Integrating active and cooperative learning strategies to a redesigned Microprocessor based system design course

Alfonso Avila, Juan Manuel Hinojosa
ITESM Campus Monterrey

Abstract

The paper presents the active-cooperative learning strategies incorporated to a redesigned Microprocessor based system design (MbSD) course. The microprocessor based system design (MbSD) area is both the target of technological innovations and a common requirement in many engineering fields. Important technological trends in this area are the availability of powerful and sophisticated tools for design and development, and the continuous improvement on both the design methodologies and the fabrication technology. To be competitive in this area, industry employers are looking for engineers with qualifications such as: self-guided learning, system design skills, team-working skills, development tools skills, and ethical behavior. Educating professionals at universities with these qualifications becomes a key to support the continuous development of the MbSD area. To do this, the authors have formally incorporated active-cooperative learning strategies based on the course *redesign framework* implemented at ITESM Campus Monterrey. This redesign framework provides the basis to adjust the curricula using valuable pedagogical principles such as: "the student is the subject and the object of the learning process" and cooperative learning. The active-learning strategies are dynamic and productive techniques to facilitate the learning process by applying and incorporating knowledge. The learning strategies introduced to the E95-857 course are: (1) learning general concepts of programming and interfacing microprocessors before learning the disposition of a specific processor, (2) learning to program and interface a specific processor through the "case of study" technique, (3) programming and simulating microprocessor based systems, and (4) building and simulating actual microprocessor based systems. The discussion of the active and cooperative learning strategies includes a description of how each of the strategies meets the redesign framework principles as well as their expected effect on educating professionals with the required qualifications to continue the development of the MbSD area.

1. Introduction

This paper describes the cooperative and active learning strategies incorporated to the microprocessor system design course, which is offered to third-year undergraduate students at ITESM Campus Monterrey. This course is specifically designed to serve engineering students enrolled in the Electronics and Communications program and has two digital systems courses as prerequisites and is the prerequisite for the Digital Systems Design Laboratory. The active learning term defines a class environment where the students act as an active learners rather than passive receptors[2]. In such an environment, the students build their own knowledge by criticizing, debating, and analyzing the course material. Active learning attempts to complement the traditional class environment where the student usually performs as a passive receptor.

Listening to the professor lecture and taking class notes are the dominant activities of the traditional class environment. Cooperative learning is by nature a part of the active learning environment. The Cooperative learning term defines the dynamic process where the students working in teams accomplish a common goal under conditions such as: positive interdependence, individual accountability, adequate use of collaborative skills, and continuous evaluation of the team performance[1]. Cooperative learning is not simply a synonym for students working in-groups; a learning activity only qualifies as cooperative learning to the extent that the listed conditions are satisfied.

2. The redesign framework at ITESM

ITESM Campus Monterrey is currently reviewing the curricula and the learning methodologies to adjust to the changes and challenges triggered by the continuous technological development. Important consequences of the technological innovation are the exposure of the new generations to large amounts of information and the need of professionals with competitive skills, attitudes, and knowledge to satisfy the new industrial and social requirement [2].

The redesign framework at ITESM incorporates active and cooperative learning methodologies and uses Lotus Notes as a technological platform. The redesign framework pursues the following eleven essential characteristics[3,9].

1. Develop the self-guided learning skills
2. Search for a deep and wide learning of knowledge
3. Develop in an intentional and a programmed approach of the skills required to generate new knowledge and to know how to apply them to reality (life-long learners)
4. Emphasize the knowledge of the reality of the Country and the World
5. Promote the use of attitudes and values required to aim for the development of the community and the country
6. Use a wide variety of didactical processes
7. Incorporate cooperative learning activities
8. Change the professor role from lecturer to learning-guide
9. Promote human interaction through technology
10. Have the students evaluate their own knowledge
11. Review the activities performed in the classroom

Lotus Notes supports the learning process of the students by using five databases: schedule, media center, course room, profiles, and assessment[10]. The schedule database keeps a calendar with the activities to be performed by the students during the sixteen-week period. The schedule database also keeps the general information of the course such as the contents, the prerequisites, the educational intentions, the course objectives, and the evaluation system. The course room database provides an interactive environment that allows the students to participate in debates among them and the professor. The course room database also presents a space for individual homework and teamwork. The media center database keeps the materials prepared by the professor as well as other support materials such as simulators, specification sheets, solutions sets, etc.

The profiles database groups the personal information and photographs of the students and the professors. The profile database helps define a better work environment. The last database, the assessment manager, allows the evaluation of the student and professor achievement. This database contains exams, quizzes, self-evaluations, professor evaluation, team evaluation, and surveys.

3. The Learning strategies

The learning strategies[4] are based upon the educational intentions and the general objectives defined for the Microprocessor Systems Design course. The educational intentions are:

1. Promote honesty and responsibility
2. Promote cooperative learning
3. Bound the knowledge with the personal experience using technological tools
4. Develop the critical reasoning and the creative solution of problems
5. Encourage the compromise of acting as change agent
6. Develop skills for fault detection and correction
7. Develop skills for managing technological tools
8. Develop skills for searching and selecting information

The following general objectives were defined according to the educational intentions. At the end of the course, the students will be capable of:

1. Learn how to program microprocessors, either individually or in teams, with different architectures based on the knowledge of the generic elements of a microprocessor
2. Design optimal systems based on microprocessors, in groups, considering the components cost, the system complexity and the fulfillment of the minimum requirements
3. Value the impact of technological tools such as Mentor Graphics, the SmartModels and processor simulators on the learning process and the design of optimal microprocessor based systems
4. Value the impact of the microprocessor fabrication and utilization on the national and international society as well as in the environment based upon personal experiences and the new acquired knowledge.

The active and cooperative learning strategies, introduced to the microprocessor systems design courses during the fall semester 1998, are also defined according to the general objectives of the course. These learning strategies are:

1. Learning general concepts of programming and interfacing microprocessors before learning the disposition of a specific processor
2. Learning to program and interface a specific processor through the “case of study” technique
3. Cooperative programming of microprocessor based systems
4. Building and simulating actual microprocessor based systems

3.1 Learning general concepts of programming and interfacing microprocessors before learning the disposition of a specific processor

The goal of the first strategy is to use cooperative learning to help the students learn the general concepts and procedures common to any processor before learning the implementation details of commercial devices. The students worked in teams and performed three activities during three one-hour sessions. The session structure for these activities required the students to evenly distribute the session work, to obtain a final answer by team, present to the class their answer, and evaluate the other teams. The professor's role was to guide the session, evaluate each team's final answer and complement the solutions presented by the students. This strategy allowed the students to create a base knowledge useful for both inferring the operation of different processors and for understanding new concepts and procedures. The first strategy was practiced in the following topics: processor based systems definition, instruction set, addressing modes, assembly language programming, and peripheral interface programming. The following example illustrates the use of the first strategy for the addressing modes concept. Table 2.1 shows a general definition of 12 common addressing modes considering the addressing mode name, the instruction format, instruction operation, elements of the operator and the effective address calculation The students activities were to identify the differences among the 12 addressing modes by focusing on the following aspects:

1. the data associated to the source operand is in either a memory location or a register
2. the calculation of effective addresses or target addresses
3. the source operator format and its elements
4. the number of instructions needed to code a matrix addition program

4.1 Learning to program and interface a specific processor through the "case of study" technique

The second strategy employed the case of study technique to enforce the previously acquired knowledge and to get knew knowledge. The two processors under study were the 68000 and 8051 processors. The strategy consisted of three activities developed in three one-hour sessions. Each group of activities met the following structure: (1) assimilate the problem, (2) get new knowledge to solve the problem, (3) solve the problem, 4) evaluate solutions. To assimilate the problem the students had to determine whether their previous knowledge was enough or not. Then, the students had to search for more knowledge to propose a problem solution. The students, finally, solved the problem after getting enough new knowledge. The professor's role for these activities was to guide the session and evaluate solutions. The second strategy was applied in the following topics: instruction set, addressing modes, assembly language programming, and peripheral interface programming. The following example illustrates the second strategy. The professor asked the students to write an instruction sequence to do a memory reference using the 8051 and 68000 addressing modes. The students had to learn by themselves the addressing modes of the 68000 and 8051 by comparing them with the 12 common addressing modes of the first strategy. The students determined the similarities and differences among addressing modes of the generic (common), 8051 and 68000 processors. Also, the students created another table with the discovered addressing modes and the similar addressing modes for all the processors under study.

The students were finally able to write the instructions sequences to solve the problem and to do an evaluation of the solutions based on the number of instructions necessary to perform the memory reference.

Addressing Mode	Format	Operations	Source operator elements	Effective Address (EA) Calculation
Register	Mov R4,R3	R4<-R3	1 Register, R3	NA
Immediate	Mov R4,#3	R4<-3	1 Constant, #3	NA
Direct	Mov R4,200	R4<-Mem[200]	1 Memory address	EA = 200
Indirect with register	Mov R4,(R1)	R4<-Mem[R1]	1 Base register, R1 2 Parenthesis	EA = R1
Indirect with displacement	Mov R4,20(R1)	R4<-Mem[R1+20]	2 Parenthesis 1 Base register, R1 1 Displacement, 20	EA = 200 + R1
Indirect with postincrement	Mov R4,(R1)+	R4<-Mem[R1] R1<-R1+DataSize	1 Base register, R1 2 Parenthesis 1 plus sign	EA = R1
Indirect with predecrement	Mov R4,-(R1)	R4<-Mem[R1] R1<-R1-DataSize	1 Base register, R1 2 Parenthesis 1 minus sign	EA = R1
Indexed	Mov R4,(R1,R2)	R4<-Mem[R1 + R2]	1 Base register, R1 1 Index register, R2 2 Parenthesis	EA = R1+R2
Indexed with displacement	Mov R4,20(R1,R2)	R4<-Mem[R1 + R2 + 20]	1 Base register, R1 1 Index register, R2 1 Displacement, 20 2 Parenthesis	EA = R1+R2
Memory Indirect	Mov R4,@(R3)	R4<-Mem[Mem[R3]]	1 Base register, R3 2 Parenthesis 1 @ sign	EA1 = R3 EA2 = Mem[R3]
Absolute	Jmp 200	PC <- 200	1 Target address	NA
Relative	Bra 200	PC <- PC + 200	1 Displacement	NA

Table 2.1 General definition of 12 common addressing modes

4.2 Cooperative programming of microprocessor based systems

The third strategy introduced technological tools to support the student's learning process. This strategy involves active and cooperative learning not only to acquire knowledge but also to develop skills and attitudes. The strategy focused on developing small programs in two class sessions and large programs as homework. The structure of these activities was as follows: 1) identify the necessary tasks to solve the problem, 2) assign tasks to each team member, 3) simulate the individual tasks, 4) simulate the complete system. The professor's role was to prepare and guide the sessions. The Course-Room database (Lotus Notes/Learning Space) allowed the professor to support and monitor the student's progress. The students used software simulators of the 68000 and 8051 to practice the writing of programs, to debug syntax errors, to debug programs. The simulator selected for 68000 was EMU68k[5] and the simulator for the 8051 was PROVIEW32[6].

The activities assigned to the students focused on the following topics: introduction to the simulators and the assemblers, addressing modes, stack and subroutines, assembly language programming, and exception processing.

4.3 Building and simulating actual microprocessor based systems

The fourth strategy focused on motivating the students and supporting their learning process by simulating and building actual microprocessor based systems. The material of these activities is naturally suitable for active and cooperative learning. The activities included the design hardware and software prototypes using technological tools and commercial components. The activities had the following structure: 1) consult in teams all the resources (specification sheets, procedures and design methodologies) available to solve the problem, 2) break the system in subsystems, 3) put the complete system together once the individual subsystems were operating correctly. The professor's role was to guide, facilitate and evaluate the students work. The students designed and simulated a microprocessor-based system using the SmartModels[7] from Synopsys running under the Mentor Graphics CAE tool. The SmartModels are an extensive library of commercial components designed to meet manufacturer specifications. The SmartModel components are capable of displaying error messages during simulation when their specifications are ignored. The students designed and simulated a development system based on the 68020 microprocessor. The students performed simulations to verify: the operating frequency of the oscillator, the operation of the memory decoder, the instruction execution of a simple program, the exception processing sequence of an auto-vectored level-5 interrupt, and the operation of the MC68230 parallel ports (A and B). The components of the development system were:

1. A fully functional 32-bit MC68020 microprocessor
2. an oscillator
3. a ROM constructed with 4 AMD IC's
4. a supervisor RAM constructed with 4 Motorola IC's
5. a user RAM constructed with 4 Motorola IC's
6. a memory decoder constructed with TTL-AS gates
7. a MC68230 timer and parallel interface

The students used the 8051 microcontroller and commercial components to build the prototype of a microcontroller based system. The 8051 was the selected processor for this strategy due to its simplicity. The reduced number of the 8051's terminals allows the designers to build a complete system in a single breadboard. A typical 8051 minimal system contains only two components: an 8051 and a ROM external chip[8]. During the semester, the students built a minimal system following three scheduled activities. The students first interfaced the microcontroller with ROM and verified the operation of the system by running a small program burned in ROM. The second activity focused on manipulating the microcontroller ports using simple programs such as: implementing logic functions, generating square waves with a specified period, code conversion, etc. The goal for the last activity is to implement a controller (state machine) using the minimal microcontroller based system. The selected projects were simple and related to previous semester topics. For instance, many students implemented semaphores, electronic locks, and fluid level controllers.

4. Assessment

This section presents the evaluation system for the course and the most significant results obtained from the fall semester 1998. Table 4.1 describes the different types of evaluations incorporated to the redesigned course. The diagnostic evaluation allows the instructor to determine whether the students are entering the course with deficient knowledge that needs to be reviewed before starting to cover the course material. The learning process evaluation provides feedback to the instructor about the effectiveness of the class activities. The self-evaluation and the co-evaluation have the purpose to provide feedback to the students about the development of their attitudes, skills and values.

Evaluation type	Description
Diagnostic	Evaluate the previous knowledge of the students
Learning process	Evaluate the learning process of the student during the semester
Self-evaluation	The student evaluates his own knowledge, attitudes, values and skills
Co-evaluation	The student evaluates the knowledge, attitudes, values and skills of his classmates
Accreditation	Evaluates the student performance in the course

Table 4.1 Description of evaluation types

The accreditation evaluation, also known as the traditional evaluation consists of the aspects such as oral presentations, information search, quizzes, self-study assignments, problems and questionnaires, partial exam (3) and a final exam. The grading scale is defined from 0 to 100 and the minimum grade for accreditation is 70. Results of the learning-process-type evaluations, applied to the students during the semester, provided to the instructors a valuable feedback about the learning strategies[4]. Approximately 70% of the student opinions indicated that the combination of the first two learning strategies facilitated the learning of the material. Also, the 72% of the students believed that the technological tools help them get a deeper understanding of concepts and procedures. Almost a 93% of students admitted that building a microcontroller based system help them understand better the system design procedures and develop the skill to do design under manufacturer specifications. Also, the 70% of the students consider the use of the SmartModels and Mentor Graphics a good way to gain a deeper understanding of the concepts associated to: instruction execution, interrupts, and peripheral interface.

5. Discussion

This section discusses the expected effectiveness of the strategies described in the previous section and the observed results from the beginning to the end of the semester. The effectiveness of the strategies is measured in terms of how close are the observed results to the general objectives and educational intentions defined for the present course. As mentioned earlier, the active and cooperative learning methodologies are the important components of both the educational intentions and the general objectives of the course.

The students experienced confusion with the combination of the first and second strategies due to processor's particularities such as the order of the source and destination operands, the nomenclature for specifying the addressing modes and the nomenclature for specifying the operation of an instruction. This situation forced the students to develop more skills for searching and interpreting instruction specifications rather than memorizing them. The base knowledge acquired with the first learning strategy worked as a useful pattern for interpreting instruction specifications, addressing mode specifications and peripheral programming procedures. With the combinations of the first two strategies, the students not only developed skills for searching and interpreting information but also they developed a deeper understanding of the selected topics. At the end of the semester, the students were aware of the limitations of both instruction (68000 and 8051) sets and were able to decide whether an instruction set was efficient and easy to use for a given application or not. Also, the study of the 68000 and the 8051 through the combination of the first two strategies allowed the students to clearly differentiate a microprocessor from a microcontroller. Thus, the first and second strategies conformed to the first general objective of the course. Also the combination of the first and second strategies relied on the active learning methodology allowing the students to build their own knowledge by consulting information rather than receiving it from the professor.

The third and fourth strategies consolidated the fourth general objectives of the microprocessor systems design course. The students relied on the software simulators to accomplish their programming assignments and to develop skills for detecting and correcting syntax and programming errors. Simulators were also useful to the students for understanding the operation of both individual instructions and complex instruction sequences. The students also developed teamwork skills when they designed the software prototypes of the development system based on the 68020. Each team member was in charge of a portion of the design and had to interact with the rest of the team to define the extent of his/her task. The team members had also to discuss with the rest of the team and use critical thinking to identify and correct design errors. Some students also experienced hard time when their simulations did not function; however, the hard time became a source of satisfaction at the end of the semester when the designs were completed. Thus, many students learnt that putting together carelessly designed subsystems created a "chaotic" system difficult to correct. The students took advantage of course room database of Lotus Notes to help each other with design problems. A team with a problem had to send a message for both the professor and the teams explaining the details and asking for help. When no team replied, the professor solved the problem and made the solution available to the rest of the teams. The students had similar experiences and developed similar skills when they designed the hardware prototype of the microcontroller based system using the 8051. Both the third and fourth strategies relied on the cooperative learning methodology to allow the students to complete their designs. Designing the hardware prototype required of an active roll of the students, but less cooperative effort compared to the effort required for the software prototype. Since the complexity of the designs defined the amount of cooperative effort. The hardware prototype was lower in complexity compared to the software prototype.

6. Conclusion and Future improvements

The results obtained in this first attempt to incorporate active and cooperative learning strategies to the Microprocessor System Design course improved the learning process of the students. The students gained deeper understanding of concepts such as addressing modes, instruction execution and instruction operations by using the combination of the first two strategies. The use of software simulators and Mentor Graphics effectively supported the learning process of the students by allowing them to bound theoretical concepts with the experience of designing a development system. Designing a software prototype of a development system also allowed the students to develop better teamwork skills. Designing a hardware prototype of a development system allowed the students to develop skills for searching information, selecting information, designing under specifications. The design of both the software and hardware development systems required from the student critical reasoning and ability to detect design errors or defective components. The interaction of the professor with the students also suffered changes. The professor spent less time lecturing and more time facilitating and guiding the students. The professor also interacted more with the students through the Lotus Notes and the course evaluations.

The time spent by the students on the coursework exceeded the originally scheduled time. The reasons for spending additional time were the lack of teamwork experience of the students and the lack of experience of the students when using the technological tools. Fortunately, these problems are will be resolved by giving additional support to the students to use the technological tools and by teaching them how to do effective teamwork. The following improvements are being considered for the spring semester 1999:

1 Build fifteen development systems for each of the two processors selected for the course to prepare the exercises focused on interrupts and peripheral interfacing
2 Prepare additional programming assignments focused on cooperative student interaction
3 Incorporate more activities for discussion and debating in the course room data base of lotus notes

7. ACKNOWLEDGMENTS

The authors would like to thank Lotus Development Corp, Cambridge, MA., Synopsys Corp, Mountain View, CA., and Mentor Graphics Corp, Portland, OR., for the software support.

Bibliography

1. D. W. Johnson, R. T. Johnson, and K. A. Smith, Active Learning: Cooperation in the College Classroom. Edina, MN: Interaction Book, 1991.
2. D. R. Woods, Problem-Based Learning: How to Gain the Most from PBL. Waterdown, ON: D. R. Woods, 1994.
3. Moreira, H. Hacia un nuevo modelo del proceso enseñanza-aprendizaje basado en la misión del Tecnologico de Monterrey para el año 2005, sistema ITESM, Abril de 1998.
4. Avila-Ortega, A., Informe the implantacion del rediseño del curso E95-857, ITESM Campus Monterrey, Diciembre de 1998.

5. Antonakos, James L. The 68000 microprocessor: Hardware and Software, Principles and Applications. Upper Saddle River, New Jersey: Prentice Hall, 1996
6. _______, Proview Reference, Franklin Software Inc., http://www.fsinc.com/reference, 1998
7. _______, SmartModel Library: Reference Manual, Synopsys, Inc., 1998.
8. MacKenzie, I. Scott. The 8051 Microcontroller. Upper Saddle River, NJ: Prentice Hall, 1995
9. _______, Distributes learning: Approaches, Technologies and Solutions, White Paper, The Lotus Institute, August 1996.
10. ______, Plataforma Tecnologica Learning Space para el rediseño de cursos, Sistema ITESM, Direccion de Desarrollo Académico, Monterrey, N. L., México, 1998.

ALFONSO AVILA
Alfonso Avila is a Professor of the Electrical Engineering Department at ITESM Campus Monterrey. He received a BS. degree in Electrical Engineering from ITESM Campus Monterrey, Mexico in 1989 and a Ph.D. from the Department of Electrical Engineering from University of Arkansas at Fayetteville in 1997. His interest reside in the area of High Performance Processor Design, Microprocessor based System Design, and the development of activities to promote active and cooperative learning in the Digital Systems curricula.

JUAN MANUEL HINOJOSA
Juan Manuel Hinojosa is a Professor of the Electrical Engineering Department at ITESM Campus Monterrey. He received his B.S. and M.S. degrees in Electrical Engineering from ITESM Campus Monterrey in 1982 and 1984. His interest reside in the area of Microprocessor based System Design and Digital Communications.