Expanding the Options for a First-Year Student Design Experience – An Improved Microcontroller for Mobile Robotics

Jeffery P. Radigan, James M. Beams, Richard J. Freuler, Craig E. Morin, Matthew S. Gates, Jeffrey J. McCune, Andrew J. O'Brien, Joanne E. DeGroat, and John T. Demel

College of Engineering, The Ohio State University

Abstract

In order to meet the rising demands of both education and logistical feasibility when using robotics as a design tool, a research group at The Ohio State University is design and testing a new micro controller for use in mobile robotics. The motivation for this microcontroller design came out of a need to give better support and flexibility to the students when building their robots. To accommodate multiple situations, the microcontroller system is comprised of one core controller attached to function specific modules via an inter integrated circuit (I^2C) bus. The core consists of a microprocessor connected to memory and the serial, I^2C, and USB communication interfaces along with a LCD output screen. Motor control, digital and analog input and output, additional memory and other application-specific modules are connected to the core controller to expand its functionality. The software user interface is designed with the same modular approach. A robust integrated development environment provides editing, version tracking, and testing capabilities such as breakpoints and memory management. Testing of the prototype will take place during the 2003-2004 academic year with the finished controllers available beginning in the 2004-2005 academic year. This paper describes the program requirements, research, design, and testing of this controller, as well as the motivations for the project and its diverse team structure.

1.0 Introduction

Over the last year, a group of Ohio State students and faculty have been designing a new microcontroller for use in the Fundamentals of Engineering for Honors (FEH) Program. The goal of this project is to design a controller that can be modified and expanded to suit the needs of many different design teams and mobile robotic applications. This paper describes the program requirements, research, design, and testing of this controller, as well as the motivations for the project and its diverse team structure.

2.0 Background

During the past ten years, The Ohio State University's College of Engineering has been aggressively evolving a traditional first-year engineering course sequence into a dual track program that retains the essential parts of those traditional courses but add in hands-on laboratory experiences that lead to design/build projects[1]. Engineering is now “up-front” and “hands-on”, and the approach has had a noticeable, positive effect on student retention[2]. A major element in this effort has been the development of a first-year engineering program with a track for honors students, the Fundamentals of Engineering for Honors (FEH) sequence, a tightly coupled three-course sequence with each course lasting the full 10 weeks of an academic quarter.

The first FEH course is named Engineering H191 (or ENG H191) and offers each student a solid foundation in the fundamentals of engineering graphics and CAD. The second course, ENG H192, presents an introduction to computer programming with the C/C++ and MATLAB languages, and engineering problem solving involving computer programs and computer tools. Both courses have hands-on lab experiences designed to further explore the engineering disciplines, and both have a mini-design/build project usually carried out by 2-person teams over a one-week period at the end of the academic quarter.

The last course in the FEH sequence is the Engineering Fundamentals and Laboratory 3, now called ENG H193[3]. Prior to taking this course, the students will also have completed as a part of the FEH program two math courses and two physics courses, all of which are coordinated with the engineering courses. As a culminating course for first-year engineering honors students, the ENG H193 course focuses primarily on the planning, execution, management, documentation, and presentation of an engineering design/build project.

The ENG H193 design project is a focal point for the FEH program. In many respects, this freshman design project course is comparable to a junior level or senior "capstone" design course in which a student might participate as part of the requirements for his chosen engineering discipline. A major difference is that the first-year ENG H193 course teaches the various planning, management,[4] documentation,[5] and presentation aspects of a design project, whereas many senior level design projects focus on the specific design problem alone, assuming some prior instruction in or knowledge of what is needed for a complete and successful engineering project.

This design project involves all aspects of planning, designing, building, testing, documenting, and demonstrating an autonomous robot that has to perform prescribed tasks within a specified time limit while operating over a specially constructed course or track. The format of the demonstration is a competition or tournament in which a champion robot is determined. This is intended to represent a real process of choosing a potential prototype for a real solution to a problem presented to a number of different competing engineering groups. In designing and building the robots, the students have to make use of the graphics, the computer programming, the engineering problem solving, the hands-on labs, the physics, and the mathematics of the previous academic quarters. Working in teams of three or four, the students are required to demonstrate and present the results of their efforts by submitting progress reports, participating in performance reviews, writing a formal project report, and making an oral presentation about their project.

The controller currently used for this robot design project is the Handy Board controller developed at the MIT Media Lab by Fred G. Martin.[6] Designed for experimental mobile robotics work, this popular Motorola 68HC11-based controller board has a variety of digital and analog input ports for interfacing with various sensors and special ports for controlling infrared transmission and reception, up to four DC motors, and up to two servos. More recently, an expansion board has been developed to extend the number of sensor ports and interfacing capabilities. The Handy Board includes only 32K of battery-backed static RAM, a connector system that allows active sensors to be individually plugged into the board, a small LCD screen, and an integrated, rechargeable battery pack.

The platform is supported with a nearly complete subset of the C programming language in an interactive Windows-based environment called Interactive C (IC).[7] A useful feature of IC is its virtual machine approach to executing programs. Most embedded systems rely on an edit-compile-link-download cycle. In contrast, IC provides a virtual machine that runs on the 68HC11 and interprets pseudo-code (called "p-code") that is produced by the compiler. This approach is similar to that employed by the Java Virtual Machine.[8] In exchange for a performance penalty because of the interpreted p-code, the virtual machine approach does provide two benefits that are especially valuable in student learning context:

1 Interpreted execution – Allows run-time error checking. Like Java, IC performs array bounds checking, as well as trapping other errors, at run-time to protect against common programming mistakes made by beginners. Another benefit of this approach is incremental development. Students can enter single lines or blocks of code into an interaction window (or console) on the host PC. The code snippets are then compiled and sent to the Handy Board, evaluated, and results returned.

2 Multi-tasking – Multi-tasking allows students to create different processes for different types of robot operations (e.g., motor control, sensing, navigation algorithms, etc.).

The Handy Board is frequently used to run robot design courses and competitions at the university and high school level, build robots for fun, and control industrial devices. Its popularity has made supporting the controller in the ENG H193 course easier because of the good availability of boards and components from vendors and the freely available contributed software from a Handy Board web site.[9] After nearly ten years of experience in using the Handy Board in the FEH program, the FEH teaching assistants approached the faculty with suggestions and plans for a new robotics microcontroller, called Project193.

The Fundamentals of Engineering for Honors program plans to introduce this new microcontroller as a replacement to the current Handy Board robotics controller. Increased speed, memory, and expandability allows for more unique designs and solutions to any given problem.

In order for a newly designed controller to integrate well with the current FEH program, it must be able to provide functionality very similar to that of the current Handy Board and at the same time offer new, enhanced capabilities. The Project193 robotics microcontroller is thus a device similar in size to the Handy Board. All input and output is available at the top of the board. It will also feature a top-mounted LCD display. The software students will use to interface with the controller minimally will have the look and feel similar to Interactive C.

3.0 Rationale for a New Microcontroller

The rationale for a new microcontroller arises first from considerations in a few key areas, including controller hardware maintenance, ease of connecting a personal computer to a controller for downloading software, improving battery life and providing battery options, and offer ease of extension and expansion.

As the FEH program has expanded, maintenance of the increasing number of Handy Boards has become a greater task each year. The Project193 design is attempting to produce a controller that will require less maintenance and intervention by a trained staff member. When a component fails on the Handy Board, it brings a robot project team to a halt while a staff member debugs and repairs the board. There are some major factors that contribute to the recurring maintenance of the Handy Board, which can be alleviated by an improved design. Most times this component failure occurs when the warnings about proper care and handling of the Handy Board go unheeded. Problems with some of the onboard circuits, such as motor chip failure, commonly occur.

RS-232 serial ports are slowly being phased out as a default port on newer PCs, including most laptops. In order to keep in step with this change, new interfaces such as USB must be supported. The introduction of a front-mounted USB panel on newer systems also makes it easier for students to connect or disconnect their devices, such as a link to a robotics controller, to or from the PC

Battery life is a problem that contributes to many robot teams' downtime, typically near the end of the robot design project when the controller is heavily used. Teams must be able to replace the rechargeable controller battery with a standard battery at these times. The ability to easily remove the battery would also allow the support staff to charge a battery pack more carefully. The FEH experience has been that several of the battery packs are ruined each year when the charging system is inattentively left in the wrong charging mode. Being able to replace the rechargeable battery pack with a standard alkaline pack on a temporary basis should result in fewer battery failures each year.

Battery monitoring circuitry that allows the student to check his or her battery status contributes to a more proactive recharge and/or replacement program. During competition, when a robot has run many times and its batteries are low, a controller board reset may occur when the robot's motors and other actuators require more current than can be supplied. This typically results in the robot stopping in the middle of a task, and the robot earning fewer competition points than it possibly could have without this problem. Offloading the motor drive power to a separate battery results in no controller resets while the power is low and motors are demanding more than can be supplied. The robot then has an opportunity to react in a proficient manner to a low battery condition.

Limited expandability results in students typically having very similar designs. The availability of more expansion to meet any student's needs results in a more unique solution.

Due to these drawbacks, there are opportunities for an improvement in the student robotics experience. Failures in the hardware provided to the student typically detract from a positive learning experience. As these failures are reduced, the robot project team can then focus primarily on the given task.

Before starting any design of a new controller, many different existing controllers and processors were examined to replace the Handy Board. Initially, there was no commitment to designing a new controller. Many different controllers were compared against the specifications that were set. Every controller that was found did not meet the specs and was not close enough to be adaptable. For example, the popular BASIC Stamp modules do not have the ability to control motors out-of-box. It also did not have the significant speed and memory increase that was deemed necessary. An addition of a daughter board to add more functionality would push the cost well above the cost of the Handy Board. Other boards, more similar to the Handy Board, were examined; most boards were prohibitively expensive, despite having the speed, memory, and I/O capabilities meeting or exceed the required specifications.

4.0 Taking a Team-Oriented, Multi-Level Approach

4.1 Brief Project History

Beginning in the early fall of 2002 three of the six eventual team members began to discuss ideas of a project to build a new controller to replace the Handy Board being used by the FEH program. This discussion quickly snowballed and a number of meetings were held to discuss the feasibility and determine the constraints of this project. A set of design parameters was produced, and a presentation was given to the FEH faculty to explain the idea in some detail. After a lengthy discussion, the idea gained acceptance and a timeline was created with a preliminary goal to have a new controller by the beginning of the 2005-2006 academic year. In order to meet this goal, the team was expanded from the original three members to a total of six.

During November and December of 2002 the preliminary design was begun and roles of team members defined. Each member began brainstorming and exploring the various options for hardware or software in their area. As the team began to evaluate more and more options, there was a decision on the part of the team to assign roles to each team member and begin prototyping the various parts of the project. A more in-depth look at the team organizational structure can be found in section 4.2.

Hardware and software development began in December 2002 and continued through the rest of the 2002-2003 academic year. During this time the team facilitated the construction of a wire wrapped prototype board to ensure that the planned configuration of components would work on printed circuit board. During this time some members of the software and hardware groups worked together to gain basic functionality on the main board, while others worked independently on aspects of the project that would come later, such as the integrated development environment and the add-on modules to the main board. Working in parallel was an instrumental part in making the team's goals a reality.

In the summer of 2003 it was decided to move to a printed circuit board prototype as the wire wrapped prototype had shown that the individual components of the board would work together. The first of the two main board prototype set was printed in July 2003 with the second of the set printed in August of 2003. Since that time the team has continued testing these prototypes and made updates and improvements as necessary.

4.2 Team Composition

From the start of the project, each team member has brought a different area of expertise to the project. With backgrounds as varied and diverse as graduate students in electrical engineering, undergraduate students in both electrical and computer engineering and computer science and engineering, industry employees, and current faculty and staff, the team has relied on each other to work in parallel using the individual strengths of the team members to advance the project. As varied as the backgrounds of the team members are, their motivations for helping with the project are just as diverse. Undergraduate students are able to count the work accomplished as credit for their senior design project. One of the graduate students is using this work as his research thesis while the other recently graduated and is continuing to work the project to see it though to completion. The team's industry professional heard about the project at its inception and committed to being a part of it without any form of payment, class or monetary. The greatest motivation of each team member is his or her commitment to the team and seeing the project succeed to benefit the FEH program.

4.3 Hardware Levels

4.3.1 Components

Hardware of the controller is divided into two sections: the main controller and peripheral modules. The main controller contains the central processor, system and user memory, and I/O interfaces between the computer, user, and peripherals. Peripheral modules interact with the main controller and may be used to complete specific tasks, such as motor control, digital and analog input and output, and data logging.

The main controller is divided into two boards, known as the top and bottom board. This division protects the components on the bottom board from contact with external objects. This prevents failure of the more expensive and difficult to replace processor and memory devices.

The division also allows the board to achieve a form factor roughly the same size as the Handy Board, which is typically the largest single device on a FEH student's robot. The top board contains all interfaces and associated logic for connections to the PC and peripheral boards along with any additional I/O the main board supplies. The top and bottom board are connected via a 40-pin connector similar to a standard IDE connector. The option of connecting the top and bottom boards via a short 40-pin ribbon cable is also available.

The bottom board contains an Intel 196-series microcontroller, 512K of FLASH memory, 128K of static RAM, and assorted support logic components. Memory is addressed via a 16-bit wide data bus. A programming header is available in case the kernel stored or the onboard FLASH becomes corrupt or improperly programmed. This connection is interfaced via a separate programming module controlled by a PIC microcontroller. The use of this programmer is expected to be very minimal and only in emergency situations since the board's kernel is both protected and not overwritten during normal use.

The top board, otherwise known as the interface board, contains the necessary voltage regulation, a RS-232 serial port, USB 1.2 slave interface, I^2C master controller, LCD module, and assorted I/O to the main CPU. RS-232 remains on this controller because even though it is a dying protocol in the PC world, a strong presence remains in the microcontroller and robotics world. RS-232 is also natively supported on the Intel microcontroller. Other ports not used for the address or data bus on the Intel microcontroller are used as additional digital inputs. The microcontroller also has available a 4-channel A/D converter and pulse-width generation on multiple digital outputs.

A graphical LCD is connected to the top board allowing students to display useful textual information. A graphical LCD is chosen over a standard textual LCD since it may also be reconfigured to display useful graphics, charts, and also allows for variable font sizes.

One of the most important interfaces of the top board is the I^2C bus interface. This interface acts as a main hub for peripheral connections. I^2C was chosen since these peripherals may be any standard I^2C chip available on the market, or with the use of an I^2C-enabled Microchip PIC microcontroller the interface possibilities are nearly endless. Most of the input and output for the controller was attached via modules on the I^2C bus. A few basic modules were initially designed using Microchip PIC microcontrollers to achieve the functionality of the Handy Board: A low-current motor controller, a high-current motor controller, and a servo controller. Other basic modules, such as digital I/O, analog to digital converters, and I^2C hubs, were available in single chip packages from Philips and other semiconductor manufacturers. Many other module designs were discussed, such as data-log memories, IR transceivers, stepper motor controllers, basic waveform generators, relay or MOSFET controllers, LCD displays, and keypads. Another I^2C peripheral included on all top boards is the LM80 voltage and temperature monitor. This microchip handles the monitoring of board temperature along with vital voltages. One planned use is so that students can keep better information as to the charge level of their batteries.

Even though the controller was originally designed for robotic applications, a large assortment of modules made the system reconfigurable for almost any application. The base system supplied to the FEH program consists of the core controller, a motor, and a servo module; this configuration mimics the Handy Board's functionality.

4.3.2 Board Design

Design of these boards began as simple concept schematic drawings. Once a CPU and memory requirements were determined, the main components around the CPU-memory core were selected. This involved looking for components that fit timing, voltage, and spacing requirements.

Determining peripherals for the top board required finding solutions that operate on a 5-volt power supply or are at least 5-volt tolerant. The peripherals must also support Intel (8080) timing. The USB and I^2C products selected operate on a 3.3-volt supply but are both 5-volt tolerant. Use of only 8-bit peripherals limits the number of traces traveling from bottom to top board, and thus allows for a 40-pin connector to interface between the top and bottom boards.

A wire wrap prototype was first built in order to prove the functionality of all devices and correct any misconceptions as to the interface between the microprocessor and peripherals. The use of a wire wrap prototype itself injected additional noise into the system that harshly affected certain components. Primarily components containing static RAM (the USB controller and SRAM chip itself) suffer the most from the lack of sufficient noise immunity. These problems led to some unreliability of the wire wrap model but looked to be purely random and removable with a PCB board design.

Design of the PCB boards was completed using CadSoft's Eagle layout tools. First the schematics corrected from testing the wire wrap board were placed into the schematic editor. Any parts not already in Eagle's parts library must also added, including a properly measured PCB-board representation of the device. Once the schematics and parts libraries were complete a board layout can be completed by switching to board layout mode. All components were then available with the correct connections made as a rat's nest. Components were then hand-placed and hand-routed since Eagle's auto route feature creates what may be a shortest-path route, but also resembles what may be thought of as spaghetti-routing.

Even with the careful checking against the wire wrapped model, the PCB was expected to be about 90% correct. After the first set of boards was manufactured, a number of fixes were made to achieve full functionality. Also not all features are finalized and various sections of the PCB board will need to be revised in order to accommodate these changes. A photograph of both the top and bottom PCB boards is shown in Figure 1.

4.4 Software Levels

The software for the Project193 micro controller is split into three levels of Kernel, Interpreter and Development environment. A diagram of how the levels interface with each other can be found in Figure 2 at the end of this paper.

4.4.1 Kernel and Interpreter

The interpreter software layer interfaces with the physical hardware on the controller board by means of a simple embedded operating system developed from scratch by the Project193 team, specifically for the Project193 system. The operating system is highly optimized for the 8XC196NT microcontroller and takes advantage of all its peripherals and features. In addition, it has a very small footprint: only about 80K of code and 6K of static data. All implementation was done in the ANSI C programming language. The design of the operating system is highly modular, with very low coupling between the various modules. This allows for a high degree of flexibility in the system, in the event that hardware or software features are added or removed for various applications.

A main goal of the Project193 operating system is to provide an interface for higher layers of software to manipulate system hardware in a well-defined and controlled environment, without the need to handle low-level hardware interfacing directly. To this end, a set of drivers was developed which provide an API for each piece of embedded hardware. For example, I/O to a connected PC host via the USB port or RS-232 serial port is exposed by means of a C-style "gstream".h API which includes calls such as fopen ( ), fprintf ( ), and so forth. Detecting and communicating with external modules on the I^2C bus is abstracted by means of probing functions, a device tree, and simple blocking I/O functions. Mechanisms are also provided for applications to act as virtual I^2C slave devices so that modules can take an active role in providing information rather than being constantly polled. Each piece of hardware in the system has a clearly defined API with a rich set of features intended to provide as much flexibility as possible to higher-level applications.

The operating system also includes a basic set of kernel services that provide multitasking, mutual exclusion, inter-process communication, and memory management. The kernel module handles the initialization of the system and provides services to all running drivers and applications. It allows for the creation of application processes and provides the multitasking environment to allow processes to operate and communicate.

For scheduling CPU time, a priority-based round-robin style algorithm is used. This provides a simple, fast system with short context-switching time, while allowing some flexibility by means of process priorities. Dynamic memory allocation is provided from a simple heap (currently 64K). The kernel provides mutual exclusion by means of simple counting semaphores. Basic message passing is provided in order to allow for inter-process communication. A flash-based file system is also included in order to allow persistent data, such as user programs, to be downloaded to the board.

This set of services and drivers provides a simple, yet powerful base upon which software for the Project193 system can be developed. In addition to the existing interpreter software that runs in the Project193 environment, there is plenty of space for custom software applications to be written which extend the capabilities of the system. Furthermore, new hardware or additional system services may be added. Since the project is open-source, the team invites developers to improve and extend the existing code.

4.4.2 Interpreter

Traditionally, the language and syntax used to program a controller are tailored specifically to the device. Because of this, students and educators are forced to spend considerable resources teaching and learning syntax and concepts which might only be applicable to the specific device. By utilizing a small, open source, well documented, and extensible interpreted language called Lua, Project193 is able to allow students to use a high level language with syntax they are familiar with. Code written in languages resembling C, BASIC, FORTRAN, or MATLAB can be easily translated to Lua which is then executed by the Lua interpreter process running above the microcontroller kernel.

Project193 is able to effectively implement this ambitious goal by utilizing the resources of the open source community. Originally, Lua started off as a graduate thesis, but grew into a project that was well suited for embedded applications. For example, the video game industry uses Lua extensively to provide an easy to use, and extensible interface into the inner-workings of their software without exposing critical information. Lua is further well suited to our project because it was designed with portability, speed, and size constraints in mind. Lua is fast since its code base is relatively small and it is implemented in pure ANSI C, so little effort was required to port the software to our embedded architecture. Lua is extensible in that it provides clear and well-documented interfaces to C functions and data structures. This extensibility allows Project193 to expose portions of the kernel programming interface to the end user in a controlled environment. For example, a debugger that will run on the controller itself is in development to assist students along the way. The on board debugger is possible through the abstraction of the students program from the kernel hardware interface.

Planning for the future, the abstraction of the hardware interface from the kernel itself will allow new hardware modules and functionality to be added without the need for firmware upgrades or other tedious tasks. A clear, documented, and robust interface will already exist that allows the new hardware to be utilized in any high level programming language already working with the interpreter. Furthermore, as new programming languages emerge, and new teaching methods are explored, Project193 will be ready to meet whatever needs arise. Overall, the interpreter process serves as an excellent bridge between the integrated development environment, and the lower level kernel, providing a dramatic increase in functionality and robustness, as well as scalability in the future.

4.4.3 Development Environment

The final layer of software is the Integrated Development Environment, or IDE. The IDE is the only software layer that the student sees. It is a collection of tools that allows them to edit, debug, and upload code to the board. The IDE itself is based on Eclipse, a popular, Java-based development framework. Fundamentally, Eclipse is a highly modular, plug-in oriented editor that is designed to be adapted to particular programming needs. To customize Eclipse for use with the new boards, plug-ins were developed to handle typical microcontroller programming tasks. For instance, there is a module that communicates with the RS-232 and USB. Additional modules provide the user with a graphical display of debugging information while other modules provide editors for a wide range of languages the student can choose from.

The IDE is oriented for educational purposes. The editors will provide instant feedback with regard to syntax and error checking. Error messages will be descriptive and an online help is easily accessible and linked to events in the editor. Furthermore, the IDE will be able to hide more complex features while allowing students to slowly turn them on as they become more comfortable. The IDE is designed to be multilingual. Each language takes the form of a module that translates a particular language into the interpreted language understood by the boards. By making the programming language flexible, the teachers can create and modify languages to suit the needs of their students and the goals of a particular class.

It is intended that the IDE will adapt steadily over time. Just like the board is modular so features can be added, so too is the IDE. Java programming modules have already been designed for Eclipse. Thus, to make changes to the IDE itself, a student needs only to turn on the Java Editing features in the IDE. The fact that the IDE can edit itself is a testament to degree of flexibility and customizability this project aims to achieve. Additionally, Eclipse is an open source project backed by large companies and a growing community. As a result, there is already a growing variety of freely available plug-ins that can be inserted into the IDE to add features.

4.5 Current Status

As of the time of writing, the team is still working under the original goal to have a working prototype of the controller in the spring of the current academic year (2003-2004). The top and bottom board prototypes are fully functional except for the LED output. Modules for motor control and servo motor control are being prototyped on a breadboard and will be moved to printed circuit boards in spring of 2004.

The three various areas of software are at different stages of advancement. The kernel is the farthest along with most of the basic methods implemented, while the Interpreter and IDE are functioning short of their intended capability. For the goal of a working prototype system demonstration by the end of the current academic year, the team has decided to implement only the methods needed for the FEH program and continue to develop such features as the language translator at a future time.

5.0 Lessons Learned

5.1 Observed Successes

Through the development process the team has seen noticeable success in many different areas. Being able to work in parallel as a team has greatly contributed to the team goals as well as allowing each team member to take "ownership" of a specific area of the project. With that responsibility is a greater concern for the accomplishments of the entire team and not just one's own work.

As none of the team member had ever worked on a project of this magnitude, many were unsure if the current timeline of development was too ambitious or unrealistic. But due to hard work and the constant motivation each team member received from their peers the project is keeping up with its proposed schedule.

In addition to the team's success at working together, this group has served as a model for professors when designing their senior design classes. As more problems are uncovered and dealt with by the Project193 team, the more knowledge about the process behind resolving these problems is gained. The Project193 team served as a model for one of the Electrical Engineering Senior Design classes in the Autumn Quarter of 2003 and is continuing to contribute to improving the design experience.

5.2 Opportunities for Improvement and Future Plans

When using the Project193 micro controller the team hopes that other groups will be able to extend the functionality of the controller to fit their needs on a case-by-case basis. To that end, the team has created different mechanisms for easy expansion and discussed a number of possible enhancements for the project in the future.

In the future, the team will distribute a module template that will allow users to easily add their own modules and interface them with the controlling software. As mentioned in the software section, most of the functionality is abstracted to the user and these templates will facilitate an easier time connecting custom hardware to the micro controller.

Another area that the team would like to see developed is a wireless module for communication purposes. Using the I^2C bus a wireless module would allow the controller greater flexibility to communicate back data to the user and work remotely.

Along with the many advantages of the modular nature of the hardware controller and a somewhat similar concept for the software, there are numerous opportunities for other software tools. The controller system will be use by freshman students. When these students begin their program of study, a learning styles inventory is completed. The results consistently show that the students learning style is significantly visually orientated. Software is the focus of the second course of the program. A tool that allows one to program either textually (traditional) or by a flow charting tool, seeing both the textual code and the flow chart regardless of the editing mode would allow students with either a visual or verbal type learning style to view the program the way that is best for them.

After the controller is finished the team plans to distribute the plans for use by other academic institutions and pursuits via the project website. The hardware board layout files and parts list will be posted online and can be used with permission. The software is being developed and released under the GPL that allows for unlimited use as long as credit is given to the original software developers. As other developers create more modules the team plans to have a centralized website with a listing of software updates and hardware and software expansions.

If there is a large demand from sources outside The Ohio State University, the team plans to distribute packages of controller boards with a set of modules, software, and documentation to other interested parties. These packages would allow others to take advantage of the Project193 micro controller without having to fabricate it themselves.

6.0 Summary and Conclusions

The Project193 micro controller will allow students in the Fundamentals of Engineering for Honors program a greater number of design options while increasing the support to allow them to focus their learning on the design experience and teamwork. By eliminating the current micro controller drawbacks and giving the students an expanded set of tools, the learning experience and freshman experience as a whole is improved significantly. To this point the team has also demonstrated the ability to multitask and work in parallel, demonstrating that a senior design team can work with little to no classroom assistance. In the future the team hopes to recount the micro controller's implementation into the FEH program.

Bibliography

1. Freuler, R.J., A.W. Fentiman, J.T. Demel, R.J. Gustafson, and J.A. Merrill, "Developing and Implementing Hands-on Laboratory Exercises and Design Projects for First Year Engineering Students", Proceedings of the 2001 American Society for Engineering Education Annual Conference, June 2001.
2. Demel, J.T., R.J. Gustafson, A.W. Fentiman, R.J. Freuler, and J.A. Merrill, "Bringing About Marked Increases in Freshman Engineering Retention", *Proceedings of the 2002 American Society for Engineering Education Annual Conference*, June 2002.
3. Freuler, R.J., M.J. Hoffmann, T.P. Pavlic, J.M. Beams, J.P. Radigan, P.K. Dutta, J.T. Demel, and E.D. Justen, "Experiences with a Comprehensive Freshman Hands-On Course – Designing, Building, and Testing Small Autonomous Robots", Proceedings of the 2003 American Society for Engineering Education Annual Conference, June 2003.
4. Fentiman, A.W., J.T. Demel, R. Boyd, K. Pugsley, and P.K. Dutta, "Helping Students Learn to Organize and Manage a Design Project", Proceedings of the 1996 American Society for Engineering Education Annual Conference, June 1996.
5. Fentiman, A. W., and J.T. Demel, "Teaching Students to Document a Design Project and Present the Results", Journal of Engineering Education, Vol. 84, No. 4, October 1995, pp. 329-333.

6. Martin, F.G., Robotic Explorations – A Hands-On Introduction to Engineering, Prentice Hall, Upper Saddle River, New Jersey, 2001.
7. Winton, C., "IC4 Programmer's Manual", Internet: http://www.kipr.org/ic/download/ICv4ProgMan.pdf, January 2002.
8. Lindholm, T., and F. Yellin, The Java Virtual Machine Specification, Addison Wesley, Mountain View, California, 1997.
9. Martin, F.G., "The Handy Board", Internet: http://handyboard.com, January 2002.

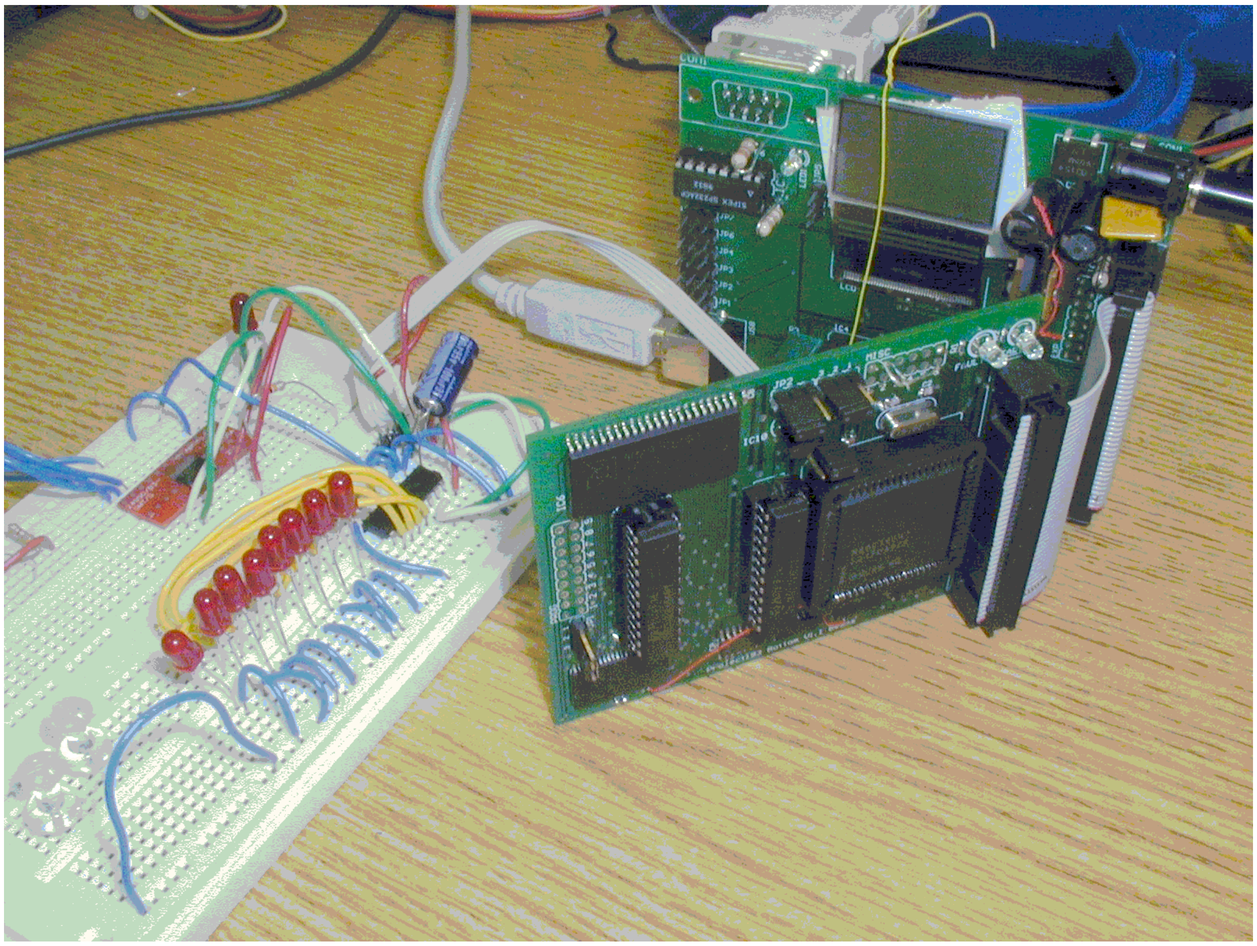

Figure 1. Project193 Top and Bottom Boards with USB, Serial and I^2C peripheral attached.

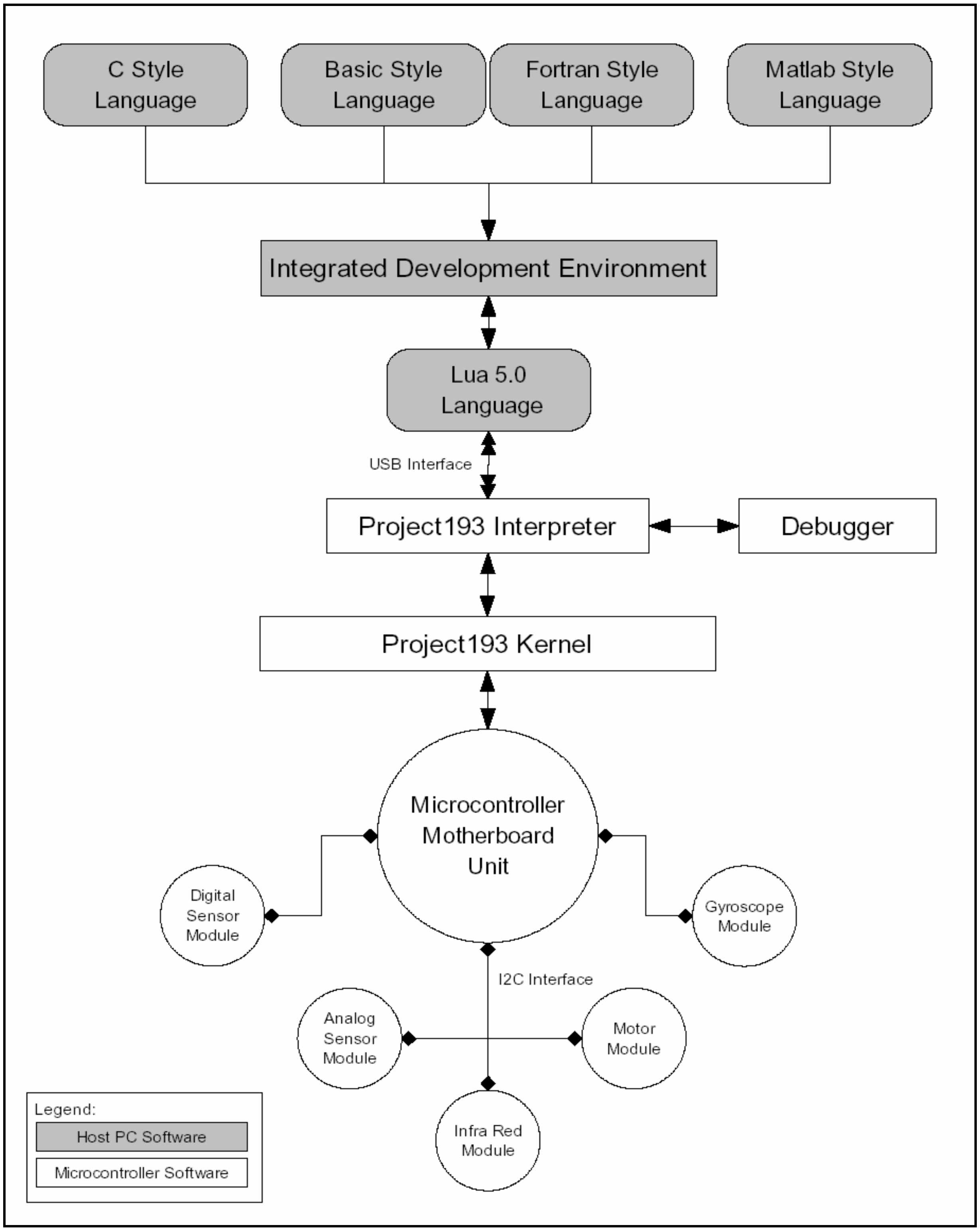

Figure 2. Software Level Interfacing

JEFFERY P. RADIGAN
Jeffery P. Radigan is a graduate student at The Ohio State University and has worked with the Fundamentals of Engineering for Honors Program (FEH) for the past four years. His research topics include the development of a robotic controller and a galloping automaton. Mr. Radigan received his B.S. in Electrical and Computer Engineering with a minor in Japanese from The Ohio State University in December 2003.

JAMES M. BEAMS
James M. Beams is a graduate student at The Ohio State University in the Electrical and Computer Engineering and a former participant of the Fundamentals of Engineering for Honors (FEH) Program. He has been a FEH teaching assistant and webmaster for five years. His research includes the development of an improved robot controller. Mr. Beams received his B.S. in Electrical and Computer Engineering from The Ohio State University in December 2002.

RICHARD J. FREULER
Richard J. Freuler is the Coordinator for the OSU Fundamentals of Engineering for Honors (FEH) Program and teaches the three-quarter FEH engineering course sequence. Dr. Freuler received B.S. and M.S. degrees in Aeronautical and Astronautical Engineering in 1974, a B.S. in Computer and Information Science in 1974, and a Ph.D. in Aeronautical and Astronautical Engineering in 1991 all from The Ohio State University.

CRAIG E. MORIN
Craig E. Morin is an undergraduate student majoring in Electrical and Computer Engineering at Ohio State and has worked as a Fundamentals of Engineering for Honors Program teaching assistant for the past three years. He works extensively with the FIRST Robotics Competition teams at OSU and does research involving robotic controllers.

MATTHEW S. GATES
Matthew S. Gates is a firmware engineer for Hewlett-Packard and a former participant and Undergraduate Teaching Associate in the Fundamentals of Engineering for Honors (FEH) Program. He has designed controller firmware for the storage-are network RAID products in the Network Storage Solutions division of HP for the past two years. Mr. Gates received his B.S. in Computer Science and Engineering from The Ohio State University in June 2002.

JEFFREY J. MCCUNE
Jeffrey J. McCune is an undergraduate student majoring in Computer Science and Engineering at Ohio State University and a previous FEH participant. Besides his class work he is currently the assistant network administrator for the Department of Linguistics at Ohio State University and enjoys mentoring high school students through the FIRST robotics program. He will receive his B.S. in Computer Science and Engineering in June 2004

ANDREW O'BRIEN
Andrew O'Brien is a senior undergraduate Electrical Engineer with a strong programming background. Besides his work with the project, he is employed as a student programmer for a variety of work, including website development and scientific programming with electromagnetism. He will receive his B.S in Electrical and Computer Engineering from OSU in March 2004.

JOANNE E. DEGROAT
Joanne E. DeGroat is a professor in the Department of Electrical Engineering and has been involved in the Freshman Engineering and FEH program for several years. Dr. DeGroat received a BS in Engineering Science from Penn State University in 1973, a MSEE from Syracuse University in 1978, and a Ph.D. in Electrical Engineering from the University of Illinois in 1991.

JOHN T. DEMEL
John T. Demel is Professor of Engineering Graphics in the Department of Civil & Environmental Engineering. Dr. Demel is the Faculty Coordinator and teaches for the First-Year Engineering Program (FEP). He helped create and develop the FEP program. Dr. Demel earned his B.S.M.E. at the University of Nebraska-Lincoln (1965) and his Ph.D. (1973) degree in Metallurgy from Iowa State University. He is a registered professional engineer in Texas.