A Custom Microcontroller System used as a platform for learning in ECE

Adriaan Smit, Donald Heer, Roger Traylor, Terri S. Fiez

Abstract: *TekBots™ is a program that was started at Oregon State University in the Electrical and Computer Engineering Department to develop Platforms for Learning™. The program is designed to assist, re-enforce and accelerate the learning process by integrating knowledge across many different courses. For each course the TekBot platform is used to closely tie the course material to 'real' engineering hardware. With these hands on materials, the students can attach a real meaning to many of the seemingly ambiguous topics learned in lecture. The TekBots platform is composed of many different sub-platforms that interact with each other to create working systems.*

The AVR microcontroller sub-platform consists of an embedded 8-bit microcontroller with features including; a liquid crystal display (LCD), analog-to-digital conversion (ADC), pulse width modulation (PWM), infrared (IR) communications, and a serial port. This powerful sub-platform is introduced during the junior level Computer Architecture and Assembly Language Programming class, taken by electrical and computer engineering students.

The AVR sub-platform is reused in later classes as a building block to bigger systems. Examples of this are Signals and Systems where the AVR becomes a simple digital signal processor, microcomputer design using the AVR as the core of a student built system, and VLSI Design where the students interface a FPGA coprocessor to the AVR.

Introduction

Teaching design and innovation to students is possibly one of the most critical challenges facing engineering education in the future [3, 4, and 5]. Faculty are continually being asked to do more in less time, while at the same time the prohibitive costs of 'real' lab experiences cause many institutions to remove physical labs from courses. Even where physical labs remain, the labs are doctored to use "pretty" or "fixed" numbers and experiments hiding the real design work from students. A solution developed at Oregon State University is Platforms for Learning™.

TekBots™, a platform for learning created for electrical and computer engineering students, is designed to assist teaching many practical engineering skills that may often be left uncovered; innovation, design, knowledge integration, and the 'real' problems of 'real' systems. With these hands on materials the students can attach a real meaning to many of the seemingly ambiguous topics presented in lecture.

In the following section the paper will present a platform for learning as it has been developed at Oregon State University. TekBots and the AVR sub-platform are then presented with in the scope of the first course they are used in the Computer Architecture and Assembly Programming course. The paper concludes with how the AVR sub-platform is to be extended through the curriculum and future work.

The Platform for Learning Concept

A Platform for Learning is a set of common, unifying objects or experiences that tie together the concepts introduced in various classes. The platform gives students a context for learning, a way to clearly observe relationships and dependencies between different materials. It provides a knowledge foundation that is expanded on through the curriculum. The active nature of the platform forces student to observe how real devices and systems differ from the standard perfect solutions commonly discussed in lecture; by being a common factor across different courses, the platform brings better continuity to the curriculum. The platform represents the application of the material taught in class, and how it relates to what was learned in previous classes. It acts to expand and integrate the entire curriculum, as indicated by Figure 1.

The ability for students to reference previous knowledge gained using that platform is key to how the platform can enable design experiences with out a large increase in faculty workload and cost to students. Students reuse larger portions of the platform and only need to create 'smaller' designs.

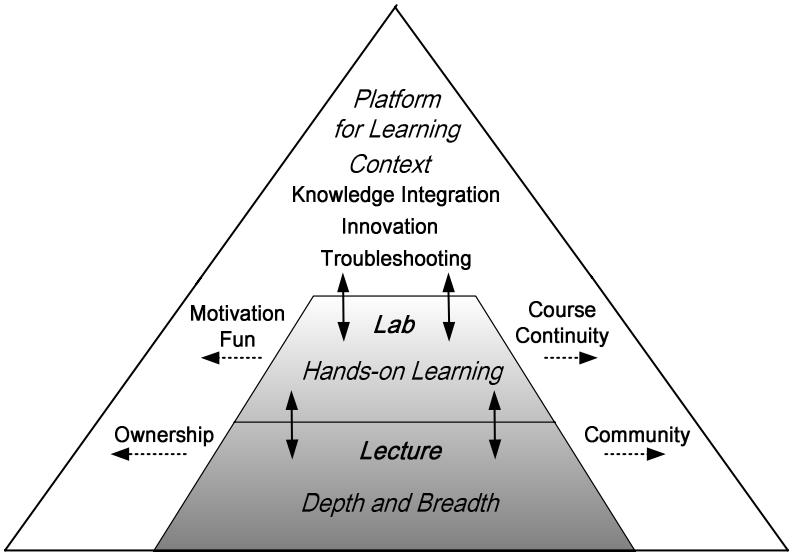

Figure 1 – A platform for learning expands the learning opportunities by providing context, knowledge integration, innovation and troubleshooting experiences. It also created ownership, motivation, community, and course continuity.

When working with the platform, students often need to choose one out of many solutions, in contrast to the end-of-chapter problems that usually have only one correct solution and rarely more than one way to find it. In this sense, the platform naturally puts students in the position of actual engineers. To solve these problems, students use many of the procedures and practices that they will need later in their professional careers. This emulation of engineering practice encourages students to use their imagination and teaches them to be innovators, rather than replicators only capable of performing exercises like ones they have already done.

Since the platform is a real object, it comes with real constraints and imperfections that are often different from the ideal/theoretical problems. In order to overcome these problems, students need to learn how to effectively troubleshoot or problem-solve, an important engineering skill. In addition, students might turn to their classmates for help, as well as consult others that work with the same platform. All students share the same platform, which helps to build a learning community around it. Finally, students actually experience making something that works, rather than just learning about it in theory, returns the excitement back in the curriculum. Even though it is often neglected, the fun-factor is extremely important in the development of future engineers.

Sub-platforms

Every platform for learning is built from many smaller sub-platforms and temporary systems. The temporary systems are used in a single class or possibly two but are not designed to be reused to the extent of sub-platforms. Sub-platforms on the other hand are complete systems that embody knowledge a student has learned. For example when a student learns the details of transistor biasing and constructs a custom biased H-bridge for driving a motor; they build this system and integrate it into their platform. When the student learns about assembly programming and computer architecture they program a microcontroller sub-platform that they can add to their platform.

TekBots and the AVR Sub-Platform in Use

The TekBots platform for learning is a simple robotic base that students control to perform many different tasks with a vast range of complexity. The platform varies from a simple 'charging capacitor' timer control to digital state machine programmed in a microcontroller. The platform has been designed for ruggedness with great care being taken to produce a flexible design. The basic TekBot is shown in figure 2 along with a Tekbot and the AVR sub-platform.

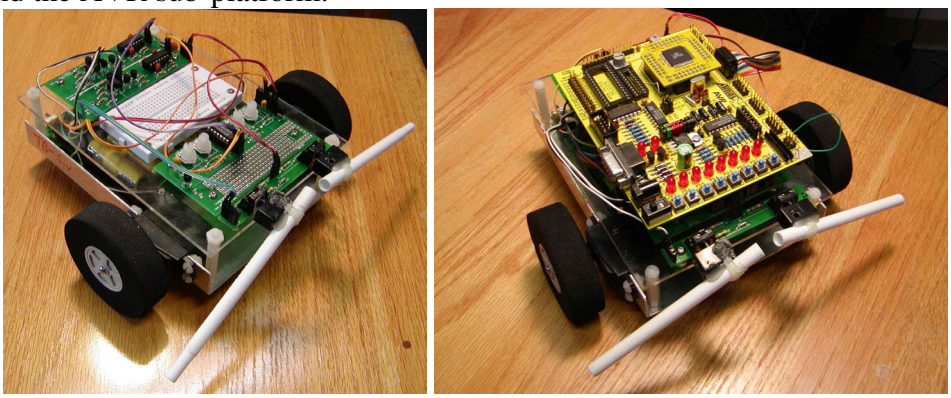

Figure 2: Basic TekBot and an AVR equipped TekBot

For the Computer Architecture and Assembly Language Programming class this platform is a custom Atmel AVR uController board designed at Oregon State University, figure 2. During this course students learn about the architecture and inner workings of a microcontroller. Every week the lecture is complemented by a lab that directly relates to the material covered in class. For example when in class the students learn about micro-operations and the working structure of an AVR microcontroller, students learn about the

assembly language instructions that performs the operations, during lab. The classroom lecture provides the groundwork for the labs while in return the labs make the lecture material more concrete.

The AVR board can also be used in more than one class. If the Atmel AVR board is programmed appropriately it can perform the same function as some of the other boards from lower level classes that were designed with discrete components. This helps students apply what they are learning in both their current class and their previous classes. The evolution of the platform shows how a problem can be solved in more than one way. For example students are asked at the beginning of each class to construct a simple 'bumper bot' that can wander around a room. The functionality of this design is well understood by each student since they must create it several times in several classes. This allows them to focus on the new information of how they will implement the design.

The Computer Architecture and Assembly Programming Course

At Oregon State University the Computer Architecture and Assembly programming course is commonly taken at the end of the sophomore or beginning of junior year. The course covers many fundamental concepts of computing and computer architecture, including; basic binary arithmetic, memory types, memory access types, instruction execution, Harvard and Von Neumann architectures, and instruction sets.

Previous to the addition of TekBots the course relied on an 8051 microcontroller simulation tools developed at Oregon State University. Students did not interact with real hardware except in the very last lab where a single system was available for the students to share and try their code on.

However after TekBots platform was integrated many changes have been made, completely updating and revitalizing the course.

Each Student owns their platform – Having a sense of ownership in this course has brought a new sense of worth to what the students are doing. Students now care if what they build doesn't work because they own it.

Real World Experience – Students are now using real hardware to see what is going on. With real hardware real problems can arise. Students have to understand that everything they do has an affect on other boards and systems.

Relevance – The restructuring and improvements helped to motivate the students to better understand the abstract materials in lecture. The new labs and Platform gave a reason to the theory.

The TekBots Experience

Lab #, Week #	Lab Description
Lab 0 Week 1 TekBot Purchase and Assembly	Students get the chance to buy their AVR boards and make sure that they work. Students also buy and assemble a TekBot if they don't already have one. Those that do, make sure that they work correctly.
Lab 1 Week 2 Lab Introduction - Become familiar with the ATmega128, AVRStudio4, and PonyProg2000	Students are provided with an assembly program that controls the TekBots to drive around a room while avoiding objects. This allows the students to become familiar with the development tools without having to know much about writing an assembly program.
Lab 2 Week 3 C->Assembly->Machine Code->TekBot Grasp a handle on assembly language through a structured programming language.	In this lab the students will write a C program that performs the same task as the assembly program from the previous lab. The compiled code from both labs are compared to point out the efficiency of writing in assembly language. Students also learn more about the TekBot and AVR hardware.
Lab 3 Week 4 Data Manipulation and LCD Display. Learn to manipulate the data in order to write your name on the LCD Display	In this lab students learn the difference between data and program memory. How to read from program memory and how to manipulate data memory. Also how to move data around using indirect addressing with the X, Y, and Z pointers. The students also learn how to set up the registers appropriately for SPI communication with the LCD display.
Lab 4 Week 5 Large Number Arithmetic Learn to manipulate numbers and data that are larger than the architect data path	The AVR is an 8bit Microcontroller. In this lab students learn how to manipulate larger than 8bit numbers using the ALU and arithmetic. Learn about carry bits and overflow bits. Also learn about functions and sub routines in assembly language.
Lab 5 Week 6 Interrupted Whiskers Learn enable and use external interrupts for the BumpBot	Students learn how and when interrupts can be used. Learn about the stack and what happens when an interrupt occurs. Also learns how to write interrupt service routines.

Lab 6 Weeks 7 & 8 Extremely Simple Computer (ESC) This lab will combine all you knowledge on AVR Assembly and Computer Organization to create an ESC Simulator in an AVR environment.	This is the only lab where the AVR board is not used. The students write a simple computer simulator that will execute a simple program written with a reduced set of instructions. This simple computer is based on the AVR architecture. Students run the program on a PC and use the debugger to verify that it is working correctly.
Lab 7 Weeks 9 & 10 ECE 375 Project Remotely Operated Vehicle Combine past knowledge and ingenuity to create this final group project.	Students work in groups of two for this lab. One AVR board is programmed to be a remote and another AVR board is used as a receiver on a TekBot. All the knowledge gained through the quarter is needed to do this lab. All the previous labs have sections of code that can be reused during this lab.

Table 1: The labs used in the Computer Architecture Course

Table 1 shows all of the labs used in the Computer Architecture and Assembly Programming course at Oregon State University. The labs begin by assuming that most students will have some basic programming experience in a higher-level language but no experience in assembly level programming.

As students progress through the labs they design various programs to perform certain specific tasks. The program structure is not dictated giving students free reign to code as they see fit.

For example in lab 4 students are asked to write a program that can perform the computation: $(X + Y)^2$, where X and Y are both 16-bit numbers. This is difficult because the AVR is only an 8-bit device. Many solutions exist and the students are not asked for any specific one. Students are simply pointed towards the AVR datasheet and told to look at the ALU registers.

All of the labs are designed so that at the end students will have a small snippet of code that they can reuse for the final lab of the course. Students are helped to understand the importance of this engineering reuse to 'rapidly develop' a new design. In the final lab students use interrupts, SPI, Push Button input and the common stack and data pointers. These are all things that they learnt to do in previous labs. Therefore the code written during the previous labs can be reused during the final project. The repetitive use of code encourages students to write clean and ordered code that can easily be transferred from one program to the next.

Evaluation of TekBots Introduction in Computer Architecture

In the computer architecture course there was a lack of faculty ownership making it difficult to sell the idea to some of our key faculty. After talking with the faculty, and after many weeks of sole searching within the TekBots team, we sought to develop a prototype hardware that would demonstrate to the faculty the possibilities for challenging the students beyond what they thought would be possible. This was particularly

important since the faculty didn't feel the students could handle a very high level of sophistication. A crude prototype of the hardware and a few examples of laboratories were developed to demonstrate to the faculty the potential of this approach. While the first couple of minutes they were very skeptical, they quickly got excited about how this could change what they could teach and the experience for the students. Once the faculty had accepted the idea, work began on developing a meaningful pilot experience.

The design of the TekBot experiences in class made the experiences progressively more complex. An early laboratory involved the students developing C and assembly code that is downloaded to the TekBot that replicates the analog controller and PLD controller previously designed in other TekBots courses. In this experiment, the microcontroller continually monitors the sensor outputs to determine if the TekBot has come in contact with a wall so that it controls the TekBot to back up and turn before moving forward again. Later in the term, students learn about interrupts in the lecture and modify their "bumper robot" so that it is interrupt driven rather than constantly monitor the sensor output. The final project challenges the students to use infrared (IR) to design one TekBot to control the other TekBot. In this project, two students must work together to make a working project.

Design became an important focus of this final lab. The initial concerns about difficulty were addressed in the development of the labs that preceded the final design problem. Students developed skills in working with the hardware and software while working on smaller design projects. For example, they developed the code that would control the movement of the TekBot as a solution to a less complex problem. The final design challenge combined that code with newer issues such as the IR communication. In addition, students were provided with a skeleton code that provided direction but still forced them to interpret that code and search for resources in online documentation and class notes to accomplish the final goal. We recognized the importance of manageable but challenging design problems in stimulating integration of theoretical knowledge.

As we were contemplating the significance of this course, we came across a critical observation. In our current curriculum, we do not encourage students to take this course until the end of the junior year. However, as we explored what the content of this course should be and how it would fit in with the rest of the curriculum, we discovered that it was actually a foundational course in the context of the platform for learning. As we look ahead to other junior courses such as signals and systems, electronics and electromagnetics, we discovered that the microcontroller will be a critical interface mechanism to, for example, add sensors so that real signals can be analyzed in signals and systems.

Conclusion

The introduction the TekBots platform for learning into the computer architecture course at Oregon State University has been very successful. Students are now able to use their platforms to understand and try out new idea and designs. Reusing pieces from previous classes that are well understood by students speeds the design process allowing student to create more complex and intriguing designs.

The TekBots program plans to integrate another three courses in the next year and to revise several courses already in place to improve the teaching and lab experiences. The new courses include Signal and Systems, DSP, and VLSI Design courses.

References

1. Roger L. Traylor, Donald Heer, and Terri S. Fiez. "Using an Integrated Platform for Learning™ to Reinvent Engineering Education". 2003. *IEEE Transactions on Education, v46, No 4,* p409-419.

2. R.W. Lawler, Computer Experience and Cognitive Development. New York: Wiley, 1985.

3. Dr. Joseph Bordogna, "Next Generation Engineering: Innovation Through Integration," Keynote Address, NSF Engineering Education Innovator's Conference, April 8, 1997. (www.nsf.gov/od/lpa/forum/bordogna/jb-eeic.htm)

4. Dr. Shirley Ann Jackson, Ph.D. "Changes and Challenges in Engineering Education." 2003 American Society for Engineering Education, Main Plenary, Nashville, Tennessee. December 26, 2003. http://www.asee.org/conferences/annual2003/speech.cfm.

5. Dr. Joseph Bordogna, "Tomorrow's Mechanical Engineers: Reengineering versus Repackaging," Dinner Address, MIT Workshop, Mechanical Engineering Undergraduate Education for the Next Twenty Years, http://www.nsf.gov/od/lpa/forum/bordogna/jbmit-96.htm