Cooperative Learning in a Digital Logic Course

Russell L. Pimmel

http://www.foundationcoalition.org

Introduction

Active and cooperative learning (ACL) techniques have proved to be effective in a wide assortment of areas.[1–6] Frequently, engineering faculty members have promoted and used ACL techniques in introductory engineering courses[7,8] and in senior capstone courses[9]; most agree that ACL has an important role in these two areas. However, acceptance of ACL in core or required engineering courses is less certain; examples in which instructors share how they have applied ACL approaches in core engineering classes may catalyze deeper conversations and promote broader acceptance of innovative approaches to teaching core engineering courses. In this document, an instructor shares an in-depth analysis of how he reshaped his digital logic course using ACL techniques. Also, he attempts to demonstrate how many ideas might be transferred to other core courses in electrical and computer engineering and other disciplines.

ACL techniques can be used in four types of activities: (1) laboratory projects, (2) extended out-of-class design, analysis, or research projects, (3) routine homework assignments, and (4) in-class exercises. This report, which focuses on in-class exercises, has two sections.

- The first section addresses general issues that might be applicable to any core engineering course.
- The second section then provides topic-specific examples of how ACL may be used in a digital logic course.

Section 1: General Issues

A. Nature of the Course

This report discusses the use of ACL in an introductory logic design course, a sophomore- or junior-level course required in electrical engineering, computer engineering, and computer science programs at the University of Alabama. The University is a residential campus with only a few commuter or part-time students. The assigned text is *Fundamentals of Digital Logic with VHDL Design* by Brown and Vranesic.[10] The course treats the first eight chapters of this text, dealing with combinational and sequential circuits using both traditional approaches (e.g., truth tables, state tables, K-maps) and more modern VHDL representations. During the academic year, the class meets three times a week for about fifteen weeks; during the summer session, it meets daily for a double period for five weeks. In addition, there are seven two-hour laboratory sessions in which students construct and demonstrate solutions to design problems.

During the academic year, the course is taught to 50 to 100 students in an auditorium-type classroom with long tables that seat fifteen or so. The classroom is equipped with a computer-projector system with an integrated document viewer. During the summer term, the course is taught to 10 to 20 students in a regular classroom with standard desks and either chalkboards or whiteboards.

B. Gaining Student Commitment

Many instructors trying ACL techniques for the first time encounter negative student reactions or resistance, or even antagonism.[11,12] Table 1 lists some approaches for overcoming student resistance and gaining their commitment.

Table 1. Strategies for overcoming student resistance and gaining commitment

Let the students know what you are doing and why	Indicate the importance of developing team skills and how ACL helps
Show data demonstrating positive gains obtained with ACL	Provide opportunities for students to express their feelings about the ACL approac

Students, like most faculty members, are comfortable with the familiar and dislike change. Spending a little time, particularly at the start of the semester, to give students a quick overview of what you will be doing and why can be important in overcoming these attitudes. I do several things. First, I let them know that we will be using a new instructional methodology that takes them out of the passive mode and involves them actively in classroom learning. In order to motivate them, I show some of Felder's data demonstrating that students taught in this way have higher GPAs in subsequent courses, higher retention and graduation rates, and better attitudes about the program, the discipline, and themselves.[6] I also show lists of attributes that employers consider important and point out that "good team skills" is always on these lists[13]. I have even used ABET EC student outcome (a)–(k) lists to emphasize the importance of team skills [14]. I then point out that enthusiastically participating in the in-class exercises will provide practice with this important skill in a technical setting.

Providing the students with opportunities to express their feelings about these ACL approaches also alleviates some of their concerns. A couple of times during the semester, I use the "one-minute paper" to get their reactions to specific issues [15]. After three or four weeks, I ask them to list one or two things that they like most about the course and one or two things that they like least. Working in teams is almost always one of the more popular positive choices; in fact, it usually appears on most lists, either as a positive or negative choice. When I show the tabulation of their comments at the next class, I remind them about improved learning with ACL techniques and importance of teaming in a technical setting.

Reporting these data and providing the associated reminder seem to increase students' acceptance of the approach. As the semester progresses, I use this process several times, asking them, for example, to indicate what they like most and least about the in-class team activities and whether or not they like switching teams and why. Results from two recent semesters are given below in the section titled "Student Reactions."

C. Nature of In-class ACL Activities

During each class, I use multiple short in-class activities. Although the structure of the class varies considerably, depending on the nature of the material and my impressions of what is working at that time, I usually lecture for five to twenty minutes, then assign a short in-class activity that will take from one to ten minutes. I then alternate these two throughout the class period. I hardly ever lecture for more than twenty minutes at a time and usually my minilectures are shorter than this. Most of the time I like short exercises, lasting less than five minutes, but there are occasions when I use longer ones. For longer exercises, I typically have "breakpoints" part of the way through the problem, where we stop so that everyone can check his/her progress and the correctness of his/her approach.

While the teams work on the exercises, I circulate around the room and eavesdrop on the team discussions. This shows my commitment and allows me to monitor the students' perceptions of the material, their approaches for dealing with the problems, and their ability to work as a team. It also provides a real opportunity for teams or individual students to ask questions and clarify any confusing issues. Continuing instructor involvement during in-class exercises is crucial.

My exercises generally fit into two categories, with several alternatives for each, as shown in Table 2.

Table 2. Types of in-class exercises

Additional examples that are very similar (useful with complicated, detailed procedures) Additional examples that expand or extend a concept	Reflective activities that deal with the organization of the course topics Reflective activities that require relating or interpreting course concepts

Additional Examples Many of my exercises use a second example that may or may not be very similar to the one that I just worked in my minilecture. If the process being taught and demonstrated is long or complex, then repeating it with a similar example may be beneficial. However, many times I find it more effective to use a second example that extends or expands the ideas demonstrated in the first example or introduces additional complexity or functionality. The second form of exercise is a reflective activity that requires the student to think about the material outside of the usual "problem" format and encourages the student to reflect on the organization of the course material or on the relationship between the present topic and others.

An example of an activity that broadens student understanding occurs when discussing encoding of signed integers. After working a straightforward example, I ask the teams to use a fixed-length code word to encode positive and negative numbers that are outside the range of the code before introducing the idea of range limitations (e.g., encode + 200 and – 200, using an 8-bit 2's complement code). Letting the students work in their teams to recognize and explain some of the inconsistencies that result (i.e., a representation for a positive number that has a 1 in the sign bit) reinforces the idea of the sign bit and allows them to recognize that any fixed-length code can encode only a limited range of values. Typically, I will make suggestions to the class as the teams encounter specific issues and generate questions, rather than letting them struggle through the whole process themselves.

Another opportunity to broaden understanding occurs in converting AND-OR diagrams into NAND-only or NOR-only diagrams. After working a straightforward example, I ask the teams to convert a second example that includes AND-to-AND or OR-to-OR connections and requires logic-level conversions using inverters, without prior discussion of this idea. Again, with some prompting, they usually see the problem and the need for the extra inverters.

When using activities to introduce functional variations or additional complexity, I may ask teams questions about the solution rather than asking for the complete solution. For example, when discussing decoders, I may ask the teams to describe the changes in the Boolean equation for one of the output signals when the input signals or output signal or both become asserted-low signals. Alternatively, I may ask them to describe the changes in the output condition for changes in signal assertion for one or two input conditions. The issue of complexity is illustrated by an activity in which teams to consider a 10-bit decoder and determine the number of output signals and equations of one or two of the output signals. Another example is to have teams determine the outputs conditions in hex notation for a few input conditions for a complex combinational circuit when the input conditions are given in hex (e.g., determine the output for an 8-bit binary subtractor when the input is B3 and FF).

Reflective Exercises The second form of exercise is a reflective activity that requires students to think about the material outside of the usual "problem" format and encourages students to reflect on the organization of the course material or on the relationship between the present topic and others. As an example of a reflective in-class activity, I frequently ask teams to determine the next topic in the course. This works particularly well in a digital logic course because of all the inherent parallelism (e.g., sum-of-products then product-of-sum forms, multiplexers then demultiplexers, decoders then encoders, additions then subtraction, combinational then sequential circuits). Other examples involve relating ideas. For example, when introducing XOR gates, I show them a two-input XOR gate with the inputs labeled "DATA" and "CONTROL" and ask them what function the circuit performs, anticipating the use of this circuit as a controlled inverter in an adder/subtractor circuit.

Rhetorical questions provide a second source of reflective exercise. Many faculty members use rhetorical questions to try to stimulate students to think about the ideas being discussed. Frequently, they do not expect the students to be able to answer the question, but they ask it to raise awareness, create interest, and stimulate some preliminary thoughts about the topic before the instructor develops the topic. Before I began using ACL techniques, I simply used rhetorical questions as transitions to new ideas and really did not expect any answers. In fact, if the truth be known, I really did not expect most students to even think about the question. Now I pose these questions as team activities and, after a brief discussion, poll a few teams for their answers. They often come up with an answer—it may not be the correct answer (i.e., the one I had in the

mind), but as a team they are more willing to take a chance and provide an answer than as an individual one. In the past, when I asked individual students these questions, most were unwilling or unable to give answers; I suspect that most were even unwilling to consider the idea and begin working toward an answer. But when an answer comes from a team rather than an individual, it reduces the risk or embarrassment.

D. Forming Teams

There are many ways of forming teams, but most experts agree that the instructor should assign teams rather than letting students self-organize [16]. Table 3 outlines my guidelines for forming in-class teams.

Table 3. Guidelines for quickly forming in-class teams
Use random team assignments—ignore race, gender, and ability issues
Use teams containing three or four students
Reorganize teams three or so times during the semester

An instructor can consider many factors in forming teams, including ability, background, race, gender, interests, and personality type. Considering each of these makes team assignment more complicated. Random assignment provides an alternate approach that is simpler and more efficient and, at the same time, considerably more effective than letting students organize themselves into teams.

When I teach in a standard classroom with individual desks, I usually use teams containing four students because face-to-face discussions are facilitated if each team's four desks form a small square. When I teach in the auditorium, I have three-student teams because, with the row seating arrangement, three is the largest number that can engage in face-to-face discussions. In forming teams, I use quasi-random assignment. On the first day of class, I have students fill in all the seats toward the front and sort them into teams using "squares" of four in a regular classroom or "rows" of three in the auditorium. I assign each team member a number (0 to 2 in the auditorium and 0 to 3 in the classroom), depending on his or her physical position in the team. I use this member-number to designate quickly a specific team member as reporter for a team activity or as the daily team recorder.

Twice during the semester, usually after each hourly exam, I reorganize teams using some structured movement based on the member-numbers. For example, in the auditorium, I have the #0 members remain in place while the #1 and #2 members move a row or two back, respectively. Students leaving the back rows move to the front. Many other similar perturbations are possible. In the standard classroom, I may have alternate rows move back two seats, with those in the last two rows moving to the front. When I reorganize the second time, I make sure that most students are working with new cohorts, but, in both reorganizations, quickness is more important than accuracy.

E. Accountability—Reporting Out

In order to motivate student involvement, having teams report the results of their in-class discussions introduces some accountability into the process. Table 4 lists suggestions to help make this an efficient process.

Table 4. Efficient strategies for having students report out after in-class exercises	
Do not have all teams report—use a sampling process	Do not have students transcribe their entire solution
Randomly select the teams and reporter at the last minute (encourages involvement)	Have students report only a part of the solution or a feature of it

Usually, having all teams report out would require an inordinate amount of class time and thus be impractical. As a result, some sort of sampling strategy is needed, and, even then, time constraints frequently limit the amount of reporting each team can do. Strategies that select teams and the specific reporter at the last minute help keep all students involved. I typically pick teams in a section of the classroom, frequently using teams in one or two rows or columns or teams in the front or back. I try not to select the same teams on a regular basis and go out of my way to choose teams that I have not heard from recently or teams that I believe are not participating fully.

Having a few students go to the board and transcribe their teams' solutions to the exercise consumes a large amount of class time, so I rarely have students do this. Fortunately, in the auditorium, I have the document viewer, so I frequently collect three or four student papers and project them for discussion. Other times—this is what I do if I do not have a document viewer with a projector—I have the teams report some part of their solutions. For example, I will have a few team reporters read the final answers their teams generated, skipping the steps leading to it, and write these on the board for discussion. Alternatively, I may have them report some aspect of their solution, such as the number of terms in the equations of a K-map minimization problem, the number of simple decoders in the hierarchical design of a more complicated decoder, or the number of states in drawing a state diagram for a specific finite-state machine.

A voting process is another interesting approach that I use when the question has a short objective, such as a number. In this approach, which can add a little fun to the class, I ask teams to report their answers verbally and write them on the board until all the answers are reported. Then, after giving them a minute or so to think about the answers, I ask students to vote for one of the solutions by raising their hands, usually insisting that everyone vote, in order to try to force each student to make a choice. Sometimes I tell them that we will decide the answer by a majority vote. If it looks as though the vote is going the wrong way, I give myself enough votes to choose the correct answer. As an example, when I ask the teams to draw the input-output model for an 8-to-1 multiplexer, I might ask them to indicate how many total input signals the device has. Alternatively, when dealing with a fairly complicated design development, I might ask how many rows are in the complete truth table or state table. This approach also works well on short exercises on integer encoding and decoding and on number conversions and arithmetic.

Another approach is to ask one reporter for a part of the answer and continue soliciting items until the total answer is obtained. In one exercise I ask teams to identify all of the changes required in a VHDL solution in order to change one or more parameters of the design—for example, convert a counter from an 8-bit device to a 24-bit device or convert an asynchronous reset signal into a synchronous one. I then poll selected teams, asking each for one for a change that must be made, and then finish the process by asking if there are any more changes. If a team suggests an inappropriate change, I might ask the next team whether or not they agree, in an effort to get them to resolve the error. Even if the change is appropriate, I sometimes ask the next team whether or not they agree, just to add variation to the process and keep it interesting.

F. Accountability—Submitting and Grading Team Products

Having random teams report out after each exercise helps keep the majority of the students involved. However, when I used in-class activities with only ungraded oral reporting, I always had some students who obviously treated the in-class activity time as recess or relaxation time and probably several others who were not fully engaged. Requiring teams to turn in their work, either at the end of class or after each exercise, introduces more accountability that encourages students to participate. Collecting and grading these solutions eliminates the slackers. Table 5 shows suggestions for dealing with students' written solutions to the in-class exercise.

Table 5. Efficient strategies for dealing with written solutions to in-class exercises

Require written solutions for in-class exercises	Require each team to prepare a single written team solution
Use a simple grading scale	Grade a simple, hopefully objective, aspect of the written solution
Use a simple approach for selecting different recorders for each class period	

I typically have the team recorder list the names of all the team members present, which provides attendance information if the instructor wants to use it. During each exercise, each recorder documents his or her team's results on a single sheet of paper, which is submitted at the end of class. Each day's recorder is designated using a simple date-based algorithm in which each team divides the date by the number of students on the team, with the remainder designating the member-number of that day's recorder. For example, if there are three-member teams and the date is March 19, then the member #1 is the recorder (19 divided by 3 yields a remainder of 1). This sounds complicated and takes a little time to explain initially, but, after the first week, the teams identify their recorder with no prompting from me.

In order to encourage the students to work seriously on the exercises and the resulting products, the instructor must grade the submitted work and include these grades in calculating the course grade. I usually count the in-class activities for 5% to 10% of the total course grade, a relatively small weight that encourages students to take these activities seriously but does not distort the purpose of the exercises and the importance of participation and teamwork.

The submitted paper will have solutions for one, two, or more exercises. In a typical class, to say nothing of an auditorium class, this could lead to a large amount of work unless the instructor develops an efficient procedure. I (or my grader) will grade only one exercise, usually the simplest one to grade. In my current scheme, I give ten points if the solution is correct and eight points if it is not, and I do not spend much time fretting over whether a solution deserves eight or ten points. Many times I simply give ten points and indicate that the papers were not graded. I use this liberal grading scheme because it is quick, while encouraging the students to try the exercise (hence the eight points for any solution). The extra two points, along with the inherent desire to get the right answer and the possibility of being selected to report during class, encourages the students to try to get the correct solutions.

G. Covering Material

Doing in-class ACL exercises reduces the amount of lecture time that the instructor has for covering material. As result, many instructors fear that they will not be able to get through the course syllabus. Although much of the ACL literature indicates that this is not a problem and that an instructor can deal with the same amount of material[12], I have not found this to be the case, and I have dropped some topics from my course that I previously included when I used the standard lecture format. However, I believe that students learn more with the ACL format than they did from my traditional lectures, in which I transcribed my notes for the students to copy, at times, rushing to make sure that I covered all the material (regardless of the students' understanding). When student learning is used as the performance measure, the ACL approaches are more efficient than lecture format, and the literature has several good references supporting this premise.[1–6]

H. Classroom Management

Using ACL techniques places different constraints on how an instructor manages classroom time and activities. Table 6 lists suggestions that will help make the ACL activities run smoothly.

Table 6. Aspects of classroom management

Use carefully planned, simple approaches to manage teams
Limit the amount of time allowed for each in-class exercises
Use efficient approaches in having students report out their solutions

ACL activities require some class time for administrative aspects to organize the teams, to describe and manage the processes involved, and then to modify the teams and processes each time the instructor makes changes. Careful planning and a little assertiveness allow these to be accomplished efficiently. Actual in-class team exercises also use class time for describing the exercise, for working on the exercise, and for reporting results.

To be fair, the added time is not excessive because the instructor normally uses most this time to work additional examples. However, turning student teams loose to work on a problem and report their results typically requires more time that it takes the instructor to work a second example. In order to minimize the added time, the instructor needs to have both a clear statement of the exercise ahead of time so that the students quickly grasp the problem and an efficient method for reporting the answers.

When it comes to actually doing the exercise, I generally do not allow enough time for most teams to complete the exercise in a manner that satisfies them. Many teams find this annoying and frustrating; providing more time for the exercises is always mentioned by a majority of the students when I ask in a "one-minute paper" about what should be changed. I justify the short time allotment, since I am not trying to get the teams to develop a complete, accurate answer to the exercise, but I do want them to move from the passive mode to think about the issue, discuss it with some cohorts, and move toward a solution. The active thinking and discussing encourages them to view ideas from different perspectives. I explain my rationale for the short time allotment often, and, even if some students begin to accept the idea, it remains an issue with many.

Having the students report out is essential, but having all students believe that they have to be prepared to report out is more important. Thus, having some students report answers (perhaps only partial answers) is important, but it is extremely critical that this be done efficiently. Inexperienced instructors using ACL techniques can waste time in the reporting-out phase. Efficiency almost always precludes full reporting, where each team gets to describe their entire solution.

I. Problems

Even though I have used ACL techniques for years and have developed some successful strategies, I still encounter problems. Here I discuss several of them and describe my approach for dealing with them.

Individuals who do not participate in the team's discussion are one problem. This was a much more common problem before I started collecting and grading a single product showing each team's solutions for that class. In spite of this extra motivation, some students, who are usually very introverted and frequently very capable academically, feel that they learn best when they work alone and participate only in a token way. If I notice an individual who is not participating while the teams are working on an exercise, I talk to him/her individually to remind him/her about the importance of learning to work as a member of a team on technical problems and that he/she still can study alone outside of class. This will reach many of them, but there will always be some holdouts, especially in the larger classes.

Another problem results from random assignment of teams, since this can lead to a few teams containing only weaker students. Sometimes, none of the members of these teams understand enough to solve the problem or even to begin working on it in a meaningful way. As I circulate around the class while they are working on an exercise, spotting these teams is easy. I will stop and talk to them, to see if I can get them started. If this fails, I suggest that they spend the exercise time trying to identify one or two questions that they could ask in order to get started. Alternatively, if other teams are short members, then I offer them the option of joining another team or even just reassign each of them to another team. Additionally, I suggest that they will need to find a way to overcome their lack of understanding and let them know that I am available to help them.

After I have organized the teams, a few students drop the course or choose not to attend regularly, leaving some teams with one or two members. If their teammates are just gone for the day, then I let two-member teams work alone, but I ask a single individual to work with another team for the day. If their teammates are absent for several days in a row, then I ask them to make the change permanent.

Another problem that is frequently discussed in the literature is the concerns of the more knowledgeable (capable) students who may feel that they do not get much out of the exercises. I have not encountered this in conjunction with the in-class exercises, but I have seen it with other types of team activities, such as extended projects. In those cases, I remind them of their need to develop team skills and suggest that they will gain additional insights if they work to deal with their teammates' questions and misunderstandings.

Sometimes, the students misunderstand or even fail to grasp at all what I want them to do in an exercise. Carefully checking the statements describing the exercises before using them in class will reduce this problem and, after making the assignment, asking for a "check for understanding" by having some individual summarize the task usually takes care of many of the other confusing exercise statements. However, I still encounter times when a substantial number of the teams are either confused or completely stumped. When I detect this situation as I walk around the room and listen to some of the teams as they work, I will stop the exercise and either clarify the problem and let then them continue or I work the exercise myself and go on with the class.

J. Student Reactions

As noted earlier, I often use one-minute papers to provide the students with an opportunity to express their feelings about these ACL approaches (along with other aspects of the course) and also to alleviate some of their concerns. I report some of these data here in order to indicate student reactions to these approaches.

Several weeks into the semester I give students a couple of minutes at the end of class to indicate what they like most about the class and what they like least. The tables below show the results obtained in two recent semesters. Data in the first table indicate that, in responding to the "like most" question, nearly half of the students indicted some aspect of the course content, while a third or so mentioned teamwork or something similar. The remaining papers contained a variety of choices, but none appeared more than twice. Data in the second table indicate that, in responding to the "like least" questions, a majority of the students felt that classroom pace, particularly when doing the in-class exercises, was too fast.

Responses to "What do you like most about this course?"		
Semester	Fall '02	Spring '03
Material (logic design, computers, circuits)	40%	49%
Teamwork and classroom format	40%	24%
Pace and challenging style	—	18%
Others	20%	9%

Responses to "What do you like least about this course?"		
Semester	Fall '02	Spring '03
Too fast, too little time for exercises	55%	65%
Nothing	—	13%
Others	45%	33%

When I showed the students these data, I made two points. First of all, I told them that I try to make the class interesting and engaging and that this requires a fast pace. I sat through many slow-paced, boring lectures as a student and want to avoid that problem; perhaps I overcompensate. Secondly, I told them that I do not expect them to leave class with a complete understanding of the material—I do not "learn" them anything. They learn themselves, by reading the text and the notes, by doing the homework, and by talking and thinking about the material. My intention is to facilitate their learning through the classroom activity and the assigned homework.

Toward the middle of the semester, I asked them to indicate how they felt about the in-class exercises and why they felt this way. The table below shows overwhelmingly that the students had a positive reaction to the team exercises.

Responses to "How do you feel about the in-class team exercises?"		
Semester	Fall '02	Spring '03
Like them	74%	82%
Indifferent or dislike them	26%	18%

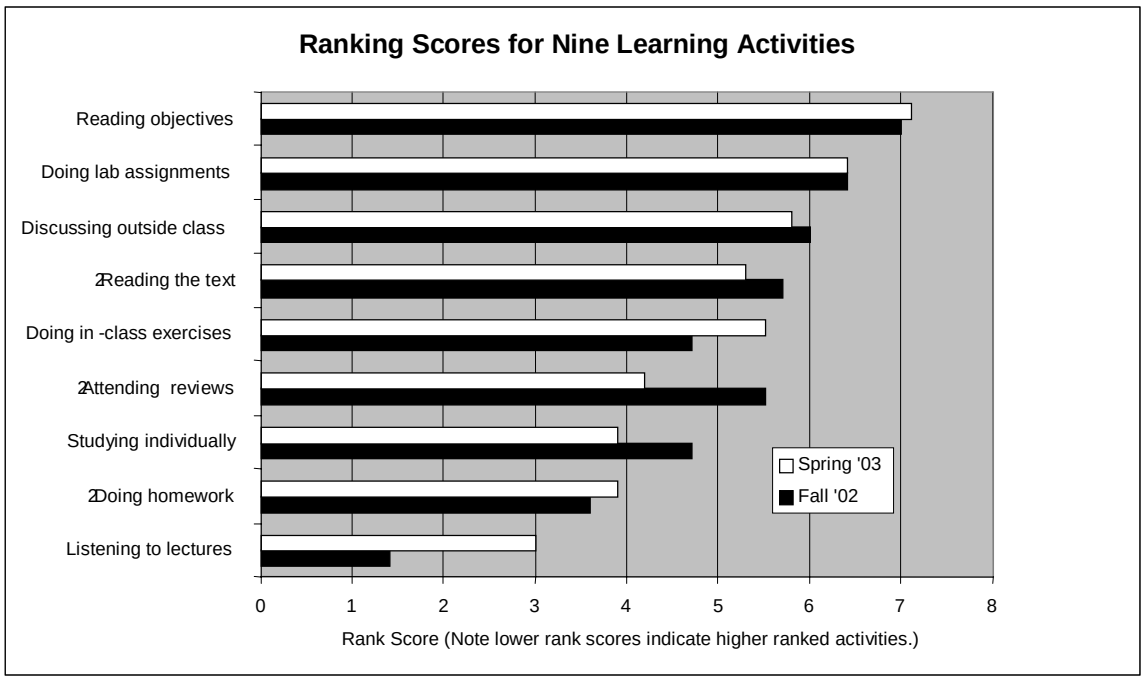
The most common type of comment indicated that these exercises help (or test) their understanding. Other popular comments concerned the need for more time for the exercises and the idea that students should be graded only on effort. A few indicated dissatisfaction because their teams were clueless.

About two-thirds of the way through the semester, after the students had changed teams twice, I asked them how they felt about changing teams and why they felt that way. The table below shows that about one-third of the students liked changing teams, about one-third disliked changing teams, and about one-third were indifferent.

Responses to "How do you feel about changing teams?"		
Semester	Fall '02	Spring '03
Like changing	31%	54%
Indifferent	31%	18%
Dislike changing	38%	28%

The comments from both the "like" and "dislike" teams were analogous. One pair of responses indicated that the student liked (or did not like) to meet or work with new people. The other pair indicated that the previous team was more (or less) effective than the new one.

At the end of the semester I ask the students to use values 1 to 9 to rank the following items in order of their importance in helping him/her learn the material: (1) attending evening review sessions, (2) completing lab assignments, (3) discussing course material with other students outside of class, (4) doing assigned homework, (5) listening to instructor's lectures, (6) reading learning objectives, (7) reading text, (8) studying lecture material individually, and (9) working with other students on in-class exercises. The figure below shows the average rank score for all nine learning activities for two recent semesters. In this figure, smaller rank scores indicate a higher ranking, so *Listening to the lecture* was the highest-ranked activity, and *Reading the objectives* was the lowest ranked. In Fall '02, *Doing in-class exercises* ranked third behind only *Listening to lectures* and *Doing homework*; while in Spring '03, the *Doing in-class exercises* ranked sixth ahead of *Discussing outside class*, *Doing lab assignments*, and *Reading objectives*. When the data from these two semesters were combined, *Doing in-class exercises ranked fifth*.



K. Instructor Commitments and Adjustments

Instructors committing to using ACL techniques in class must expect to make three important adjustments: they will spend more time in preparing classes, at least initially; they will have to give up some control in the classroom; and they will have to be flexible, adjusting to the interaction in the classroom in real-time. When an instructor decides to try ACL approaches for the first time, she or he must expect to encounter a leaning curve. Participating in a workshop or a conducting careful review of some of the literature [17–21] or Web sites [22–24] would provide a good start. Collaborating with colleague(s) who also are beginning to use ACL techniques in the classroom or collaborating with experienced instructor(s) can provide both intellectual and emotional support.

Even with a good introduction and supportive collaborators, an instructor unfamiliar with ACL approaches should plan to spend extra time preparing classes. Converting an existing set of class notes from a lecture format to an ACL format will take a novice instructor a substantial amount of time, perhaps as much as half the amount of time originally required to prepare the notes. If a novice instructor decides to incorporate in-class ACL activities as she or he develops a new set of course material, then he or she must plan on more preparation time, perhaps as much as 50% more. As an instructor develops experience with ACL techniques, the difference between preparing lecture-format vs. ACL-format class material decreases. For example, I can prepare new ACL class material as quickly as I can prepare lecture material, and I can modify an existing set of material in a short time. The extra preparation time required for ACL class material may deter some faculty members from trying these techniques, but I (and most of the other ACL-experienced instructors with whom I have discussed this issue) believe that the increased student involvement and interest make up for the extra preparation time.

Using in-class ACL techniques requires that an instructor give up some control in the classroom when he or she turns the student teams loose to complete a task. When assigning a task, that the students will proceed along the path that the instructor intended or that they will understand what the instructor expects is never certain. Well-designed activities minimize these risks, but even an experienced ACL instructor, to say nothing of the novice, should expect some surprises and have to make some real-time adjustments. This uncertainty, caused by the active participation of the students, changes the classroom from a static, predictable environment, in which the instructor controls content, pace, and direction, to a dynamic one in which unpredictable and, usually, interesting interchanges happen. This is never dull, and that is why instructors who stick with ACL classroom techniques for a few semesters say that they will never teach a traditional lecture class again. The first time I tried ACL techniques in an engineering course, I was also teaching a second course using traditional lectures. After several weeks of classes, when I became more comfortable with the ACL techniques, I found myself becoming bored about twenty minutes into the lecture in the second course, while the ACL class remained interesting and challenging for the entire period. I suspect that the students' reaction was similar.

I have found that a set of class material, including the ACL activities, may work one semester and fail the next, not unlike my prior experiences with the traditional lecture format, in which I found that a lecture from the same notes would work well sometimes and not well other times. Since there are so many uncontrollable factors (e.g., a test for most students in the period before or after your class, a big game the night before, the weather, your health and stress level) that influence the instructor-student interaction in a class, some of your classes will be exciting and some will be mediocre. The difference is that, with the ACL method, the students are involved in the turmoil, whereas, with the lecture method, you may be very aware of the shortcomings of a particular lecture, but the students are probably totally unaware, unless it is a real disaster. Using ACL techniques requires flexible instructors who are willing to interact with their students, observe their students' reactions, and make adjustments on the fly.

Section 2: Sample Exercises

In the following subsections I describe some of examples of the types of exercises that I use for each phase of the course.

A. Course Introduction

- On the first day of class, when I hand out a description of course policies, processes, and expectations of the course, I use interactive teams to increase their involvement. I ask them to read a specific section and discuss it in their teams. After each discussion, I ask a few teams to report what they read that was important, different, interesting.
- Also on the fist day, I ask teams to talk about rationale for success strategies rather than lecturing on them. I ask the teams to discuss specific questions (e.g., why they are taking this course, why it is relevant to their career goals, what they have to do to succeed in the course, why have digital approaches replaced analog ones). After selected teams report, I briefly give them my views.

B. Combinational Design and Analysis with AND, OR, and NOT Gates

- After lecturing on the symbol, truth table, and equation for the NOT, 2-input AND, and 2-input OR, I ask the teams to draw the symbol, truth table, and equation for a 3-input OR or AND gate.
- After showing them how to derive the logic equation using a multilevel AND/OR circuit with alternating gates and no embedded NOT gates, I ask the teams to write the equation for a circuit with embedded NOT gates or the same gate types connected (i.e., an OR gate connected to an OR gate). I use a similar approach for the reverse problem (i.e., drawing the circuit corresponding to an equation). A variation on this second step is to have the teams work on the reverse problem with no instruction or examples from the instructor. After they develop a solution, I show them a systematic approach for getting to the solution.
- When introducing timing diagrams for simple combinational circuits, I present an example involving a 2-input AND gate connected along with one external signal to a 2-input OR gate. Then I ask the teams to repeat the process with a slightly more complicated circuit (e.g., one with a NOT gate between the two gates or with the OR and AND gates interchanged). Usually, I start with idealized gates with no propagation delay and then ask them to indicate how the diagram changes because of the gate delays.

- When introducing min-term and max-term forms, I show examples for converting the Σ or Π form to the expanded form written as an explicit function of the input variables. Then I ask the teams to do the reverse problem (i.e., converting the expanded form into Σ or Π form). Next I ask them to expand an equation containing five or six arbitrary input variables (e.g., *UP*, *RST*, *LD*, *DAT*, and *CLR*) that do not have the "natural ordering" of variable designated by single letters (e.g., *A*, *B*, *C*, …) or subscripted letters (e.g., X_0, X_1, X_2 …). As a third type of exercise, I ask them to determine the number of rows in the complete truth table and subsequently to draw the first few rows in the truth table for this equation.
- When discussing the process for converting verbal description into truth tables, I give a typical example, such as an automobile's seatbelt, key-in, and door-open alarm system. I usually follow this with a 2-bit adder problem (two 2-bit inputs) and ask teams to determine the number of bits needed for the output. I follow this with a series of exercises in which I ask them to consider an *n*-bit adder, multiplier, and incrementor (where *n* is a specific number, such as four) and determine the number of rows in the truth table, the number of bits required for the output, and the output for a few selected inputs.
- In the first class on K-maps, I offer a few straightforward examples and then ask the teams to find the expressions for maps in which a naive user might include an extra term [e.g., the map for F(A, B, C) = Σ m(1, 4, 5, 6)], in which there is more than one optimal solution [e.g., F(A, B, C) = Σ m(1, 3, 4, 6, 7)] or in which the optimal solution includes unusual terms, such as the "four corners" in a 4-variable map.
- When introducing "don't care" conditions, I present an example with some unspecified input conditions (e.g., the BCD-to-7-segment decode). I work with the class to define the output for the specified input conditions and then ask the teams to determine what the outputs should be for one or more of the unspecified cases. I usually ask them to do a second example in which I give them a min-term expression that includes the "don't care" conditions and ask them to find the minimum cost 2-level implementation—a charge that requires them to compare the optimum SOP and POS forms.
- In an introduction to VHDL, in which ENTITY and ARCHITECTURE structures and the simple assignment statement are described, I develop the VHDL for a 3-input-2-output system described by Boolean equations. I then ask the teams to determine the Boolean equations from a VHDL description containing parentheses; next I ask them to write the equations when all the parentheses are removed.

C. Other Gate Types

- When introducing NAND and NOR gates, I draw the symbol, truth table, and equation for one of these gate types and then ask the teams to do the same for the other type.
- When discussing conversion of multilevel AND-OR-NOT diagrams to NAND- or NOR- only circuits, I work an example in which conversion is straightforward because all interconnection involves alternate gates types (i.e., all ANDs connected to ORs and all ORs connected to ANDs). Finally, I ask the teams to convert an AND-OR-NOT in which at least one connection involves the same gate type (i.e., an AND-to-AND connection or an OR-to-OR connection). Next, I might ask them to convert a NAND-only diagram to a NOR-only diagram.
- When introducing the XOR and XNOR gates, I present the symbol, truth table, and equation for the XOR and ask teams for the XNOR versions. I give an example showing the manipulation to obtain an SOP equation from a circuit containing XORs, XNORs, or both; then I ask teams to determine whether two circuits containing these gates are equivalent. I ask them to determine what function an XOR gate performs when one input is treated as a "data signal" and the other is considered a "control signal." The answer is a "controlled inverter" because, when the "control signal" is active, the gate inverts the input data signal and, when the "control signal" is inactive, it simply passes the signal straight through. This function of the XOR is used later in an adder/subtractor circuit in which the complement of one input is added to perform the subtraction.
- When introducing the tri-state gate, I describe the gate and show how it is used to construct a 2-to-1 multiplexer. As an exercise, I show the circuit for a 4-to-1 multiplexer with the inputs designated by arbitrary symbols (i.e., A through G) and then ask the teams to associate these signals with the standard symbols for the 4-to-1 multiplexer circuit (e.g., I 3:0, and S1:0). As a second type of exercise, I ask the teams to write the equations for a circuit containing several tri-sate gates.

D. Number Representation and Arithmetic

- When discussing number representation and various integer codes, I ask the teams to decode a particular binary pattern, perhaps expressed in hex (e.g., decode A4C) without indicating the code. After a short time, many will begin asking what the code is, and this reinforces the idea that a binary pattern can have many interpretations and that a particular integer can have many binary representations. I then have them decode the pattern using several codes. Next I ask them to use a specific number of bits to encode an integer that is out of range with the specified number of bits and let them try to resolve this before talking to them about the relationship between the number of bits and the range of encoded values.
- After showing them how to add two multibit numbers and illustrating that this approach can yield the correct result for integers encoded using the 2s complement code (make sure that there is no overflow in these examples). I then give the teams two additions to do and ask them if the results make sense—one will yield the correct result and the other will not, because of overflow. With this as an introduction, I discuss the overflow issue.
- After discussing the full-adder and drawing gate diagrams for it, I show the class a diagram for a multibit adder constructed with full-adders and ask the teams to compute the time delay for the adder.
- After I discuss the full-adder and the multibit ripple carry adder, I ask the teams to use a multibit adder to build a multibit subtractor. Alternatively, I show them a diagram of a multibit adder with one set of one inputs inverted and an initial carry (C_0) equal to logic 1 and ask them to describe the function of the circuit.

E. Combinational Modules

- When introducing multiplexers, decoders, and so on, I typically start with devices with all asserted high signals and draw the input-output model, construct a functional description table, determine the output pattern for one or more input patterns, and write the equation for one or more output signals. I may ask the teams to indicate the changes in one or more of these representations when size of the device changes. Following this, I might ask them to complete one or more of these four tasks with some of the signals changed to asserted-low signals.
- When discussing approaches for cascading relatively small multiplexers or decoders to create more complex circuits, I show an example using 2-bit devices as a building block to produce 4-bit devices. I then ask the teams to design a 16-bit multiplexer using several 4-bit (or 2-bit) devices. Initially, I may have them speculate on the number of devices needed before they actually start drawing the circuit.
- In discussing VHDL representations of standard combinational modules, I present an example (frequently one from the textbook) and then ask the teams to list all the changes that have to be made for one or more specific modifications (e.g., increase the number of inputs, convert some of the signals to asserted-low signals, adding carry and overflow signals to a multibit adder).

F. Flip-flops

- When talking about the operation of the SR-flip-flop using two interconnected NOR gates, I discuss what happens when R = 1 and S = 0 and then ask the teams to determine what happens when R = 0 and S = 1 and when R = 1 and S = 1. The last condition leads into the reasons for "Not Allowed" input patterns.
- After defining various flip-flop types and clocking options, I ask the teams to complete timing diagrams for several flip-flops and clocking options. After they complete routine diagrams for SR-type and D-type flip-flops, I have them consider the timing diagram for a T-type flip-flop in which the last active transition of the clock coincides with a change in the T-signal. After they struggle with this for a few minutes, I introduce the idea for a protected time window around the clock transitions that depends on the set-up, hold, and other timing parameters of the flip-flop.
- When introducing the VHDL representation of flip-flop circuits, I show the class the representation for the SR flip-flop and ask the teams to indicate the changes in the inner IF-statement needed to convert the device into a JK flip-flop. Since we have used only the IN and OUT mode in VHDL up to this time, the toggle transition presents a problem. Inevitably students' answers will be incorrect because they will not account for the limitation of IN and OUT mode variables. I point out that their solution is incorrect and let them work with it a little longer and then give them a hint to look at the variable declaration in light of their experiences with C++. Next, I introduce the BUFFER mode to represent signals that appear on both the right-hand and left-hand sides of assignment statements.

G. Standard Sequential Modules

- In discussing standard sequential modules (i.e., shift registers and counters), I describe the various types of input signals that are possible with a shift register and show examples of each. I ask the teams to list all the control signals that they can think of for a counter.
- I may give them an initial state and timing diagram showing the clock and input signals and ask the teams to draw the waveforms of the output signals or simply to list the sequence of states for a shift register or a counter module. In a slightly more complicated variation, I might include some other logic with the shift register or counter, such as an AND gate that detects a certain count and resets or loads the counter when it reaches that count.
- After I show them the design of either a parallel-to-serial or a serial-to-parallel converter using several cascaded smaller shift registers, I ask the teams to design the other. Alternatively, I may ask them to indicate the changes in the design when the size of the converter or the shift register is changed or when the ordering of the serial bits is reversed.
- In discussing VHDL representations of standard sequential modules, I present an example (often one from the textbook) and then ask the teams to list all the changes that have to be made for one or more specific modifications (e.g., increase the number of inputs, convert the reset from synchronous to asynchronous, convert from a rising-edge clock to a falling-edge clock, or reverse the priority of two control signals, such as the ENABLE and LOAD signals for a shift register).

H. Sequential Circuits Design and Analysis

- When discussing analysis of sequential circuits, I present an example using standard configurations with AND-OR logic controlling the two flip-flops. I ask the teams to analyze a more complicated circuit from a later chapter in the book (e.g., a shift register with feedback containing five flip-flops and a couple of XOR gates).
- When discussing sequential design, I give them a simple symbolic state diagram (e.g., one with two input signals and four states or one with one input signal and eight states) and ask the teams to determine the minimum number of flip-flops in the circuit. Subsequently, I give them a specific state assignment and ask them to determine the number of rows in the state table and then the entries in the first few rows.
- After completing a design example using a state table with a particular flip-flop type, I might ask the teams to determine what parts of the table change when a different flip-flop type is used and then ask them to determine some of the entries in the table for this type of flip-flop. I might ask them to obtain minimum equations for one or more of these signals and to draw the logic diagram showing the connections between the flip-flops and the gates generating these control signals.
- When discussing the development of a symbolic state table for a finite-state machine implementing various simple sequential circuits (e.g., sequence detectors), I draw the Mealy-machine (or Moore-machine) diagram for a particular problem and ask the teams to draw the other. Before they start drawing, I might ask them to estimate the number of states in the new diagram. Alternatively, I may ask them to indicate how the diagram changes if they are allowed (or not allowed) to reuse bits or switch from a bit stream to a bit packet (or vice versa).

References

1. Springer, L., Stanne, M.E., and Donovan, S.S. (1999). Effects of small-group learning on undergraduates in science, mathematics, engineering, and technology: A meta-analysis. *Rev. Ed. Resch.*, 69:21–51. Available on the World Wide Web at <http://www.wcer.wisc.edu/nise/cl1/cl/resource/scismet.pdf>.
2. Johnson, D.W., Johnson, R.T., and Smith, K.A. (1998). "Cooperative learning returns to college: what evidence is there that it works?" *Change*, Jul/Aug 1998.
3. Terenzini, P.T., Cabrera, A.F., Colbeck, C.L., Parente, J.M., and Bjorklund, S.A. (2001). "Collaborative learning vs. lecture/discussion: students' reported learning gains," *J. Engr. Ed.*, 90:1, 123–130. Available on the World Wide Web at < http://www.asee.org/jee/papers/content.cfm?name=EE007-TERENZINI-123.pdf>.
4. Hake, R.R. (1988). "Interactive-engagement vs. traditional methods: a six-thousand-student survey of mechanics test data for introductory physics courses," *Am. J. Phys*. 66, 64–74. Available on the World Wide Web at <http://www.physics.indiana.edu/~sdi/ajpv3i.pdf>.
5. Wright, J.C., Millar, S.B., Kosciuk, S.A., Penberthy, D.L., Williams, P.H., and Wampold, B.E. (1998). "A novel strategy for assessing the effects of curriculum reform on student competence," *J. Chem. Ed.*, 75:8, 986–992. Available on the World Wide Web at <http://www.pkal.org/documents/Assessment_Wright_et_al.pdf>.
6. Felder, R.M., Felder, G.N., and Dietz, E.J. (1998). "A longitudinal study of engineering student performance and retention. V. comparisons with traditionally-taught students," *J. Engr. Ed.*, 87:4, 469–480.
7. DiBiasio, D., and Groccia, J.E. (1995). "Active and cooperative learning in an introductory chemical engineering course," *Proceedings*, FIE Conf. Available on the World Wide Web at <http://fie.engrng.pitt.edu/fie95/3c2/3c25/3c25.htm>.
8. Hart, F.L., and Groccia, J.E. (1994). "An integrated, cooperative learning oriented freshman civil engineering course: computer analysis in civil engineering," presented at Freshman Year Experience Conf., Columbia, SC. Available on the World Wide Web at <http://cee.wpi.edu/ce_faculty/faculty_pages/hart_pdf_files/COOP%20LEARN.pdf>.
9. Pimmel, R. (2001). "Cooperative learning instructional activities in a capstone design course," *J. Engr. Ed.*, 90:3, 413–421.
10. Brown, S.D., and Vranesic, Z.G. (1999). *Fundamentals of Digital Logic with VHDL Design,* New York: McGraw Hill.
11. Felder, R.M., and Brent, R. (1996). "Navigating the bumpy road to student-centered instruction." *College Teaching,* 44:*2,* 43–47. Available on the World Wide Web at <http://www.ncsu.edu/felder-public/Papers/Resist.html>.
12. Cooper, J.L., MacGregor, J., Smith, K.A., and Robinson, P. (2000). "Implementing small-group instruction: insights form successful practitioners," in Jean MacGregor, James L. Cooper, Karl A. Smith, Pamela Robinson (eds.) *Strategies for Energizing Large Classes: From Small Groups to Learning Communities, New Directions in Teaching and Learning*, vol. 81, San Francisco: Jossey-Bass Pub.
13. Carnevale, A.P., Gainer, L.J., and Meltzer, A.S. (1988). Workplace basics: skills employers want. Washington D.C.: Am. Soc. Training and Devel. and U.S. Dept. Labor, Employment and Training Admin.
14. ABET. Available on the World Wide Web at <http://www.abet.org>.
15. Angelo, T.A., and Cross, K.P. (1993). *Classroom Assessment Techniques: A Handbook for College Teachers, 2d ed.* San Francisco: Jossey-Bass Pub.
16. Forming Student Teams. Available on the World Wide Web at <http://www.foundationcoalition.org/publications/brochures/2002-Mar-01_Forming_Teams.pdf>.
17. Felder, R., and Brent, R. (1994). Cooperative learning in technical courses: procedures, pitfalls, and payoffs. ERIC Doc. Repro. Serv., ED 377038. Available on the World Wide Web at <http://www.ncsu.edu/felder-public/Papers/Coopreport.html>.
18. Felder, R., and Brent, R. (2001). "Effective strategies for cooperative learning." *J. Cooperation & Collaboration in College Teaching*, 10(2), 69–75. Available on the World Wide Web at <http://www.ncsu.edu/felder-public/Papers/CLStrategies(JCCCT).pdf>.
19. Johnson, D.W., Johnson, R.T., and Smith, K.A. (1998). *Active Learning: Cooperation in the College Classroom,* 2d ed. Edina, MN: Interaction Book Co.
20. Millis, B.J., and Cottell Jr., P.G. (1998). *Cooperative Learning for Higher Education Faculty*. Phoenix, AZ: Oryx Press.
21. Jacobson, D., Davis, J., and Licklider, B. (1998). "Ten myths of cooperative learning in engineering education," *Proceedings, FIE Conf.* Available on the World Wide Web at <http://vulcan.ee.iastate.edu/~davis/papers/FIE-TenMyths-1998.pdf>.
22. Active/Cooperative Learning: Best Practices in Engineering Education. Available on the World Wide Web at <http://clte.asu.edu/active/main.htm>.
23. Felder, R., "Active learning and cooperative learning." Available on the World Wide Web at <http://www.ncsu.edu/felder-public/Cooperative_Learning.html>.
24. Smith, K.A., "Getting started with cooperative learning." Available on the World Wide Web at <http://www.wcer.wisc.edu/nise/CL1/CL/story/smithkar/TSKSA.htm>.

***Russ Pimmel** completed this work while on the faculty in the Electrical Engineering Department of the University of Alabama. He has previously held faculty appointments at the University of Missouri Columbia, University of North Carolina Chapel Hill, and Ohio State University; he has worked for Emerson Electric, Battelle Northwest Labs, and McDonnell Douglas. Currently he is a Program Officer in the Division of Undergraduate Education at the National Science Foundation.*

Whether you are just getting started or looking for additional ideas, the Foundation Coalition staff would like to help you incorporate active/cooperative learning into your engineering classes through workshops, Web sites, lesson plans, and reading materials. For suggestions on how to start, see our Web site at

<http://www.foundationcoalition.org> or contact Jeffrey Froyd at froyd@tamu.edu or at 979-845-7574.