TEACHING MECHANICAL ENGINEERS ABOUT EMBEDDED PROGRAMMING

Peter H. Meckl[1]

Abstract *? This paper describes an elective course on Microprocessors in Electromechanical Systems, taught in Purdue's School of Mechanical Engineering. The course uses a Motorola 68HC12 microcontroller in a sequence of laboratory exercises to introduce students to modular embedded programming. Students can develop embedded code to control a variety of electromechanical systems, including an air-driven engine, a controllable refrigeration cycle, an inverted pendulum, and an active mass damper applied to a small building model. Some of these hardware projects will be described in more detail.*

INTRODUCTION

For many years, Purdue's School of Mechanical Engineering has been instrumental in making mechanical engineers aware of electronics and control. For almost two decades, the School of ME has offered an elective course in Microprocessors in Electromechanical Systems to expose mechanical engineers to programming and interfacing of microcomputers. In the past year, this course has been significantly updated, replacing an Intel 80C188EB microprocessor with the Motorola 68HC12 microcontroller. This new microcontroller contains on-board FLASH memory, where students can store code that remains viable even after power is turned off. And with sophisticated integrated software tools, FLASH programming can be accomplished with little more than the click of a button.

COURSE OVERVIEW

The current course on Microprocessors in Electromechanical Systems (ME 586) 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, undergraduates are encouraged to participate. Thus, both lectures and laboratory assignments have been designed to provide the necessary background to make sure that both undergraduates and graduate students make similar progress. Supplementary references are provided as necessary.

Since mechanical engineering curricula typically do not include courses on digital logic circuits, assembly language programming, and microprocessor interfacing, this course is designed to introduce these topics to mechanical engineers. In this way, they can develop their expertise for implementing microprocessor-based 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.

Lectures have been designed to introduce students to various basic concepts that are crucial to programming and interfacing with microprocessors. These topics include digital logic, 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, some time is devoted to a discussion of stepper motors and drivers, incremental optical encoders for position measurement, and issues of digital sampling and aliasing.

MOTOROLA 68HC12 MICROCONTROLLER

The laboratory includes 8 stations, each of which consists of a Pentium 4 personal computer, a Motorola 68HC12-based microcontroller board with its own analog I/O capabilities, a proto-board, and a mixed-signal oscilloscope. In addition, a variety of digital logic chips and discrete electronic components are available.

The 68HC12-based microcontroller board is an Adapt912 board purchased from Technological Arts [1], as shown in Fig. 1. It consists of a Motorola 68HC912B32 microcontroller, along with facilities for serial communication via an RS-232 connection, a Background Debug Mode (BDM) connector for downloading and debugging assembly code, and circuitry for stepping up the input to 12 volts to program the FLASH memory. A companion board provides op-amp circuitry to scale, bias, and buffer the eight analog inputs and to low-pass filter the two pulse-width-modulated outputs to generate true analog output signals.

[1] Peter H. Meckl, School of Mechanical Eng., Purdue University, 585 Purdue Mall, West Lafayette, IN 47907-2088, meckl@ecn.purdue.edu.

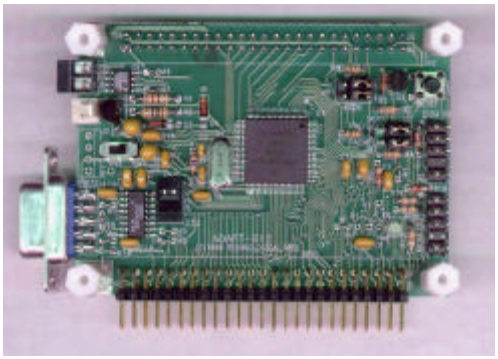

FIGURE 1
TOP VIEW OF THE ADAPT912 MICROCONTROLLER BOARD
(PHOTO COURTESY OF TECHNOLOGICAL ARTS)

The Motorola 68HC12 is a 16-bit microcontroller with 1 KB RAM, 768 bytes EEPROM, and 32 KB FLASH memory on chip. It also contains an 8-channel 10-bit analog-to-digital converter and two 16-bit pulse-width-modulated outputs. In addition, it supports an RS232 serial communication interface and Background Debug Mode (BDM).

The Adapt912 board and associated circuitry has been enclosed in a box with BNC connectors to make it easy to interface external signals with the microcontroller. A top view of the front panel of the box is shown in Fig. 2. Connections for eight analog inputs are on the upper left, connections for two analog outputs are on the upper right. In addition, a DB9 serial connector provides serial I/O with the PC. A BDM cable from P&E Microcomputer Systems [2] provides a parallel interface between the 68HC12 and the PC for program downloading and debugging. Finally, a jack is provided in the back of the box for an interrupt input.

The Motorola 68HC12 microcontroller was chosen instead of other competitive products (from Intel, Infineon, and Microchip) for several important reasons. First and foremost was the small size of the chip itself, no bigger than the size of a thumbnail. This permits the entire board to be no larger than the size of a credit card, making it feasible to integrate a microcontroller into a variety of hardware devices for true embedded control. Motorola microcontrollers use the so-called Princeton architecture, which means that both code and data share the same memory space. This makes it consistent with the ubiquitous Intel-based microprocessors found in most personal computers. Also, Motorola is the leading brand of microcontrollers in many key industries, such as automotive, making it likely that students would find themselves working with a Motorola chip in their future employment. Finally, the Background Debug Mode interface makes it simple to communicate with the microcontroller for downloading code into the FLASH memory and debugging programs once installed in memory.

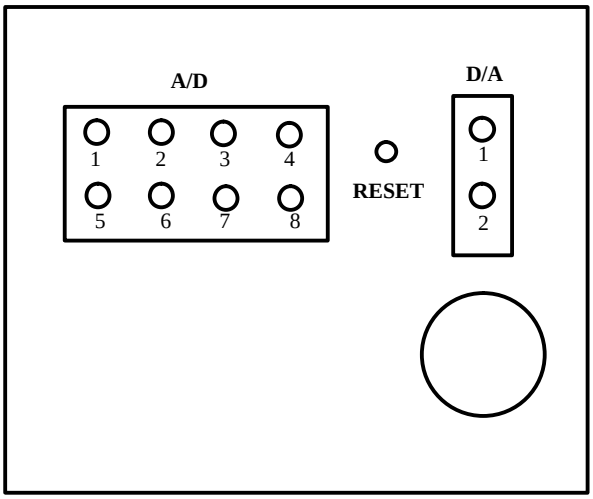

FIGURE 2
FRONT PANEL FOR THE ADAPT912 CIRCUIT BOX.

The 68HC12 microcontroller in particular was chosen based primarily on the 16-bit data wordlength. This allows the microcontroller to be easily programmed in C, the programming language of choice for embedded control applications. A new integrated development software package from IAR Systems [3] permits program development in both assembly and C code on the PC, with seamless connection to the IAR C-SPY debugger for downloading and remote debugging of code running on the 68HC12. This debugger permits true source-level debugging, whether programs are written in assembly or C code. Thus, the capabilities and programming ease of a high-level language like C is coupled with the embedded control environment provided by stand-alone processors.

EMBEDDED CONTROL LABORATORY

The embedded control laboratory complements the lectures by providing hands-on experiences for learning the course concepts. Students have opportunities to actually build circuits and to program microcontrollers to achieve desired control objectives. In the laboratory, students use the PCs for software development and user interfaces for their embedded controllers. They use the 68HC12-based microcontroller board for interrupt-driven programming and implementation of most of their control schemes. With its analog input/output capabilities, the 68HC12 can be used as a stand-alone processor to control a variety of electromechanical systems. The Adapt912 microcontroller board is an excellent vehicle to introduce students to the lowest level of real-time control programming, providing the ability to "burn" program code into FLASH memory for direct control of the microprocessor. As such, it represents

an analog of the type of stand-alone electronic control module found, for example, on automotive engines.

All students perform the first seven labs, which are designed to introduce the fundamentals of microcomputer hardware, software, and interfacing. After that, students are free to select three projects from a list of different electromechanical project setups. These include a two-link robotic manipulator, a dc motor speed control, a heating/cooling system, an inverted pendulum, a four-cylinder air-driven engine, a controllable refrigerator, and an active mass damper.

In the first laboratory assignment, students are acquainted with the hardware and software environment that they will be using for the semester. Even though they have not yet been exposed to assembly language programming, it is essential that students become familiar with the IAR Embedded Workbench (EW) programming environment that they will be using for the rest of the semester. Thus, students are provided with a partial assembly program, with a listing of what the completed code should look like. They then use the editor within IAR EW to finish typing up the assembly code. Once they set the required options for "building the project," they compile and link their program. If any errors occur, they can click on the error in the Message Window, which takes them right back to the editor, with the cursor positioned at the code statement where the error occurred. Once they get an error-free executable file, they can run the program directly, or trace through its execution within IAR C-SPY to debug their logic. All steps of the program development process are integrated within IAR EW as one seamless software package.

Also in the first lab, students are exposed to C programming, the language that will be used for developing most of the higher-level operations of their embedded control code. They learn that the same software development tool can be used for both assembly and C programming, and they can see the actual assembly instructions corresponding to their C programs to appreciate the underlying low-level commands.

The second laboratory assignment focuses on digital logic circuits, both combinational and sequential. It is designed to expose students to the fundamental binary operations that take place in the heart of the microcontroller. First, students are asked to build simple half-adder and full-adder circuits out of TTL logic devices to construct a 4-bit adding machine. They also build a logic circuit that sets several different outputs, depending on whether two binary input numbers are equal, even or odd, or positive or negative. Finally, they use several NOR gates with feedback to construct an SR flip-flop, which they then use to construct a memory register.

Laboratory Assignment 3 introduces students to Motorola CPU12 assembly language programming. During the first week, students look at some simple code, perform a program assembly and disassembly, and become familiar with the operations associated with several instructions using different addressing modes. They use the C-SPY Simulator within the IAR EW environment to examine the microcontroller registers before and after each operation. They are then asked to write a simple program that takes a set of integers and finds the average. This program is to run successfully on the C-SPY Simulator, which emulates the 68HC12 on the PC.

During the second week of Lab 3, students learn about the Adapt912 microcontroller board and its associated memory map. Their task this week is to convert their earlier assembly program to a form that will run on the Adapt912. Most of the required changes have to do with relocating the data and code into appropriate memory locations. Once these changes have been made, students can assemble and link their code using IAR EW as before. Once built, they can use the C-SPY debugger to download the code directly to FLASH on the Adapt912 and then run the program. A final task is to write a program that mimics the digital logic circuit that was written in Lab 2. In this way, it exercises their ability to handle digital I/O with the Adapt912.

Laboratory Assignment 4 focuses on serial communication. The ultimate objective is to link the microcontroller with the PC via the serial line so that programs can exchange information in both directions. This is a valuable contribution, since eventually, students will be programming their microcontroller to perform real-time control while the PC is configured as a user interface to send setpoints and display signals. Initially, students write simple subroutine modules for the 68HC12 that initialize the communication settings (baud rate, data format, etc.) and establish serial I/O. This exercise teaches students about programming on-board peripherals using status and control registers. Their code must check whether data has been received on the serial communication line before reading it in, and also must check that the transmit buffer is empty before sending data out. Once these modules are working, the students generate code that uses specific key presses to increment and decrement a counter, which is displayed on the PC screen via HyperTerminal. This exercise also familiarizes students with conversion from binary to ASCII characters.

Laboratory Assignment 5 exposes students to interrupts and interrupt programming. Students use a clock signal of varying frequencies as an external interrupt input to the microcontroller. They develop an interrupt service routine that, at each cycle of the external clock, increments a counter. An internal timer, set to a specified frequency, generates another interrupt, which instructs the microcontroller to store the counter associated with the external clock signal. The number of counts divided by the time programmed into the timer gives the external clock frequency. This lab illustrates the ability to use external events to alter program execution, and develops the

programming skills necessary to establish interrupt-driven timing for real-time control. It also helps reinforce the notion of modular programming, since the same serial I/O routines from Lab 4 are used to display the counter value.

So far, students have done all their programming in CPU12 assembly language. However, it would be unreasonable to expect them to generate all embedded control code for the projects in assembly code. Therefore, Lab 6 introduces students to mixed-language programming. They essentially create C-callable serial I/O modules that mimic those from Lab 4. Thus, the low-level access of status and control registers is still done in assembly, but the results are passed on to C so that C can be used for higher-level processing. They then redo the up-down counting experiment from Lab 4 in C code. This allows them to compare the resulting assembly code generated after compiling C code with the original assembly code from Lab 4 to gain an appreciation for the efficiency of assembly programming.

Laboratory Assignment 7 introduces students to analog-to-digital (ADC) and digital-to-analog conversion (DAC). After wiring up a simple 8-bit DAC, students use the digital input on the 68HC12 together with the DAC to develop a counting ADC. Once they understand the hardware associated with analog/digital conversion, they are given a set of C-callable assembly routines for the 68HC12 to access its own analog inputs and outputs. These routines then serve as building blocks for writing control code for the electromechanical projects.

ELECTROMECHANICAL PROJECTS

Once all seven of these labs have been completed, students begin their selected projects. Each setup is an electromechanical system, complete with required sensors and actuators. Most of the interfacing circuitry has already been incorporated to allow students to focus on real-time control programming. Each system has inherent nonlinearities that must be addressed during control design and implementation. For each project, students must calibrate the sensors, perform some system identification to obtain approximate system models, design a suitable controller, simulate that controller with their system model in Simulink [4], and finally, implement their controller using the 68HC12 microcontroller. Students use timer interrupts to establish a uniform sample rate at which to run their digital controllers.

A brief description of several of the projects follows. The two-link robotic manipulator has a cylindrical base joint that allows rotation about a vertical axis and a second rotational joint that permits motion in a vertical plane. Both axes are driven by stepper motors. Students must develop the appropriate open-loop commands to the stepper motors to achieve a desired trajectory. In addition, they must coordinate the motions of the two axes to ensure that the desired trajectory is followed accurately.

The dc motor includes an incremental optical encoder that generates a pulse train as function of angular rotation. Students must develop the software to convert this signal to a velocity measurement, and then must develop a suitable control scheme to maintain motor speed in the presence of torque disturbances.

The heating/cooling system represents a simple model of a heating and air-conditioning system that would be used in a commercial building. A resistance heater is used to heat up a small aluminum block, while a small computer fan is used to achieve cooling. Both actuators only work in one direction; the heater only heats, while the fan only cools. Thus, students must develop a strategy that uses the temperature of the block as single input to determine the action of the two different actuators. A block diagram illustrating this control system is shown in Figure 3. Students must achieve desired temperature settings while minimizing energy consumption.

The inverted pendulum is a classic project, consisting of a two-foot rod hinged at the bottom and connected to a movable cart, powered by a dc motor (see Fig. 4). Potentiometers at the hinge and along the track indicate rod angle and cart position. Students must use these two signals to generate an appropriate control strategy for the cart motor that stabilizes the unstable dynamics of the inverted pendulum while ensuring that the cart does not run off the ends of the track.

The four-cylinder engine, shown in Fig. 5, represents a table-top model that operates on shop air. A computer-controlled valve regulates the air pressure to the individual cylinders so as to maintain desired speed. A dc generator, mounted to the engine crankshaft, can be used as a dynamometer to absorb varying loads from the engine. The objective of the project is to adjust the air valve so as to maintain speed in the presence of torque disturbances from the generator. The challenge is the friction associated with the valve, requiring that students add a local feedback loop around the valve itself to ensure reliable operation. A typical engine speed response is shown in Fig. 6. The engine starts at 1000 RPM when a torque disturbance is applied at around 2 sec, which initially slows down the engine. However, within 1 sec, the controller manages to bring the speed back up to 1000 RPM. The same thing happens when the torque disturbance is removed at 6 sec.

The controllable refrigerator is a conventional refrigerator compressor, condenser, and coil, but it uses a computer-operated expansion valve in place of the capillary tube (see Fig. 7). In this way, the flow of coolant can be controlled so as to achieve maximum efficiency while maintaining desired cold temperatures. Several temperature and pressure sensors are distributed throughout the coolant circuit. Students must decide which of the sensors to use to achieve their feedback control strategy.

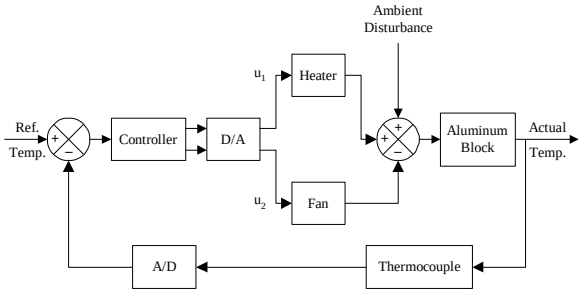

FIGURE 3
BLOCK DIAGRAM OF THE HEATING-COOLING CONTROL SYSTEM.

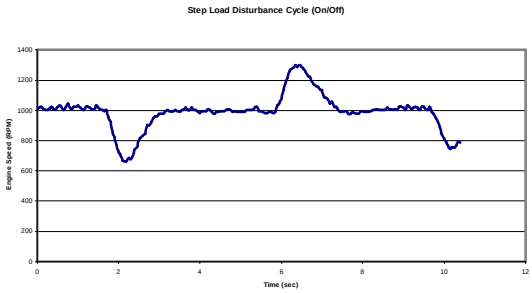

FIGURE 6
ENGINE SPEED RESPONSE FOR 4-CYLINDER ENGINE WHEN TORQUE DISTURBANCE IS APPLIED AT 2 SEC AND REMOVED AT 6 SEC.

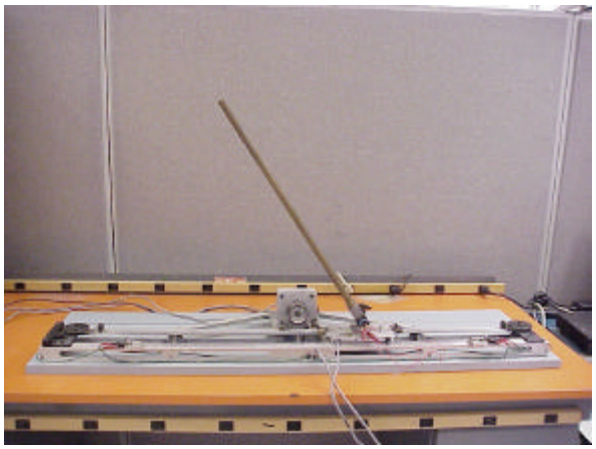

FIGURE 4
PHOTOGRAPH OF THE INVERTED PENDULUM SYSTEM.

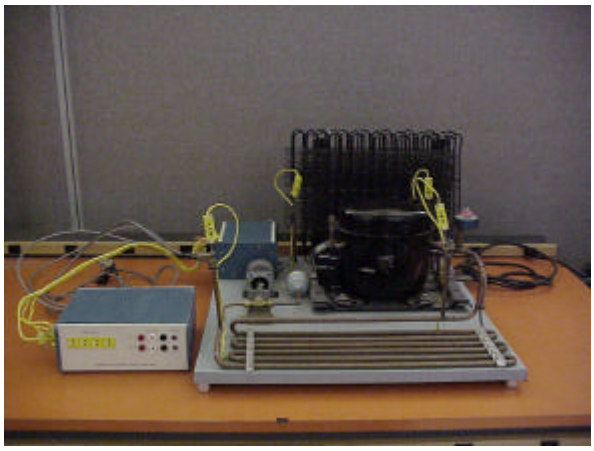

FIGURE 7
PHOTOGRAPH OF THE CONTROLLABLE REFRIGERATOR SYSTEM.

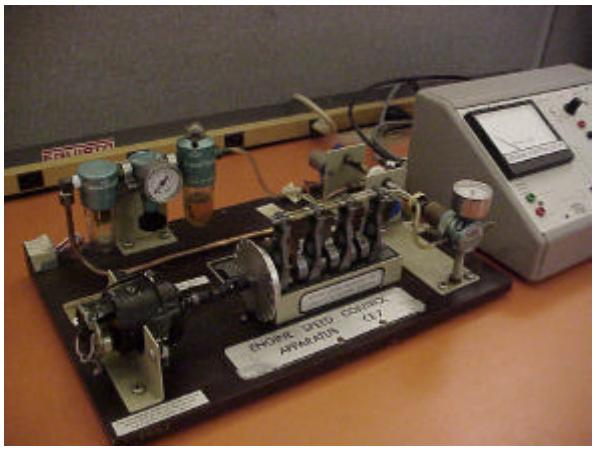

FIGURE 5
PHOTOGRAPH OF THE 4-CYLINDER AIR-DRIVEN ENGINE SYSTEM.

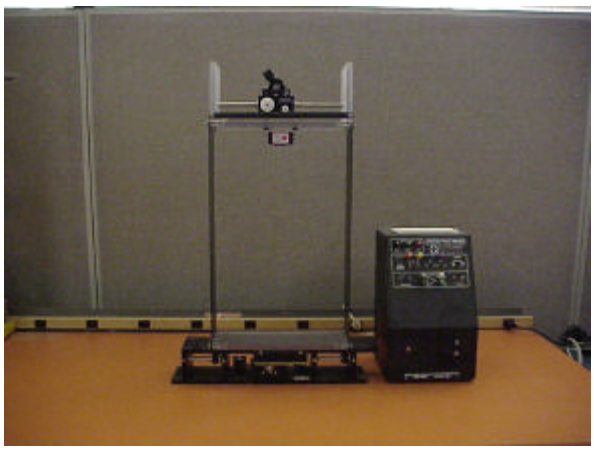

FIGURE 8
PHOTOGRAPH OF THE ACTIVE MASS DAMPER SYSTEM.

The active mass damper, shown in Fig. 8, is designed to emulate the vibration absorbing systems used in some high-rise buildings. It consists of a pair of thin metal plates that represent tall "building" walls and a powered mass that can traverse across the top floor of the "building." An accelerometer measures the vibration at the top of the "building" when the bottom floor is excited by a shaker that simulates an earthquake. The control objective is to use the accelerometer measurement to control the motion of the mass so that it moves opposite the motion of the "building" and effectively damps out the vibration.

CONCLUSIONS

A microprocessor course for mechanical engineers at Purdue University has been described. It emphasizes programming and interfacing of a microcontroller, with ultimate application to several electromechanical control projects. A 68HC12 microcontroller board is used for all embedded programming. An integrated development software package makes it easy to write and debug programs in both assembly and C code. Students are encouraged to exercise modular programming by preparing low-level serial I/O routines and other utility programs that are used for other programs later. The course culminates in a series of projects that allow students to design controllers for various electromechanical systems to achieve desired control objectives. These projects include an inverted pendulum, heating/cooling system, air-driven four-cylinder engine, controllable refrigerator, and active mass damper.

ACKNOWLEDGMENT

The author gratefully acknowledges IAR Systems (San Francisco, CA) for donating the IAR Embedded Workbench and C-SPY software at cost.

REFERENCES

[1] Technological Arts, Toronto, Ontario, Canada, http://www.technologicalarts.com/.

[2] P & E Microcomputer Systems, Inc., Woburn, MA, http://www.pemicro.com/

[3] IAR Systems, San Francisco, CA, http://www.iar.com/.

[4] Simulink, The MathWorks, Inc., Cambridge, MA, http://www.mathworks.com/.