120th ASEE Annual Conference & Exposition
FRANKLY, WE DO GIVE A D*MN
June 23-26, 2013

ATLANTA

Paper ID #6559

Microcontrollers for Mechanical Engineers: From Assembly Language to Controller Implementation

Mr. Noah Salzman, Purdue University, West Lafayette

Noah Salzman is a graduate student in engineering education at Purdue University. He received his B.S. in engineering from Swarthmore College, his M.Ed. in secondary science education from University of Massachusetts, Amherst, and his M.S. in Mechanical Engineering from Purdue University. He has worked as an engineer and has taught science, technology, engineering, and mathematics at the high school level. His research focuses on the intersection of pre-college and undergraduate engineering programs.

Dr. Peter H. Meckl, Purdue University School of Mechanical Engineering

Peter H. Meckl is a Professor in the School of Mechanical Engineering, where he has served since 1988. Peter obtained his BSME from Northwestern University and MSME and Ph.D. degrees from MIT. His research interests are primarily in dynamics and control of machines, with emphasis on vibration reduction, motion control, and engine diagnostics. His teaching responsibilities include courses in systems modeling, measurement systems, and control. In addition, he teaches a course entitled Technology and Values, which introduces students to the social and environmental impacts of technology through a series of readings and discussions. Peter received the Ruth and Joel Spira Award for outstanding teaching in 2000. He spent a semester in the Institute of Measurement and Control Engineering at the University of Karlsruhe, Germany, in spring 2005, conducting research and teaching on autonomous vehicles.

Microcontrollers for Mechanical Engineers: From Assembly Language to Controller Implementation

From automobiles to robotics to process automation, the electronic control of mechanical systems is ubiquitous in modern control engineering. These controls are often implemented using microcontrollers, making understanding and learning how to work with microcontrollers an important part of the mechanical engineering curriculum related to control theory. For over 25 years[1,2], a course on utilizing microcontrollers to control electromechanical systems has existed at The University.

Many different microcontroller platforms have been utilized in the course, including the Intel 8085, the Intel 80188, and the Motorola 68HC12. Having used the 68HC12 platform for the past 10 years, and with various hardware and software problems leading to student frustration, the decision was made to upgrade to a new hardware and software platform. After evaluating tools from Microchip, Texas Instruments, and ARM the decision was made to use the STM32VLDISCOVERY board utilizing a Cortex-M3 microcontroller from ST Microelectronics[3] along with the Keil[4] development environment from ARM. To facilitate easy connections to existing laboratory equipment, a custom enclosure and interface circuitry were developed. This provides easy access to digital and analog I/O and a serial interface to communicate with a terminal program running on a PC.

These tools were chosen for several reasons. Keil makes a free version of its development tools available for download, with the only major limitation being the size of the code able to be compiled. By choosing a free set of development tools, students were able to install the tools on their personal computers to work on outside of the lab and utilize for their own projects after completing the course. The STM32VLDiscovery board is readily available and inexpensive (<$15) allowing students to have their own hardware to work with as well. With over 20 billion ARM based chips shipped to date[5], it is one of the most commonly utilized microcontroller platforms currently available and is utilized across a variety of industries.

Course Description

The current course focuses on the use of microcontrollers for control system implementation. The lectures and laboratory assignments have been designed to satisfy the following three objectives: (1) provide a basic knowledge of microprocessors, their architecture, and their programming; (2) provide the tools for interfacing microprocessors with peripheral devices, including digital I/O, analog I/O, and serial communication; and (3) provide experiences in utilizing microprocessors for real-time measurement and control. Although this course is a graduate course, upper-division undergraduates are encouraged to participate. While prior experience with programming and control theory are desirable, the course does not have any formal prerequisites.

Since mechanical engineering curricula typically do not include courses on assembly language programming and microcontroller interfacing, this course is designed to introduce these topics to mechanical engineers. In this way, they can develop their expertise for implementing controllers toward the end of the semester, without having to take several courses that cover these topics in depth in the electrical engineering department. The ultimate goal is to have the students control actual electromechanical systems with microcontrollers. Appendix A contains the sequence of topics included in the course and accompanying laboratory activities.

Lectures have been designed to introduce students to various basic concepts that are crucial to programming and interfacing with microprocessors. These topics include microprocessor architecture, assembly language programming, digital I/O, serial communication, interrupts, and analog-to-digital conversion. In addition, since not all students have the same level of control background, some lectures are devoted to fundamentals of classical control design, including controller gain selection, integrator windup, and digital implementation. Since actuators and sensors represent a vital connection between the microprocessor controller and the system to be controlled, several lectures are devoted to a discussion of stepper motors and drivers, incremental optical encoders for position measurement, and issues of digital sampling and aliasing.

Numerous lab exercises complement the lectures, culminating in the design and implementation of real-time controllers for a variety of electromechanical systems. The labs begin with a basic introduction to the development environment and assembly language programming, followed by working with the various peripherals of the microcontroller. These include digital I/O, serial I/O, timer and external interrupts, and analog to digital and digital to analog conversion. The first third of the exercises focus exclusively on assembly programming, followed by an exercise in mixed C and assembly programming, with the rest of the course including the controller implementation done using the C programming language.

The last six weeks of the course are dedicated to a series of two-week projects implementing controllers on several different electromechanical systems. All of the students design a controller for a common apparatus (most recently a modified version of the Aeropendulum[6]), and then choose two additional setups. Options include an inverted pendulum, a heating and cooling apparatus, speed control of a DC motor, a cruise control for an air-powered engine, control of a refrigeration system, and an active mass damper. Each of these projects involves system characterization, modeling in Simulink, and developing an appropriately-tuned controller to achieve a set of performance specifications.

Given the somewhat unorthodox use of assembly programming in a controls course for mechanical engineers and the recent update to the microcontroller platform, we set out to answer the following research questions:

1) What aspects of the course, both content and pedagogy, did the students find the most valuable?
2) How does learning assembly language programming help students to understand and use a microcontroller?

Methods

To help answer these questions, a survey was written to collect three different kinds of information. Students were asked to provide demographic information on their status (graduate or undergraduate), major, graduate degree program, and prior experience with coursework in controls and working with microcontrollers. The second part of the survey consisted of three Likert-type questions to measure students' perceptions of the value of the various content areas presented in the course, their confidence at utilizing skill and knowledge acquired in the course, and the usefulness of the different learning activities employed in the course. Finally, the students were asked to provide their views on the usefulness of learning assembly programming, the most positive aspects of the course and opportunities for improvement via open-response items. The survey was administered online using the Qualtrics survey software, and descriptive statistics were generated using Microsoft Excel.

Results

The type of degree being pursued by the respondents and their majors are shown in Tables 1 and 2, respectively. Table 3 shows the prior controls coursework experience, and shows that the majority of respondents have taken at least one course in controls prior to this course. However, 3 of the respondents had no prior controls experience. Five of the respondents had worked with a variety of microcontrollers prior to taking this course, as shown in Table 4.

Table 1: Type of degree of respondents (n=13)

Degree Type	**Count**
MS Non-thesis	5
MS Thesis	4
Ph.D.	2
Undergraduate	2

Table 2: Major of respondents (n=13)

Major	**Count**
Mechanical Engineering	9
Mechanical Engineering Technology	1
Civil Engineering	1
No Response	2

Table 3: Prior controls coursework reported by respondents (n=13)

Number of Prior Control Courses	**Count**
None	3
1	4
2	3
3 or more	3

Table 4: Prior microcontrollers used by respondents (n=5)

Microcontroller	**Count**
Arduino	2
AVR	2
Texas Instruments MSP430	1
Intel 8051	1
Cypress PSOC	1
PIC	1

Figure 1 indicates the respondents' perceptions of the helpfulness of various content areas presented in the course. With the exception of microcontroller architecture, which is not a strong focus area of the course, the respondents indicated that areas that focused on the low-level functioning of the microcontroller (Interrupts, Interfacing, and Assembly Programming) were the most helpful aspects of the course. Least helpful was modeling and simulation, a topic that is a significant part of the project activities but receives relatively little formal treatment in the lectures.

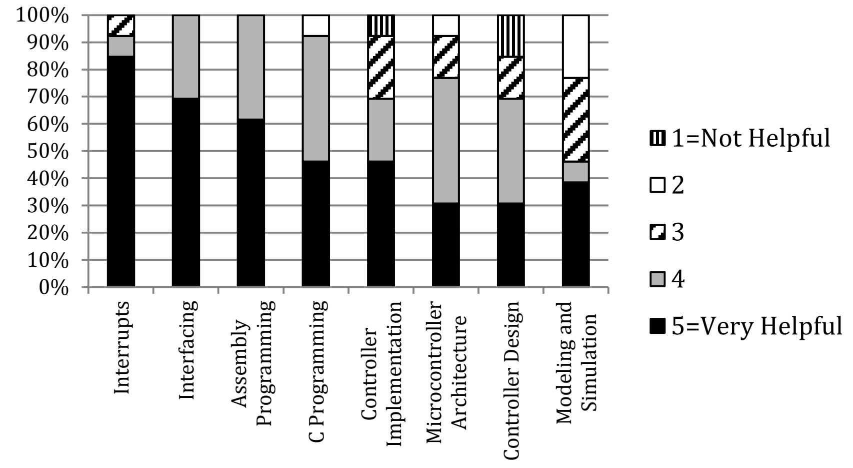

Figure 1: Helpfulness of various content areas reported by respondents (n=13)

Figure 2 shows the respondents' confidence in various skills learned in the course. The homework, labs, and projects all have a strong focus on programming, and students reported the most confidence in their ability to complete tasks related to programming. The students reported the least confidence in their ability to model a system and design a controller, suggesting that there is room for improvement in the instruction related to these areas.

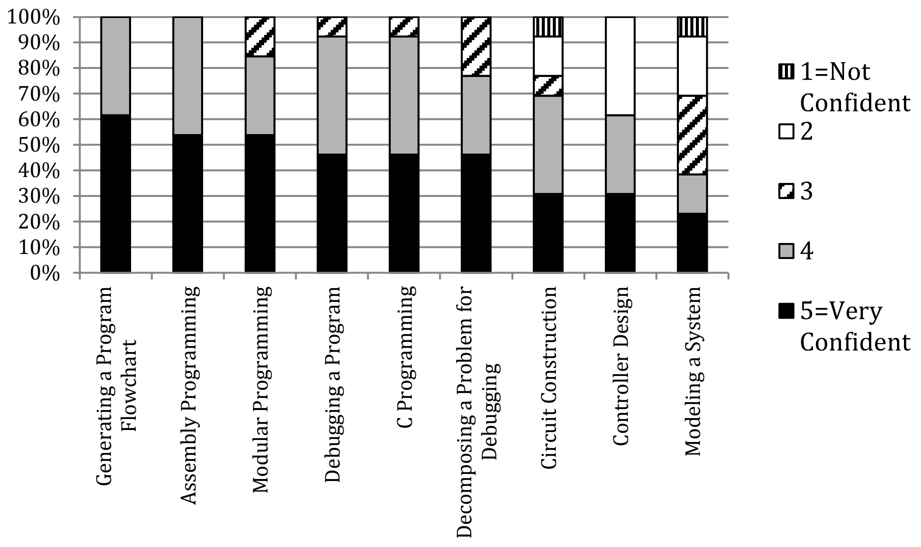

Figure 2: Confidence in skills learned during the course reported by respondents (n=13)

Figure 3 shows the helpfulness of various instructional activities reported by the respondents. Labs, lectures and projects were all considered helpful, with 90% or more of the respondents rating these a 4 or 5 on a five point scale. Homework was slightly less helpful, probably because the homework assignments generally involved writing assembly or C code to prepare for that week's lab without having access to the lab equipment.

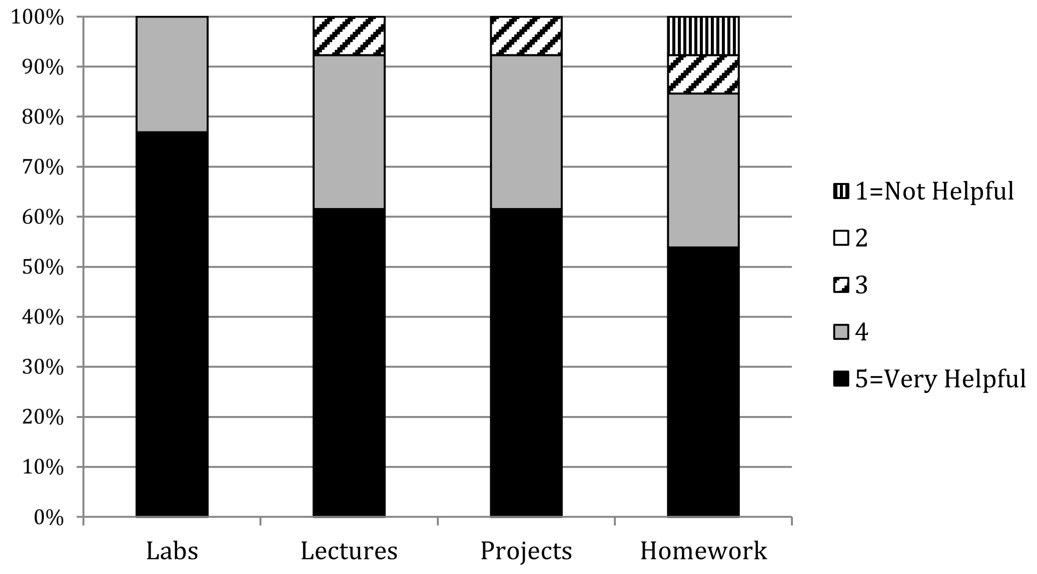

Figure 3: Helpfulness of different learning activities reported by respondents (n=13)

Open Response Results

In addition to the demographic and Likert-type questions included on the survey, the students were invited to share their thoughts via open response questions on a variety of topics. These included students' perceptions of the value of learning assembly programming, overall assessment of the value of the course, and an opportunity to describe challenging aspects of the course and suggestions for improvement.

When asked how learning assembly programming helped them to understand how the microcontroller worked and their ability to connect assembly programming to the rest of the course material, all 12 of the respondents indicated that they felt it was valuable in some way. They said it "connects more to the working of the microcontroller than C language", "helps me to better visualize how microcontrollers process in reality through memory and registers", and "assembly coding help me understand the basic operation of a microcontroller." In addition to the respondents indicating that learning assembly helped them to generally understand the operation of the microcontroller, many also indicated specific low-level functions of the microcontroller that learning assembly helped them to understand, such as "how serial and digital IO works", "how a C-program is decomposed into machine level code", and "how various components help in functioning of microcontroller like timers, interrupts, IO etc. and how best we can utilize them." Only one respondent felt that the assembly programming was "a little bit disconnected" from the rest of the course.

Most of the respondents felt that the best part of the course was the ability to apply the material presented in the lectures the solution of actual control problems in the labs and projects. One respondent wrote: “The best parts of the work is the interaction of class and lab, we can learn from lecture and get further idea of intuition from labs. When on lecture, professor tries to make us better understood the materials…so that we can not only try to understand them from the powerpoint but also can see the structure of what he tries to describe(sic).” Others appreciated the labs and projects “because they were able to draw a connection to most of the topics covered”, and “it enables us to put what we learn in class into practice.” Another respondent wrote, “I have taken many controls courses…but none of them bridged the gap between the actual implementation of controls on the physical system. Different components that I have learnt together seem to be more connected after this course.” Finally, one of the respondents also appreciated the ability to work with an ARM Cortex-M3 microcontroller, as the “ARM Cortex-M3 is popular right now.”

The time required to complete the labs and projects was the most frequently described challenge presented by the course with one student writing “a lot of time needed to prepare for the lab as well as debugging.” Another student wrote “The most challenging part to me is programming language. I have learned C++ during my undergraduate courses long ago, and I have forgotten them totally.” Others found the control theory included in the course challenging, writing: “Controls systems was new to me and it took a lot of time before I understood it.” Several also indicated modeling in Simulink was challenging. These results are consistent with those shown in Figure 1. Finally, several respondents also indicated some frustration dealing with the hardware and software bugs that accompanied moving to a new microcontroller platform requesting that we “ensure that all of the supplied code indeed functions properly. The most frustrations and waste of time stemmed from finding errors in the library code. Once you lose confidence that the supplied code was functioning properly it was increasingly difficult to efficiently debug new programs.” Other suggestions for improvement mostly focused on providing more examples or scaffolding for the content areas that they felt were challenging.

Conclusion

Based on the results of the survey, the respondents felt that this was a useful course and appreciated the opportunities that it provided to apply control theory to the solution of real-world, electromechanical control problems. By learning assembly language programming, students were able to understand the underlying function of the microcontroller, and they felt it was a useful learning experience. The results of the survey also suggest that as the course exists right now, there is not enough time spent on modeling and controller design, especially for students with limited prior controls experience. Based on these results, future versions of the course will contain more structured assignments and activities to help students become more confident in these areas.

Bibliography

1. Shoureshi, R. & Meckl, P. H. Control Education in the Era of Microprocessors. in *Proceedings of the 1992 American Control Conference* 455–460 (1992).
2. Meckl, P. H. Teaching Mechanical Engineers about Embedded Programming. in *Proceedings of the ASEE 2003 IL/IN Sectional Conference* (2003).
3. STM32VLDISCOVERY - STMicroelectronics. at <http://www.st.com/internet/evalboard/product/250863.jsp>
4. Keil Embedded Development Tools. at <http://www.keil.com/>
5. Company Profile - ARM. at <http://www.arm.com/about/company-profile/index.php>
6. Enikov, E. T. & Campa, G. USB-powered portable experiment for classical control with MATLAB Real-Time Windows Target. in *119th ASEE Annual Conference and Exposition, June 10, 2012 - June 13, 2012* (American Society for Engineering Education, 2012).

Appendix A: Course Sequence and Laboratory Activities

Session	Topic	Laboratory Activity
1	Introduction to Microprocessors and Microcontrollers; Number Systems, Binary Arithmetic	
2	Digital Logic; Memory and Memory Addressing	
3	Microprocessor Architecture and Internal Operations	
4	Instruction Set Overview, Addressing Modes	PC Familiarization, Software Development Environment
5	Assembler Directives; Branching and Looping	
6	Subroutines & the Stack, Passing Parameters	
7	Peripherals, I/O Interfaces	Assembly Language Programming, Digital I/O
8	Parallel I/O Ports	
9	Serial Communications, UART, ASCII Conversion	
10	Interrupts	
11	Interrupts (cont.); Timers	
12	Mixed-Language Programming	Keyboard & Console I/O Operations, Serial Communications
13	Digital-to-Analog Conversion	
14	Analog-to-Digital Conversion	Interrupts: Application to Frequency Counting
15	Review of Systems Concepts	
16	Review of Classical Controller Design	Mixed-Language Programming
17	Midterm Exam	
18	PID Control	
19	Ziegler-Nichols Tuning & other Tuning Algorithms	Digital-to-Analog and Analog-to-Digital Conversion
20	Discrete Controller Implementation	Digital Controller Implementation
21	Sampling, Sampled Data Systems	
22	Aliasing, Anti-Aliasing Filters	
23	Saturation and Anti-Windup	
24	Digital Filtering	Electromechanical Control Project I
25	A More Detailed Look at Peripherals	
26	Actuators, Stepper Motors	
27	Stepper Motors (cont.)	
28	Stepper Motor Drivers	
29	Optical Encoders	Electromechanical Control Project II
30	Microcomputer Peripherals, Bus Standards	

Electromechanical Control Projects

Engine Cruise Control	Inverted Pendulum	Heating/Cooling System
DC Motor Control	Adaptive Refrigeration Cycle	Active Vibration Control