AC 2011-192: A LABORATORY PROJECT INTRODUCING BASIC MICROPROCESSOR HARDWARE AND SOFTWARE FOR AN INTRODUCTORY UNDERGRADUATE ECE CLASS FOR NON-MAJORS

Brennan T. Ashton, Worcester Polytechnic Institute

Sophomore in Electrical and Computer Engineering at Worcester Polytechnic Institute.

Paul Malmsten, Worcester Polytechnic Institute
Gautam Vallabha, MathWorks

Gautam K. Vallabha received the B.S. (1995) degree in Electrical and Computer Engineering from Carnegie Mellon University, Pittsburgh, U.S.A, and the Ph.D. (2003) degree in Complex Systems and Brain Sciences from Florida Atlantic University, Boca Raton, U.S.A. From 2003 to 2007, he was a postdoctoral researcher at the Center for the Neural Basis of Cognition at Carnegie Mellon University and at the Department of Psychology at Stanford University, specializing in neural network models of speech perception and language learning. He is currently employed at MathWorks (Natick, U.S.A).

Sergey N. Makarov, Worcester Polytechnic Institute

Submitting author: Sergey N. Makarov earned his B.S./M.S./Ph.D./Dr. Sci. degrees at the State University St. Petersburg (Leningrad), Russian Federation Faculty of Mathematics and Mechanics. Dr. Makarov joined Institute of Mathematics and Mechanics at State St. Petersburg University in 1986 as a researcher and then joined the Faculty of State St. Petersburg University where he became a full professor in 1996. In 2000 he joined the Faculty of Department of Electrical and Computer Engineering at Worcester Polytechnic Institute, MA. His current teaching interests include fundamental ECE classes.

A laboratory project introducing basic microprocessor hardware and software for an introductory undergraduate ECE class for non-majors

Most electrical and computer engineering departments in the United States and abroad typically offer a fundamental one or two semester course in ECE for non-major students. Sometimes, this course is offered to both majors and non-majors. In general, it is a very difficult task to teach complex electrical engineering concepts, including circuit theory, semiconductor fundamentals and digital fundamentals in one course. Therefore, the intro class for non-majors (or for both majors and non-majors) is frequently devoted only to circuit fundamentals.

This scenario creates a visible dissatisfaction, especially among Mechanical Engineering (ME) majors, who wish to be exposed to modern microprocessor basics as early as possible, even in their first and perhaps only ECE class. The same trend is observed for other ECE majors when only a generalized introductory course is offered. The exposure to microprocessors is not only beneficial for the intro class, but it may also stimulate further long-term interest in ECE and Robotics. On the other hand, the systematic study of digital fundamentals in the intro class may not be possible as it would require a significant extension of an already tight syllabus. What could be done to resolve this dilemma?

This paper reports on our pedagogic method of introducing the microprocessor material on one particular class laboratory in an engaging yet technically correct way that is particularly appealing to non-major students. The laboratory syllabus includes the following steps:

1. The base circuit is a single power MOSFET connected in series with a small DC motor and a protection diode.
2. A PIC microcontroller is the source of control signals for the power MOSFET.
3. Students are asked to demonstrate basic open-loop control of the attached motor. By using Pulse Width Modulation (PWM), they should be able to ramp up the average voltage applied to the motor, hold it constant, and ramp it down.
4. Using high-level MATLAB® statements, students describe the behavior required of their microcontroller, including in-hardware generation of a PWM signal and ramping. Students then compile and download these instructions to an appropriate PIC microcontroller using our custom MATLAB® toolbox.
5. Students place the microcontroller in their circuit and demonstrate the expected motor control behavior.

The heart of this laboratory is a custom MATLAB® toolbox that allows a student to write a MATLAB® program and execute it on an inexpensive Microchip PIC microcontroller. We shall describe our experience with the toolbox in an introductory course offered during Fall 2010/Spring 2011 and the corresponding results. In addition, we will discuss the basic design of the toolbox (which will be made freely available) and how it may be used for more advanced courses as well.

The following topics will be discussed:

1. Case Study: Open-Loop Motor Control
2. Overview of Toolbox Design
3. Application of Toolbox
4. Extending the Toolbox

1. Case Study: Open-Loop Motor Control

To help bridge the gap between mechanical engineering and electrical engineering, the basics of open-loop motor control, in particular the H-Bridge, were discussed during the fall 2010 introductory ECE course. Two major topics were introduced in a series of lectures and laboratories: the steady-state DC motor model and electronic switching with transistors.

DC Motor Model

To begin any discussion of DC motor theory, an understanding of the Lorentz force in necessary. Fundamental in this discussion is the relationship between current moving through a conductive armature in a magnetic field and the resulting linear acceleration of the armature, seen in Fig 1 and defined by:

$$F = lI_aB \qquad (1)$$

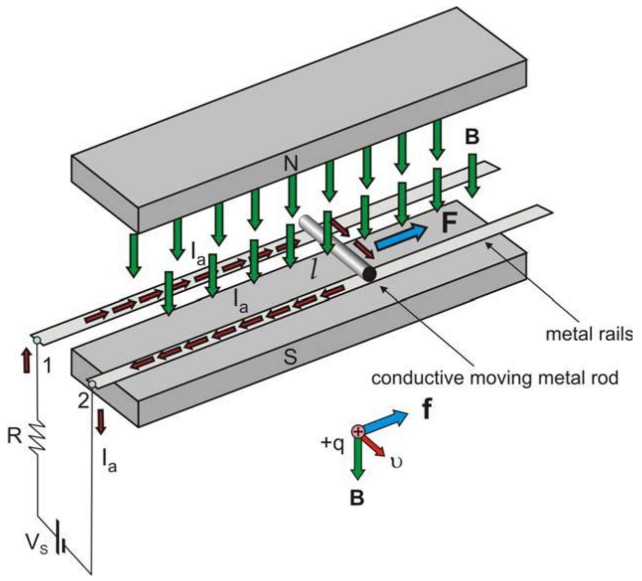


Fig 1. The basic linear accelerator using Lorentz force

This is extended to discuss how applying a load to the current path can create braking effect on the armature, an effect that can be particularly applicable to controlling systems from a mechanical perspective.

Once the students have understood how the Lorentz force can be used to move armatures linearly, its application in a rotational armature in the form of a brushed DC motor, such as the one in Fig 2, is discussed.

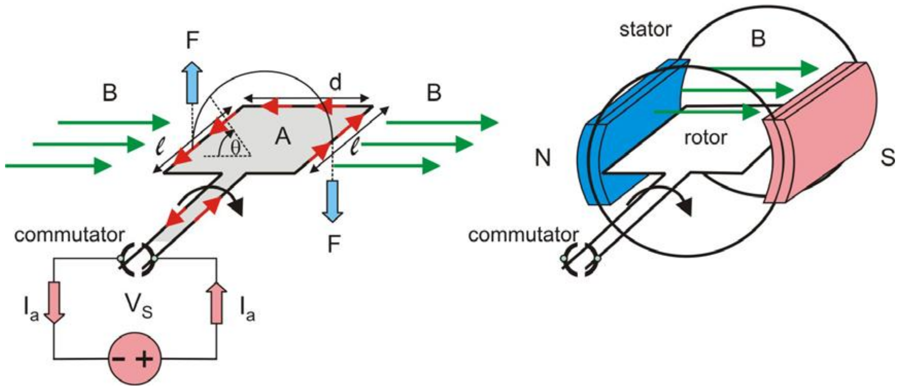


Fig 2. The basic brushed DC motor

At this point, the relationship between current and motor torque becomes apparent, especially for mechanical engineering majors who may need to include motor selection in design considerations. By using the a DC motor data sheet along with torque equations, students are able to develop torque and efficiency plots for the motors such as the one shown in Figure 3.

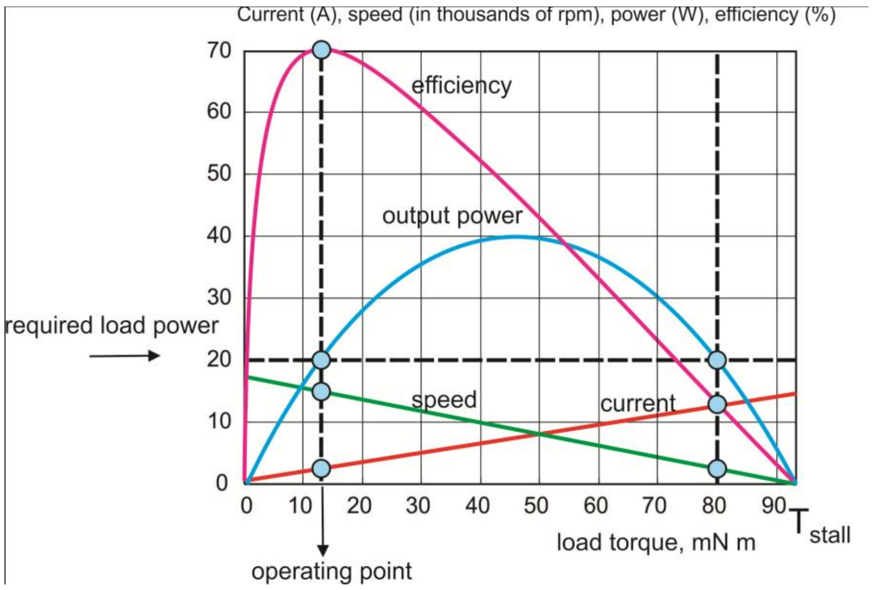


Fig 3. A set of motor torque curves

With a basic understanding of the physics and mechanics of DC motors, it is important that a portion of the class is devoted to controlling motors using an H-Bridge. To introduce this, the basics of electronic switching using transistors must be discussed.

Electronic Switching for H-Bridge

As entire classes are devoted to the study of transistors and their properties, the study of transistors in an introductory course is limited to the simplified field-effect transistor model, in which transistors conduct or do not conduct depending on discrete voltage thresholds as applied at the transistor's gate. From this model, the half H-Bridge is first introduced to form the motor control circuit, in Figure 4, which allows for three motor states: forward, free, and braking.

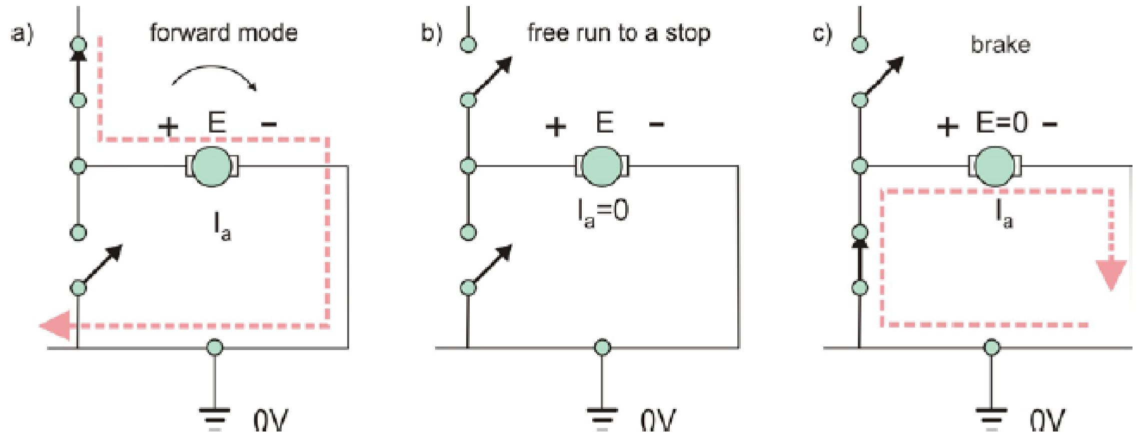


Fig 4. Half H-Bridge motor control

Students are then able to develop the full H-Bridge, in Figure 5, which adds the capability to run a motor in reverse.

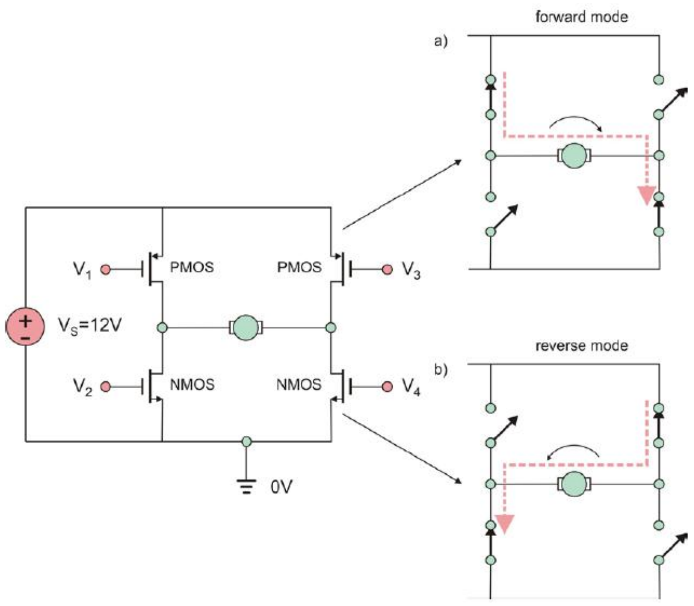


Fig 5. Full H-Bridge motor controller

In the laboratory, students are able to construct and observe the full control of the H-Bridge by supplying control signals from a computer using a National Instruments DAQ. At this point, students are encouraged to add some digital logic to take the 16 possible combinations of control signals, some of which could damage the controller, and restrict them to the four useful combinations using two control lines for input.

With direction and braking control established, the next question is how can the speed of the motor could be controlled. At this time, speed control using Pulse Width Modulation (PWM) is introduced. By switching the motor on and off at a varying duty cycle, the effective voltage applied to the motor can be controlled. Students extend their circuit by adding an additional transistor to their H-Bridge, as in Figure 6, and demonstrate voltage control of a motor using a PWM signal generator.

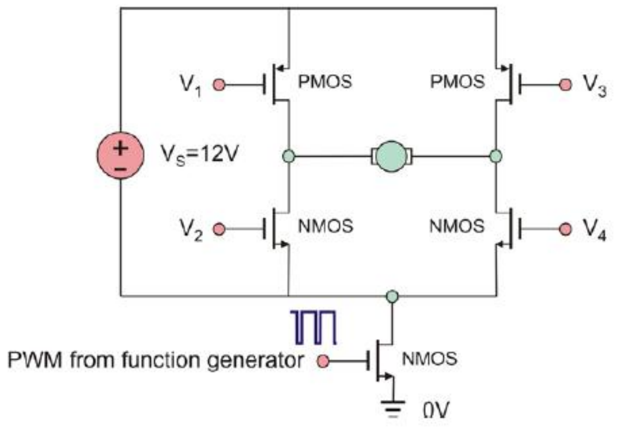


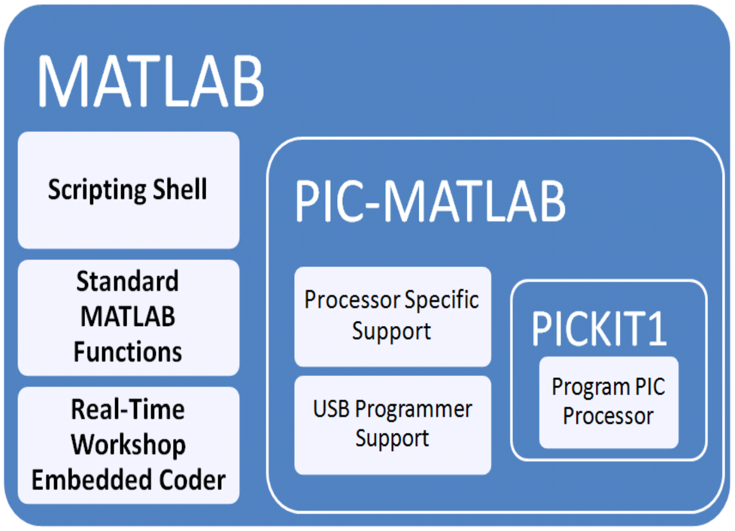
Fig 6. H-Bridge with voltage control.

At this point, students have developed an understanding of what is necessary to control a motor. However, the source of these control signals in a practical application remains unclear. This leads naturally to the discussion of microcontrollers which are a common source of such control signals. Unfortunately, microcontrollers can be quite complex, and using them effectively often requires multiple courses worth of discussion. To reduce the required knowledge background to something suitable for an introductory course, we have developed a toolbox for MATLAB® which facilitates simple microcontroller development with little if any prior programming experience required.

2. Overview of Toolbox Design

In an educational setting, simplicity and convenience are paramount. Based on these objectives, we developed a toolbox which automates the entire process of transforming and downloading a program for execution on a PIC microcontroller. Employing this toolbox in a laboratory scenario only requires the following actions from a student participant:

1. Add the toolbox to MATLAB's execution path (so that it can be executed from the MATLAB® command interpreter).
2. Write an Embedded MATLAB® function, named matlab_main.m, that uses a simple set of functions for pulse width modulation, digital input and output control, and analog to digital conversion. ("Embedded MATLAB" is a subset of the MATLAB® language and libraries that can be converted to ANSI C code using MathWorks' Real-Time Workshop).
3. Execute the toolbox build function.


MATLAB
Scripting Shell
Standard MATLAB Functions
Real-Time Workshop Embedded Coder
PIC-MATLAB
Processor Specific Support
USB Programmer Support
PICKIT1
Program PIC Processor
Fig 7. Toolbox Architecture

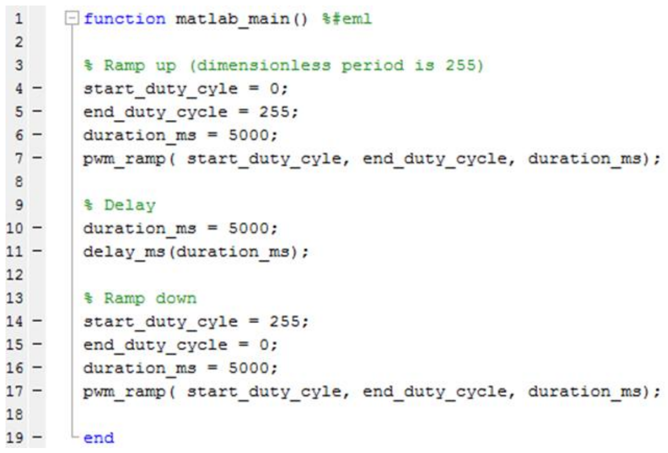
With a PICKit 1 board connected to the host computer with an appropriate PIC installed, the toolbox fully automates the following processes:

1. The MATLAB® source code in matlab_main.m is transformed into standard C code using the Real-Time Workshop.
2. The generated C code and the toolbox library routines are compiled into instructions for the target PIC device.
3. The resulting instructions are written to the program memory of the target PIC device through the PIC Kit 1 programmer board via USB.

Once complete, the target PIC may be used directly on the PIC Kit 1 development board or moved to a custom circuit.

3. Application of the Toolbox

This toolbox was first used to extend the H-Bridge laboratory by introducing a microcontroller with a programmed speed profile to provide control signals. To begin, students brought their previously built circuit to a lab bench with the toolbox installed. After discussing the structure of an example voltage profile MATLAB script, shown in Figure 8, the students customized the script to a profile of their choosing, such as the one in Figure 9. In a matter of minutes, students are able to demonstrate their new voltage profile in a working circuit.



```
function matlab_main() %#eml

% Ramp up (dimensionless period is 255)
start_duty_cyle = 0;
end_duty_cycle = 255;
duration_ms = 5000;
pwm_ramp( start_duty_cyle, end_duty_cycle, duration_ms);

% Delay
duration_ms = 5000;
delay_ms(duration_ms);

% Ramp down
start_duty_cyle = 255;
end_duty_cycle = 0;
duration_ms = 5000;
pwm_ramp( start_duty_cyle, end_duty_cycle, duration_ms);

end
```

Fig 8. Sample motor voltage profile script

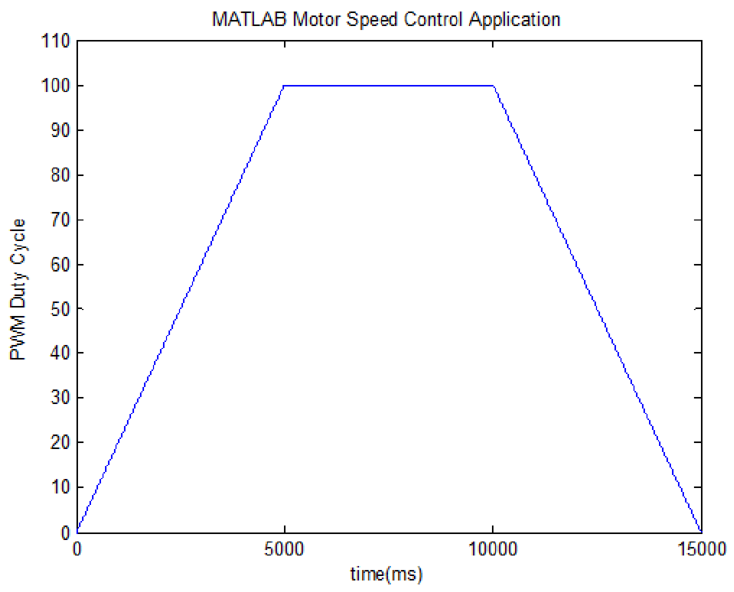


Fig 9. Custom motor voltage profile

By monitoring the PWM output from the microcontroller with an oscilloscope, students are able to watch the PWM duty cycle change in real time according to their profile.

One may notice that the sample program above does not appear to utilize the full features of an H-Bridge; it only demonstrates the generation of PWM signals. This was done to only simplify the original laboratory circuit used during its initial testing. Appropriate routines for digital output are included in the toolbox, and their usage to control the direction or braking of a motor with an H-Bridge would only require a handful of additional lines of code.

This initial laboratory was a very simple demonstration of the toolbox's flexibility; it only just begins to tap the capability of the toolbox. We are devising additional laboratories to introduce more complex concepts, such as audio storage and retrieval, and harness the full potential of the toolbox as a teaching tool. Even with our simple initial laboratory, however, our students were able to interact with a digital system and gain a concrete understanding of basic digital concepts in a short time.

Total 23 non-major students (volunteers) participated in this experimental laboratory during the fall of 2010. They were able to accomplish the MATLAB® task and demonstrated an interest in the subject matter.

4. Extending the Toolbox

In general, any valid Embedded MATLAB® commands and library functions may be run on a target device, subject only to its code memory limitations. Consequently, custom MATLAB® functions or libraries can be easily added without modifying the toolbox.

However, for laboratory scenarios which require fine-grained control over the code executing on the target processor, the toolbox includes several custom C library routines such as a PWM ramp function. This allows students to avoid time-consuming debugging during a laboratory and focus on implementing higher-level behaviors.

The toolbox is also portable to a variety of 8 bit PIC processors beyond the two that we initially experimented with (the PIC12F675 and PIC16F684). In this manner, the toolbox accommodates the goals of a laboratory without unduly restricting them.

Additionally, the toolbox could be extended to provide basic emulation of a target device. Although likely out of scope for an introductory ECE course, such emulation would simplify the development of advanced concepts like closed-loop control systems, as it would allow for verification of expected behavior entirely within MATLAB® before deployment to hardware.

Conclusions

By providing a practical toolbox for MATLAB® which enables freshman ECE students to describe high-level behaviors of a microcontroller, we have enhanced the careful balance between theory and pragmatism which is coveted in an introductory course. In this manner, students not only learn the theory behind analog circuit components and their associated uses, but also how to interface them with a common microcontroller and achieve a remarkable level of flexibility in a laboratory setting. During the laboratory all students were successful in completing the project , and many were interested in what was happening in the background. Some even chose to go beyond the requirements and create various other more complex PWM profiles and were able to do with ease after spending only a short period of time with the toolbox. In addition to being an tool for stimulating interest in ECE studies the toolbox also encourages further interest in learning MATLAB® (and the C language) by all engineers. The development and use of this toolbox will be continued with a larger class size starting in the Fall of 2011 with possible applications including an audio recording device.

References

[1] MATLAB Real-Time Workshop, http://www.mathworks.com/products/rtwembedded/
[2] MATLAB Embedded C, http://www.mathworks.com/products/featured/embeddedmatlab/
[3] Microchip PICKIT1, ww1.microchip.com/downloads/en/DeviceDoc/40051D.pdf