Teaching the Nonscience Major: EE101—The Digital Information Age

Roman Kuc, *Senior Member, IEEE*

***Abstract*—EE 101—The Digital Information Age, a course taught for the past six years to nonscience majors and freshmen considering electrical engineering as a major, is one the largest courses at Yale with a cumulative enrollment of approximately 2700 students. The goal is to describe how common-place digital information systems work and *why* they work that way by illustrating clever engineering solutions to technological problems. The course considers the following topics: information sources, logic gates, computer hardware and software, measuring information using entropy, error detection and correction coding, compression, encryption, data transmission and data manipulation by computer. Earlier versions of EE101 included both hardware and software projects. The hardware project was to implement a bean counter using digital logic modules. The software project involved writing a personal World Wide Web page and developing a Web page for a Yale-affiliated organization. Recent versions replaced the hardware project with additional Internet projects that receive data from a Web page viewer and that measure transmission times and the number of nodes between a source and destination. Having completed the course, students feel that they have an appreciation for the digital information systems they encounter on a daily basis.**

***Index Terms*—Course for nonscience majors, freshman engineering course, introductory engineering course.**

I. Introduction

THE PROBLEMS with teaching science and technology to nonscience majors have been actively discussed [1]–[4]. The main problem is dealing with the wide spectrum of the students' experience in math and science. A secondary problem is finding projects that are meaningful, instructive, and satisfying. This paper describes the author's attempt to solve these problems.

EE101 is a course for nonscience majors and freshmen considering electrical engineering as a major. In addition to teaching students about electrical engineering, the student is invited to be an engineer for one semester—to think quantitatively, to understand the design of simple digital systems, and to develop pages on the World Wide Web. The course has been taught for six years (1995 enrollment of 195, 1996—486, 1997—789, 1998—382, spring 1999—335, fall 1999—235, 2000—267). The large initial 1995 enrollment, with no course history, indicates that there is a need for such a course. Over its six-year history EE101 has undergone several modifications from its initial version to its current form. The significant changes are described along with their motivations.

The course attempts to teach technology in the least stressful manner to allow *poets*, who would not normally have access to this material, to take the course. Among the difficulties with teaching a 100-level course are that there are no prerequisites and the course itself is not usually a prerequisite for follow-on courses. In a student's time and effort, it competes with courses in the major. Teaching such a course introduces a challenge to make the course accessible to the liberal arts major, while still making it interesting for the potential engineering major.

This paper describes the successes and shortcomings in six years of teaching this course. In Section II, we give an overview of the course material. Section III describes the educational objectives for the course. Section IV summarizes student impressions as provided on course evaluations, followed by the instructor's impressions. The paper concludes with a discussion on whether an EE101-type course is appropriate for your school.

II. Overview of Course

EE101 lasts 13 weeks and consists weekly of two 75-minute lectures and an optional recitation session, during which students get additional help and can prepare for tests. The course materials include the text written by the instructor [5], a scientific calculator, and a 20-page pamphlet describing a systematic procedure to compose and publish a simple Web page on the Yale computer system. Students are predominantly liberal arts majors and tend to be equally distributed over the freshman to senior classes.

A. Topics

The topics for the course were selected by starting with the digital systems students encounter daily, such as digital clocks, bar codes, credit cards, proxy-cards, keyboards, modems, encryption, faxes, cell phones, and IR remote controls, and then including the physical and mathematical principles that explain their operation. Each topic forms a self-contained concept that can be explained in one lecture. The course is a collection of such concepts which are unified by the digital information theme. Physical and mathematical principles that can be applied to several topics are favored over those that have more limited application.

The following topics are covered in the lectures, with each topic corresponding to a chapter in the text. Selected equations are included to indicate the technical level.

1) Digital information sources. Information is defined as a quantity that a system needs to complete a task. The me-

Manuscript received January 23, 1997; revised November 28, 2000. The equipment for demonstrations was provided by a grant from the Paul Moore Memorial Fund at Yale.

The author is with the Department of Electrical Engineering, Yale University, New Haven, CT 06520-8284 USA (e-mail: Roman.Kuc@Yale.edu).

Publisher Item Identifier S 0018-9359(01)03872-9.

chanical switch, familiar to all students, is treated as a source of digital (binary) information. An open pair of contacts represents a logic 1 and a closed pair a logic 0. Push-button, mercury, and magnetic reed switches are employed in applications including lighting a lamp when the car door opens, and inflating an air bag in a collision. Configuring switches in a matrix reduces the number of wires needed to determine the unique identity of the depressed switch. A computer key board illustrates the operation of a switch matrix.

Commonly-encountered optical detectors produce a switch closure when the detected light power exceeds a threshold. The inverse square law explains optical sensor operation in the far field as the power produced by a detector at range r being equal to

$$P_D(\alpha r_o) = \frac{P_D(r_o)}{\alpha^2}$$

where α is a scaling factor and r_o is a reference range where a calibration value is available. A beam-interrupt optical sensor counts customers entering a store, and an infrared remote control illustrates optical transmission of binary data. Bar code scanners and auto focus cameras employ reflectance sensors.

2) Analog-to-digital conversion. Analog-to-digital conversion is the other main source of digital data and transforms analog waveforms into bit sequences. Sampling and quantization are illustrated using audio waveforms and images. The Nyquist criterion specifies the minimum sampling frequency, f_s, to be greater than twice the maximum frequency in the waveform $f_{\max}$, or $f_s > 2f_{\max}$. If this is not the case, aliasing occurs. Sinusoids of frequency f_o illustrate aliasing. When $f_s \leq 2f_o$, the alias frequency f_a results, with

$$f_a = |f_s - f_o|.$$

A selection of popular rock songs is under-sampled to cause unintelligibility (due to aliasing) and students try to identify the song as the sampling frequency is increased.

Quantization is the process of expressing sample values in a finite number of bits. The binary number system is introduced to specify sample data values. The noise power level for a quantizer having step size Δ equals

$$\sigma_n^2 = \frac{\Delta^2}{12}.$$

Signal power level is defined in terms of the measured rms-value of the signal, S_{rms}, as

$$\sigma_s^2 = S_{\text{rms}}^2.$$

This leads to the definition of signal-to-noise ratio σ_s^2/σ_n^2, and its evaluation in decibels units, $10\log_{10}\sigma_s^2/\sigma_n^2$.

Digital-to-analog conversion illustrates how bits generate analog waveforms. Audio waveforms from children's talking books are displayed on a digital storage oscilloscope and analyzed to determine the sampling rate, quantizer resolution, number of bits, and memory size.

3) Digital logic. Combinational and sequential logic gates are the building blocks that implement simple logic circuits. The elemental combinational AND, OR, and NOT gates are used to design a logic circuit that recognizes a particular binary pattern. A truth table is implemented as a logic circuit by OR-ing elemental logic circuits that recognize each input binary pattern that produces a logic 1 in the truth table output. One example implements the exclusive-OR (ExOR) gate, which occurs in binary addition and later for encrypting data. Another implements the truth table that converts binary-coded decimal (BCD) patterns to drive a seven-segment display. Techniques to reduce the gate count illustrate the cleverness in the design process.

Set–reset and toggle flip-flops are the sequential logic gates discussed, the former being the elemental memory unit and the latter being the elemental counting unit. Timing diagrams illustrate their behavior. Both logic types are combined to form a modulo-N counter, an addressable memory, a keyboard switch matrix decoder, and a digital clock.

4) Computers. A brief overview of computer hardware and software is presented. The smart card provides an example of an elementary computer, meaningful since students use a contactless smart card (a *proxy-card*) for gaining access to their residential college. Multiprocessor computer systems speed up the search of multiple databases and efficiently serve a large number of simultaneous users.

5) Information measurement. Elemental probability theory is used to model information sources. The relative frequency of a symbol produced by a source motivates the use of probability. The *entropy* $\mathcal{H}$ computes the information content of data generated by a source producing M symbols, $X_1, X_2, \ldots, X_M$, with their associated probabilities, and equals

$$\mathcal{H} = -\sum_{i=1}^{M} P[X_i]\log_2 P[X_i] \text{ bits/symbol.}$$

(Yes, $\sum$ has uses other than for fraternities.) This idea extends to the information content of data files. The *effective probability* of symbol X_i occurring N_{X_i} times in a data file containing a total of N symbols equals its relative frequency

$$P_e[X_i] = \frac{N_{X_i}}{N}.$$

The *effective entropy* uses the effective probability values to predict the degree of compression possible. Later, a Huffman coding procedure approaches this degree of compression.

A computer uses modulo-N arithmetic to simulate a sequence of random numbers with the pseudorandom number generator formula

$$Y_n = [A \times Y_{n-1} + B]_{\mathrm{mod}(N)}$$

where A, B, and N are large integers. Using $N = 256$ generates random byte-sized bit patterns for encryption.

6) Information coding. The block diagram of a data transmission system illustrates that the channel characteristics determine the coding requirements: Channels with noise require redundancy for error detection and correction; noise-free channels allow compression; and nonsecure channels often require encryption.

Adding redundancy to information permits error detection, and adding more provides error correction capability. The simplest form of redundancy is data repetition, and a more efficient form computes additional parity bits and checks sums from the data. This latter method works properly only when errors are rare (the single error assumption), which is true for most commercial systems. Constant-length codes, digital watermarks, variable-length codes, and encryption are included in this topic. ASCII, credit card codes and bar codes illustrate constant-length codes. The U.S. Postal Service and Universal Product Code (UPC) bar codes employ error detection and correction techniques. Code-word structures permit error detection and check-sum digits allow error correction. Data extraction comes from the magnetic stripe on credit cards, along with the self-clocking feature, which allows the card to be swiped at different, and even nonconstant, speeds through the reader.

Huffman coding illustrates variable-length coding by assigning the shortest code words to the most probable symbols. The average number of bits per symbol achieved with Huffman coding approaches the entropy. A spinning pointer and a set of unequal pie cuts model a source generating random symbols having different probabilities. Theoretical and experimental results are compared and the differences are explained. Huffman codes compress data files by computing effective probabilities of the symbols. Huffman coding run-lengths make facsimile machines commercially viable.[1]

Encryption is modeled as an ExOR operation on data with a pseudorandom binary sequence. The properties of modulo-N arithmetic permit transmitting the encryption key securely over a nonsecure channel.

7) **Information transmission.** Data transmission techniques are presented for real-time and non real-time operation. Systems experience delay when they generate data at a rate faster than can be transmitted over a channel. The source data rate $\mathcal{D}$ equals

$$\mathcal{D} = R \log_2 M \text{ bits/second}$$

where R is the symbol generation rate in symbols/second and M is the number of different symbols. A modem signal constellation illustrates the case for $M = 64$. The channel capacity $\mathcal{C}$ equals

$$\mathcal{C} = B \log_2 \left(1 + \sigma_s^2/\sigma_n^2\right) \text{ bits/second}$$

where B is the bandwidth in Hz. Real-time operation can occur only when $\mathcal{C} \geq \mathcal{D}$. The effect of signal-to-noise ratio on the channel capacity is illustrated. The inverse square law quantifies the decrease in σ_s^2, and consequently in $\mathcal{C}$, as a data gathering probe ventures out into space. A similar argument determines the maximum spacing between antennas in a cell-phone network.

Asynchronous data transmission is examined by observing waveforms in one-way transmission systems (IR remote control) and two-way transmission systems (modems). Internet data packet formats and protocols are investigated by using TCP/IP for acknowledged data transmission and UDP/IP for best effort transmission when retransmission is impractical, such as telephony over the Internet. We compare the proposed asynchronous transfer mode (ATM) system using a dedicated connection with packet switching over the current Internet.

8) Information manipulation. This chapter illustrates what a computer can and cannot accomplish. The computational complexity of solving a problem with a computer algorithm was examined by discussing problems of size N having different complexities. Exponential complexity (2^N typical of tree searches) and factorial complexity (N! typical of searches for optimum routing) define problems as being *difficult* and incapable of practical solution when N is large but still encountered in practice. Game trees and the traveling salesman problem (TSP) illustrate difficult problems. The minimax procedure determines the best path through a simple game tree. Evaluation functions choose the best move in large game trees that cannot be fully evaluated. The game of tic-tac-toe illustrates the procedure. For difficult problems, suboptimal solutions are suggested (*The boss needs an answer*). For the TSP involving nine cites, first the shortest tour was found by exhaustive search. Then this optimal tour was compared with two suboptimal solutions. The first determines the tour by traveling to the nearest unvisited city, with the paths to the last few remaining cities being typically long. The second proceeds by continually guessing a random tour whose length is evaluated quickly, comparing it with the current shortest tour, and keeping the shorter of the two. This method has recently been proposed as a type of evolutionary computing [6]. The convergence properties of this approach motivate the need for faster computers and larger memories.

9) The future. The course concludes with a discussion of technology trends that are likely to continue (faster processors, cheaper memory, larger bandwidth, increased connectivity, more efficient human/machine interfacing), along with the advantages and disadvantages (loss of

[1]It amazes the students that the basic patents on facsimile transmission devices date from 1854.

personal privacy) that will occur with expansion of information technology. The class discusses the importance of an informed citizenry in a technological age.

Lectures take place in a 450-seat auditorium. (To accommodate the large enrollment in 1996–97, the instructor taught two consecutive 75-minute sessions of identical lectures. This was beyond the call of duty, but the lure of teaching a record-breaking course overcame sanity.) To assist the lecturer, 12 teaching assistants (TA) are assigned to EE101, one TA to each residential college in Yale. Course-related announcements were posted on the EE101 class Web page, which included a Newsgroup to discuss questions and answers and other points of interest.

B. Projects

The initial three offerings of EE101 (1995–97) required students to do both hardware and software projects. The hardware project was dropped after the third offering because of the difficulty of troubleshooting the often-obscure circuit problems. Hardware projects that were simpler to manage did not provide the requisite educational value. More sophisticated software projects that probed the Internet replaced the hardware projects in current versions.

Hardware Project: The initial vision for the hardware project had groups of two or three students first construct one of a set of logic modules and then interconnect the modules to implement a digital system. Subsequent course offerings utilized these modules, eliminating the troublesome construction phase. Students designed a bean counter. A pendulum breaking an IR beam-interrupt sensor provided a counting process. Each student was assigned a number to count between 17 and 56, with the binary value of each number containing two or three 1-valued bits. In preparation for the project, students designed two binary counters on paper. The first uses the conventional approach that recognizes the full binary pattern representing the number. The second is a simplified version that exploits the counting sequence and uses only the 1s in the binary pattern. Students implemented the simplified system in the lab. Accommodating 500 students for the hardware project was one of the most challenging (and in the end, overwhelming) tasks in the course: The educational impact did not merit the effort.

Software Projects: Initial course versions had students write a personal Web page and a page for a Yale-affiliated organization. Each student has an account on the Yale computer system, which operates under UNIX. A pamphlet containing instructions for writing a simple Web page using the *pico* text editor, creating a public_html directory, and assigning read privileges to files resides on the class Web page. This was sufficient to get students started. Students were allowed to use Web page composition packages, as long as they understood the basic HTML tags. Each residential college has a PC computer cluster and a Computing Assistant (CA) to help students with basic computer questions. In addition, an optional weekly recitation section was held in a computer cluster to assist students with their Web pages. It was satisfying to see a process of mentoring develop spontaneously, with experienced students often helping those with less experience.

The personal Web page was to contain five components at a minimum: an email tool, a link to a file in the student's directory, a link to another Web page, an image in JPEG format, and some form of animation, blinking or marquee feature. Even with their varied backgrounds, most students were able to compose a satisfying Web page. Many found this to be an appealing project to demonstrate their creative skills by implementing advanced techniques, such as novel background patterns, a counter indicating the number of hits, audio tracts, and video clips. No extra credit was given for these. Students typically spent more time than was necessary to complete the assignment to explore additional features of Web page design. Little lecture time is spent on this topic since students easily grasp it and were self-motivated to create a quality product.

The organizational Web page assignment had two purposes. The first was to give the students some experience with applying their newly acquired talents in a client/server context. The second was to give university organizations that were not yet on the Web an opportunity to be exposed to it. Students were encouraged to approach their favorite organization (department, course, sports team, fraternity/sorority, singing group, etc.) and offer to write a Web page. A list of Yale organizations interested in having a Web page was also posted. The default organization was the EE101 course. Page development was typically done on the student's personal computer account. If the client was happy with the result, the code is transferred from the student account to the organization account for permanent accessibility. Many students became Web masters for their organizations.

By the fourth course offering the market for organizations needing Web pages was depleted and having a Web page was no longer a novelty. The organizational Web page and hardware project were then replaced by additional Web-based projects and a report that integrated these activities:

1) **Dot-com Web page.** This Web page illustrates how information travels in the opposite direction, from viewer to owner. The class received a simple CGI-script written in perl, which emails information entered by the viewer to the Web page owner. The operation of the script, gaining access to it, and security issues were discussed. A sample Web page template required modification to meet the specifications. Students create a product or service to be marketed over the Web. The Web page was to contain a radio button selection of options, a pull-down menu of allowed credit card types, and text fields for entering name, address, and fictitious credit card number. The entered text and option selections, when submitted, were emailed to the student's account.
2) **Internet mapping projects.** Two additional projects introduced students to the topology of the Internet. Two Microsoft programs operating under MS-DOS were employed: *tracert* and *ping*. *Tracert* (trace route) lists the Internet router nodes between the source and a specified Web address. For example, typing *tracert www.yale.edu* lists the node names and IP addresses in the path between your location and Yale's Web server. *Ping* responds with the round-trip packet transmission time between the source and specified Web addresses. Students find a Web URL beginning with the first initial of their last name on

each of five continents, and list the nodes and the transmission times from Yale to each URL. Students print a world map, indicate the URL locations, and determine their distances from Yale. They plot the average ping times and number of nodes as a function of distance. They then explain why the transmission times and number of nodes are not proportional to distance, which requires students to analyze their data.

3) **Report.** In recent versions of EE101, the four Web projects (personal Web page, dot-com Web page, ping, and tracert) are combined into a report in which the student provides an overview of the procedures employed and the knowledge gained. Hard copies of the Web pages, the corresponding HTML printouts, and data from ping and tracert were included in an appendix. The instructions indicate that the report should be understandable by the students' grandchildren, when they find it in the attic and wonder what the grandparent learned at Yale. The TAs graded the reports, and the instructor issued the final grade. This activity was instructive for determining what students gleaned from the projects and what they found most appealing. It was also entertaining to read the various spins used by humanities majors in describing their technological projects.

C. Grading

How grading is done is among the first questions asked by students. Grading is difficult for a course in which the students' prior experience varies widely. The intended audience for EE101 includes students with little prior experience who are willing to expend a reasonable amount of time attempting to grasp the material. Obviously, these will have a harder time than students more experienced in math and science. Since all students learn novel aspects about technology in the course, grading was lenient in the early versions of EE101, which partially explains their large enrollments. Non-science majors often prefer not to have a curve, as they are afraid that a few gifted students or science majors would spoil the grade for the rest. There should also be a mechanism by which a student can compensate for a poor grade on a test due to improper preparation in a strange new subject.

After six years experience, the following grading procedure is most effective. The grade is based on four tests, each counting 21 points, and the report, also counting for 21 points. Obvious trivial mistakes were penalized by giving half credit. There was no curve: 83–105 = A, 80–82 = A−, 77–79 = B+, 73–76 = B, 70–72 = B−, …, 50–52 = D− and 0–49 = F. Expanding the A interval upward allowed inclusion of a challenging (*top-bit*) question on each test and a point for creative expression in the report. Students getting the rare perfect test score earned bragging rights and received a prize—a plastic pocket protector pen holder.

Adding scores is preferable to averaging because when averaging, inevitably students whose scores fall 0.1 below a threshold, especially the one dividing B+ and A−, come to discuss, argue, or plead for that tenth of a point. Students falling near a threshold have an option. They could take the optional final exam that covers the entire course. The exam score then replaces one text score in a way that maximally helps or minimally hurts the student in computing the final grade. This approach follows this instructor's view that a student should be able to work toward achieving a desired grade.

The TAs grade the tests of students in their recitation sections and email the grade to each student, usually the evening of the test. Averages for the top five residential colleges are ranked in class to spawn some friendly competition. The test difficulty is set so that the average corresponds to about a B: The A-level students are mostly happy and students that fail the course have little reason to complain. Over six years, about 60% earned A's, and five students failed. Students doing poorly in the course withdrew before the grade was issued, so that most students completing the course gained the knowledge expected by the instructor.

III. Educational Objectives

The course material is described above. But what are the students expected to learn? To help us answer this question the author utilized Bloom's Taxonomy ([7], or search the Web), which classifies educational goals into six categories of increasing desirability: 1) Knowledge (tested with questions, such as *What is an error correcting code?*); 2) Comprehension (*How is combinational logic different from sequential logic?*); 3) Application (*What is the maximum range of a detector that obeys the inverse square law?*); 4) Analysis (*Explain why the following data file cannot be compressed.*); 5) Synthesis (*Design an error correction code that can correct a maximum of three errors per code word.*), and 6) Evaluation (*Compare redundancy by repetition to the redundancy by adding parity bit, and check sum in terms of error detection and correction coding performance.*).

Recent 75-minute tests contain 20 questions (plus a top-bit question) with short fill-in answers and simple logic circuit designs. Questions covering the Web projects were included by giving simple HTML code containing a possible error and asking whether the intended operation will occur and, if not, why not. Each student brings a scientific calculator and one sheet of handwritten notes to the test.

Since students have a sheet of notes, questions in the first category are rarely asked. The sixth category motivates the typical top-bit question. The majority of questions fall into categories 2 and 3, with a few on each test in 4 and 5. This distribution tests a breadth of knowledge and a depth in selected areas that the students find personally interesting. It also provides an measure of competence for students considering engineering as a major.

Being equipped with talents acquired in EE101, several liberal arts majors have applied them effectively:

- One history major worked on a summer job for a U.S. Senator investigating conducting an election over the Web.
- A graduate majoring in English impressed her new employers by composing a Web page for the firm.
- Three students majoring in different fields started a company to link all college students in the U.S. (They have since sold their enterprise to a large communications company.)

IV. Impressions

Students taking EE101 often suggest course modifications and improvements. The last test includes the standard Yale course evaluation form soliciting student comments. About 90% of the students respond in some fashion, while about half provide comprehensive remarks, summarized by the following:

1) About three-quarters found the course worthwhile, feeling empowered by having gained some understanding about how digital information systems operate. Several stated that it was the most worthwhile course they had taken at Yale. There will inevitably be students who are dissatisfied. Some will find the material overwhelming; others trivial. Maybe not surprising, a few better prepared students complained that the grading was too easy, and that A's were given too freely. Unfortunately, these latter students tend to write the most caustic evaluations. After recovering emotionally, the instructor usually divines the source of the saddle burr, which then leads to improvements in the next offering. However, the objective of the course, to reach a wide nontechnical audience, invites this type of criticism.
2) The Web projects were an almost unanimous hit. Students appreciated the usefulness and timeliness of what they had learned in class.
3) The test format was appreciated. The four tests allowed students to study the material in reasonable two-chapter chunks. An optional final exam rewarded good students by allowing them more time to study for other courses. Students appreciate the novelty of taking an optional final exam to replace a low test score. In many cases, students review and learn material that they did not grasp during their initial exposure.
4) During early offerings of the course, some students felt that the tests were too *quantitative*, testing mathematical manipulation rather than conceptual skills. With a huge class it is difficult to grade test questions that have other than numerical values. Current enrollments permit short written answers to more conceptual questions to be included to indicate whether students understand the concepts.

The following summarize the impressions of the instructor:

1) An EE instructor must resign himself to the fact that liberal arts students are different than engineering students, but not necessarily better or worse, just different. Unlike engineering students, however, who faithfully attend every class, come on time, and wait until the end of class to leave, liberal arts students attend classes selectively and then tend to come and leave sporadically during class. This instructor found this tendency to be quite disconcerting, especially when a student leaves just before an important point is to be made. (Maybe the point was important only for the instructor?) On the positive side, teaching nonengineering majors allows the instructor to reexamine basic principles and engage in technical discussions from a novel point of view. The reward of seeing a student *get it* makes it all worthwhile.
2) In a service course such as EE101, a detailed syllabus describing the topics covered in each lecture is important. Non-science students use the syllabus to select the lectures that they need or want to attend.
3) In the initial course offerings students registered for the course either for a letter grade or a Pass/D/Fail grade. About one-third of the students selected the latter. With the Pass/D/Fail option, the students who cause frustration fall into one of three categories:
    a) those who aim at a C− grade in order to achieve a Pass by expending the minimal amount of effort;
    b) those who, having attained the necessary number of points to achieve their desired grade, stop coming to class;
    c) those who feel they have enough math and science background to pass the tests without attending lectures.

   The Pass/D/Fail option was dropped after 1997, possibly accounting for the reduction in enrollment.
4) Initial course offerings employed overhead projection slides taken from the figures and examples in the book, the idea being that the students would then not need to take laborious notes. But this proved to be a failure. It did not illustrate the thought process that an engineer undergoes in reaching a design solution. Each lecture in the current version of EE101 starts with an empty black board and develops the problem and solution in class to illustrate design and thought processes.
5) A course for nonscience majors and freshmen engineering students is different than courses in conventional engineering degree programs. Successfully teaching of nonscience majors requires that each lecture provides a concept that students find interesting enough to discuss with fellow students in the dining hall. The course should become a collection of such concepts which are unified by a common theme.
6) Mixing nonscience majors with engineering students allows a mentoring relationship to develop among students. Engineering students are typically willing to help liberal arts students in return for the recognition and admiration they receive, with the amount of admiration being roughly proportional to the amount of help given.

V. Is EE101 for Your School?

The question asked most often is: Is the course a success because of the material or the personality of the instructor? It is our opinion that it is a combination of the two, but the material covered should certainly be timely and relevant. Teaching such a course will elicit different reactions at different universities depending on the student body. It will take a few teaching cycles to adapt the course to a particular setting, but, once established, it should become a popular course that meets the needs of students who live in an increasingly technological age, and one that is satisfying and fun to teach.

ACKNOWLEDGMENT

The author would like to thank Dr. M. Ewing for providing technical assistance for the Web page and Internet projects, and Dr. J. Szinger for providing tutorial assistance for Web page design.

REFERENCES

[1] F. S. Barnes, "Engineering education under attack," *IEEE Trans. Educ.*, vol. 37, pp. 1–2, 1994.

[2] J. R. Lohmann, "Myths, facts and the future of U.S. engineering and science education," *Eng. Educ.*, pp. 365–371, Apr. 1991.

[3] S. B. Sample, "Engineering education and the liberal arts tradition," *IEEE Trans. Educ.*, vol. 31, pp. 54–57, 1988.

[4] M. A. Shamos, *The Myth of Scientific Literacy*. New Brunswick, NJ: Rutgers Univ. Press, 1984.

[5] R. Kuc, *The Digital Information Age*. New York: PWS, 1999.

[6] D. B. Fogel, "What is evolutionary computation?," *IEEE Spectrum*, vol. 37, no. 2, pp. 26–32, Feb. 2000.

[7] B. S. Bloom, Ed., *Taxonomy of Educational Objectives: The Classification of Educational Goals by a Committee of College and University Examiners*. New York: McKay, 1956.

Roman Kuc (S'75–M'77–SM'89) received the B.S.E.E. degree from the Illinois Institute of Technology, Chicago, and the Ph.D. degree in electrical engineering from Columbia University, New York.

His engineering career started at Bell Telephone Laboratories, where he investigated efficient digital speech coding techniques. As a Postdoctoral Research Associate in the Department of Electrical Engineering, Columbia University, he applied digital signal processing to diagnostic ultrasound signals to characterize liver disease. In 1979, he joined the Department of Electrical Engineering at Yale University, New Haven, CT, where as the Director of the Intelligent Sensors Laboratory he has been pursuing research in robotics and bioengineering. Current projects investigate the biosonar systems of bats and dolphins, in order to mimic their behavior with man-made systems and to implement biomimetic embedded systems for the disabled. At Yale, he has been Director of Undergraduate Studies in Electrical Engineering for many years. He is currently Director of Educational Affairs in the Faculty of Engineering. He is the author of *Introduction to Digital Signal Processing* (New York: McGraw-Hill, 1988)and *The Digital Information Age* (New York: PWS, 1999).

Dr. Kuc is a past Chairman of the Instrumentation Section of the New York Academy of Sciences. In 1998, he received the Sheffield Distinguished Teaching Award, and in 1999, he was inducted into the Grand Order of the Golden Bulldog for service to Yale alumni. He was recently elected as Academician (Hon.) in the Academy of Higher Education, Academy of Sciences of Ukraine.