 1. ROM resident software 2. Operating systems (e.g., MS-DOS, UNIX, OS2) 3. Diagnostic software 4. User software (installation and disk organization)
PREREQUISITES	CS2
REQUISITE FOR	CS4, CS5, CS6
ACTIVITIES	Closed Laboratory Use of diag. software to test computer systems and the installation and organization of user software. Location of simple system faults using diagnostic software, and the repair of faults by component replacement or correcting system configuration.

SUBJECT NAME	**COMPUTING SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**CS4: COMPUTER PERIPHERALS**
DESCRIPTION	Descriptions of common peripherals used with computer system. Includes hardware interface and software driver requirements
TOPICS	Minimum number of hours: 6 1. Bus standards (ISA, EISA, Microchannel) 2. Printers 3. Mouse systems 4. Scanners 5. Interface boards
PREREQUISITES	CS3
REQUISITE FOR	None

ACTIVITIES	Closed Laboratory Interfacing of printers, mouse systems and scanners to a computer system.

SUBJECT NAME	**COMPUTING SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**CS5: COMPUTING ENVIRONMENTS**
DESCRIPTION	Application environments such as real-time, distributed, time sharing and multiprocessor systems. Hardware environments such as mainframe, minicomputers, microcomputers, supercomputers, and embedded controllers.
TOPICS	Minimum number of hours: 3 1. Application environments real-time, distributed, time shared 2. Mainframes 3. Minicomputers and workstations 4. Microprocessors including laptops and embedded controllers
PREREQUISITES	CS3
REQUISITE FOR	CS7
ACTIVITIES	None

SUBJECT NAME	**COMPUTING SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**CS6: APPLICATIONS SOFTWARE**
DESCRIPTION	Installation and use of commercial software applications products. Examples include wordprocessors, spreadsheets, and database software.
TOPICS	Minimum number of hours: 12 1. Wordprocessors 2. Spreadsheets 3. Databases
PREREQUISITES	CS3
REQUISITE FOR	None
ACTIVITIES	Open Laboratory Use of application software to create reports, tables, and graphs.

SUBJECT NAME	**COMPUTING SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**CS7: NETWORK IMPLEMENTATION**
DESCRIPTION	Hardware and software requirements to implement a network.
TOPICS	Minimum number of hours: 16 1. Hardware requirements 2. Software requirements 3. Network peripherals (printers, plotters, etc.) 4. Interconnections to other networks 5. Layout design considerations
PREREQUISITES	CS5
REQUISITE FOR	CS8
ACTIVITIES	Open Laboratory Design, implement and test a computer network

SUBJECT NAME	**COMPUTING SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**CS8: NETWORK MANAGEMENT**
DESCRIPTION	An introduction to the management of computer networks.
TOPICS	Minimum number of hours: 12 1. Establishing user accounts 2. System security 3. Updates and backups 4. User training 5. Record keeping and documentation
PREREQUISITES	CS7
REQUISITE FOR	None
ACTIVITIES	Open Laboratory Develop and test a secure user account system with appropriate

documentation for an existing network.

SUBJECT NAME	**DATA COMMUNICATIONS**
KNOWLEDGE UNIT TAG AND NAME	**DC1: DATA TRANSMISSION BASICS**
DESCRIPTION	Overview of systems used for data communications; direct wire, optical, telephone network, radio, microwave, satellite. Digital vs analog communications, digital data transmission codes, asynchronous and synchronous, concept of duplex, error detection and correction.
TOPICS	Minimum number of hours: 3 1. Telephone, direct wire, optical fiber, radio links, and satellite communications 2. Analog and digital data transmission 3. Codes - ASCII, EBCDIC 4. Asynchronous and synchronous, concept of duplex 5. Error detection and correction
PREREQUISITES	DG6
REQUISITE FOR	DC2, DC3
ACTIVITIES	None

SUBJECT NAME	**DATA COMMUNICATIONS**
KNOWLEDGE UNIT TAG AND NAME	**DC2: THE TELEPHONE SYSTEM**
DESCRIPTION	Basic telephone operation, telephone system characteristics, phone system characteristics, phone system parameters; frequency response, distortion, impedance, channel conditioning, private line service
TOPICS	Minimum number of hours: 3 1. Physical characteristics of the telephone and the telephone system - pulse & tone dialing 2. Transmission line characteristics 3. Channel conditioning 4. Noise and hits
PREREQUISITES	DC1

REQUISITE FOR	None
ACTIVITIES	None

SUBJECT NAME	**DATA COMMUNICATIONS**
KNOWLEDGE UNIT TAG AND NAME	**DC3: DATA TRANSMISSION VIA ANALOG SIGNALS**
DESCRIPTION	Use of analog signals for data transmission. Includes modulation, error rates, noise, multiplexing
TOPICS	Minimum number of hours: 3 1. Basic modem operations & definitions 2. Frequency shift keying 3. Phase shift keying: PSK, QPSK, OQPSK, 8PSK, QAM, DBPSK 4. Synchronization 5. Signal quality 6. Software for data transmission
PREREQUISITES	DC1
REQUISITE FOR	DC4, DC7
ACTIVITIES	Closed Laboratory Construction and testing of an analog data transmission system to interconnect two or more computers.

SUBJECT NAME	**DATA COMMUNICATIONS**
KNOWLEDGE UNIT TAG AND NAME	**DC4: DATA TRANSMISSION VIA DIGITAL SIGNALS**
DESCRIPTION	Use of digital signals for data transmission. Includes parallel vs serial, Standards PCM, TDM, compounding, delta modulation, CODECS, error rates, noise
TOPICS	Minimum number of hours: 6 1. Serial data transmission, basic definitions such as baud rate 2. Serial transmission standards (RS232, 422, 423, 449) 3. Parallel transmission standards (Centronix, IEEE 488) 4. Multiplexing (PCM, TDM) 5. Analog data encoding schemes (compounding, Delta modulation CODECs) 6. Error rates, noise
PREREQUISITES	DC3

REQUISITE FOR	DC5, DC6, DC7
ACTIVITIES	Closed Laboratory Construction and testing of a digital data transmission system to a computer.

SUBJECT NAME	**DATA COMMUNICATIONS**
KNOWLEDGE UNIT TAG AND NAME	**DC5: PROTOCOLS**
DESCRIPTION	Rules for data transmission; commonly used system protocols
TOPICS	Minimum number of hours: 3 1. Data link control characters & message format 2. Bit oriented vs. byte oriented 3. Error checking 4. OSI model
PREREQUISITES	DC4
REQUISITE FOR	None
ACTIVITIES	Closed Laboratory Design, construct and test a digital link between two computers. Write software to service the link which includes hand-shaking protocol for data interchange and error correction scheme.

SUBJECT NAME	**DATA COMMUNICATIONS**
KNOWLEDGE UNIT TAG AND NAME	**DC6: OPTICAL DATA TRANSMISSION**
DESCRIPTION	Characteristics of optical fibers and their use in data transmission
TOPICS	Minimum number of hours: 3 1. Fiber characteristics 2. Light transmission 3. Types of optical fibers & their parameters 4. Losses 5. Sources and detectors
PREREQUISITES	DC4
REQUISITE FOR	None
ACTIVITIES	Closed Laboratory Construction and testing of an interconnection of two or more computers

using a fiber optic cable and appropriate connectors.

SUBJECT NAME	**DATA COMMUNICATIONS**
KNOWLEDGE UNIT TAG AND NAME	**DC7: LOCAL AREA NETWORKS**
DESCRIPTION	Types and characteristics of local area networks. Includes equipment used and system considerations.
TOPICS	Minimum number of hours: 9 1. Network configurations--advantages and disadvantages 2. IEEE Standard 802 3. Network types with an expanded discussion on Ethernet 4. Network considerations and calculations 5. Networking software
PREREQUISITES	DC3, DC4
REQUISITE FOR	None
ACTIVITIES	Closed Laboratory Configure and install a local area network connecting a file server to two or more computers and one output device.

SUBJECT NAME	**DEVICES**
KNOWLEDGE UNIT TAG AND NAME	**DV1: AMPLIFIER CONCEPTS AND CHARACTERISTICS**
DESCRIPTION	Amplifier concepts and characteristics including gain, bandwidth, input and output impedance
TOPICS	Minimum number of hours: 2 1. Amplifier concepts 2. Operational characteristics
PREREQUISITES	CR8, CR9
REQUISITE FOR	DV2
ACTIVITIES	None

SUBJECT NAME	**DEVICES**
KNOWLEDGE UNIT TAG AND NAME	**DV2: OPERATIONAL AMPLIFIERS**
DESCRIPTION	Characteristics, models, and circuit applications of operational amplifiers.
TOPICS	Minimum number of hours: 6 1. Operational characteristics of op amps 2. Models 3. Linear applications (inverting and non-inverting amplifier, filters) 4. Non-linear applications (comparators, waveshaping, waveform generation)
PREREQUISITES	DV1
REQUISITE FOR	DV3, LC1, LC2
ACTIVITIES	Closed Laboratory Design, construct, and test circuits containing operational amplifiers.

SUBJECT NAME	**DEVICES**
KNOWLEDGE UNIT TAG AND NAME	**DV3: LINEAR INTEGRATED CIRCUITS**
DESCRIPTION	Linear integrated circuits including comparators, voltage regulators and timers
TOPICS	Minimum number of hours: 3 1. Comparators 2. Voltage regulators 3. Timers
PREREQUISITES	DV2
REQUISITE FOR	LC1
ACTIVITIES	Closed Laboratory Construct and test circuits which include one or more of the following

comparators, voltage regulators, and/or timers.

SUBJECT NAME	**DEVICES**
KNOWLEDGE UNIT TAG AND NAME	**DV4: DIODES**
DESCRIPTION	Diodes used for rectification and voltage regulation
TOPICS	Minimum number of hours: 3 1. Diode operation and characteristics 2. Semiconductor diode types 3. Rectifier circuits 4. Zener diode applications
PREREQUISITES	CR6
REQUISITE FOR	DV5, LC1
ACTIVITIES	Closed Laboratory Construction of rectifier and voltage regulator circuits.

SUBJECT NAME	**DEVICES**
KNOWLEDGE UNIT TAG AND NAME	**DV5: TRANSISTORS**
DESCRIPTION	Transistors used in switching applications
TOPICS	Minimum number of hours: 4 1. Transistor operational characteristics (bipolar and field-effect transistors) 2. Switching applications
PREREQUISITES	DV4
REQUISITE FOR	DV6
ACTIVITIES	Closed Laboratory Construct and test transistor switching circuits.

SUBJECT NAME	**DEVICES**
KNOWLEDGE UNIT TAG AND NAME	**DV6: OPTO-ELECTRONIC DEVICES**
DESCRIPTION	Opto-electronic devices and their application. Includes light-emitting diodes, photo-couplers.
TOPICS	Minimum number of hours: 2 1. Light-emitting diodes 2. Photodiodes 3. Phototransistors 4. Opto-couplers
PREREQUISITES	DV5
REQUISITE FOR	DV7
ACTIVITIES	Closed Laboratory Construction and testing of circuits containing opto-electronic devices.

SUBJECT NAME	**DEVICES**
KNOWLEDGE UNIT TAG AND NAME	**DV7: DEVICE MODELING AND SYSTEM SIMULATION**
DESCRIPTION	Modeling of electronic devices and computer analysis of systems containing these devices.
TOPICS	Minimum number of hours: 6 1. Device specifications and computer models 2. System simulation software 3. Simulation applications
PREREQUISITES	DV6
REQUISITE FOR	LC6
ACTIVITIES	Closed Laboratory Comparison of system simulation results to actual experimental data.

SUBJECT NAME	**DIGITAL CIRCUITS AND SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**DG1: NUMBER SYSTEMS**
DESCRIPTION	Representation of quantities in common number systems. Includes discussion of the characteristics of number systems and conversions between number systems.
TOPICS	Minimum number of hours: 3 1. Characteristics of number systems 2. Binary number system 3. Binary coded decimal 4. Hexadecimal number system 5. Conversions between numbers systems
PREREQUISITES	None
REQUISITE FOR	DG2
ACTIVITIES	None

SUBJECT NAME	**DIGITAL CIRCUITS AND SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**DG2: BOOLEAN ALGEBRA**
DESCRIPTION	Presents a mathematical system to describe the operation of logic circuits. Includes Boolean algebra rules, evaluation of logic expressions, and the use of logic expressions in describing logic circuit operation.
TOPICS	Minimum number of hours: 3 1. Boolean algebra rules 2. Evaluating Boolean expressions 3. Simplifying Boolean expressions
PREREQUISITES	DG1
REQUISITE FOR	DG3

ACTIVITIES None

SUBJECT NAME	**DIGITAL CIRCUITS AND SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**DG3: LOGIC FUNCTIONS AND DEVICES**
DESCRIPTION	Basic logic functions used in the design and implementation of digital circuits. Includes operating characteristics, schematic representation, and logic analysis techniques.
TOPICS	Minimum number of hours: 4 1. Logic function definitions 2. Function tables 3. Logic schematic symbols 4. Analyzing multiple gate circuits 5. Developing logic expressions from schematic 6. Developing schematics from logic expressions
PREREQUISITES	DG2
REQUISITE FOR	DG4, DG6
ACTIVITIES	Closed Laboratory Build and test simple combinational logic circuits. Design simple combinational circuits.

SUBJECT NAME	**DIGITAL CIRCUITS AND SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**DG4: LOGIC SIMPLIFICATION METHODS**
DESCRIPTION	Circuit reduction techniques to reduce the size of logic circuits, including both manual and computer-based techniques.
TOPICS	Minimum number of hours: 3 1. Karnaugh mapping techniques 2. Computer simplification tools
PREREQUISITES	DG3
REQUISITE FOR	DG5
ACTIVITIES	Closed Laboratory Use of computer-based logic simplification software.

SUBJECT NAME	**DIGITAL CIRCUITS AND SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**DG5: COMPLEX COMBINATIONAL CIRCUITS**
DESCRIPTION	Complex combinational circuits as building blocks of larger digital systems. Includes operational characteristics and applications of multiplexers and demultiplexers, encoders and decoders, and programmable logic devices.
TOPICS	Minimum number of hours: 6 1. Multiplexers and demultiplexers 2. Encoders and decoders 3. Programmable logical devices (PLDs) 4. Applications of complex combinational circuits
PREREQUISITES	DG4
REQUISITE FOR	DG8
ACTIVITIES	Closed Laboratory Construction, and testing of circuits using complex combinational circuits. Use of computer simulation tools.

SUBJECT NAME	**DIGITAL CIRCUITS AND SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**DG6: SEQUENTIAL LOGIC DEVICES AND THEIR APPLICATIONS**
DESCRIPTION	Basic latches and flip-flop circuits. Includes operational characteristics and analysis techniques. Design, construction, and testing of sequential logic circuits; e.g., counters and registers.
TOPICS	Minimum number of hours: 6 1. Latches, Flip-flops (D, R-S, J-K) 2. Timing parameters 3. Monostable multivibrators 4. Applications (counters and registers)
PREREQUISITES	DG3
REQUISITE FOR	DG7, DG8, DG9, DC1, LC3, MS1
ACTIVITIES	Closed Laboratory

Design, build, and test sequential logic circuits. Use oscilloscopes and logic analyzers to test and verify circuit operation. Use of computer simulation tools.

SUBJECT NAME	**DIGITAL CIRCUITS AND SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**DG7: DATA CONVERSION**
DESCRIPTION	The conversion of information between analog and digital formats. Includes A/D and D/A conversion techniques and their operational characteristics.
TOPICS	Minimum number of hours: 4 1. Digital-to-Analog conversion 2. Analog-to-Digital conversion 3. Commercial D/A and A/D converters 4. D/A and A/D specifications
PREREQUISITES	DG6
REQUISITE FOR	None
ACTIVITIES	Closed Laboratory Prototype and test D/A and A/D converter systems.

SUBJECT NAME	**DIGITAL CIRCUITS AND SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**DG8: STATE MACHINES**
DESCRIPTION	Synchronous state machines as a basic building block of digital systems. Includes design techniques, hazards, and test techniques.
TOPICS	Minimum number of hours: 6 1. Synchronous state machine models 2. Design and implementation techniques 3. Hazard conditions 4. Testing state machines
PREREQUISITES	DG5, DG6
REQUISITE FOR	None
ACTIVITIES	Closed Laboratory

Construct simple state machines using PLDs. Use of computer simulation tools. Design, build and test more complex sequential circuits.

SUBJECT NAME	**DIGITAL CIRCUITS AND SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**DG9: LOGIC ANALYSIS TECHNIQUES**
DESCRIPTION	Logic analysis techniques used in the design, testing, and troubleshooting of digital systems. Includes computer simulation and timing, and state analysis techniques.
TOPICS	Minimum number of hours: 6 1. Timing analysis 2. State analysis 3. Computer simulation 4. Schematic capture 5. Device programming techniques
PREREQUISITES	DG6
REQUISITE FOR	MS3
ACTIVITIES	Closed Laboratory Use of logic analyzers and computer-based simulation schematic capture and device programming software.

SUBJECT NAME	**LINEAR INTEGRATED CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**LC1: VOLTAGE AND CURRENT REFERENCES**
DESCRIPTION	Characteristics and applications of devices used as voltage and current references.
TOPICS	Minimum number of hours: 3 1. Zener diodes 2. Bandgap references 3. Constant current devices 4. Circuits for voltage and current sources
PREREQUISITES	DV2-DV4
REQUISITE FOR	None
ACTIVITIES	Open Laboratory

Construction and testing of voltage and current reference circuits.

SUBJECT NAME	**LINEAR INTEGRATED CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**LC2: FILTER CONCEPTS AND CIRCUITS**
DESCRIPTION	Introduction to passive and active filter circuits and their characteristics.
TOPICS	Minimum number of hours: 6 1. Filter types and characteristics 2. Passive filters 3. Active filters 4. Switched capacitor filters
PREREQUISITES	DV2, CR10
REQUISITE FOR	LC3, LC4
ACTIVITIES	Open Laboratory Construction and testing of passive, active and switched capacitor filters.

SUBJECT NAME	**LINEAR INTEGRATED CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**LC3: PHASE LOCKED LOOPS**
DESCRIPTION	Basic concepts and operation of phase locked loops (PLL).
TOPICS	Minimum number of hours: 3 1. PLL concepts 2. PLL design considerations 3. PLL applications
PREREQUISITES	LC2, DG6
REQUISITE FOR	None
ACTIVITIES	Open Laboratory Completion and testing of a phase locked loop based project.

SUBJECT NAME	**LINEAR INTEGRATED CIRCUITS**
KNOWLEDGE UNIT TAG AND NAME	**LC4: OSCILLATORS**
DESCRIPTION	Basic oscillator concepts, different types of oscillators and their characteristics.
TOPICS	Minimum number of hours: 3 1. Oscillator operation 2. Relaxation oscillators 3. RL and LC oscillators 4. Crystal oscillators
PREREQUISITES	DV2, LC2
REQUISITE FOR	None
ACTIVITIES	Open Laboratory Construction and testing of various oscillator circuits.

SUBJECT NAME	**LINEAR SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**LC5: GROUNDING AND SHIELDING**
DESCRIPTION	An examination of interference sources and methods for eliminating interference
TOPICS	Minimum number of hours: 4 1. Types and sources of interference 2. Grounding techniques 3. Shielding 4. Isolation techniques
PREREQUISITES	CR9
REQUISITE FOR	LC6
ACTIVITIES	Closed Laboratory

Demonstration of proper grounding and shielding techniques and problems encountered when improper techniques are used.

SUBJECT NAME	**LINEAR SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**LC6: MODELING AND SIMULATION**
DESCRIPTION	The use of computer simulation software to analyze circuits which include discrete devices and linear integrated circuits.
TOPICS	Minimum number of hours: 8 1. Models of diodes and transistors 2. Models of linear integrated circuits 3. DC analysis 4. AC analysis 5. Frequency response analysis
PREREQUISITES	LC5, DV7
REQUISITE FOR	None
ACTIVITIES	Open laboratory Modeling, simulation and testing of circuits containing linear integrated circuits.

SUBJECT NAME	**MICROPROCESSOR SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**MS1: MACHINE ORGANIZATION**
DESCRIPTION	Building blocks of a microprocessor system and their interconnections including basic machine operation.
TOPICS	Minimum number of hours: 5 1. µP system elements (CPU, ROM, RAM, I/O) 2. Bus structure 3. Fetch execute operation 4. Register concepts 5. I/O operations
PREREQUISITES	DG6
REQUISITE FOR	MS2

ACTIVITIES	Closed Laboratory Examination of signals in a microprocessor system.

SUBJECT NAME	**MICROPROCESSOR SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**MS2: ASSEMBLY LANGUAGE PROGRAMMING**
DESCRIPTION	Examination of a microprocessor instruction set and how the instructions are used.
TOPICS	Minimum number of hours: 9 1. Instruction types (arithmetic and logical, data moves, control & I/O) 2. Addressing 3. Flags 4. Use of the stack 5. Assemblers
PREREQUISITES	MS1
REQUISITE FOR	MS3
ACTIVITIES	Closed Laboratory Coding and testing of short assembly language programs that use flags, the stack, and different addressing modes.

SUBJECT NAME	**MICROPROCESSOR SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**MS3: HARDWARE SPECIFICATIONS**
DESCRIPTION	An examination of the specifications of the various elements of a microprocessor system and their relationship to system performance.
TOPICS	Minimum number of hours: 3 1. Memory - types, organization, and speed 2. CPU timing and speed 3. I/O operation
PREREQUISITES	DG9, MS2
REQUISITE FOR	MS4, MS6
ACTIVITIES	Open Laboratory Use of an oscilloscope and logic analyzer to examine signals within a

microprocessor system.

SUBJECT NAME	**MICROPROCESSOR SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**MS4: INTERFACING**
DESCRIPTION	Examination of hardware and software interfacing considerations
TOPICS	Minimum number of hours: 12 1. I/O devices and their operation 2. Interrupt operation 3. Synchronous and asynchronous communication devices 4. Direct memory access
PREREQUISITES	MS3
REQUISITE FOR	MS5
ACTIVITIES	Open Laboratory Design, build and test system operation using several different types of I/O devices.

SUBJECT NAME	**MICROPROCESSOR SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**MS5: SYSTEM LEVEL IMPLEMENTATION**
DESCRIPTION	A project oriented unit where the student is required to design, build, write the software, and test a microprocessor based system to perform a specific task.
TOPICS	Minimum number of hours: 6 1. System specifications 2. System implementation 3. Emulators, development systems, evaluation boards
PREREQUISITES	MS4
REQUISITE FOR	None
ACTIVITIES	Open Laboratory

Design, build and test of a microprocessor based system.

SUBJECT NAME	**MICROPROCESSOR SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**MS6: ALTERNATIVE ARCHITECTURES**
DESCRIPTION	An examination of other types of architectures for microprocessor systems.
TOPICS	Minimum number of hours: 3 1. Microcontrollers 2. Bit slice 3. Digital signal processors
PREREQUISITES	MS3
REQUISITE FOR	None
ACTIVITIES	None

SUBJECT NAME	**OPERATING SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**OS1: OPERATING SYSTEM MODELS**
DESCRIPTION	Introduction to concepts of virtual machines, resources and operating system types.
TOPICS	Minimum number of hours: 3 1. Virtual machine software layers and resources 2. Personal computing operating systems 3. Operating system for real time and embedded systems 4. Time sharing operating systems
PREREQUISITES	PM5
REQUISITE FOR	OS2
ACTIVITIES	None

SUBJECT NAME	**OPERATING SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**OS2: PROCESSOR MANAGEMENT**
DESCRIPTION	The control scheduling and allocation of the processors in a computing system.
TOPICS	Minimum number of hours: 6 1. Multiprogramming and multiprocessing concepts 2. Concurrence control mechanisms and deadlocks 3. Multitasking 4. Scheduling
PREREQUISITES	OS1
REQUISITE FOR	OS3
ACTIVITIES	None

SUBJECT NAME	**OPERATING SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**OS3: DEVICE CONTROL**
DESCRIPTION	Control scheduling and allocation of input/output equipment in a computing system.
TOPICS	Minimum number of hours: 6 1. Input/Output control, device sharing 2. Synchronization 3. Buffering 4. Scheduling
PREREQUISITES	OS2
REQUISITE FOR	0S4
ACTIVITIES	None

SUBJECT NAME	**OPERATING SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**OS4: MEMORY MANAGEMENT**
DESCRIPTION	Memory allocation and control in a computer system.
TOPICS	Minimum number of hours: 6 1. Address interpretation, relocation, segmentation 2. Allocation of memory 3. Swapping and paging 4. Virtual memory
PREREQUISITES	OS3
REQUISITE FOR	OS5
ACTIVITIES	Open Laboratory Examine actual memory allocation in a computer system using system diagnostic software or memory dump.

SUBJECT NAME	**OPERATING SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**OS5: FILE SYSTEM MANAGEMENT**
DESCRIPTION	Organization, management and control of disk storage space in a computing system.
TOPICS	Minimum number of hours: 6 1. Disk storage and organization 2. File systems 3. Scheduling and optimization 4. Allocation and access control
PREREQUISITES	OS4
REQUISITE FOR	OS6
ACTIVITIES	Open Laboratory Use diagnostic software to examine the file naming and disk allocation of a

computer.

SUBJECT NAME	**OPERATING SYSTEMS**
KNOWLEDGE UNIT TAG AND NAME	**OS6: OPERATING SYSTEM CASE STUDY**
DESCRIPTION	Examination of the components, organization, user interface and operation of an actual operating system.
TOPICS	Minimum number of hours: 12 1. System components and organization 2. User interface 3. Commands processing 4. System processing 5. Memory, input/output, files and devices management
PREREQUISITES	OS5
REQUISITE FOR	None
ACTIVITIES	Open Laboratory A series of laboratory assignments examining the use, structure and operation of an actual operating system.

SUBJECT NAME	**PROGRAM DESIGN TECHNIQUES**
KNOWLEDGE UNIT TAG AND NAME	**PD1: SOFTWARE ENGINEERING CONCEPTS**
DESCRIPTION	Introduction to terminology, concepts and methodology of software engineering.
TOPICS	Minimum number of hours: 3 1. The software development cycle 2. Structured programming: modularization, interaction and top-down design 3. Maintainability, testability and documentation
PREREQUISITES	PM5
REQUISITE FOR	PD2
ACTIVITIES	None

SUBJECT NAME	**PROGRAM DESIGN TECHNIQUES**
KNOWLEDGE UNIT TAG AND NAME	**PD2: REQUIREMENTS, SPECIFICATION AND SOFTWARE DEVELOPMENT**
DESCRIPTION	Methodology for developing a software design from requirements and specifications.
TOPICS	Minimum number of hours: 6 1. The planning cycle: development of a process model and choice of method 2. Tools of structured design: structure charts, specification language, flow charts, hierarchical diagrams 3. Input/output specifications 4. Functional specification
PREREQUISITES	PD1
REQUISITE FOR	PD3
ACTIVITIES	Open Laboratory Program exercises which emphasize development of specifications from an explicit statement of requirements.

SUBJECT NAME	**PROGRAM DESIGN TECHNIQUES**
KNOWLEDGE UNIT TAG AND NAME	**PD3: ALGORITHMS AND DATA STRUCTURES**
DESCRIPTION	Illustration of algorithms and data structures together with methods for their manipulation and use.
TOPICS	Minimum number of hours: 8 1. Array manipulation 2. Sorting and searching 3. Use of queues and stacks 4. Use of sequential storage and linked lists 5. Use of trees and traversals
PREREQUISITES	PD2
REQUISITE FOR	PD4

ACTIVITIES	Open Laboratory Program exercises which emphasize the development of an algorithm and its translation into symbolic code.

SUBJECT NAME	**PROGRAM DESIGN TECHNIQUES**
KNOWLEDGE UNIT TAG AND NAME	**PD4: SOFTWARE DESIGN, DOCUMENTATION AND IMPLEMENTATION**
DESCRIPTION	Translation of functional design, specification and chosen algorithm into commented testable and maintainable code.
TOPICS	Minimum number of hours: 8 1. Order of modular development 2. Isolation of modules 3. Design for testability, reliability and maintainability 4. Coding and documentation
PREREQUISITES	PD3
REQUISITE FOR	PD5, ST1
ACTIVITIES	Open Laboratory Program exercises which emphasize standards, modules, testability and proper documentation.

SUBJECT NAME	**PROGRAM DESIGN TECHNIQUES**
KNOWLEDGE UNIT TAG AND NAME	**PD5: SOFTWARE INTEGRATION, TESTING AND VALIDATION**
DESCRIPTION	Testability, failure analysis, debugging and integration of programming modules.
TOPICS	Minimum number of hours: 8 1. Testing: modular levels and integration 2. Choice of test data 3. Analysis of failure modes 4. Debugging, corrections, tools and aids
PREREQUISITES	PD4
REQUISITE FOR	PD6
ACTIVITIES	Open Laboratory Program exercises which emphasize failure analysis, choice of test data,

testability and the use of trouble-shooting aids.

SUBJECT NAME	**PROGRAM DESIGN TECHNIQUES**
KNOWLEDGE UNIT TAG AND NAME	**PD6: TEAM PROGRAMMING MANAGEMENT**
DESCRIPTION	An introduction to software management and working as part of a production team. A capstone study is intended here.
TOPICS	Minimum number of hours: 12 1. Project phases and planning cycle, standards and efficiency 2. Configuration control, revisions, deviations and changes 3. Standards for coding, documentation, testability 4. Teamwork and presentations
PREREQUISITES	PD5
REQUISITE FOR	None
ACTIVITIES	Open Laboratory Modification and use of a large scale software project.

SUBJECT NAME	**PROGRAMMING METHODOLOGY**
KNOWLEDGE UNIT TAG AND NAME	**PM1: HISTORY OF PROGRAMMING LANGUAGES**
DESCRIPTION	A brief historical survey of major developments in programming languages, beginning with the evolution of procedural high-level languages. An overview of contemporary programming paradigms and their related languages, including procedural, functional, logic, object-oriented, and parallel programming.
TOPICS	Minimum number of hours: 2 1. Early languages ALGOL, FORTRAN, COBOL 2. The evolution of procedural languages (e.g. ALGOL, C, Modula-2, Pascal, PL/1) 3. Non-procedural paradigms and languages; functional (e.g., Lisp), logic (e.g., Prolog), object-oriented (e.g., Smalltalk), and parallel (e.g., Occam)
PREREQUISITES	None

REQUISITE FOR	PM2
ACTIVITIES	None

SUBJECT NAME	**PROGRAMMING METHODOLOGY**
KNOWLEDGE UNIT TAG AND NAME	**PM2: INTRODUCTION TO A PROGRAMMING LANGUAGE**
DESCRIPTION	An introduction to the syntactic and execution characteristics of a contemporary programming language (such as Pascal, C, or Ada) and their use in the construction and execution of complete programs that solve simple algorithmic problems.
TOPICS	Minimum number of hours: 12 1. Type declarations integer, real, Boolean, character, and string 2. Arithmetic, assignment statements, conditional statements, and loops 3. Procedures, functions and parameters 4. Arrays and records 5. Overall program structure
PREREQUISITES	PM1
REQUISITE FOR	PM3
ACTIVITIES	Open Laboratory Students should gain experience with compiling, finding and correcting syntax errors, and running a minimum of three programs that solve elementary algorithmic problems.

SUBJECT NAME	**PROGRAMMING METHODOLOGY**
KNOWLEDGE UNIT TAG AND NAME	**PM3: DATA TYPES**
DESCRIPTION	Elementary and structured data types. Creation of user-defined data types & applications.
TOPICS	Minimum number of hours: 3 1. Choice and representation of elementary data types; integer, real, Boolean, and character 2. Specification and representation of structured data types; arrays, records, sets, pointers and dynamic storage allocation
PREREQUISITES	PM2
REQUISITE FOR	PM4

ACTIVITIES	Open Laboratory Solve a programming problem requiring a structured, dynamically-allocated variable. Student gains experience with language support for structured, dynamic data types.

SUBJECT NAME	**PROGRAMMING METHODOLOGY**
KNOWLEDGE UNIT TAG AND NAME	**PM4: ALGORITHMS AND PROGRAM DESIGN**
DESCRIPTION	Design and implementation of solutions to problems using algorithmic processes and top-down design.
TOPICS	Minimum number of hours: 6 1. Definition of algorithm 2. Use of algorithms in problem solving 3. Program implementation
PREREQUISITES	PM3
REQUISITE FOR	PM5
ACTIVITIES	Open Laboratory Design and develop programs to solve algorithmic-type problems.

SUBJECT NAME	**PROGRAMMING METHODOLOGY**
KNOWLEDGE UNIT TAG AND NAME	**PM5: PROGRAM DOCUMENTATION, TESTING AND VERIFICATION**
DESCRIPTION	General requirements for program documentation. Elements of software engineering as related to program testing and verification.
TOPICS	Minimum number of hours: 3 1. Use of program headers 2. Program comments 3. Data traps and filters 4. Verification of program performance
PREREQUISITES	PM4
REQUISITE FOR	OS1, PD1, QC1
ACTIVITIES	Open Laboratory Assign a programming project that is vulnerable to faulty execution, and

which ties together the knowledge units of program methodology.

SUBJECT NAME	**QUALITY CONTROL**
KNOWLEDGE UNIT TAG AND NAME	**QC1: RELIABILITY AND MAINTAINABILITY**
DESCRIPTION	Introduction to basic concepts of hardware and software reliability and maintainability
TOPICS	Minimum number of hours: 3 1. Hardware reliability 2. Hardware maintainability 3. Software reliability 4. Software maintainability
PREREQUISITES	PM5
REQUISITE FOR	QC2
ACTIVITIES	None

SUBJECT NAME	**QUALITY CONTROL**
KNOWLEDGE UNIT TAG AND NAME	**QC2: STATISTICAL METHODS**
DESCRIPTION	Introduction to statistical methods for the collection and analyses of data from a population.
TOPICS	Minimum number of hours: 8 1. Probability 2. Statistical distributions 3. Sampling 4. Statistical process control
PREREQUISITES	QC1
REQUISITE FOR	QC3

ACTIVITIES	Closed Laboratory Collect data from a population or a sample from a population and develop appropriate statistics from that data.

SUBJECT NAME	**QUALITY CONTROL**
KNOWLEDGE UNIT TAG AND NAME	**QC3: RELIABILITY EVALUATION**
DESCRIPTION	Methods of evaluating reliability of hardware and software elements and systems.
TOPICS	Minimum number of hours: 9 1. Hardware reliability parameters (MTBF, failure rate) 2. Hardware reliability testing (Burn-in, thermal cycling, MIL STD 217) 3. Hardware reliability data sources (MIL STD 883) 4. Software reliability considerations (verification, validation, error recovery)
PREREQUISITES	QC2
REQUISITE FOR	QC4
ACTIVITIES	Closed Laboratory Perform validation and verification including error recovery for a system which has been deliberately flawed.

SUBJECT NAME	**QUALITY CONTROL**
KNOWLEDGE UNIT TAG AND NAME	**QC4: FAILURE MODE EFFECTS ANALYSIS**
DESCRIPTION	Examination of failures and their effect on system operation
TOPICS	Minimum number of hours: 6 1. Failure modes and types 2. Effects of failures on system operation 3. Failure avoidance and minimization of catastrophic failures 4. Software failure modes
PREREQUISITES	QC3
REQUISITE FOR	None
ACTIVITIES	Closed Laboratory Failure mode analysis of a simple digital system under a variety of failure

modes.

SUBJECT NAME	**SOFTWARE TESTING AND VALIDATION**
KNOWLEDGE UNIT TAG AND NAME	**ST1: CONCEPTS OF TESTING AND VALIDATION**
DESCRIPTION	An introduction to the terminology and concepts of software testing and validation.
TOPICS	Minimum number of hours: 3 1. Errors and the cost of software 2. Designing reliable software 3. Types of testing 4. Specifications and the choice of test data 5. Analysis of failure modes 6. Theory of bugs
PREREQUISITES	PD4
REQUISITE FOR	ST2
ACTIVITIES	None

SUBJECT NAME	**SOFTWARE TESTING AND VALIDATION**
KNOWLEDGE UNIT TAG AND NAME	**ST2: MODULE TESTING**
DESCRIPTION	Designing tests for, and testing program modules. Standards which enhance testability.
TOPICS	Minimum number of hours: 6 1. Internal module (White-box) testing 2. External module (Black-box) testing 3. Test case design 4. Documentation and standards
PREREQUISITES	ST1
REQUISITE FOR	ST3

ACTIVITIES	Open Laboratory Programming exercise in which the student is given a program module and its specifications, and designs logic and I/O testing for that module.

SUBJECT NAME	**SOFTWARE TESTING AND VALIDATION**
KNOWLEDGE UNIT TAG AND NAME	**ST3: INTEGRATION AND SYSTEM TESTS**
DESCRIPTION	Various types of testing for integration of modules and complete systems are described and illustrated.
TOPICS	Minimum number of hours: 6 1. Inter-module connections 2. Top-down and bottom-up testing 3. Cause-effect graphing 4. System integration 5. Tools
PREREQUISITES	ST2
REQUISITE FOR	ST4
ACTIVITIES	Open Laboratory Given modules and system specifications, the student integrates and the modules and designs testing for the system.

SUBJECT NAME	**SOFTWARE TESTING AND VALIDATION**
KNOWLEDGE UNIT TAG AND NAME	**ST4: OTHER TYPES OF TESTS AND VALIDATION**
DESCRIPTION	Testing and validation of software systems
TOPICS	Minimum number of hours: 6 1. Acceptance tests 2. Installation testing 3. Debugging and tools 4. Corrections
PREREQUISITES	ST3
REQUISITE FOR	None
ACTIVITIES	Open Laboratory Given a software system, student groups modify it to meet appropriate standards and design acceptance and installation testing.

PART III

CURRICULA IMPLEMENTATION SAMPLES

1.0 CURRICULA OVERVIEW

It is a purpose of this report to allow individual institutions of higher learning to develop their own courses and curricula based on the prescribed knowledge units. Flexibility is essential to insure the recommendations of this report do not conflict with the character and mission of each institution. This flexibility allows individual institutions to balance the needs of industry and transferability to upper-division programs.

1.1 Curricula Course Structure

The subject areas and knowledge units listed in Part II this report specify the scope of topics all students should study in a particular discipline of computing and engineering technology. The lecture hours associated with each knowledge unit gives an approximate indication of the time required to achieve a depth of knowledge beyond conceptual understanding (a level of at least *3* as defined in Section 1.3). The prerequisite structure suggests that some sequencing is required in courses developed from the knowledge units.

The knowledge units in one subject area do not have to be grouped into one course or a series of courses, but may be distributed among several courses with proportional time spans. If the minimum number of lecture hours for a particular knowledge unit is *n*, then it is possible to split the knowledge unit among two courses such that one course has *x* lecture hours while the other course has *y* lecture hours for the same knowledge unit, with $x + y = n$. Curriculum designers at each institution must ensure that the full treatment of the knowledge unit is covered within the total curriculum.

Some knowledge units may be covered either in a standard class setting or in a laboratory setting. However, all knowledge units of a discipline should be included in the program of study for all students. Thus, the knowledge units can be combined in various ways to form courses. In this activity, the following guidelines should be followed.

- Knowledge units should be combined so that the composite subject matter of a course forms a coherent body of topics for the student.
- The combined set of courses that comprise the discipline should have a prerequisite structure that is consistent with the prerequisite structure among the constituent knowledge units.
- The combined set of courses must cover all of the knowledge units that make up the curriculum.

Implementations may, of course, exceed the minimum by assigning additional depth of coverage or topics beyond those suggested. Also, not every knowledge unit of each subject area need be covered in each curriculum.

The design of courses contain the following components: (See Figure III-1)

Course Title
A clear but brief name of the course (not subject area) with course code or number.

Number of Semester Hours
To include the total number of semester credits and hours, preferably with the format:
Lecture: (number of credits : number of hours)
Laboratory: (number of credits : number of hours)

Prerequisites
Courses that are to be completed prior to taking the course being described.

Goal or Purpose of Course
A brief statement, one or two sentences, expressing what the course should accomplish.

Objectives for Students
Knowledge, skills and/or capabilities expected from students who successfully complete the course. Conventionally, these abilities are enumerated and preceded by the phrase: *Upon completion of the course, the student should be able to ...*

EET 130 Digital Circuits Lecture (2:2)
Laboratory (1:3)

Prerequisite:
Intermediate algebra

Goal:
A fundamental course in combinational circuits, flip-flops, and latches.

Objective:
Upon completion of the course, the student should be able to:
- Analyze combinational logic circuits in terms of function and timing.
- Express circuit function in terms of Boolean expressions.
- Design, implement, and test combinational circuits.
- Analyze, implement, and test flip-flop and latch circuits.
- Use multimeters, oscilloscopes, and logic analyzers to analyze and troubleshoot digital circuits.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
CR1	1/3	2	T
CR2	1/6	3	T,A
CR3	1/3	2	T
CR4	1/6	3	T,A,D
DG1	3/3	3	T
DG2	3/3	4	T
DG3	4/4	4	T,A,D
DG4	3/3	4	T,A
DG5	6/6	4	T,A,D
DG6	6/6	4	T,A,D

Activities:
Analyze, build, test, and troubleshoot combinational and sequential logic circuits and systems. Operate and use multimeters, oscilloscopes, and logic analyzers to verify circuit function and timing. (Consult *Activities* component in the respective knowledge units.)

Figure III-2 Course Format from Knowledge Units

Subject Matter
A table with the following four columns:

(a) Knowledge Unit Tag.

(b) Fraction of integers, *x/n*, depicting the number of hours *x* of the knowledge unit devoted to the course compared the minimum number of hours required *n*.

(c) Depth of knowledge expected to be achieved by the knowledge unit within the course as given by the *depth indicator*, an integer 1 through 5. This number is not necessarily the exit competency expected from the curriculum.

(d) Emphasis of knowledge manifested by an emphasis on theory, analysis or design.

Activities Component
An explanation of the need for laboratory or exercise experiences, if any; description of the experiences needed, equipment or materials required. Suggestions for the type of experience should be provided, including whether the laboratory uses an open or closed setting.

It should be noted that the activities component is an essential experience, and should be included in all courses of the curriculum.

Figure III-1 shows an example of how a course is constructed from knowledge units. Many other possibilities do exist and should reflect local institutional needs.

1.2 Emphasis of Knowledge

Courses which are developed from the knowledge units must indicate a particular emphasis according to the paradigm used. Emphasis is divided into three categories: *theory*, *analysis* and *design*. *Theory* corresponds to the fundamental knowledge base of the particular unit and is defined as that which is generally supported by fact or accepted understanding. *Analysis* corresponds to the scientific method used in verifying or testing a theoretical phenomenon, or in discovering a certain principle regarding the subject. With this paradigm, one forms a hypothesis, constructs a model and makes predictions, designs an experiment and collects data, and finally analyzes the results. *Design* corresponds to building an entity (hardware or software) that demonstrates the general theme of the knowledge unit. The knowledge units used to form a course are appended with an *emphasis code* formed by any combination of letters of T, A or D, representing theory, analysis and design, respectively.

As an example, suppose two courses are constructed from the same variety of knowledge units. One course may be entitled, *Computer Programming* while another is called, *Programming Concepts*. The first course may have an emphasis on designing programs while the second course may have an emphasis on program methodology. The material of Section 1.3 further illustrates the concept of course emphases.

1.3 Depth of Knowledge

It is important that those who use the knowledge units to develop courses include an indication of the expected depth of knowledge. This *depth indicator* should be identified by an integer 1 through 5 which is included in the course description. The interpretation of these integers is as follows:

1 Awareness as expressed by general knowledge, definition, or recognition.

2 Description as evidence of conceptual understanding.

3 Differentiation as expressed by the ability to compare/contrast or to make connections with related topics.

4 An application as evidence of using predefined principles, methods or tools in a structured environment.

5 Judgment as evidence of innovative decision making based upon analysis, synthesis and/or evaluation.

The depth of knowledge or understanding as prescribed is considered minimal for each knowledge unit in its course setting. The depth concept is based in part on the taxonomy developed by the Software Engineering Institute [7].

1.4 Exit Competencies

Students completing a curriculum in the area of computing and engineering technology are expected to achieve a level of competency commensurate with those of technical professionals. The index used to indicate the depth of achievement for a curriculum is the same as those used for the depth of knowledge as defined in Section 1.3, which was called a *depth indicator*, an integer 1 through 5. Thus, an institution of higher learning can offer a curriculum composed of knowledge units within courses that would have not only subject content, but an expected depth of understanding for the content.

Consider the CET and EET curricula. Suppose a college distributes these knowledge units among any variety of courses. It is important that those courses include an indication of the desired exit competency or achievement level. This level should be indicated a *depth indicator* for each knowledge unit. For example, among the knowledge units of these curricula would appear the following:

	Exit Competency	
KU Tag	CET	EET
...	...	...
CR4	3	4
PM3	4	-
LC2	-	3
...	...	...

In the first situation (CR4) both curricula contain the same knowledge unit but the expected exit competency for each curriculum is not the same -- *differentiation* (3) for CET and *application* (4) for EET. The second and third situations show exit competencies for distinct knowledge units associated with these curricula. Section 7 contains a listing of typical exit competencies for the EET, CET and SET curricula. Each institution should formulate its own list based on the mission of the institution, student preparation, and community needs.

1.5 Curriculum Model

A curriculum should contain components which state to the public not only a name and course content, but also the purpose of the curriculum, and the subject content with expected achievement upon completion of the curriculum. Each curriculum can be thought of as a *major* which relates to a specific outcome for students and which gives the necessary knowledge and skills to be acquired. The following components should be part of any curriculum.

Curriculum Title and Degree
A brief name for the curriculum which properly depicts the intended purpose of the program. It is important to specify the degree or certificate which will be obtained when the requirements for that program are satisfactorily met.

Purpose
An explanation of the purpose or mission of the curriculum. It should include explanations for each option, if any. The discussion should emphasize the end-product, or outcome that can be expected by the student who completes the curriculum.

Curriculum Course Content
The list of courses that are required in the curriculum, including different options where applicable, followed by the list of courses to be chosen from electives, followed by the list of courses to be chosen from other disciplines.

Subject Matter Content
The list of knowledge units required in each curriculum and for each option in the curriculum, if any. Indication of the required exit competency for each knowledge unit within the curriculum is essential.

Course Sequence
A table or chart giving the anticipated sequence of courses required for the curriculum.

For example, a purpose of a curriculum entitled, *A.S. in Computer Engineering Technology* might be to prepare students for a career in computer system installations. This major may have two options: Manufacturing and Sales. A knowledge unit from this curriculum, for example, CR4, must illustrate, using the *depth indicator*, the exit competency expected for each option in the curriculum. Within the listing for the curriculum, the subject content might show

Curriculum	- - - - - -Options - - - - - -
Knowledge Unit	Manufacturing Sales
CR4	4 3

This would indicate that for this knowledge unit, a greater exit depth of knowledge would be expected from the Manufacturing Option than from the Sales Option.

1.6 Curricula for Programs in Computing and Engineering Technology

The sample courses in Sections 3, 4, and 5 were formulated from the knowledge units described in this report, however the courses are not intended to be prescriptive. They show only one of many ways in which the knowledge units detailed in Part II can be mapped into courses. In turn, these typical courses, or a subset of them, can be mapped into a curriculum.

Figures III-2, III-3 and III-4 that follow demonstrate how courses created from knowledge units can be used to build a curriculum in computer engineering technology, electronic engineering
technology and software engineering technology, respectively. The structure of a curriculum and the courses that are used should reflect the mission of the program at each institution and the needs of its constituency. This is an important feature in designing a curriculum and is necessary to enable each institution and program to preserve its own identity.

Courses in each curriculum are identified by generic names. These names are followed by a notation of the form *(n : x, y)*, where *n* indicates the suggested number of total semester credit hours assigned to the courses, *x* indicates the suggested number of semester credit hours assigned to lectures, and *y* indicates the suggested number of semester credit hours assigned to laboratories. In each case, the sum of the lecture and laboratory credits equals the total number of credits; that is, $n = x + y$.

The three sample curricula are based on semester credits. However, since courses are composed of knowledge units, the curricula may be implemented to other formats such as a quarter-based program of study.

1.7 Commonalities among Curricula

The three curricula have the distribution of subject areas as shown in Table III-1. Section 3.0 shows the sample curricula showing how knowledge units from these subject areas can be combined into courses that form a curriculum. This may result in similar courses having different tag names. (For example, CET 101 is the same as SET 101.) This is done to allow institutions to implement a specific curriculum. Institutions offering more than one curriculum may minimize duplication by combining courses whereby one course can satisfy more than one program.

Subject Area	**EET**	**CET**	**SET**
Circuits	X	X	X
Computer architecture	X	X	X
Computing systems	X	X	X
Data communications	X	X	
Devices	X	X	X
Digital circuits and systems	X	X	X
Linear integrated circuits	X		
Microprocessor systems	X	X	X
Operating systems			X
Program design techniques			X
Programming methodology	X	X	X

Quality control	X	X	X
Software testing and validation			X

Table III-1 **Distribution of Subject Areas**

2.0 UNDERLYING THEMES

The disciplines of computing and engineering technology can convey a wholeness by directly acknowledging and discussing underlying themes as they appear at different times during the students' educational experiences. Done properly, the following benefits can occur:

- Minimizing the perception that this discipline is a fragmented collection of unrelated topics.
- Learning will be facilitated by the presence of generalizations and analogies.

Since underlying themes span all subject areas, they serve to unify the discipline. Furthermore, given a reference to an understood concept, students can assimilate new ideas more easily. The ability to generalize and transfer concepts is important not only in moving into new subject areas in course work, but also embarking on new areas of professional activity.

The *underlying themes* for computing and engineering technology are identified as:

Modeling	Trade-offs
Input/output characteristics	Reuse
Ordering in time	Testability
Levels of analysis	Component interaction
Signal flow analysis	Documentation

Each theme occurs throughout the three disciplines of computing and engineering technology and have a high degree of technological independence. They are more fundamental than any occurrence of their use. These ideas, principles, and processes help to unify the academic discipline, and must be communicated throughout the curriculum so that the student develops an appreciation for the pervasiveness of these concepts and the ability to apply them in appropriate contexts.

The ten themes which occur throughout the three disciplines of computing and engineering technology are described as follows:

> Modeling: The representation of devices in terms of their ideal characteristics and in terms of their practical characteristics. Device models may be mathematical models, equivalent circuit models, and software models.
>
> Trade-offs: Technical trade-offs recognize that there are multiple solutions to a design or engineering problem. These solutions may involve both hardware and software.
>
> Input/output Characteristics: The description of devices and circuits in terms of their signal processing characteristics where the device or circuit receives an input, processes the input information in a prescribed manner, and produces an output signal that characterizes the output information.
>
> Reuse: The process that allows the reapplication of concepts, principles, devices,

and circuits to new problems, thereby building up a knowledge base that serves as the resource for future engineering solutions.

Ordering in time: The ordering of events and processes in time showing the orderly and sequential operation of electronic devices, circuits, and systems as well as critical sequential processes.

Testability: The planning and implementation of the process by which the correct behavior of devices, circuits, and systems are verified using an array of electronic test instruments, computer-based instruments, and test algorithms.

Levels of abstraction: The process of viewing electronic systems at various levels of detail. For example, in terms of hardware: component level, circuit level, subsystem level, and network level, or in terms of software: machine level, assembly level and high-level languages.

Component interaction: How hardware/software work together to effect a predictable desired result.

Signal flow analysis: The tracing and measurement of signals within a system as these signals move from one functional block, device or other signal processing unit to another.

Documentation: The process of recording experimental results, procedures and other information pertinent to the design, implementation, debugging, testing, or other processing of a circuit, system or program.

There may well be other concepts that may be categorized as underlying themes. The concepts may be included in a particular program. The important understanding, however, is that programs are to be designed so that the themes become an integrated part of the student's education and that institutions of learning foster those themes which they consider necessary for the proper implementation of their program.

3.0 SAMPLE CURRICULUM AND COURSES FOR COMPUTER ENGINEERING TECHNOLOGY

The sample curriculum for computer engineering technology assumes a semester-based program consisting of 67 credits. Of these 67 credits, 16 are mathematics/science credits, 6 are oral/written communication credits, 6 are humanities/social science/general education credits, and the remaining 39 credits are technical courses of which 6 credits are electives.

Figure III-2 shows the general structure of the curriculum. To illustrate how to build a program of study in computer engineering technology, each component of the curriculum has been translated into a sample course. The course descriptions for the CET curriculum follow. Notice the prefix affixed to the course number. For example, a course for the *Circuits* component of the curriculum might be named CET 101, Basic Electronics. Similarly, a course for the computer architecture component might be named CET 160, Computer Architecture. Table III-2 lists the total number of lecture hours of subject areas within each CET course.

Each brief course description includes the prerequisite knowledge required of the student upon entering the course, a description of the course, course objectives, subject matter as defined by the set of knowledge units to be covered, and suggested laboratory activities.

	CR	CA	CS	DC	DV	DG	MS	PM	QC
CET 101	31	0	0	0	6	0	0	0	0
CET 110	0	8	19	0	0	0	0	0	0
CET 120	0	0	0	0	0	0	0	26	0
CET 130	0	0	0	0	2	25	0	0	0
CET 150	0	0	0	0	0	0	38	0	0
CET 160	0	27	0	0	0	0	0	0	0
CET 210	0	0	0	26	0	0	0	0	0
CET 220	0	0	0	0	0	0	0	0	26
CET 230	0	0	25	0	0	0	0	0	0
CET 235	0	0	26	0	0	0	0	0	0
Totals	31	35	70	26	8	25	38	26	26

Table III-2 **Subject Area Content (by Lecture Hours) in CET Courses**

Computer Engineering Technology

SEMESTER 1		**SEMESTER 2**	
CET 101 Basic Electronics [CR1-4, CR7-9, DV1-2]	(4:3,1)	CET 120 Intro. to Programming [PM1-5]	(3:2,1)
CET 110 Intro. to Computers [CA1-2,4, CS1-3,6]	(3:2,1)	CET 160 Computer Architecture [CA1-6]	(3:2,1)
CET 130 Digital Circuits [DG1-6, DV2]	(3:2,1)	CET 150 Microprocessors [MS1-6]	(4:3,1)
Mathematics I	(4:4,0)	Mathematics II	(4:4,0)
Communications I	(3:3,0)	Communications II	(3:3,0)
SEMESTER 3		**SEMESTER 4**	
CET 220 Quality Control [QC1-4]	(3:2,1)	CET 210 Data Communications [DC1-7]	(3:2,1)
CET 230 Computer Systems I [CS1-5, 7]	(4:2,2)	CET 235 Computer Systems II [CS7-8]	(4:2,2)
Technical Elective	(3:-,-)	Technical Elective	(3:-,-)
Mathematics III/Science	(4:4,0)	Science	(4:3,1)
General Education	(3:3,0)	General Education	(3:3,0)

Figure III-3 Typical Curriculum for Computer Engineering Technology

CET 101 Basic Electronics Lecture (3:3)
Laboratory (1:3)

Prerequisite:
Intermediate algebra

Goal:
A first course in electric circuits and amplifier principles that introduces a student to electrical parameters, basic laws of electricity, circuit configurations, and amplifier principles.

Objective:
Upon completion of the course, the student should be able to:
- Analyze electric circuits.
- Measure electric quantities.
- Build circuits from a schematic diagram.
- Operate electronic test instruments.
- Analyze simple operational amplifier circuits.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
CR1	3/3	3	T,A
CR2	6/6	4	T,A
CR3	3/3	3	T,A
CR4	6/6	4	T,A,D
CR7	3/3	3	T
CR8	4/4	3	T,A,D
CR9	6/6	3	T,A,D
DV1	2/2	3	T
DV2	4/6	4	T,A,D

Activities:
Build and test electric circuits and use basic test instruments such as power supplies, multimeters, function generators, and oscilloscopes. (Consult *Activities* component in the respective knowledge units.)

CET 110 Introduction to Computers Lecture (2:2)
Laboratory (1:3)

Prerequisite:
Intermediate algebra

Goal:
An introduction to computer systems, the operating system, and application software packages.

Objective:
Upon completion of the course, the student should be able to:

- Identify the major parts of a computer system.
- Understand the function of and use the command of the operating system to create files, store files, and execute programs.
- Use application software packages such as word processors and spreadsheets.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
CA1	4/6	3	T
CA2	2/4	3	T,A
CA4	2/4	3	T,A
CS1	1/3	1	T
CS2	2/9	2	T
CS3	4/9	2	T,A
CS6	12/12	4	T,A,D

Activities:
Use of commercial applications packages on a personal computer to prepare reports, tables and graphs. (Consult *Activities* component in the respective knowledge units.)

CET 120 Introduction to Programming Lecture (2:2)
Laboratory (1:3)

Prerequisite:
College algebra
CET 110 Introduction to Computers

Goal:
A first course in programming methodology and a programming language. The course covers program development techniques, algorithm design, coding, testing and program documentation.

Objective:
Upon completion of the course, the student should be able to:

- Implement algorithms in a programming language.
- Design algorithms from program specifications.
- Create user-defined data types.
- Test and debug algorithms and programs.
- Produce a complete documentation package for an algorithm or program.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
PM1	2/2	3	T
PM2	12/12	4	T,A,D
PM3	3/3	4	T,A,D
PM4	6/6	4	T,A,D
PM5	3/3	4	T,A,D

Activities:
Design, code, test and debug, and document algorithms and programs. (Consult *Activities* component in the respective knowledge units.)

CET 130 Digital Circuits Lecture (2:2)
Laboratory (1:3)

Prerequisite:
Intermediate algebra

Goal:
A fundamental course in combinational circuits, flip-flops, and latches.

Objective:
Upon completion of the course, the student should be able to:

- Analyze combinational logic circuits in terms of function and timing.
- Express circuit function in terms of Boolean expressions.
- Design, implement, and test combinational circuits.
- Analyze, implement, and test flip-flop and latch circuits.
- Use multimeters, oscilloscopes, and logic analyzers to analyze and troubleshoot digital circuits.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
DG1	3/3	3	T
DG2	3/3	4	T
DG3	4/4	4	T,A,D
DG4	3/3	4	T,A
DG5	6/6	4	T,A,D
DG6	6/6	4	T,A,D
DV2	2/6	3	T,A,D

Activities:
Analyze, build, test, and troubleshoot combinational and sequential logic circuits and systems. Operate and use multimeters, oscilloscopes, and logic analyzers to verify circuit function and timing. (Consult *Activities* component in the respective knowledge units.)

CET 150 Microprocessors Lecture (3:3)
Laboratory (1:2)

Prerequisite:
CET 101 Basic Electronics
CET 130 Digital Circuits

Goal:
A first course in microprocessors which introduces a student to the basic concepts of microprocessors including operation, programming, and their use in a complete system.

Objective:
Upon completion of the course, the student should be able to:

- Describe the various building blocks of a microcomputer system and explain how they are interconnected.
- Demonstrate an understanding of the instruction set of a microprocessor and the use of an assembler.
- Describe and explain the important system element specifications.
- Demonstrate an understanding of the various aspects of input and output and the associated hardware and software aspects of I/O.
- Design, build and test a microprocessor based system.
- Describe the various alternative architectures which are used for microprocessor systems.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
MS1	5/5	3	T,A
MS2	9/9	4	T,A,D
MS3	3/3	3	T,A,D
MS4	12/12	3	T,A,D
MS5	6/6	4	T,A,D
MS6	3/3	3	T

Activities:
Writing assembly language programs for and the hardware testing of a microprocessor system. The design, build and test of a microprocessor system to perform a specific function. (Consult *Activities* component in the respective knowledge units.)

CET 160 Computer Architecture Lecture (2:2)
Laboratory (1:3)

Prerequisite:
CET 110 Introduction to Computers

Goal:
A course that examines different computer architectures in terms of CPU structure, memory, data transfer and control.

Objective:
Upon completion of the course, the student should be able to:

- Describe different types of computer architectures.
- Discuss the advantages and disadvantages of various computer architectures.
- Analyze physical implementations of memory systems.
- Contrast methods of handling interrupts, DMA, and parallel and serial I/O.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
CA1	2/6	3	T
CA2	2/4	3	T,A
CA3	12/12	4	T,A,D
CA4	2/4	4	T,A,D
CA5	5/10	4	T,A,D
CA6	4/4	3	T,A,D

Activities:
Various computer architectures will be used in the laboratory to compare computer system functions. (Consult *Activities* component in the respective knowledge units.)

CET 210 Data Communications Lecture (2:2)
Laboratory (1:2)

Prerequisite:
CET 130 Digital Circuits
CET 230 Computer Systems I

Goal:
A first course in data communications which introduces a student to the basic means and methods for the communication of data.

Objective:
Upon completion of the course, the student should be able to:

- Describe the various media used for data communications.
- Demonstrate an understanding of the telephone system and its characteristics particularly as they apply to data communications.
- Describe and explain the use of analog signals for data transmission.
- Demonstrate an understanding of the generation and use of digital signals for data communications.
- Demonstrate an understanding of the various types of protocols and methods of error checking.
- Describe and explain the use and application of optical methods for data communication.
- Demonstrate an understanding of the equipment, software and operation of a local area network.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
DC1	3/3	3	T
DC2	3/3	3	T
DC3	3/3	3	T,A,D
DC4	6/6	4	T,A,D
DC5	3/3	3	T,A,D
DC6	3/3	3	T,A,D
DC7	5/9	4	T,A,D

Activities:
The design, build and test of various circuits for data transmission using analog and digital methods. (Consult *Activities* component in the respective knowledge units.)

CET 220 Quality Control Lecture (2:2)
Laboratory (1:3)

Prerequisite:
CET 120 Introduction to Programming

Goal:
An introductory course in quality control concepts and methods.

Objective:
Upon completion of the course, the student should be able to:
- Collect data from a population or sample.
- Use statistical methods to analyze data.
- Compare methods of evaluating reliability.
- Perform failure mode analyses.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
QC1	3/3	3	T
QC2	8/8	4	T,A,D
QC3	9/9	3	T,A,D
QC4	6/6	3	T,A,D

Activities:
Data collection, data analysis, and the design of procedures and practices to improve quality in a process or product. (Consult *Activities* component in the respective knowledge units.)

CET 230 Computer Systems I Lecture (2:2)
Laboratory (2:6)

Prerequisite:
CET 120 Introduction to Programming
CET 160 Computer Architecture

Goal:
The first course in a two-course sequence on the design, implementation, and maintenance of computer systems and computer networks.

Objective:
Upon completion of the course, the student should be able to:

- Assemble a personal computer system.
- Load resident software.
- Test and verify proper system operation.
- Interface peripheral units such as printers and mouse systems.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
CS1	2/3	3	T
CS2	7/9	4	T,A,D
CS3	5/9	4	T,A,D
CS4	6/6	4	T,A,D
CS5	3/3	3	T
CS7	2/16	2	T

Activities:
This is the capstone sequence in the computer engineering technology program. Students will construct complete personal computer systems and then interconnect these computers as a network. Network administration and management will be included. (Consult *Activities* component in the respective knowledge units.)

CET 235 Computer Systems II Lecture (2:2)
Laboratory (2:6)

Prerequisite:
CET 230 Computer Systems I

Goal:
This is the second course in a two-course sequence. This course focuses on the networking of computer systems and the management of the network.

Objective:
Upon completion of the course, the student should be able to:

- Design the physical layout of a computer network.
- Interconnect the computer systems comprising the network.
- Install the network software.
- Test and verify proper operation of the network.
- Manage the network, including the training of new users.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
CS7	14/16	4	T,A,D
CS8	12/12	4	T,A,D

Activities:
A computer network will be designed, implemented and tested. (Consult *Activities* component in the respective knowledge units.)

4.0 SAMPLE CURRICULUM AND COURSES FOR ELECTRONIC ENGINEERING TECHNOLOGY

The sample curriculum for electronic engineering technology assumes a semester-based program consisting of 66 credits. Of these credits, 16 are mathematics/science credits, 6 are oral/written communication credits, 6 are humanities/social science/general education credits, and the remaining 38 credits are technical courses of which 6 credits are electives.

Figure III-3 shows the general structure for the curriculum. To illustrate how to build a program of study in electronic engineering technology, each component of the curriculum has been translated into a sample course. The course descriptions for the EET curriculum follow. Notice the prefix affixed to the course number. For example, the digital component from the electronic engineering technology program might be given the name EET 130, Digital Circuits I. Likewise, the linear devices component in the third semester might be called EET 200, Linear Devices. Table III-3 lists the total number of lecture hours of subject areas within each EET course.

Each brief course description includes the prerequisite knowledge that the student must possess upon entering the course, a description of the course, course objectives, subject matter as defined by the set of knowledge units to be covered, and suggested laboratory activities.

	CR	CA	CS	DC	DV	DG	LC	MS	PM	QC
EET 105	36	0	0	0	0	0	0	0	0	0
EET 110	0	10	19	0	0	0	0	0	0	0
EET 120	0	0	0	0	0	0	0	0	26	0
EET 130	4	0	0	0	0	25	0	0	0	0
EET 135	0	0	0	0	0	16	0	12	0	0
EET 140	0	0	0	0	26	0	0	0	0	0
EET 155	0	0	0	0	0	0	0	26	0	0
EET 200	0	0	0	0	0	0	25	0	0	0
EET 210	0	0	0	30	0	0	0	0	0	0
EET 220	0	0	0	0	0	0	2	0	0	26
Totals	40	10	19	30	26	41	27	38	26	26

Table III-3 Subject Area Content (by Lecture Hours) in EET Courses

Electronic Engineering Technology

SEMESTER 1		**SEMESTER 2**	
EET 105 Basic Electronics [CR1-10]	(4:3,1)	EET 140 Electronic Devices [DV1-7]	(3:2,1)
EET 110 Intro. to Computers [CA1-2; CS1-3,6]	(3:2,1)	EET 120 Intro. to Programming [PM1-5]	(3:2,1)
EET 130 Digital Circuits [CR1-4; DG1-6]	(3:2,1)	EET 135 Adv. Digital Circuits [DG7-9; MS1-2]	(3:2,1)
Mathematics I	(4:4,0)	Mathematics II	(4:4,0)
Communications I	(3:3,0)	Communications II	(3:3,0)
SEMESTER 3		**SEMESTER 4**	
EET 200 Linear Devices [LC1-6]	(3:2,1)	EET 210 Data Communications [DC1-7]	(3:2,1)
EET 155 Microprocessors [MS2-6]	(3:2,1)	EET 220 Quality Control [QC1-4; LC6]	(3:2,1)
Technical Elective	(3:-,-)	Technical Elective	(3:-,-)
Mathematics III/Science	(4:4,0)	Science	(4:3,1)
General Education	(3:3,0)	General Education	(3:3,0)

Figure III-4 Typical Curriculum for Electronic Engineering Technology

EET 105 Basic Electronics Lecture (3:3)
Laboratory (1:3)

Prerequisite:
Intermediate algebra

Goal:
A first course in electric circuits that introduces a student to electrical parameters, basic laws of electricity, circuit configurations, circuit analysis techniques, and simulation and measurement procedures.

Objective:
Upon completion of the course, the student should be able to:
- Analyze electric circuits.
- Measure electric quantities.
- Build circuits from a schematic diagram.
- Operate electronic test instruments.
- Use simulation software.
- Model electric circuits.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
CR1	2/3	4	T,A
CR2	5/6	4	T,A
CR3	2/3	4	T,A
CR4	5/6	4	T,A,D
CR5	3/3	4	T,A
CR6	3/3	5	T,A
CR7	3/3	3	T
CR8	4/4	4	T,A
CR9	6/6	4	T,A,D
CR10	3/3	4	T,A

Activities:
Build and test electric circuits and use basic test instruments such as power supplies, digital multimeters, function generators, and oscilloscopes. (Consult *Activities* component in the respective knowledge units.)

EET 110 Introduction to Computers Lecture (2:2)
Laboratory (1:3)

Prerequisite:
Intermediate algebra

Goal:
An introduction to computer systems, the operating system, and application software packages.

Objective:
Upon completion of the course, the student should be able to:

- Identify the major parts of a computer system.
- Understand the function of and use the command of the operating system to create files, store files, and execute programs.
- Use applications software packages such as wordprocessors and spreadsheets.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
CA1	6/6	3	T
CA2	4/4	3	T,A
CS1	1/3	1	T
CS2	2/9	2	T
CS3	4/9	2	T,A
CS6	12/12	4	T,A,D

Activities:
Use of commercial applications packages on a personal computer to prepare reports, tables, and graphs. (Consult *Activities* component in the respective knowledge units.)

EET 120 Introduction to Programming Lecture (2:2)
Laboratory (1:3)

Prerequisite:
College algebra
EET 110 Introduction to Computers

Goal:
A first course in programming methodology and a programming language. The course covers program development techniques, algorithm design, coding, testing, and program documentation.

Objective:
Upon completion of the course, the student should be able to:

- Implement algorithms in a programming language.
- Design algorithms from program specifications.
- Create user-defined data types.
- Test and debug algorithms and programs.
- Produce a complete documentation package for an algorithm or program.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
PM1	2/2	3	T
PM2	12/12	4	T,A,D
PM3	3/3	4	T,A,D
PM4	6/6	4	T,A,D
PM5	3/3	4	T,A,D

Activities:
Design, code, test and debug, and document algorithms and programs. (Consult *Activities* component in the respective knowledge units.)

EET 130 Digital Circuits Lecture (2:2)
Laboratory (1:3)

Prerequisite:
Intermediate algebra

Goal:
A fundamental course in combinational circuits, flip-flops, and latches.

Objective: Upon completion of the course, the student should be able to:
- Analyze combinational logic circuits in terms of function and timing.
- Express circuit function in terms of Boolean expressions.
- Design, implement, and test combinational circuits.
- Analyze, implement, and test flip-flop and latch circuits.
- Use multimeters, oscilloscopes, and logic analyzers to analyze and troubleshoot digital circuits.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
CR1	1/3	2	T
CR2	1/6	3	T,A
CR3	1/3	2	T
CR4	1/6	3	T,A,D
DG1	3/3	3	T
DG2	3/3	4	T
DG3	4/4	4	T,A,D
DG4	3/3	4	T,A
DG5	6/6	4	T,A,D
DG6	6/6	4	T,A,D

Activities:
Analyze, build, test, and troubleshoot combinational and sequential logic circuits and systems. Operate and use multimeters, oscilloscopes, and logic analyzers to verify circuit function and timing. (Consult *Activities* component in the respective knowledge units.)

EET 135 Advanced Digital Circuits Lecture (2:2)
Laboratory (1:3)

Prerequisite:
EET 130 Digital Circuits

Goal:
An advanced digital circuits course that covers data conversion, state machine analysis and design, and an introduction to microprocessor technology.

Objective:
Upon completion of the course, the student should be able to:

- Design, analyze, and test complex digital circuits.
- Analyze and implement data conversion circuits.
- Use PC-based design tools to create and implement simple state machines.
- Analyze the hardware of a simple single-board microprocessor system.
- Write assembly language programs for a microprocessor trainer.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
DG7	4/4	4	T,A
DG8	6/6	4	T,A,D
DG9	6/6	3	T,A
MS1	5/5	3	T,A
MS2	7/9	3	T,A,D

Activities:
Build and test complex digital circuits. Design and implement simple state machines. Analyze the operation of digital circuits using logic simulation software. Write and debug simple assembly language programs for a microprocessor trainer. (Consult *Activities* component in the respective knowledge units.)

EET 140 Electronic Devices Lecture (2:2)
Laboratory (1:2)

Prerequisite:
EET 105 Basic Electronics

Goal:
A course covering the common devices used in electronic circuits, their characteristics and applications.

Objective:
Upon completion of the course, the student should be able to:

- Describe and define the concepts of an amplifier.
- Describe the characteristics of an operational amplifier and analyze a circuit containing an operational amplifier.
- Understand the operation of comparators, voltage regulators, and timers.
- Understand the operation of diodes and their applications.
- Describe the operation of BJT and FET transistors and how they can be used as switches.
- Describe the operation of various opto-electronic devices and apply these devices in the laboratory.
- Use available software to analyze circuits containing the above devices.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
DV1	2/2	2	T
DV2	6/6	3	T,A,D
DV3	3/3	4	T,A
DV4	3/3	4	T,A
DV5	4/4	3	T,A
DV6	2/2	3	T,A
DV7	6/6	4	T,A,D

Activities:
Design, build and test circuits containing various electronic devices. Use various measuring instruments and computer simulation to verify and simulate circuit operation. (Consult *Activities* component in the respective knowledge units.)

EET 155 Microprocessors Lecture (2:2)
Laboratory (1:2)

Prerequisite:
EET 135 Advanced Digital Circuits

Goal:
A continuation of Advanced Digital Circuits where the concept of using a microprocessor in a system is developed.

Objective:
Upon completion of the course, the student should be able to:

- Describe and explain the important system element specifications.
- Demonstrate an understanding of the various aspects of input and output and the associated hardware and software aspects of I/O.
- Design, build and test a microprocessor-based system.
- Describe the various alternative architectures which are used for microprocessor systems.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
MS2	2/9	3	T,A,D
MS3	3/3	3	T,A,D
MS4	12/12	3	T,A,D
MS5	6/6	4	T,A,D
MS6	3/3	3	T

Activities:
The design, build and test of a microprocessor system to perform a specific function. (Consult *Activities* component in the respective knowledge units.)

EET 200 Linear Devices Lecture (2:2)
Laboratory (1:2)

Prerequisite:
EET 140 Electronic Devices

Goal:
A course covering the more common specialized linear devices used in electronic circuits, their characteristics and applications.

Objective:
Upon completion of the course, the student should be able to:

- Describe and apply common voltage and current references.
- Design, build and test different types of active filters.
- Understand the operation and application of phase-locked loops.
- Describe the various types of oscillators and design, build and test a specific oscillator circuit.
- Explain the causes and sources of interference and demonstrate an understanding of how interference can be reduced.
- Use an available computer program for the analysis of a system containing one of more of the above devices.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
LC1	3/3	3	T,A
LC2	6/6	4	T,A,D
LC3	3/3	4	T,A,D
LC4	3/3	4	T,A
LC5	4/4	2	T
LC6	6/8	3	T,A

Activities:
Design, build and test circuits containing various linear electronic devices. Use various measuring instruments and computer simulation to verify and simulate circuit operation. (Consult *Activities* component in the respective knowledge units.)

EET 210 Data Communications Lecture (2:2)
Laboratory (1:2)

Prerequisite:
EET 130 Digital Circuits
EET 155 Microprocessors

Goal:
A first course in data communications which introduces a student to the basic means and methods for the communication of data.

Objective:
Upon completion of the course, the student should be able to:

- Describe the various media used for data communications.
- Demonstrate an understanding of the telephone system and its characteristics particularly as they apply to data communications.
- Describe and explain the use of analog signals for data transmission.
- Demonstrate an understanding of the generation and use of digital signals for data communications.
- Demonstrate an understanding of the various types of protocols and methods of error checking.
- Describe and explain the use and application of optical methods for data communication.
- Demonstrate an understanding of the equipment, software and operation of a local area network.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
DC1	3/3	2	T
DC2	3/3	2	T
DC3	3/3	3	T,A
DC4	6/6	3	T,A
DC5	3/3	2	T,A
DC6	3/3	3	T
DC7	9/9	4	T,A

Activities:
The design, build and test of various circuits for data transmission using analog and digital methods. (Consult *Activities* component in the respective knowledge units.)

EET 220 Quality Control Lecture (2:2)
Laboratory (1:3)

Prerequisite:
EET 120 Introduction to Programming

Goal:
An introductory course in quality control concepts and methods.

Objective:
Upon completion of the course, the student should be able to:
- Collect data from a population or sample.
- Use statistical methods to analyze data.
- Compare methods of evaluating reliability.
- Perform failure mode analyses.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
LC6	2/8	3	T,A
QC1	3/3	3	T
QC2	8/8	4	T,A,D
QC3	9/9	3	T,A,D
QC4	6/6	3	T,A,D

Activities:
Data collection, data analysis, and the design of procedures and practices to improve quality in a process or product. (Consult *Activities* component in the respective knowledge units.)

5.0 SAMPLE CURRICULUM AND COURSES FOR SOFTWARE ENGINEERING TECHNOLOGY

The sample curriculum for software engineering technology assumes a semester-based program consisting of 65 credits. Of these 65 credits, 16 are mathematics/science credits, 6 are oral/written communication credits, 6 are humanities/social science/general education credits, and the remaining 37 credits are technical courses of which 6 credits are electives.

Figure III-4 shows the general structure of the curriculum. To illustrate how to build a program of study in software engineering technology, each component of the curriculum has been translated into a sample course. These course descriptions for the SET curriculum follow. Notice the prefix affixed to the course number. For example, a course for the *Programming* component in the second semester of the curriculum might be named SET 120, Introduction to Programming. Similarly, a course for the microprocessor component in the second semester might be named SET 150, Microprocessors or SET 150, Microprocessor Systems. Table III-4 lists the total number of lecture hours of subject areas within each SET course.

Each brief course description includes the prerequisite knowledge required of the student upon entering the course, a description of the course, course objectives, subject matter as defined by the set of knowledge units to be covered, and suggested laboratory activities.

	CR	CA	CS	DG	DV	MS	OS	PD	PM	QC	ST
SET 101	31	0	0	0	6	0	0	0	0	0	0
SET 110	0	12	19	0	0	0	0	0	0	0	0
SET 120	0	0	0	0	0	0	0	0	26	0	0
SET 130	0	0	0	25	2	0	0	0	0	0	0
SET 150	0	0	0	0	0	38	0	0	0	0	0
SET 160	0	28	0	0	0	0	0	0	0	0	0
SET 210	0	0	0	0	0	0	39	0	0	0	0
SET 220	0	0	0	0	0	0	0	0	0	26	0
SET 230	0	0	0	0	0	0	0	38	0	0	0
SET 250	0	0	0	0	0	0	0	7	0	0	21
Totals	31	40	19	25	8	38	39	45	26	26	21

Table III-4 **Subject Area Content (by Lecture Hours) in SET Courses**

Software Engineering Technology

SEMESTER 1		**SEMESTER 2**	
SET 101 Basic Electronics [CR1-4; CR7-9; DV1-2]	(4:3,1)	SET 120 Intro. to Programming [PM1-5]	(3:2,1)
SET 110 Intro. to Computers [CA1-2,4; CS1-3,6;]	(3:2,1)	SET 160 Computer Architecture [CA3-6]	(3:2,1)
SET 130 Digital Circuits [DG1-6; DV2]	(3:2,1)	SET 150 Microprocessors [MS1-6]	(4:3,1)
Mathematics I	(4:4,0)	Mathematics II	(4:4,0)
Communications I	(3:3,0)	Communications II	(3:3,0)
SEMESTER 3		**SEMESTER 4**	
SET 210 Operating Systems [OS1-6]	(4:3,1)	SET 250 Software Test. & Val. [ST1-4, PD5-6]	(3:2,1)
SET 230 Adv. Programming [PD1-6]	(3:2,1)	SET 220 Quality control [QC1-4]	(3:2,1)
Technical Elective	(3:2,1)	Technical Elective	(3:-,-)
Mathematics III/Science	(4:4,0)	Science	(4:3,1)
General Education	(3:3,0)	General Education	(3:3,0)

Figure III-5 Typical Curriculum for Software Engineering Technology

SET 101 Basic Electronics Lecture (3:3)
Laboratory (1:3)

Prerequisite:
Intermediate algebra

Goal:
A first course in electric circuits and amplifier principles that introduces a student to electrical parameters, basic laws of electricity, circuit configurations, and amplifier principles.

Objective:
Upon completion of the course, the student should be able to:

- Analyze electric circuits.
- Measure electric quantities.
- Build circuits from a schematic diagram.
- Operate electronic test instruments.
- Analyze simple operational amplifier circuits.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
CR1	3/3	3	T,A
CR2	6/6	4	T,A
CR3	3/3	3	T,A
CR4	6/6	4	T,A,D
CR7	3/3	3	T
CR8	4/4	3	T,A,D
CR9	6/6	3	T,A,D
DV1	2/2	3	T
DV2	4/6	4	T,A,D

Activities:
Build and test electric circuits and use basic test instruments such as power supplies, multimeters, function generators, and oscilloscopes. (Consult *Activities* component in the respective knowledge units.)

SET 110 Introduction to Computers Lecture (2:2)
Laboratory (1:3)

Prerequisite:
Intermediate algebra

Goal:
An introduction to computer systems, the operating system, and application software packages.

Objective:
Upon completion of the course, the student should be able to:

- Identify the major parts of a computer system.
- Understand the function of an operating system.
- Use the command of an operating system to create files, store files, and execute programs.
- Use applications software packages such as wordprocessors and spreadsheets.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
CA1	6/6	3	T
CA2	4/4	3	T,A
CA4	2/4	3	T,A
CS1	1/3	1	T
CS2	2/9	2	T
CS3	4/9	2	T,A
CS6	12/12	4	T,A,D

Activities:
Use of commercial applications packages on a personal computer to prepare reports, tables, and graphs. (Consult *Activities* component in the respective knowledge units.)

SET 120 Introduction to Programming Lecture (2:2)
Laboratory (1:3)

Prerequisite:
College algebra
SET 110 Introduction to Computers

Goal:
A first course in programming methodology and a programming language. The course covers program development techniques, algorithm design, coding, testing, and program documentation.

Objective:
Upon completion of the course, the student should be able to:

- Implement algorithms in a programming language.
- Design algorithms from program specifications.
- Create user-defined data types.
- Test and debug algorithms and programs.
- Produce a complete documentation package for an algorithm or program.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
PM1	2/2	3	T
PM2	12/12	4	T,A,D
PM3	3/3	4	T,A,D
PM4	6/6	4	T,A,D
PM5	3/3	4	T,A,D

Activities:
Design, code, test and debug, and document algorithms and programs. (Consult *Activities* component in the respective knowledge units.)

SET 130 Digital Circuits Lecture (2:2)
Laboratory (1:3)

Prerequisite:
Intermediate algebra

Goal:
A fundamental course in combinational circuits, flip-flops, and latches.

Objective:
Upon completion of the course, the student should be able to:

- Analyze combinational logic circuits in terms of function and timing.
- Express circuit function in terms of Boolean expressions.
- Design, implement, and test combinational circuits.
- Analyze, implement, and test flip-flop and latch circuits.
- Use multimeters, oscilloscopes, and logic analyzers to analyze and troubleshoot digital circuits.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
DG1	3/3	3	T
DG2	3/3	4	T
DG3	4/4	4	T,A,D
DG4	3/3	4	T,A
DG5	6/6	4	T,A,D
DG6	6/6	4	T,A,D
DV2	2/6	3	T,A,D

Activities:
Analyze, build, test, and troubleshoot combinational and sequential logic circuits and systems. Operate and use multimeters, oscilloscopes, and logic analyzers to verify circuit function and timing. (Consult *Activities* component in the respective knowledge units.)

SET 150 Microprocessors Lecture (3:3)
Laboratory (1:3)

Prerequisite:
SET 130 Digital Circuits

Goal:
A first course in microprocessors which introduces a student to the basic concepts of microprocessors including operation, programming, and their use in a complete system.

Objective:
Upon completion of the course, the student should be able to:

- Describe the various building blocks of a microcomputer system and explain how they are interconnected.
- Demonstrate an understanding of the instruction set of a microprocessor and the use of an assembler.
- Describe and explain the important system element specifications.
- Demonstrate an understanding of the various aspects of input and output and the associated hardware and software aspects of I/O.
- Design, build and test a microprocessor based system.
- Describe the various alternative architectures which are used for microprocessor systems.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
MS1	5/5	3	T,A
MS2	9/9	4	T,A,D
MS3	3/3	3	T,A,D
MS4	12/12	3	T,A,D
MS5	6/6	4	T,A,D
MS6	3/3	3	T

Activities:
Writing assembly language programs for and the hardware testing of a microprocessor system. The design, build and test of a microprocessor system to perform a specific function. (Consult *Activities* component in the respective knowledge units.)

SET 160 Computer Architecture Lecture (2:2)
Laboratory (1:3)

Prerequisite:
SET 110 Introduction to Computers

Goal:
A course that examines different computer architectures in terms of CPU structure, memory, data transfer and control.

Objective:
Upon completion of the course, the student should be able to:

- Describe different types of computer architectures.
- Discuss the advantages and disadvantages of various computer architectures.
- Analyze physical implementations of memory systems.
- Contrast methods of handling interrupts, DMA, and parallel and serial I/O.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
CA3	12/12	4	T,A,D
CA4	2/4	4	T,A,D
CA5	10/10	4	T,A,D
CA6	4/4	3	T,A,D

Activities:
Various computer architectures will be used in the laboratory to compare computer system functions. (Consult *Activities* component in the respective knowledge units.)

SET 210 Operating Systems Lecture (3:3)
Laboratory (1:3)

Prerequisite:
SET 160 Computer Architecture

Goal:
A course that covers the theory, concepts and components of current operating systems with emphasis upon resource allocation, input/output, memory and software system organization.

Objective:
Upon completion of the course, the student should be able to:

- Describe the components of an operating system.
- Discuss system resources and resource allocation within a computing system.
- Install, configure and use a current operating system.
- Use system diagnostic software to examine memory allocation, file naming, disk usage in a computing system.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
OS1	3/3	2	T
OS2	6/6	3	T
OS3	6/6	3	T
OS4	6/6	4	T,A
OS5	6/6	4	T,A
OS6	12/12	4	T,A,D

Activities:
Installation, configuration and use of a current operating system. Use of diagnostic software to examine memory allocation and file management. (Consult *Activities* component in the respective knowledge units.)

SET 220 Quality Control Lecture (2:2)
Laboratory (1:3)

Prerequisite:
SET 120 Introduction to Programming

Goal:
An introductory course in quality control concepts and methods.

Objective:
Upon completion of the course, the student should be able to:
- Collect data from a population or sample.
- Use statistical methods to analyze data.
- Compare methods of evaluating reliability.
- Perform failure mode analyses.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
QC1	3/3	3	T
QC2	8/8	4	T,A,D
QC3	9/9	3	T,A,D
QC4	6/6	3	T,A,D

Activities:
Data collection, data analysis, and the design of procedures and practices to improve quality in a process or product. (Consult *Activities* component in the respective knowledge units.)

SET 230 Advanced Programming Lecture (3:3)
Laboratory (1:3)

Prerequisite:
SET 120 Introduction to Programming

Goal:
A second course in programming methods and practices, from requirements and design through implementation, documentation and testing.

Objective:
Upon completion of the course, the student should be able to:
- Develop software system design requirements.
- Develop a program design using appropriate algorithms.
- Implement the design in a programming language.
- Test and document the system implementation.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
PD1	3/3	2	T
PD2	6/6	3	T,A,D
PD3	8/8	3	T,A,D
PD4	8/8	3	T,A,D
PD5	6/8	4	A,D
PD6	7/12	4	A,D

Activities:
Specification, design, implementation, documentation and testing of software systems. (Consult *Activities* component in the respective knowledge units.)

SET 250 Software Testing and Validation Lecture (2:2)
Laboratory (1:3)

Prerequisite:
SET 230 Advanced Programming

Goal:
A course in methods and practices in software test and validation.

Objective:
Upon completion of the course, the student should be able to:
- Describe current software testing strategies.
- Implement testing plans at module and system levels.
- Analyze and interpret test data to locate and correct software bugs.
- Incorporate failure mode analysis in the software design process.

Subject Matter:

KU Tag Code	Portion of Hours	Depth	Emphasis
PD5	2/8	3	A,D
PD6	5/12	3	A,D
ST1	3/3	2	T
ST2	6/6	3	T,A,D
ST3	6/6	3	T,A,D
ST4	6/6	3	T,A,D

Activities:
Design of testing procedures, implementation of test plans, design of code for ease of testing, and analysis of test data for troubleshooting system problems. (Consult *Activities* component in the respective knowledge units.)

6.0 PROGRAM COMMONALITIES

The courses and knowledge units for the three areas of computer engineering technology, electronic engineering technology and software engineering technology share certain commonalities. For example, in Section 4, the course entitled, *Introduction to Computing* was the same for each of the three curricula. Naturally, institutional needs may prescribe three separate courses which foster different emphases for this topic. As another example, the course named *Computer Architecture* is shared by two curricula, namely, CET and SET. Table III-5, which is arranged and alphabetized within groups, shows commonalities among the three curricula. This is not a prescription for the development of a curriculum. It is one suggestion among many about how this can be achieved. In this instance, a conscious effort was made to avoid duplication of similar courses which may be a consideration for a particular institution. Program mission and constituency needs should always be an important part in the design of a curriculum.

Course	**EET**	**CET**	**SET**
Basic Electronics	X	X	X
Digital Circuits	X	X	X
Introduction to Computers	X	X	X
Introduction to Programming	X	X	X
Microprocessors	X	X	X
Quality Control	X	X	X
Computer Architecture		X	X
Data Communications	X	X	
Advanced Digital Circuits	X		
Advanced Programming			X
Computer Systems I		X	
Computer Systems II		X	
Linear Devices	X		
Operating Systems			X
Software Testing and Validation			X

Table III-5 **Commonalities among the three curricula**

7.0 CURRICULA EXIT COMPETENCIES

The following is a listing of typical exit competencies for the EET, CET, and SET curricula. Each institution is encouraged to formulate its own list based on the mission of the institution, student preparation, and community needs.

	Knowledge Units	**EET**	**CET**	**SET**
CA1:	History of Computing	1	1	1
CA2:	Machine Organization	3	3	3
CA3:	Assembly Language Programming	3	4	5
CA4:	Memory Systems	5	4	2
CA5:	Interfacing	4	4	2
CA6:	Alternative Architectures	4	3	3
CR1:	Electrical Parameters	2	2	2
CR2:	Electrical Relationships	4	4	2
CR3:	Electrical Signals	3	3	2
CR4:	Circuit Configurations	4	3	2
CR5:	Circuit Reduction	4	3	1
CR6:	Circuit Simulation and Analysis	4	4	2
CR7:	Capacitance and Inductance	4	3	2
CR8:	Transient Analysis	4	3	2
CR9:	Frequency Response	4	3	2
CR10:	Spectral Analysis	3	2	1
CS1:	System Organization	4	3	2
CS2:	Hardware Specifications	4	4	2
CS3:	System Software	3	4	5
CS4:	Computer Peripherals	4	4	2
CS5:	Computing Environments	3	3	3
CS6:	Applications Software	4	4	4
CS7:	Network Implementation	4	5	3
CS8:	Network Management	2	3	4
DC1:	Data Transmission Basics	3	3	-
DC2:	The Telephone System	2	3	-
DC3:	Data Transmission via Analog Signals	3	4	-
DC4:	Data Transmission via Digital Signals	3	4	-
DC5:	Protocols	2	4	-
DC6:	Optical Data Transmission	2	3	-
DC7:	Local Area Networks	2	4	-
DG1:	Number Systems	4	4	4
DG2:	Boolean Algebra	3	2	2
DG3:	Logic Functions and Devices	4	4	2
DG4:	Logic Simplification Methods	3	2	2
DG5:	Complex Combinational Circuits	4	3	2

DG6:	Sequential Logic Devices and Their Applications		4		4		3
DG7:	Data Conversion	3		3		2	
DG8:	State Machines	3		2		1	
DG9:	Logic Analysis Techniques	4		3		2	

	Knowledge Units (continued)	**EET**		**CET**		**SET**	
DV1:	Amplifier Concepts and Characteristics		2		2		2
DV2:	Operational Amplifiers	4		3		2	
DV3:	Linear Integrated Circuits	4		3		2	
DV4:	Diodes	4		2		2	
DV5:	Transistors	4		3		2	
DV6:	Opto-Electronic Devices	3		3		1	
DV7:	Device Modeling and Systems Simulation		3		3		1
LC1:	Voltage and Current References		3		-		-
LC2:	Filter Concepts and Circuits	3		-		-	
LC3:	Phase Locked Loops		3		-		-
LC4:	Oscillators	2		-		-	
LC5:	Grounding and Shielding	3		-		-	
LC6:	Modeling and Simulation	3		-		-	
MS1:	Machine Organization	3		3		3	
MS2:	Assembly Language Programming	3		4		4	
MS3:	Hardware Specifications	4		4		2	
MS4:	Interfacing	4		4		2	
MS5:	System Level Implementation		5		5		3
MS6:	Alternative Architectures	3		3		3	
OS1:	Operating System Models	-		-		3	
OS2:	Processor Management	-		-		4	
OS3:	Device Control	-		-		4	
OS4:	Memory Management	-		-		4	
OS5:	File System Management	-		-		4	
OS6:	Operating System Case Study		-		-		3
PD1:	Software Engineering Concepts		-		-		3
PD2:	Requirements, Specifications and Software Development		-		-		3
PD3:	Algorithms and Data Structures		-		-		4
PD4:	Software Design, Documentation and Implementation		-		-		5
PD5:	Software Integration, Testing and Validation	-		-		4	
PD6:	Team Programming Management	-		-		3	
PM1:	History of Programming Languages	2		2		2	
PM2:	Introduction to a Programming Language		3		3		3
PM3:	Data Types	3		4		4	
PM4:	Algorithms and Program Design	3		4		5	
PM5:	Program Documentation, Testing and Verification		3		3		4
QC1:	Reliability and Maintainability		3		3		-
QC2:	Statistical Methods		4		4		-

QC3:	Reliability Evaluation	3		2		-		
QC4:	Failure Mode Effects Analysis		3		2		-	
ST1:	Concepts of Software Testing and Validation	-		-		3		
ST2:	Module Testing	-		-		4		
ST3:	Integration and System Tests		-		-		4	
ST4:	Other types of Tests and Validation	-		-		3		

PART IV

SUPPORTIVE INFORMATION

ENDNOTES

1. ACM Curriculum Committee on Computer Science. Curriculum '68: Recommendations for Academic Programs in Computer Science. *Comm. ACM 22*, 3 (Mar. 1968), 151-197.

2. ACM Curriculum Committee on Computer Science. Curriculum '78: Recommendations for the Undergraduate Program in Computer Science. *Comm. ACM 22*, 3 (Mar. 1979), 147-166.

3. IEEE Computer Society. Recommendations and Guidelines for Associate Degree Programs in Computer Systems Technology. *IEEE Computer*, (1982).

4. ACM/IEEE-CS Joint Curriculum Task Force. *Computing Curricula 1991*. ACM Press and IEEE Computer Society Press (1991).

5. Accreditation Board for Engineering and Technology, Inc. Criteria for Accrediting Programs in Engineering Technology - 1991/92. (1991).

6. Hull, Dan, and Dale Parnell. *Tech Prep Associate Degree: A Win/Win Experience*. The Center for Occupational Research and Development, 199.

7. Ford, G. and N. Gibbs. A Master of Software Engineering Curriculum: Recommendations from the Software Engineering Institute. *IEEE Computer*, (Sept. 1989), 63.

BIBLIOGRAPHY

Accreditation Board for Engineering and Technology, Inc. *Criteria for Accrediting Programs in Engineering Technology - 1991/92*. (1991).

ACM Curriculum Committee for Community and Junior College Education. *Recommendations for a Two-Year Associate Degree Career Program in Computer Programming*. Association for Computing Machinery (1981).

ACM Curriculum Committee on Computer Science. Curriculum '78 -Recommendations for the Undergraduate Program in Computer Science. *Commun. ACM 22*, 3 (March 1979), 147-166.

ACM/IEEE-CS Joint Curriculum Task Force. *Computing Curricula 1991*. ACM Press and IEEE Computer Society Press (1991).

American Society for Engineering Education. *Quality of Engineering Education: Final Report of the Quality of Engineering Education Project*. (1984).

Barker, et al. Laboratory Experiences in Computer Science and Engineering. *Computer Science Education* 1, (1988), 1-10.

Beidler, John, et al. Computing Programs in Small Colleges. *Commun. ACM 28*, 6, (June 1985), 605-611.

Freedman, Alan, editor. *The Computer Glossary, Fourth Edition*. The Computer Language Company (1989).

Data Processing Management Association. *Associate-Level Model Curriculum in Computer Information Systems, First Edition*. Data Processing Management Association (October 1985).

Data Processing Management Association. *CIS '86, The DPMA Model Curriculum for Undergraduate Computer Information Systems*. Data Processing Management Association (1986), 1-29.

Data Processing Management Association. *The DPMA Associate-Level Model Curriculum in Computer Information Systems*. Data Processing Management Association (1985), 1-23.

Data Processing Management Association. *Information Systems the DPMA Model Curriculum for a Four-Year Undergraduate Degree*. Data Processing Management Association (1990), 1-52.

Denning, Peter J. The Science of Computing. *American Scientist* 73, 16-19.

Denning, et al. Computing as a Discipline. *Commun. ACM*, (1989), 9-23 and 63-70.

Rosenberg, Jerry, editor. *Dictionary of Computers, Information, and Telecommunications, Second Edition*. John Wiley & Sons, Inc. (1987).

Drysdale, et al. Computer Science in Liberal Arts Colleges. *Computer Science Education* 1, (1988), 11-35.

Hull, Dan and Dale Parnell. *Tech Prep Associate Degree: A Win/Win Experience*. The Center for Occupational Research and Development.

IEEE Computer Society Press. *Recommendations and Guidelines for Associate Degree Programs in Computer Systems*

Technology. IEEE Computer Society Press (1982).

Koffman, et al. Recommended Curriculum for CS1. *Commun. ACM 27*, 10, (1984), 998-1001.

Koffman, et al. Recommended Curriculum for CS2. *Commun. ACM 28*, 8, (1985), 815-818.

Occupational Projections and Training Data, 1986 Edition. U.S. Government Printing Office (1986).

Parnell, Dale. *The Neglected Majority*. The Community College Press (1985).

Ralston, Anthony. The First Course in Computer Science Needs a Mathematics Corequisite. *Commun. ACM 27*, 10 (1984), 1002-1005.

Shaw, Mary, editor. *The Carnegie Mellon Curriculum for Undergraduate Computer Science*. Springer-Verlag (1985).

Struble, George. Most Effective Lab Exercises. *Computer Science Education* 1 (1990), 85-87, 177-180, 273-275, 375-386.

Tucker, A.B. and D. K. Garnick. *A Breadth-First Approach to the Introductory Curriculum in Computing*. Bowdoin College (1990), 1-35.

Welsh, William. *Employment Outlook for NTID Data Processing Graduates*. Office of Post-Secondary Career Studies and Institutional Research, Rochester Institute of Technology, (January 1988).

Welsh, William et al. *Employment Outlook for Graduates of a Major in Microcomputing Applications*. National Technical Institute for the Deaf, Rochester Institute of Technology, (1988).

DEFINITION OF TECHNICAL TERMS

[Taken from *IEEE Standard Dictionary of Electrical and Electronic Terms; Computer Glossary* (Freedman, Fourth Edition); and the *Dictionary of Computers, Information and Telecommunications* (Wiley, Second Edition.)]

Active filter -- A filter network containing one or more voltage-dependent or current-dependent sources in addition to passive elements.

Algorithm -- A prescribed set of well-defined rules or processes for the solution of a problem in a finite number of steps.

Analog-to-digital converter -- A circuit whose input is information in analog form and whose output is the same information in digital form.

Analog signals -- Pertaining to data in the form of continuously variable physical quantities.

Architecture -- The structure and relationship, among the components of a system.

Assembly language programming -- A machine specific language whose instructions are usually in one-to-one correspondence with computer instructions.

Asynchronous transmission -- A transmission process such that between any two significant instants in the same group, there is always an integral number of unit intervals. Between two significant instants located in different groups, there is not always an integral number of unit intervals.

Backups -- Provisions made for the recovery of data files or software and for restart of processing or use of alternative computer equipment after a system failure or disaster.

Bandwidth -- The range of frequencies within which performance, with respect to some characteristic, falls within specific limits. For systems, bandwidth is commonly defined at the points where the response is 3 decibels less than the reference value.

Bode plots -- A plot of log-gain and phase-angle values on a log-frequency base.

Boolean algebra -- Pertaining to the processes used in the algebra formulated by George Boole.

Bus -- One or more conductors used for transmitting signals or power from one or more sources to one or more destinations.

Capacitance -- The property of a system of conductors and dielectrics which permits the storage of electrically separated charges when potential differences exist between the conductors.

CISC -- Complex instruction set computer.

Coaxial conductor -- An electric conductor comprising outgoing and return current paths having a common axis, one of the paths completely surrounding the other throughout its length.

Combinational logic function -- A logic function wherein for each combination of states of the input or inputs, there corresponds one and only one state of the output or outputs.

Computer networks -- An interconnection of assemblies of computer systems, terminals and communication facilities.

Concurrence -- Processes that may execute in parallel on multiple processors or asynchronously on a single processor. Concurrent processes may interact with each other, and one process may suspend execution pending receipt of information from another process or the occurrence of an external event.

CPU -- central processing unit.

Data base -- A set of data, part or the whole of another set of data, and consisting of at least one file that is sufficient for a given purpose or for a given data processing system.

Data transmission -- The sending of data from one place to another or from one part of a system to another.

Debugging -- The process of locating, analyzing and correcting suspected faults.

Demodulation -- A modulation process wherein a wave resulting from previous modulation is employed to derive a wave substantially the characteristics of the original modulating wave.

Digital signals -- Information in the form of one of a discrete number of codes.

Data types -- A class of data characteristics by the members of the class and the operations that can be applied to them, for example, integer, real, logical.

Diagnostic software -- A routine designed to locate either a malfunction in the computer or a mistake in coding.

Digital-to-analog conversion -- A device that converts an input number sequence into a function of a continuous variable.

Digital data communications -- Transmission of data in the form of digits or interval quantities.

Diode -- A semiconductor device having two terminals and exhibiting a nonlinear voltage-current characteristic.

Distributed parameters -- A parameter that is spread out over an electrically significant length or area.

Emulation -- The imitation of all or part of one computer system by another, primarily by hardware, so that the imitating computer system accepts the same data, executes the same programs, and achieves the same results as the imitated system.

Encode -- To express a single character or a message in terms of a code.

Error detection/correction -- The process where a code in which each data signal conforms to specific rules of construction so that departures from this construction in the received signals can be automatically detected, and permits the automatic correction, at the received terminal, of some or all of the errors.

Failure mode and effects analysis -- The identification of significant failures, irrespective of cause, and their consequences.

Frequency response -- The frequency-dependent relation, in both gain and phase difference, between steady-state sinusoidal inputs and the resultant steady-state sinusoidal outputs.

Frequency shift keying -- That form of frequency modulation in which the modulating wave shifts the output frequency between predetermined values, and the output wave has no phase discontinuity.

Hardware -- Physical equipment used in data processing, as opposed to computer programs, procedures, rules, and associated documentation.

Inductance -- The property of an electric circuit by virtue of which a varying current induces an electromotive force in that circuit or in a neighboring circuit.

Instruction set -- The set of instructions of a computer, of a programming language, or of the programming languages in a programming system.

Interface -- A shared boundary involving the specification of the interconnection between two equipments or systems. The specification includes the type, quantity and function of the interconnection circuits and the type and form of signals to be interchanged via those circuits.

Interrupt -- A suspension of a process such as the execution of a computer program, caused by an event external to that process, and performed in such a way that the process can be resumed.

Kirchhoff's Current Law -- The algebraic sum of the currents toward any point in a network is zero.

Kirchhoff's Voltage Law -- The algebraic sum of the products of the current and resistance in each of the conductors in any closed path in a network is equal to the algebraic sum of the electromotive forces in that path.

Light emitting diode (LED) -- A pn junction semiconductor device that emits incoherent optical radiation when biased in the forward direction.

Lumped parameters -- Parameters effectively concentrated at a single point.

Modulation -- The process by which some characteristic of a carrier is varied in accordance with a modulating wave.

Multiplexing -- The combination of two or more signals into a single wave (the multiplexed wave) from which the signals can be individually recovered.

Multitasking -- Special methods and systems designed to achieve concurrency by separating programs into two or more interrelated tasks that share the same code.

Norton's Theorem -- States that a linear time-invariant one-port is equivalent to a circuit which consists of the driving-point admittance of the one-port shunted by the short-circuit current of the one-port.

Ohm's Law -- The current in an electric circuit is inversely proportional to the resistance of the circuit and is directly proportional to the electromotive force in the circuit.

Operating systems -- Software that controls the execution of programs. An operating system may provide services such as resource allocation, scheduling, input/output control, and data management.

Operational amplifiers -- An amplifier, usually a high-gain dc amplifier, designed to be used with external circuit elements to perform a specified computing operation or to provide a specified transfer function.

Optical couplers -- An optical coupler provides isolation using a short length, optical path.

Oscillators -- A non-rotating device for producing alternating current, the output frequency of which is determined by the characteristics of the device.

OSI model -- Stands for Open System Interconnection. The use of standardized procedures to enable the interconnection of data processing systems in networks.

Passive filters -- A filter network containing only passive elements, such as inductors, capacitors, resistors, and transformers.

Peripherals -- Equipment external to a basic unit. A tape unit, for example, is peripheral equipment to a computer.

Phase locked loops -- A circuit for synchronizing a variable local oscillator with the phase of a transmitted signal.

Phasor -- A complex number.

Photodiodes -- A diode designed to produce photocurrent by absorbing light.

PLD's -- Programmable logic devices.

Protocols -- A formal set of conventions governing the format and relative timing of message

exchange between two communications terminals.

Queues -- A list that is accessed in a first-in, first-out manner.

Rectifier circuits -- A circuit bounded by two circuit terminals that has the characteristic of conducting current substantially in one direction only.

Reliability -- The ability of an item to perform a required function under stated conditions for a stated period of time. Note: The term reliability is also used as a reliability characteristic denoting a probability of success, or a success ratio.

RF communications -- A radio frequency communication channel or channels used to connect a transmitter and one or more receivers.

RISC -- Reduced instruction set computer.

Schematic capture -- The process of inputting a diagram showing the scheme of a circuit or system using graphic symbols in a computer program.

Searching -- To scan a set of items for those that have a desired property.

Sequential logic function -- A logic function in which there exists at least one combination of input states for which there is more than one possible resulting combination of states at the outputs.

Series connections -- A connection of two-terminal circuit elements such that only one current path exists through the connection.

Simulation -- The representation of an actual or proposed system by the analogous characteristics of some device easier to construct, modify, or understand.

Sinusoidal signal -- A function of the form A sin(x,a) where A is the amplitude, x is the independent variable, and a the phase angle.

Software -- Computer programs, procedures, rules, and possibly associated documentation and data pertaining to the operation of a computer system.

Sorting -- The process of arranging data or items in an ordered sequence by applying specific rules.

Spectral analysis -- The process of determining the distribution of the amplitude (and sometimes phase) of the components of a wave as a function of frequency.

Spectrum analyzer -- An instrument generally used to display the power distribution of an incoming signal as a function of frequency.

Spreadsheets -- A screen-oriented, interactive program enabling a computer user to layout

financial or other numeric data on a screen.

State -- The values of a minimal set of functions, which values contain information about the past history of a system sufficient to determine the future behavior, given knowledge of future inputs.

Statistical test model -- A model that relates program faults to the input data set (or sets) which cause them to be encountered. The model also gives the probability that these faults will cause the program to fail.

Statistical process control -- The use of statistical methods and techniques as a means of controlling the quality of a product or process.

Superposition Theorem -- States that the current that flows in a linear network, or the potential difference that exists between any two points in such a network, resulting from the simultaneous application of a number of voltages distributed in any manner whatsoever throughout the network is the sum of the component currents at the first point, or the component potential differences between the two points, that would be caused by the individual voltages acting separately.

Thevenin's Theorem -- States that the current that will flow through an impedance Z', when connected to any two terminals of a linear network between which there previously existed a voltage E and an impedance Z, is equal to the voltage E divided by the sum of Z and Z'.

Three-phase systems -- A combination of circuits energized by alternating electromotive forces which differ in phase by one-third of a cycle (120 degrees).

Transfer function -- A response function for which the variables are measured at different ports (terminal pairs). The variables are usually considered to represent an input signal and a response to that excitation.

Transducer -- A device to receive energy from one system and supply energy, of either the same or of a different kind, to another system, in such a manner that the desired characteristics of the energy input appear at the output.

Transient response -- The time response of a system or device under test to a stated input stimulus.

Transistors -- An active semiconductor device with three or more terminals. It is an analog device.

User interface -- The portion of an interactive computer program that issues messages to and receives commands from a terminal user.

Virtual memory -- The storage space that may be regarded as addressable main storage by the user of a computer system in which virtual addresses are mapped into real addresses. The size of virtual storage is limited by the addressing scheme of the computer system and by the amount of

auxiliary storage available and not by the actual number of main storage locations.

Wordprocessor -- A text editing unit or software program designed for the preparation, storage, and dissemination of text initiated on typewriter-like devices.

Workstation -- A high-performance, single-user microcomputer or minicomputer that has been specialized for graphics, computer-aided design (CAD), computer-aided engineering (CAE), or scientific applications.

Zener diode -- A class of silicon diodes that exhibit in the avalanche-breakdown region a large change in reverse current over a very narrow range of reverse voltage. Note: This characteristic permits a highly stable reference voltage to be maintained across the diode despite a relatively wide range of current through the diode.

KNOWLEDGE UNIT INDEX

Arranged Alphabetically by Tag Name

KNOWLEDGE UNIT INDEX

Arranged Alphabetically by Knowledge Unit Name

INDEX OF COURSES

Arranged Alphabetically by Course Number

INDEX OF COURSES

Arranged Alphabetically by Course Name