Transforming a Microprocessors Course through the Progressive Learning Platform

Sohum Sohoni, David Fritz, Wira Mulia
Oklahoma State University

Abstract

This paper describes an innovative learning platform called the Progressive Learning Platform (PLP), and its use in an introductory microprocessors course. The discussion covers the overall transformation of the course from the examination and modification of existing course objectives or Be-Able-To's (BATS), to the development of laboratories and other curricular materials for a highly collaborative active-learning approach. Decisions made by the instructors during this process, and the reasons behind those decisions are discussed to provide guidance to those wishing to revamp their 'Microprocessors' courses to follow this model. Our modified course will be first offered in the Fall 2011 semester. The paper also presents metrics and methods that will be used to measure the effects that PLP has on student learning and student development.

1. Introduction

The Progressive Learning Platform (PLP)[1] is a system designed to facilitate computer engineering education while decreasing the overhead costs and learning curve associated with existing solutions. The PLP system is a System on a Chip design with accompanying tools reflecting a contemporary CPU architecture. It is unique in that it can be used in a number of courses (Digital Logic Design, Microcomputer Principles, Computer Architecture, Compilers, Embedded Systems) as students progress through a Computer Engineering curriculum. The system consists of a fully pipelined, MIPS-like processor with surrounding support hardware. The support hardware includes a programmable interrupt controller, VGA controller and framebuffer, UART, memory controller, simple cache, timer, and GPIO hardware. All components are written in Verilog HDL, are open-source, and are freely available. To support the hardware components, a unified assembler, cycle accurate simulator, and board interface software package is included. The software is written in Java, works on Linux, Windows, and Mac OS, is open-source, and is freely available from the project website[1].

With only a brief learning curve on the PLP system, students can work on course objectives immediately. The system and accompanying curriculum emphasize inter- and intra- team collaborative learning by dividing components of the design process used in lab to individual teams. The goal is to expose students to a less controlled environment representative of real-world design practice. Student teams are responsible for the design decisions of their assigned component, as well as ensuring that components are compatible for use in the larger, class-wide system. Other highlights of the PLP system are: a 'hands-on' experience with real hardware early in the computer engineering curriculum, low overall cost for students and institutions, and cross-

course application of concepts. The latter is of great importance since students often fail to see how concepts learned in one course apply to another.

We assert that using a single, unified system throughout these courses provides an invaluable and cohesive framework that students can use to transfer knowledge and skills from one course to the next. With an overarching system like PLP, where different aspects of it are taught and used in a variety of courses, students can make direct connections and see how concepts in computing are related. Details of PLP are available online[1] and are discussed in other publications[2, 3].

In this paper we present a major revision of ENSC 3213, Computer Based Systems (henceforth referred to as CBS), a course that covers microprocessor theory and applications. The revision focuses on hands-on active and collaborative learning developed on PLP.

CBS has been traditionally taught as a 3 credit-hour course with two hours of lecture and one 2-hour lab per week. It is an important course for our department, and has been carefully developed over the years to include several interesting laboratory assignments, paired-programming, teamwork, and in-class activities. The last revision of the course was in 2008 when it was transformed from being an ECEN course required for all Electrical and Computer Engineering students to and Engineering Science (ENSC) course recommended for all engineering majors. Although, an ENSC course, it is still taken mostly only by ECE students. The course has about 20-40 students each semester, with an even distribution of second semester sophomores and first semester juniors. One of the goals of the course revision discussed in this paper is to incorporate skills and topics important to non-ECE engineering students. These include the ability to interface with peripherals for data acquisition and processing, and to control various peripheral devices.

Students have generally found the course to be interesting and worthwhile (based on student feedback over several semesters). To give the reader a better idea of what the course entails, we have listed below some questions that drive the content of the course. Course objectives are listed in section 3.

a. What occurs in the CPU during the execution of an instruction?
b. How does an embedded system differ from a general purpose computer?
c. How is data represented in a CPU and what are the basic operations that are performed on data in a CPU?
d. What are the steps to creating a working program?
e. What are the different parts of a program and why are they important?
f. What are different addressing modes, how do they differ from one another and why do we need more than one addressing mode?
g. What are some things to watch out for when storing data in a finite number of bits- i.e. do we lose precision and how do we deal with that?
h. How do we communicate with the devices that are connected to the CPU?
i. What are some of the important things to consider when communicating with other devices or with data from the point of view of time and space required for storage?
j. Why do we need subroutines and how is the stack used with subroutines?
k. Why would we use interrupts and how do we implement them into a program?

Leading up to 2011, the textbook used was Jonathan Valvano's "Introduction to Embedded Microcomputer Systems: Motorola 6811/6812 Simulations[4]". We made extensive use of the TExaS[5] simulator provided with the book for programming the 6811. The course was extensively revised since 2005, and included extensive laboratory assignments which tried to relate to the real world. For example, there was a 3-lab series where students programmed the operation of a cruise control system found in most modern automobiles. Students started with a barebones system and implemented their algorithm using simulated switches and LEDs through TExaS. Next, they interfaced with the simulated keypad to get user inputs for speed settings and the (simulated) 7-segment display to show the current and desired speeds. Finally, they connected to the DC-motor (again simulated via TExaS) to control its speed as if it were the car's engine, and built a more realistic system with more features such as suspending the cruise control and resuming the last set speed.

While the course has been popular, and the labs have motivated the students to explore and learn the features of the 6811, there have been consistent complaints regarding the lack of real hardware. From the instructor's perspective, although the simulation environment is helpful, it presents a steep learning curve for the students, does not translate easily to real systems, and above all, does not lend itself to class-wide collaboration or large teams. Research shows that students learn best from experience gained programming real systems[6]. Without this, students have difficulty relating concepts to real-world systems. These issues, along with the advantages of PLP stated earlier have prompted us to revise the course around the new platform.

Section 2 describes simulators and platforms similar the PLP system, and presents the unique features of PLP. Section 3 describes the approach taken by the instructors for transforming Engineering Science 3213, Computer Based Systems. Section 4 discusses how the course serves as a platform for educational research, and what studies will be set up to gauge the effectiveness of PLP. Section 5 presents our conclusions, ongoing work, and future plans. The Appendix provides some documents that have resulted from the transformation of CBS.

2. Related Work

Many universities use simulators to teach Computer Engineering concepts to students. Some simulators feature visual representation of the hardware to better convey the systems being studied. Examples of this include WebMIPS[7], RaVi[8] and MipsIt[9]. Other simulators such as MARS[10], SPIM[11] and TExaS[5] provide an integrated development environment and debugging features for students to develop programs for the target hardware. These systems have much less emphasis on the inner working of the processor. Hades[12] is a Java-based logic simulator with an extensive library of logic components and a powerful visualization of the circuit simulation. Lastly, LC-3[13] is an Instruction Set Architecture (ISA) with an assembler and simulator suite that students may use in learning Computer Architecture.

What differentiates PLP from the systems described above is the tight integration between the software tool and the hardware implementation, and how PLP integrates with multiple Computer Engineering courses. PLP is also an open source and community-developed project, licensed under GPLv3, and can be easily obtained from its website[1].

There exist several FPGA based System on a Chip designs. Holland et al.[14] suggest a reduced MIPS instruction set design for use in the classroom. The design uses an 8 instruction variant of MIPS, and allows full observability and controllability through a host tool. It, however, does not provide a host-independent product and, thus, requires a host tool for running the processor in the classroom (for tasks that reach beyond programming). The PLP system provides a rich set of I/O, requiring the host tool for nothing more than initial programming. Additionally, the PLP MIPS design is more robust, while still simple enough for design and implementation in the classroom. Nagaonkar and Manwaring[15] discuss a complete FPGA and micro-controller based SoC for use in research and academia. Their design uses a very flexible custom hardware design. While its use in the classroom is mentioned, no explicit educational use is defined. The PLP system is specifically designed for use in the classroom, with special emphasis on exposing students to critical foundational components of Computer Engineering curriculum.

3. Transforming CBS: The Approach

While the questions listed in Section 1 are the driving force behind course content, the pedagogy and delivery of the course is driven by research on how people learn. Figure 1 shows a pictorial view of the overall process. The approach was guided by the instructor's experience, various national and regional workshops on effective teaching including the National Effective Teaching Institute (NETI)[16] and OSU's Institute for Teaching and Learning Expertise (ITLE)[17]. It was also driven by inputs from faculty in the college of education who are co-PI's on the grant that funds the development of PLP and from the TAs who are participating in a 12 credit-hour university faculty preparation certificate[18]. The PLP system and associated curriculum are based upon a particular set of teaching philosophies that include social constructivism[19] and cooperative learning[20]. The PLP system facilitates a collaborative effort by a significant number of students for design and implementation. This is facilitated by the use of a course Wiki, which is also used as a primary assessment tool, code management software, issue tracker, and special team assignments.

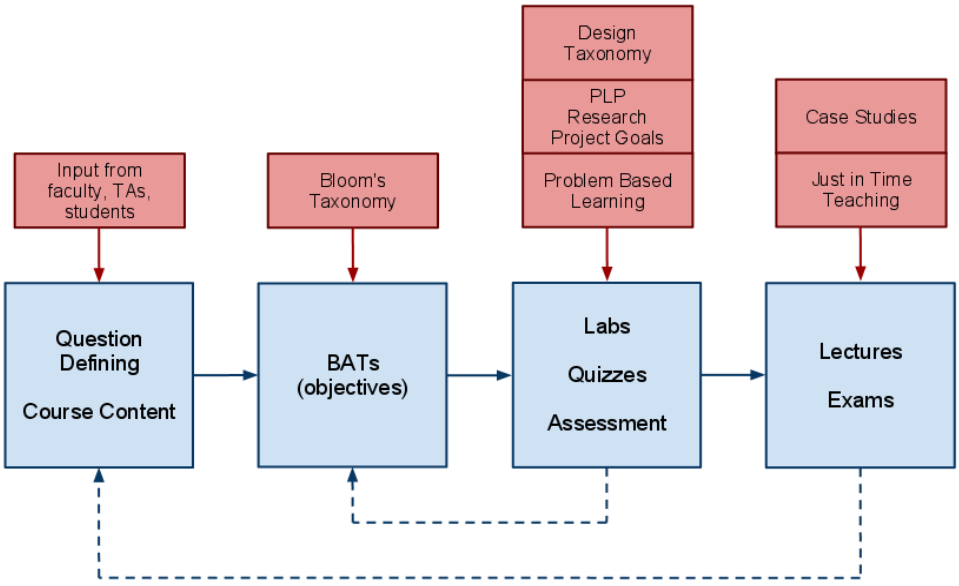

Figure 1: Course revision approach. This was an iterative process that started with questions that defined the course content, and was shaped by external inputs and known best practices

With these goals in mind, we designed the following set of course objectives:

1. List the main parts of a microprocessor
2. Draw a block diagram of CPU internals, and label each part
3. Process data presented in various representations
4. Differentiate between general purpose microprocessors and embedded processors w.r.t to limitations, styles of programming and resources
5. Explain the effect of storing all information in the form of a finite number of bits (quantization)
6. Design an algorithm to solve a given problem
7. Write a correct program in assembly language from a flowchart or algorithm
8. Debug their programs (or a program given to them) and fix the errors
9. Use their knowledge of advanced concepts such as stacks, subroutines, I/O and interrupts in order to write programs that accomplish complex tasks
10. Work with sensors to acquire and process data
11. Evaluate different algorithms to solve the same problem and explain why one is better than the other
12. Work together in teams (divide tasks, co-operate, provide feedback)
13. Transfer their knowledge from the PLP simulation environment to a real microprocessor development board
14. Evaluate the contributions of their teammates in a constructive and professional manner, for example, using an instructor mediated peer review.
15. Effectively communicate to a technical and non-technical audience the results of lab projects through oral, written, or visual media.

3.1 Laboratory Assignments and Final Project

Once we had the course objectives mapped out, we worked on how to assess them through laboratory assignments, quizzes and exams. The most important component of the course and hence of the assessment was found to be the labs, since socio-constructivism (on which PLP is abased) argues that knowledge and meaning is constructed in the context of one's environment. We designed the first lab, 'Lab 0' to be an introduction to the PLP programming environment and board. The next four labs were designed to cover BATS 3, 5, 6, 7, and 8. Each lab is a week-long pair-programmed[21, 22] lab, with a goal of not only covering a particular BAT, but to get students familiar and comfortable with PLP. This prepares them for the course project which is described below in detail.

Students are initially grouped into self-selected pairs. For the first 4 lab assignments, students work in these pairs, without much collaboration across different pairs. For the final project, teams of 4-5 students come together to work on a class-wide design project. The collaborative effort is shown in Figure 2. Emphasis is placed on inter- and intra- team collaboration. Communication and documentation officially takes place on the student driven course Wiki. Communication is facilitated by a student project manager, elected by the students. The project manager has final administrative rights over all design and implementation decisions. Moving

the control of these components to the project manager enables the instructor to further assume the role of facilitator and assures both real and perceived authenticity of learning (that which mimics the real world) for students [23].

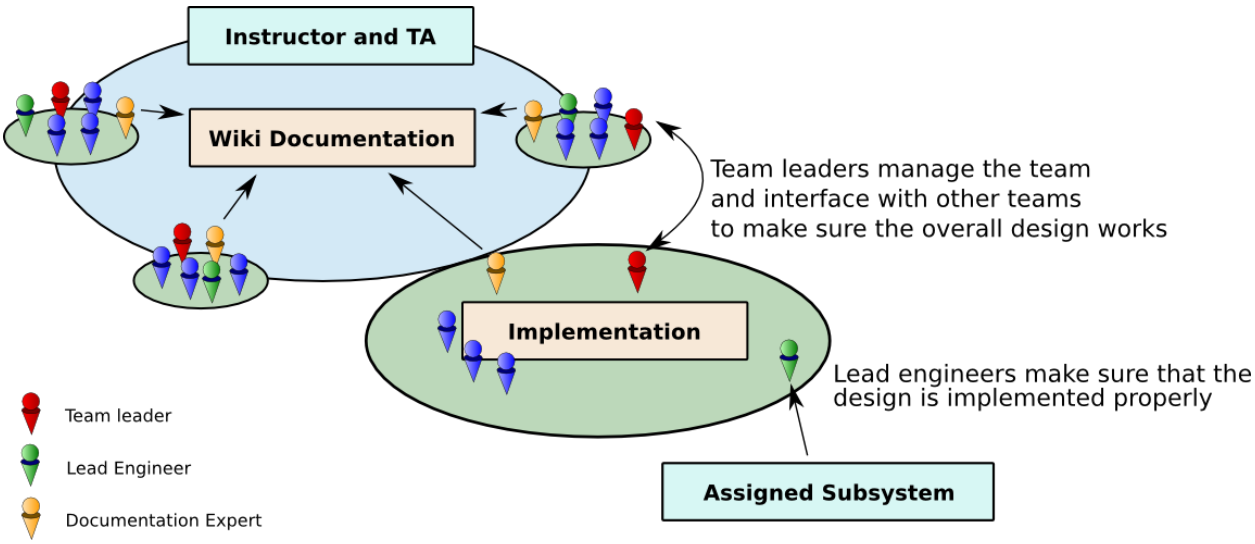

Figure 2: Teams in ENSC 3213 (CBS). Figure shows how PLP will be used in ENSC 3212 and how the class is organized around the collaborative project.

Each team has a team leader, documentation expert, and lead engineer. All team members must have at least one distinct role. Team leaders meet regularly to ensure that proper communication of design efforts. The project manager's role is made explicit, and he or she serves as the final decision maker on conflicts in design decisions. This provides a unique opportunity to extend the authenticity of student learning to address the interpersonal communication challenges of working in a tem, which is a critical real-world skill that is too often not addressed in engineering education.

The project, which lasts for most of the semester, is split into three phases: research, implementation, and integration. In this project, pairs of teams collaboratively work to complete a remote controlled robot project. The goal of the project is to program a robot that is outfitted with the PLP system to navigate an obstacle course manually and autonomously (see appendix for project description).

The research phase lasts for approximately two weeks and teams learn in great detail the aspects of their part of the overall design. Teams are asked to research information relevant to their part of the design, create block diagrams, fully define signals and protocols that impact the other team, and document all of their work on the course Wiki. At the end of the research phase, teams deliver formal presentations of their findings. Other students, as well as an assessment board made up of the course instructor, other knowledgeable instructors, and some graduate students, are also present for the presentation. The assessment board is responsible for assessing the team on the effectiveness and clarity of communication of their part of the design, as well as their understanding of the overall design. Assessment follows a rubric that is provided and explained

to the students in advance. Other students are encouraged to ask questions as well, particularly about how that team's design and design decisions impact their own design.

The implementation phase is the longest phase of the design; within this phase, students implement their designs from the research phase using the PLP system. Communication is also critical in this phase, as changes can have significant impact on the work of other teams. All teams provide up-to-the minute documentation on the course Wiki, which enables the teams to use and build upon each other's work. All implementations must meet specification in order to move on to the integration phase.

The integration phase is the final phase of the project. In this phase, students complete necessary integration of their overall designs, create a project video, finalize all documentation, and create a demonstration for the end-of-semester College of Engineering Design Day where students demonstrate their semester projects in the hallways of the Engineering College.

3.2 Student Assessment

Assessment practices in the course are based on the ability of students to effectively communicate their understanding of the design and its implementation. This is accomplished through four major communication metrics:

1. Documentation of all work on a publicly accessible Wiki,
2. In-class demonstrations of the outcomes of each phase,
3. An end-of-term video detailing the course project, and
4. An end-of-term, high quality demonstration for the College of Engineering Design Day.

Our revised course emphasizes the labs and course project, which carry a combined 50% of the course grade. Quizzes are used to monitor ongoing learning (formative assessment), and carry 20% of the course grade. A mid-term exam and a final carry 10% weight each. Additionally, there are many in-class assignments for bonus points, an in-lab journal and a final reflective writing assignment that carry 20% of the grade. The course has a built-in 10% bonus, so the points add up to 110%.

The primary assessment metric for the course project is the course Wiki. A Wiki is a website that is driven by a powerful and simple markup language and is intended for rapid development of deeply connected content. Due to the rapid development of content and ease of use, Wiki software is used in numerous contexts including project development portals, documentation efforts, and in education. Wiki software facilitates collaborative development, as anyone with access to the Wiki can edit it. This allows for information to develop in an evolutionary way from multiple users. Side discussions about the development of particular Wiki articles often develop as students work to resolve conflicts of information among users. Additionally, Wiki software saves revision history of every edit to an article, allowing users to revert a particular edit to any previous point in time.

In an educational context, Wiki software can facilitate student learning by leveraging the social constructivist and cooperative learning paradigms inherent to the collaborative nature of Wikis

[24-27]. Cooperative learning in engineering design courses is an established and well received practice[28]. Students document their progress on the Wiki, allowing others to learn from the material. Consequently, students edit existing information, further facilitating learning in a social context. Wiki software also aids in assessment since it can measure design ability. Cheville et al. suggest that the ability to communicate the process and details of a design is a reliable measure of overall design ability[29]. This stems from the positive correlation of effective communication with performance on design projects[30], and the use of verbal communication as a means of assessment on design projects[31]. Wiki software records individual contributions on a per-user level and records every revision to existing articles. This provides an effective way to measure individual contributions as well as team contributions.

In cooperative learning, students collectively work towards a common goal that supports the learning of both the individual and the team. Cooperative learning facilitates a "positive interdependence of group members, individual accountability, face-to-face interaction, appropriate use of collaborative skills, and regular self-assessment of team functioning"[26]. The use of Wiki software facilitates cooperative learning by providing a powerful tool for rapid, asynchronous communication, discussion, revision, and scaffolding of information. The use of Wiki software also facilitates cooperative learning in a multi-generational context, as previous student's work can be retained for future semesters to build upon.

Lectures, which are usually the starting point for most faculty members designing a course, and which are often included with textbooks take a backseat in the active learning approach. We let the labs and the course project drive our lectures, and plan them as we go. PowerPoint slides from prior implementations of the course serve as a reference on content that should be covered, but the content is covered in the order required by the labs and the course project. Exams are oriented to cover those aspects of the course which will not be captured through the course project wiki.

4. A Platform for Research in Engineering Education

PLP and the transformation of CBS to use PLP will serve as a vehicle for conducting research to answer the following questions:

1. How effective is PLP in changing student motivation?
2. How does PLP improve student learning?

The answers to these questions will come from a number of studies performed over the next two years as part of an NSF funded project. We will take an integrative approach to the assessment of the effect PLP has on student learning, motivation, cognition, and performance. Effective design and appropriate implementation of PLP is expected to contribute to enhanced motivation, learning, and achievement. Motivation researchers, particularly those taking the social cognitive perspective, suggest that students' goals and beliefs are shaped by their perceptions of the learning environment. Therefore, it is essential to examine how students' achievement goals and motivational beliefs are affected by the PLP learning environment. To assess the many factors contributing to learning, we have carefully selected multiple evaluation methods, which taken together will provide meaningful and reliable data. Both qualitative and quantitative studies will

be designed to facilitate a longitudinal perspective. The research to answer these questions begins with the work described in this paper.

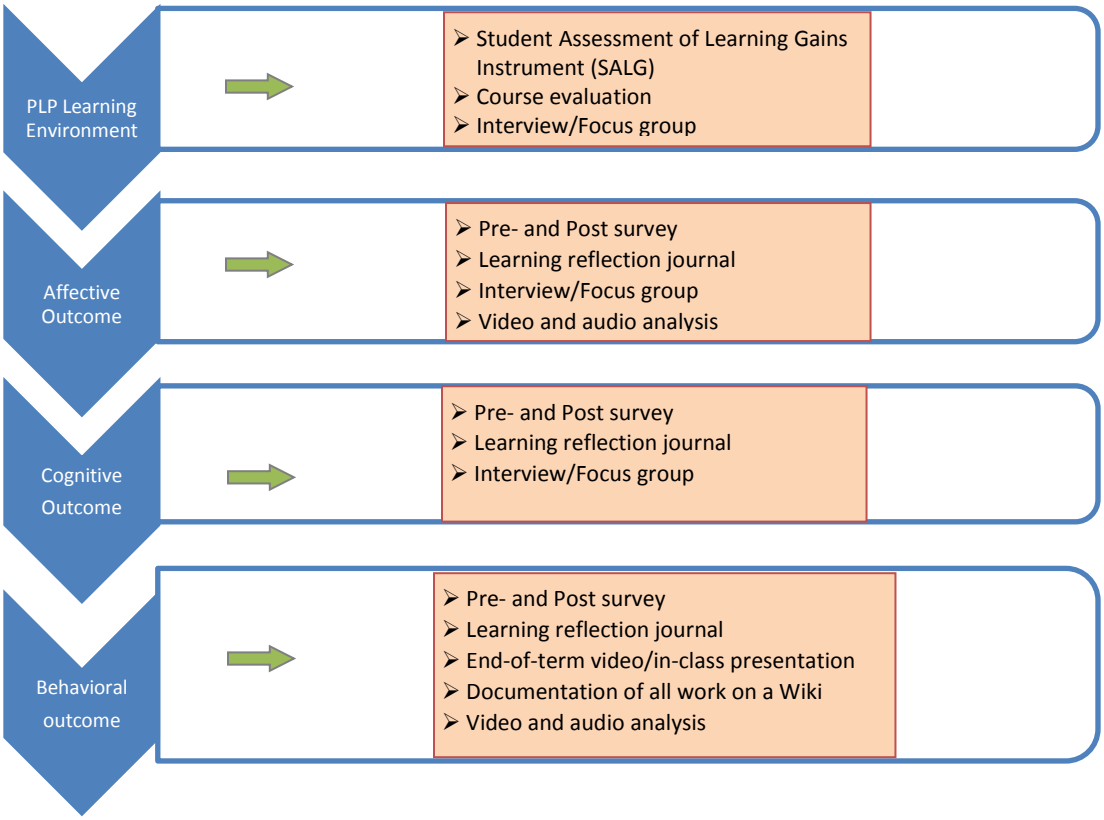

Figure 3: Assessment. The outcomes and the different activities planned for assessing them.

Figure 3 shows the particular instruments that will be used to measure the impact of PLP on each facet of student learning and development in CBS. Below we describe each of these instruments.

Student Assessment of Learning Gains (SALG): The SALG [32] instrument, used since 1997 and nationally validated, will be used to gather information on how students evaluate various course elements and individual gains from the PLP based courses. It provides a method for students to clearly rank the elements of the courses with respect to their learning of concepts and skills and appreciation of the subject and its application. The instrument can be configured to include questions on how the PLP based courses contributed to the students' overall learning in a course, or ask even more specific questions regarding particular characteristics of the course.

Interview/Focus Group: Students taking CBS will be interviewed by a faculty member outside ECEN to capture student responses that might not be captured through the SALG of other methods, due to the free-form and interactive setting of interviews. A similar focus group was convened for a trial implementation of PLP in a senior level computer architecture course with excellent results. Questions such as those listed below will be used as starting points and to steer the discussion.

1. Describe your learning experience in this class.
   1.1. How does it compare to other learning experiences in your engineering coursework?
2. In what ways were you able to link learning from other classes to this one?
   2.1. What are the areas that applied?
   2.2. How did you apply your previous knowledge?
3. How would you describe your interest level within and across the time of this class?
   3.1. What caused your interest to increase or decrease at varying points in the class?
4. Describe the level of collaboration that was required with your classmates.
   4.1. In what ways was this alike or different from experiences in your other coursework?
   4.2. What do you see as the effects or impacts?
5. In what aspects did this class help you to prepare to work in engineering in the "real world"?

Pre-and-Post Surveys- Knowledge Growth: To evaluate the effectiveness of PLP a one group pretest posttest design will be used. This design involves a single group that is pretested, exposed to a treatment, and then tested again[33]. The success of the treatment is determined by comparing pretest and posttest data. Specifically, we are interested in finding out if the use of the PLP system impacts students' knowledge of microprocessors in general, and if the use of the PLP system in the classroom impacts knowledge of concepts explicitly covered by PLP. To analyze data under this design, a dependent t-test statistical test will be utilized. A dependent-samples *t* test assesses whether the mean difference between paired/matched observations is significantly different from zero. That is, the dependent-samples *t* test procedure evaluates whether there is a significant difference between the means of the two variables (test occasions or events). The questions on the test originate from a 'Quiz 0' that has been administered on the first and last day of class to assess student learning for several semesters. Since the test was designed to capture overall student learning over the different outcomes for CBS, some of the questions do not address topics that are directly impacted by PLP. We will thus measure pre and post comparisons for the overall scores, as well as the scores for those questions on which we expect PLP to have a direct impact. The sample test is included in the appendix.

Pre-and-Post Surveys- Motivation and Mindset: Surveys will also be administered for gauging student mindsets at the beginning of the course and at the end. This survey includes questions from the Implicit Theories of Intelligence Scale[34], the Achievement Goal Questionnaire[35] the Academic Efficacy Scale[36], the Intrinsic Value Scale[36], the Engagement Scale[37], and the Classroom Community Inventory[38]. Participants will respond to the following items on a 7-point Likert scale ranging from (1) Not at All True of me to (7) Very True of me.

Analysis of Video and Audio Recordings in Lab: Video-taping of observations allows for multiple researchers to analyze the data at multiple points in time. Researchers in communication will transcribe the audio and video and analysis the body language as well as the linguistics. This will help us gauge students attitudes towards PLP and their fellow students, to determine whether these attitudes change over the course of the semester.

Learning Reflection Essays or Journals: Students will be assigned a reflection paper to summarize their outlook on what they learned from the course. Note that this will be done before the focus group and without any input questions or hints, so that we can capture their understanding and perspective on what happened. In addition, the instructor and the TAs will write their own essays to capture their perspective on the course. These reflections essays will be analyzed by a faculty member outside the department, who has the qualifications and prior experience analyzing such artifacts.

End-of Semester Videos: Over the past few semesters, we have had great success with student developed videos that are demonstrated at our end-of-semester design day event. These videos not only allow students to express themselves through a different medium, but also provide a wealth of information if analyzed by the right experts. Again, faculty outside engineering with the relevant qualifications will analyze these videos to gauge student attitudes.

5. Conclusions and Future Work

This paper presents a view of our transition to collaborative learning using the progressive learning platform. It also provides some details on how we plan to assess the effectiveness of our course. We hope that it will serve as a model for others aiming for a similar transition. We encourage these adoptions, and will facilitate them by making all our findings as well as our curricular materials available.

Future work for PLP includes the introduction into other courses at OSU and a longitudinal study of students using PLP over a number of semesters.

6. Obtaining the Progressive Learning Platform

The PLP system is licensed under the GNU GPL version 3 license. Media components, including recorded lectures from the classroom, lecture slides, in-class assignments, and other documents such as syllabi are licensed under a Creative Commons Attribution license. All are cost-free to use and modify.

The project is hosted at http://plp.okstate.edu

7. References

1. *The Progressive Learning Platform*. Available from: http://plp.okstate.edu/.
2. D. Fritz, W. Mulia, and S. Sohoni, *The Progressive Learning Platform*, in *Workshop on Computer Architecture Education*. 2011: San Antonio, TX.
3. David Fritz, et al., *The Progressive Learning Platform for Computer Engineering*, in *ASEE National Conference and Expo*. 2011: Vancouver, Canada.
4. J. Valvano, *Introduction to Embedded Microcomputer Systems: Motorola 6811/6812 Simulations*. 2003: Thomson-Engineering 480.
5. J. Valvano. *Texas: Test Execute and Simulate*. Available from: http://www.ece.utexas.edu/~valvano/sim.html.

6. R.E. Bryant and D.R. O'hallaron, *Introducing Computer Systems from a Programmer's Perspective*, in *Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education*. 2001, ACM: Charlotte, North Carolina, United States. p. 90-94.
7. I. Branovic, R. Giorgi, and E. Martinelli, *Webmips: A New Web-Based Mips Simulation Environment for Computer Architecture Education*, in *Proceedings of the 2004 workshop on Computer architecture education: held in conjunction with the 31st International Symposium on Computer Architecture*. 2004, ACM: Munich, Germany. p. 19.
8. *Ravi*. Available from: http://ls12-www.cs.tu-dortmund.de/de/teaching/download/ravi/index.html.
9. M. Brorsson, *Mipsit: A Simulation and Development Environment Using Animation for Computer Architecture Education*, in *Proceedings of the 2002 workshop on Computer architecture education: Held in conjunction with the 29th International Symposium on Computer Architecture*. 2002, ACM: Anchorage, Alaska. p. 12.
10. K. Vollmar and P. Sanderson, *Mars: An Education-Oriented Mips Assembly Language Simulator.* SIGCSE Bull., 2006. **38**(1): p. 239-243.
11. J. Larus. *Spim: A Mips32 Simulator*. Available from: http://spimsimulator.sourceforge.net/.
12. *Hades Interactive Simulation Framework.* Available from: http://tams-www.informatik.uni-hamburg.de/applets/hades/webdemos/index.html.
13. *Lc-3 Simulator*. Available from: http://highered.mcgraw-hill.com/sites/0072467509/student_view0/lc-3_simulator.html.
14. M. Holland, J. Harris, and S. Hauck. *Harnessing Fpgas for Computer Architecture Education*. in *Microelectronic Systems Education, 2003. Proceedings. 2003 IEEE International Conference on*. 2003.
15. Y. Nagaonkar and M.L. Manwaring, *An Fpga-Based Experiment Platform for Hardware-Software Codesign and Hardware Emulation*, in *Proceedings of 2006 International Conference on Computer Design (CDES'06/ISBN #:1-60132-009-4/CSREA)*. 2006, CSREA Press: Las Vegas, Nevada. p. 169-174.
16. *The National Effective Teaching Institute*. Available from: http://www4.ncsu.edu/unity/lockers/users/f/felder/public/NETI.html.
17. *Oklahoma State University Institute for Teaching and Learning Excellence*. Available from: http://itle.okstate.edu/.
18. *Certificate in University Faculty Preparation*. Available from: http://grad.okstate.edu/programs/ufp_cert/ufp.htm.
19. M. Cakir, *Constructivist Approaches to Learning in Science and Their Implications for Science Pedagogy: A Literature Review.* International Journal of Environmental & Science Education, 2008. **3**(4): p. 193-206.
20. J. Shimazoe and H. Aldrich, *Group Work Can Be Gratifying: Understanding & Overcoming Resistance to Cooperative Learning.* College Teaching, 2010. **58**(2): p. 52-57.
21. C. Mcdowell, et al., *The Effects of Pair-Programming on Performance in an Introductory Programming Course.* SIGCSE Bull., 2002. **34**(1): p. 38-42.
22. L. Williams and R.R. Kessler, *Pair Programming Illuminated*. 2003: Addison-Wesley Professional.
23. D.R. Hannah and R. Venkatachary, *Putting "Organizations" into an Organization Theory Class: A Hybrid Cao Model for Teaching Organization Theory.* Journal of Management Education, 2010. **34**(2): p. 200-223.
24. M. Notari, *How to Use a Wiki in Education: 'Wiki Based Effective Constructive Learning'*, in *Proceedings of the 2006 international symposium on Wikis*. 2006, ACM: Odense, Denmark. p. 131-132.
25. J.T. Chao and K.R. Parker, *Wiki as a Teaching Tool.* Interdisciplinary Journal of Knowledge and Learning Objects, 2007. **3**: p. 57-72.
26. S. Schaffert, et al. *Learning with Semantic Wikis*. in *Workshop on Semantic Wikis*. 2006.
27. B. Mcmullin, *Putting the Learning Back into Learning Technology.* Emerging issues in the practice of university learning and teaching, 2006: p. 67-76.
28. A. Cheville, C. Co, and B. Turner. *Improving Team Performance in a Capstone Design Course Using the Jigsaw Technique and Electronic Peer Evaluation*. in *American Society for Engineering Education Annual Conference and Expo*. 2007. Honolulu, Hawaii.
29. A. Cheville, C. Co, and B. Turner. *Communication as a Proxy Measure for Student Design Ability in Capstone Design Courses*. in *American Society for Engineering Education Annual Conference and Expo*. 2007. Honolulu, Hawaii.
30. A. Dong, A.W. Hill, and A.M. Agogino, *A Document Analysis Method for Characterizing Design Team Performance.* Journal of Mechanical Design, 2004. **126**(3): p. 378-385.
31. C. Atman, *Verbal Protocol Analysis as a Method to Document Engineering Student Design Processes.* 1998. **87**.

32. *Student Assessment of Learning Gains*. Available from: http://www.wcer.wisc.edu/salgains/instructor/salgains.asp.
33. L.R. Gay, G.E. Mills, and P. Airasian, *Educational Research* 9ed. 2009, Upper Saddle River, New Jersey: Pearson Education.
34. C.S. Dweck. *Mindset*. Available from: http://mindsetonline.com/.
35. A.J. Elliot and K. Murayama, *On the Measurement of Achievement Goals: Critique, Illustration, and Application*. Journal of Educational Psychology, 2008. **100**(3): p. 613-628.
36. P.R. Pintrich, et al., *A Manual for the Use of the Motivated Strategies for Learning Questionnaire*, T.R.o.t.U.o. Michigan, Editor. 1991: Ann Arbor, MI.
37. J.G. Wellborn, *Engaged and Disaffected Action: The Conceptualization and Measurement of Motivation in the Academic Domain*, University of Rochester: Rochester, NY.
38. A.P. Rovai, M.J. Wighting, and R. Lucking, *The Classroom and the School Community Inventory: Development, Re-Finement, and Validation of a Self- Report Measure for Educational Research.* Internet and Higher Education, 2004. **7**(4): p. 263-280.

Biographical Information

SOHUM SOHONI
Dr. Sohum Sohoni is an Assistant Professor in Electrical and Computer Engineering at Oklahoma State University. He received his PhD in computer engineering from the University of Cincinnati in 2004 and his Bachelors in electrical engineering from COEP, Pune University in 1998. His research interests are in STEM Education and Computer Architecture.

DAVID FRITZ
David Fritz is a Doctoral Candidate in Electrical and Computer Engineering (ECE) at Oklahoma State University (OSU). He received his M.S. in ECE from OSU in 2008. His research interests are in Computer Engineering Education and Computer Architecture, with an emphasis in memory systems and virtualization. He is lead developer for the Progressive Learning Platform.

WIRA MULIA
Wira Mulia is a Doctoral Candidate in Electrical and Computer Engineering (ECE) at Oklahoma State University (OSU). He received his M.S. in ECE from OSU in 2009. His research interests are in Computer Architecture and Systems. He has been instrumental to the development of the Progressive Learning Platform, and has led the work on PLPTool.

Appendix A: Syllabus- ENSC 3213: Computer Based Systems

Class Time and Location: Tuesday and Thursday 9-9:50 COR 127. Labs in ES 103.

Instructor: Sohum Sohoni (ES 407, sohum.sohoni@okstate.edu)

Office Hours: Wednesday and Thursday 3:30-4:30 p.m. Also by appointment.

TAs: Wira Mulia and Pranav Pathak (Office hours will be posted on OC, location ES 408)
wira.mulia@okstate.edu, pranav.pathak@okstate.edu

Prerequisite: CS1113 or ENSC1412. These two courses cover some elementary computer programming, a skill that is essential for getting the most out of ENSC 3213.

Textbook: Embedded Microcomputer Systems: real time Interfacing, Jonathan W. Valvano, ISBN- 978-1-111-42625-5. This is a reference book, and not a required text.

Course Website: All course material as well as the syllabus, TA hours, announcements etc. will be up on OC (online classroom). All assignments should be dropped into the relevant dropbox on OC by the due date. Please do not turn in paper copies unless specifically asked to do so. The course will also use a tool called PLP (Progressive Learning Platform). Information on the tool, and access to some additional materials will be at the PLP site: plp.okstate.edu.

Course Goals (things you will learn in this course): Embedded computing, binary number systems, assembly language programming, organization and design of a microcomputer. Essentially, you will have a detailed answer to the question- 'What is a microprocessor and how does it work?' The course will be taught in an active learning environment, with team projects that will put the science and concepts covered by the course material in a practical context.

The course is based around the following questions that you will be expected to be able to answer at the end of the semester.

1. What are the different components of a computing system and how do they interact with each other to accomplish a given task?
    - The anatomy and physiology of a computing system, system-level block diagrams, views of sub-elements within the central processing unit.
    - System architecture, choice of storage modes, theory behind the choice of addressing modes and implications of allowing various addressing modes (RISC versus CISC models of Instruction Set Architecture design).
    - Types of memory, types of storage, characteristics.
2. How does a microprocessor represent information, and how does it make decisions?
    - The concept of binary representation of text and numbers, and the reasoning behind it – the connections to and implications for underlying hardware.
    - The effect of a finite number of bits for representation - accuracy and precision.
    - The stored program model and instruction execution.
    - Phases of execution - fetch, decode, execute, write-back.
3. How does the microprocessor communicate with the devices around it, and how can we program it to control these devices?

- Operations involved in source code compilation and assembly.
- Conceptual difference between High-level and Assembly-language programming.
- Assembly language programming concepts - syntax, conditional statements, loops, subroutines, stack manipulation, modular code, relocatable code.
- Interrupts- Vectored and Polled
- Serial versus parallel communication
- Digital/analog conversions
- Sampling theory and Nyquist Criterion
- Data acquisition (ADC/DAC), interfacing, signal conditioning
- Basics of Servo-motor control

4. How can we take an engineering problem and present it to the computer in a way that it can solve that problem?
    - Engineering problem solving
    - Block-diagram design approach
    - Team-dynamics and team-building
    - Design-build-test cycle
    - Algorithms, order of growth, and big-O notation.
    - Debugging – functional and performance.

Grading: Grades will be determined as follows. There will be no curve in this course.

1 mid-term exam (50 minute in-class)	100
Final Exam (2 hours during finals week)	100
Quizzes	200
Lab Assignments (4 labs)	200
Course Project	300
In-lab Journal and end-of-semester Reflective Essay	200

A	900 and above
B	800 – 899
C	700 - 799
D	600 - 699
F	below 600

Insurance Policy: Throughout the course, you will be allowed to accrue extra points with the total equivalent of 100 points of the grade. View these points as disaster insurance- if you have a bad day on the day of the midterm, or have to miss a lab session when you were expected to demonstrate your working program, you can use the points you accrue to make sure that your grade does not suffer. If you accrue enough points through the semester, you can breathe easy during finals week.

Make-up Exams: With the above insurance policy in place, no makeup exams will be given under ordinary circumstances. If you miss the final, you will receive an incomplete grade and will be

allowed to take the final the next semester, unless you notify me that you have accrued enough points to make the grade you desire, and do not wish to receive an incomplete grade.

Late homework/lab: 20% penalty within one week after the due date. No credit thereafter. Start early and ask the TAs for help to avoid turning in the assignment late, and to avoid any temptation to copy someone else's code. This will apply to all labs, and all components of a lab. For a multi-part lab, if you demonstrate one of the parts late, the late penalty will apply only to the part that was demonstrated late, and not to the entire lab.

Peer Evaluation: Throughout the semester a number of peer evaluations will be administered. These peer evaluations give each student a chance to rate team member performance (in the form of a grade) and provide comments to the instructor. Peer evaluation comments and grades will not be disclosed to students. Peer evaluations may be used at the end of the semester to weight course grades. Weighting is at the discretion of the instructor, and will be applied by averaging peer review scores from your team members. Weighting will not fall outside the range of 50% to 150% of your raw score. Peer evaluations will be taken on D2L.

Policy on Academic Integrity: I do not hold the view that most of you are lazy, and are just looking to copy assignments and programs from someone. I also encourage you to discuss the assignments and help your friends and fellow students understand the course material. However, there are some things that I will certainly not allow. You will not share your program code in any way with someone who is not your lab partner, or someone who is not on your team (if we form teams). You will not pass text messages, or communicate in any way with your peers during quizzes or exams. You will not copy the solution for a take-home assignment. It is your responsibility to prevent your work from being copied. Both the student doing the copying and the student allowing the copying will be punished to ensure fairness to those who don't. I have written up students for violations of the academic integrity policy in the past. It is without a doubt, the most stressful and unpleasant part of my job as a faculty member. Please do not make me write you up. If you do, you will see my respect for the entire class and my enthusiasm for teaching go down a notch, and if I have to go through it year after year, I will turn into one of the cranky old professors who hate all their students and allow only the top 10% of the class to pass. The policy on academic integrity is in place not just to discourage bad behavior, but to ensure that those who spend the hours trying to get their programs to work are rewarded for their efforts. If you have any specific questions on what is allowed and what isn't, ask me.

Special Note: Read this syllabus carefully. There will be a quiz on the syllabus. You have to take the quiz and get a passing grade on it before you start your first lab assignment. You can take the quiz multiple times until you pass. I am not kidding.

Appendix B: CBS Fall 2011 Project Description

REMOTE ROBOT CONTROL

Your group of engineers are tasked to design a remote control scheme for the PLP bot to navigate an obstacle course. The robot is equipped with an IP camera, optical sensors, rotary feedback sensors, an XBee wireless communication module, and of course, the PLP board. You need to utilize all of these components to meet the requirements listed in the next section.

PROJECT REQUIREMENTS

1. The robot must be able to complete the first leg of the obstacle course with manual control. Then the robot must be able to autonomously navigate the second leg by relying on its sensors to follow a line.
2. The robot must record its movements in the first two legs and use this data to autonomously return to its home base.
3. The robot has to be controlled via wireless communication during the first leg.
4. The controller station must indicate the current speed of the robot at all times. This indication must be updated every 50ms or faster.
5. The robot must record a running average of its speed during the run.
6. The robot must indicate a loss-of-signal whenever it loses signal from the controller station during its run.
7. The robot must stop when it is in the loss-of-signal mode.
8. A video documenting the development and showcasing the result of the project.

TEAMS

The class will form four teams for the project: the manual control team, the sensor team, the autonomous team, and the controller station team. The manual control and the sensor teams are responsible for the first and second legs of the obstacle course, respectively. The autonomous team is responsible for recording and replaying the robot's movements to allow it to return to its home base. The controller station team is responsible for designing and implementing the control software on the PC and the base station for the robot.
Each team will have a project manager, a lead engineer, and a documentation expert.

- The project manager is responsible for managing the team and making sure that the goals for the project are met. The project manager also holds the final authority on all design decisions.
- The lead engineer is the team member who is responsible for making the case for the team's technical design decisions to help the project manager decide what course of action to take. The lead engineers are also expected to collaborate with the other lead engineers to make sure that the different subsystems will work together.
- The documentation expert is responsible for updating the team's documentation.

PROJECT PHASES

This project has three phases as described below:

1. Research

In this phase, your team will have to research, plan, and document the steps that you will need to complete the project. You will have to research the design component that has been assigned to your team and gain complete understanding on what is required to meet the project requirements, and be able to communicate your understanding to the other teams and the rest of the class. The documentation that you produce in this phase will guide your group for the rest of the project.

Requirements:

1. Documentation of detailed description on the planned design.
2. An implementation plan for the next phase of the project: timeline and objectives.
3. Each team will give a presentation on the result of the research phase.

2. Implementation

In this phase, each subsystem team will implement their component and document their progress and design derivations. The final product of this phase is a working subsystem that is ready to be integrated with the rest of the project and a detailed documentation of the subsystem.

Requirements:

1. A complete, working subsystem that meets the specifications from the research phase
2. Documentation of the team's design process and the implementation itself
3. Demonstration of the subsystem to the instructor and TA

3. Integration

The final phase involves integrating all of the subsystems to make a complete robot control system. The class as a whole is expected to give a demonstration of the system, finalize documentation, and submit a class video.

Appendix C: ENSC3213 Pre and Post Quiz- Fall 2011

1. Are you a sophomore, junior or senior?

2. Have you participated in any courses that have required you to work in teams? If yes, which one(s)?

3. Have you written any computer programs? If yes, which programming languages have you used?

4. Have you taken ECEN 3233 (digital logic design) or any other digital design course?

5. Given a choice between designing hardware or software what would you pick? Please check one of the following:

Hardware	Software
I like both equally	I don't know enough to choose

6. Of the following words, please check all those that you are familiar with

- Logic Gate
- EPROM
- Diode
- D Flip-flop
- Hexadecimal
- DAC (Digital to Analog Conv.)
- ROM (Read Only Memory)
- Data structures
- Binary
- Boolean logic
- I/O port
- CPU registers

7. How would you rate your interest in Computer Engineering on a scale of 1-5? (5 being the highest)

8. Do you plan to take additional Computer Engineering courses?

9. Do you plan to pursue graduate studies in Electrical or Computer Engineering?

10. What is <u>your idea</u> of an embedded microcomputer system? If you have no idea, just say so.

Name:

11. List the main parts of a microprocessor

12. 0x24a5002a is the machine code to add 42 to register 5. How does the microprocessor know to perform this action based on the instruction?

13. What is the result of 0110 AND 0011? 0110 XOR 0011?

14. Can the PLP processor correctly add the numbers 1241390809987 and 378902930? If not, why not?

15. What is the sequence of events that automatically execute when an interrupt occurs?

16. What are the three conditions necessary to cause an interrupt?

17. What is the difference between serial and parallel communication?

18. Given that all the serial UART send register is empty, what's the maximum number of bits that we can write without overwriting our data before a single bit is transmitted?

19. A microprocessor communicates with the outside world through ……

20. In PLP, what does the .org command do?

21. What are breakpoints? What can they be used for?

22. How are input and output devices accessed by the microprocessor?