Using a Web 2.0 Approach for Embedded Microcontroller Systems

J. O. Hamblen[1] **and G. M. E. Van Bekkum**[1]
[1]School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, Georgia, USA

Abstract *- This paper describes our experiences using a new approach for teaching an embedded systems design course and the associated laboratory. A cloud-based C/C++ compiler and file server are used for software development along with a low-cost 32-bit microcontroller board. Student resources include an eBook, web-based reference materials and assignments, an online user forum, and wiki pages with sample microcontroller application code. In laboratory assignments, breadboards are used to rapidly build prototype systems using the microcontroller, networking, and other I/O subsystems using small breakout boards with a wide variety of sensors, displays, and drivers. Software development is done in any web browser, all student files are stored on the web server, and downloading code to the microcontroller functions in the same way as a simple USB flash drive.*

Keywords: Embedded Systems, Design Project, Cloud Compiler, Microcontroller, Microprocessor

1 Introduction

Many schools offer an embedded systems design course. They can be found in electrical engineering, computer engineering, and in many systems oriented computer science undergraduate degree programs. For historical reasons, the course title can be somewhat different such as “design with microcontrollers” or “microprocessor based design” but they all have a lot in common.

These courses started out with low-cost 8-bit processors, but most of the development effort in industry has moved on to System on-a-Chip (SOC) 32-bit devices that contain a processor, volatile and non-volatile memory, and a wide assortment of I/O interfaces on a single chip. For software development in the embedded systems industry, the C/C++ family of languages is still the most widely used according to annual industry surveys [1]. Many new embedded devices now utilize networking even including those with microcontrollers.

This paper describes our experiences in developing such a course, updating it with new technology, placing an increased focus on networking, and utilizing an increased number of web-based resources. We will focus primarily on the technologies used in the associated student instructional laboratory for this course.

2 Embedded Systems Laboratory

This type of course requires a laboratory and is also an ideal place in the undergraduate curriculum to include design projects. Textbooks are always a bit problematic for such a course as once they include the details needed on a particular hardware setup for examples and laboratory assignments they quickly become out of date. We developed a textbook for the course that includes an overview of embedded systems design process with an emphasis on I/O systems, interfacing to external devices, and software development. This is where most of the development effort is focused for embedded products, now that a single chip microcontroller contains the processor, memory, and numerous I/O interfaces. The course textbook is distributed freely to students in an electronic format and is updated each semester. Students are required to own a notebook PC at our school and are allowed to use the notebook on tests as an eBook reader. Only a small percentage of students print or purchase a hardcopy of the textbook.

We use a microcontroller for the first half of the semester in the student laboratory assignments. The textbook covers the embedded development process, popular I/O interfaces, sensors, drivers, and networking. All materials for the laboratory assignments are provided on the web and can be updated each semester [2].

3 Laboratory Equipment and Tools

Selection of the hardware and software for a student laboratory is always a difficult decision. The desire to use the latest technology pushes course instructors to constantly update such a course. The cost of new equipment and software tools always makes the process more daunting. By adopting the some of the newer approaches being used in industry, students should be more productive and able to produce prototypes of complex embedded devices in less time.

For every desktop computer, there are around one hundred times as many processors found in embedded products worldwide. So job opportunities for students are vast in the embedded arena.

ARM processors are the most widely used processors in embedded devices. ARM does not make processor chips, but they license their processor design to over one hundred

semiconductor manufacturers. They would be a natural choice for such a course. The Keil tools C/C++ compiler and emulator is one of the more popular development platforms for ARM processors in industry, but it is still somewhat expensive for schools. There are also some open source options for ARM C/C++ compilers.

We considered several options for the laboratory projects. One option we had considered initially was the new Cypress PSOC 5 ARM-based processor, but it had production delays and was not available in time for our course. We finally chose the mbed module seen in Fig. 1[3-5]. The small low-cost mbed module contains an NXP LPC1768 SOC processor. The new LCP1768 contains an ARM Cortex M3 processor, 64K RAM memory, a 512K Flash memory, a network controller, and a wide range of I/O interfaces such as USB, SPI, I2C, GPIO, ADC, DAC, RS232, and PWM. The price of the mbed module is significantly less than most textbooks, so it is even possible to consider the option of students purchasing their own mbed module. They are currently available for purchase from a number of web-based electronics distributors such as Digikey and Sparkfun.

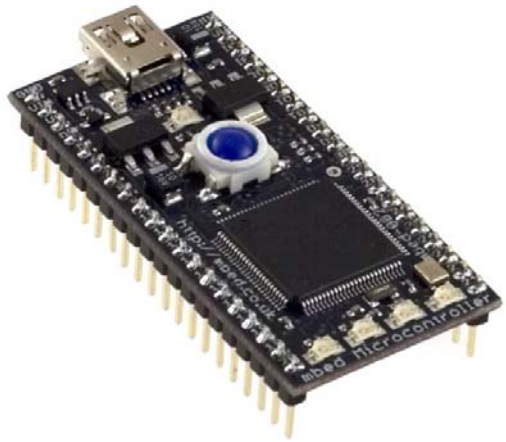

Fig. 1. The small low-cost mbed module plugs into a standard student breadboard.

The mbed module grew out of an internal research project at ARM to make the embedded development process easier both in industry and at schools [4, 5]. The module itself can plug into a standard student solderless breadboard. A USB cable connects it to a PC and for downloading code during software development it functions just like a USB flash drive. Power is provided by the USB cable. The USB interface can also function as a virtual com port allowing mbed programs to perform "printfs" or "scanfs" using any terminal application running on the PC.

For many years, schools have moved away from using solderless breadboards since many new ICs are only available in surface mount packages and not the older one tenth inch DIP style IC packages that plug directly into breadboards. Another problem with breadboarding was the large number of wires required to build up a realistic prototype of an embedded system. There is some pedagogical value in having students actually build up a circuit rather than using a large pre-assembled board. A breadboard also allows students to add their own custom hardware.

In the past couple of years, two factors have combined that make breadboards an interesting option to consider again for student laboratory work. Modern SOC processors already have sufficient internal memory and I/O interfaces on-chip. Most external I/O sub systems for embedded devices (i.e., sensors, displays, drivers, and networks) now use a serial interface that requires only a few wires. Due to a greatly increased level of hobbyist activity with the new generation of inexpensive single-chip microcontrollers, a large assortment of low-cost external I/O devices are commercially available pre-assembled on small printed circuit boards (i.e. breakout boards) that contain new surface mount ICs. They have pins that will plug directly into a standard student breadboard as seen in Fig.2.

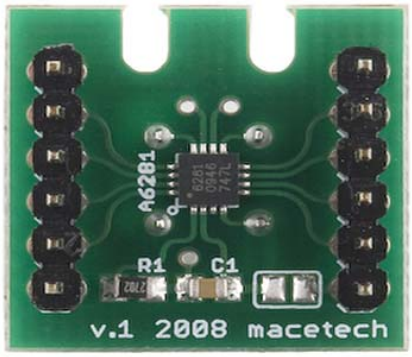

Fig. 2. A surface mount IC on a small breakout board that plugs into a student breadboard.

We compiled an extensive list of over one hundred commercial breakout boards with sensors, displays, drivers, and I/O connectors and posted it on the mbed wiki site [6]. A student breadboard project built with breakout boards is seen in Fig. 3.

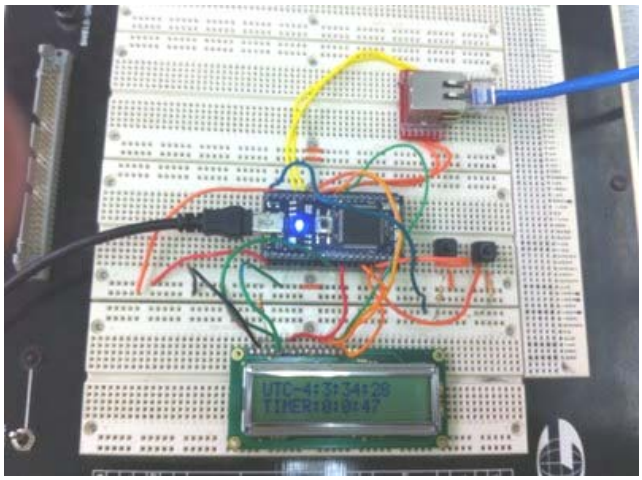

Fig. 3. An Internet Clock student project built using mbed and breakout boards on a breadboard. The time on the LCD is automatically synchronized with a NTP server via the Internet.

For those that want a pre-assembled board option there are also a number of commercial baseboards available for the mbed module [7]. The mbed module plugs into the baseboard. The baseboard typically provides an Ethernet network connector, microSD card slot, and a USB connector perhaps a sensor and small prototyping area as seen in Fig.4. On most baseboards, the prototyping area is limited. If students are going to design and add significant custom hardware in their laboratory projects, the breadboard may be a better choice.

Fig. 4. One of the commercial baseboards with network and I/O connectors designed for the mbed module.

4 Software Development

One of the more novel approaches of the mbed project was the decision to develop and support a cloud-based compiler for C/C++ software development. The compiler can run in any web browser as seen in Fig 5.

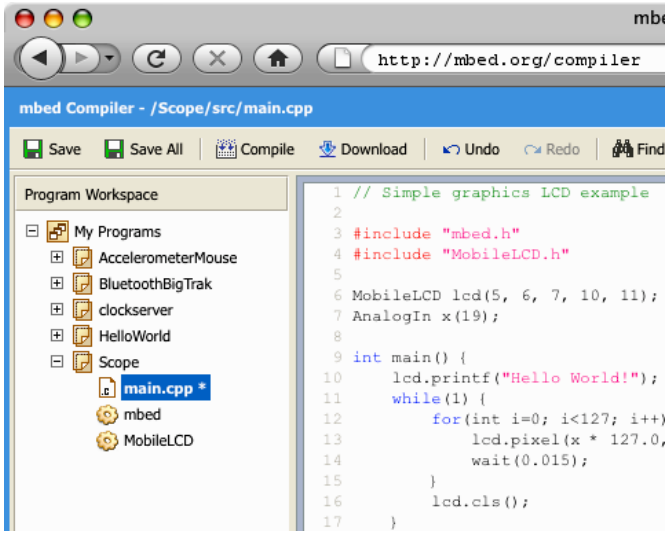

Fig. 5. The cloud compiler for mbed runs in any web browser. It is based on the Keil tools compiler.

Since the mbed module mounts just like a PC USB flash drive, it is easy to save new executable files to program the device. It runs the file with the newest date whenever you push the reset pushbutton on the mbed module. Another smaller ARM processor provides the USB flash drive interface and controls the main processor chip for debug. The firmware in this second interface processor cannot be modified by the user.

All students in the class can get a free password to setup an account on the cloud compiler with space for file storage [7]. Course instructors can request and obtain these passwords via email. Source files and documentation are saved on the mbed server. This means that there is no software to install and maintain, and development can move anywhere to any machine with a web browser. So students can easily work in the lab or at home on their projects. Most students can have a "hello world" application running on mbed in under five minutes.

In addition to the cloud compiler, the API support is also innovative. Network drivers, basic file system drivers, and easy to use APIs were developed for the NXP1768s on chip I/O features. These add higher level support for networking, files, PWM, SPI, I2C, Analog I/O, timers, delays, and RS232 serial ports using simple C++ object oriented library API calls.

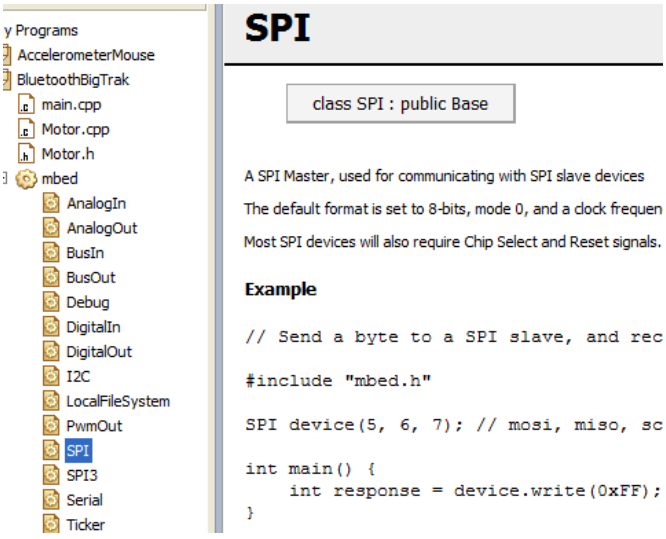

Fig. 6. The mbed API web-based documentation for an SPI interface and a code example.

Pin names can be used to specify the use of individual pins on the mbed module. Most pins can have four different programmable functions on the processor. The C++ object oriented APIs are able to automatically configure multifunction pins based solely on the pin names and the APIs used. Most students are able to hookup new I/O device hardware without ever checking the detailed data sheet for the device.

Along with the cloud compiler, the mbed website contains a number of helpful resources for students. Each student gets a notebook area where they can save documentation about a project. The handbook page contains an online API reference manual for the mbed I/O functions [9]. The documentation page for the SPI library API is seen in Fig. 6. There is an active user forum where students can post questions [10]. An extensive Wiki site contains documentation and code examples provided by users [6]. Code examples and projects can be easily imported from other users on the mbed web server with a few mouse clicks.

5 Laboratory Assignments and Projects

The topics for laboratory work will vary significantly at each school depending on the student's background and the goals of each individual course. Our class was oriented to embedded systems design and it contained a mixture of EE, CmpE, and a few CS undergraduates. Students had previously taken a digital logic design class, introduction to computer architecture, and C/C++ programming. Each laboratory assignment is two weeks for a total effort of around 6 hours. Students work in teams of two. Our institution provides Tsquare for web-based distribution of course materials and grades. Portions of Tsquare are based on the widely used Sakai Project. Tsquare was used to distribute mbed passwords, announcements, laboratory assignments, course materials, and grades.

In our course, we use mbed for two laboratory assignments followed by a short design project. We started out with a basic introductory lab where students used mbed for digital I/O, used PWM to dim an LED, added an I/O port expander, and used power management to reduce power levels by adapting C/C++ examples from the mbed wiki pages. For extra credit, they could add a watchdog timer or go back and use ARM assembly language instead of C/C++ for the basic digital I/O LED blink demo. Using assembly language made them appreciate the productivity gains using the C/C++ compiler and mbed's I/O API support. ARM assembly language development is possible using standard *.*s* files in the cloud compiler [7].

The free evaluation version of the Keil tools compiler can also be used for I/O emulation and assembly language debugging offline. It is limited to 32K code size. The cloud-based compiler is based on the Keil tools compiler, so projects can be moved offline and back to the cloud compiler.

In the second laboratory experiment, students connected a number of different interfaces and devices by adapting several of the C/C++ cookbook Wiki code examples (i.e., RS-232, I2C, SPI, Analog in and out, USB, Ethernet, LCD text display, and a DC motor using PWM) and demoed each one working to the TA on a breadboard. The mbed's higher-level C/C++ I/O APIs really save some time here. In the class lectures at this time, we were talking about different I/O interfaces, so the lab was a good fit with the lectures.

After two introductory labs using mbed, we allowed students the freedom of a team-based design project where they could pick the idea. Instructor and TA approval and guidance regarding the scope of the project was required. Students post their project documentation using the mbed web site's notebook feature and they are encouraged to include photos and video clips. It is surprising the array of different ideas that were seen, and students also are more motivated when working on their own project idea.

6 Conclusions

There were very few software related issues to resolve. The cloud compiler with students keeping files on the server worked out better than many of our locally running tools and no support was required other than initially handing out student passwords and enabling network access for the mbed modules.

One concern we had initially was the support for debugging. Hardware breakpoints are not currently supported. To debug, the mbed module has four user LEDs, and "printfs" can print to a terminal application program running on the PC. Most of our problems occurred from students not wiring up all of the jumper wires correctly and not issues with debugging program code.

It is also possible to compile, set breakpoints, and emulate code and I/O offline using the Keil Tools compiler. Only a couple of students doing low-level hardware coding for their design project have needed to use this approach. The editing features available in the browser-based cloud compiler are a bit limited, but they seem to be rapidly improving with time.

Students need to pick a design project idea early in the term to allow time for any custom parts that might have to be ordered to arrive. It is also a good idea to remind students to check and only order parts that are in stock so that they arrive in time.

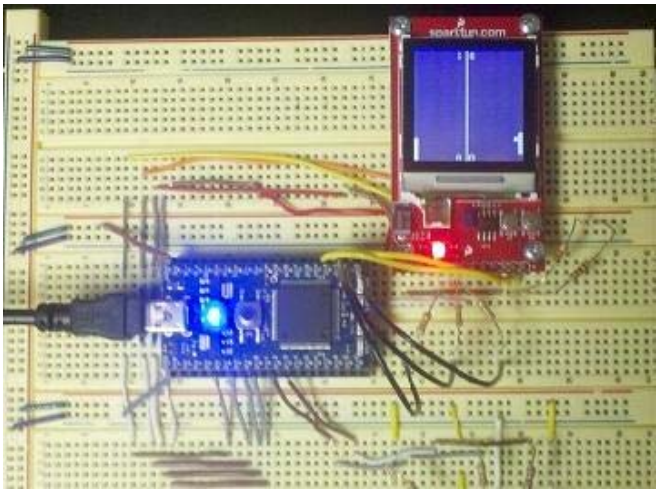

Fig 7. A student pong game project using a color LCD breakout board. The LCD is similar to those found in many cell phones.

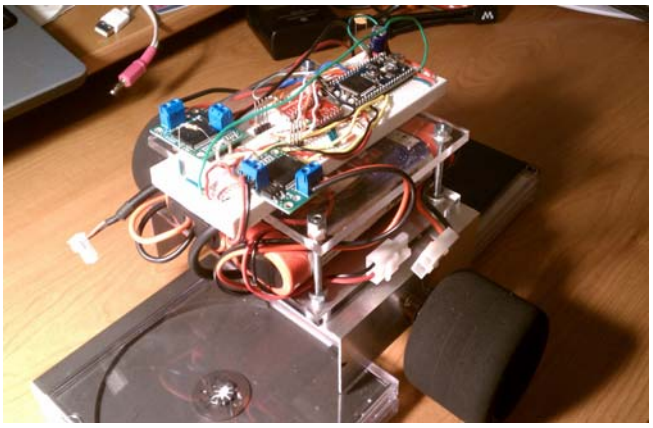

Fig 8. A two-wheel self-balancing robot project using DC motors with H-bridge breakout boards using PWM speed control, quadrature encoders, and a MEMs gyro and accelerometer breakout board with a PID control loop.

Fig. 9 An mbed-based internet radio built using Ethernet, USB, stereo audio jack, and MP3 audio decoder breakout boards.

Supporting all of the different design projects will require a larger assortment of sensor, drivers, and breakout boards than a more structured lab. The good news is that all of the parts unplug and can be reused for next term minus the few that get destroyed or lost. Surprisingly, so far with almost a hundred students no one has destroyed an mbed module. We did have a USB cable short out and lost a couple of the breakout boards.

Several examples of student design projects are seen in Figs. 7, 8, and 9. They include a pong game using a color LCD, a self-balancing robot, and an internet radio. All were built using the mbed module on a student breadboard using only commercially available breakout boards and jumper wires. Examples of many of the design projects from the class can be found in the mbed site's wiki pages under Student Projects [11].

A wide range of interesting devices were successfully prototyped for the design projects and approximately half of the projects used the internet. A number of students have also chosen to use mbed again in their team-based senior design project after taking the class.

7 References

[1] M. Barr, "Real men program in C", Embedded Systems Design, 2009 [Online]. Available: http://www.embedded-systems.com/design/218600142

[2] ECE 4180 Embedded Systems Design [Online] Available: http://www.ece.gatech.edu/~hamblen/4180

[3] Ashlee Vance, "You Too Can Join the Internet Of Things", New York Times, September 20, 2010. Available: http://bits.blogs.nytimes.com/2010/09/20/you-too-can-join-the-internet-of-things/

[4] S. Ford, Rapid Prototyping for Microcontrollers,[Online]. Available: http://mbed.org/media/press/mbed_whitepaper.pdf

[5] ARM University Program [Online]. Available: http://www.arm.com/support/university/

[6] J. Hamblen, "IC Sensor and Driver Breakout Boards" [Online]. Available: http://mbed.org/cookbook/IC-Sensor-and-Driver-Breakout-Boards

[7] Mbed Cookbook Wiki [Online]. Available: http://mbed.org/cookbook/Homepage

[8] Mbed Educational Program [Online]. Available: http://mbed.org/handbook/Education

[9] Mbed Handbook [Online]. Available: http://mbed.org/handbook/Homepage

[10] Mbed Forum [Online]. Available: http://mbed.org/forum/

[11] J. Hamblen, "Mbed Student Projects", 2011 [Online]. Available: http://mbed.org/cookbook/Student-Projects