Introducing Embedded Systems in the first C/C++ Programming Class

James O.Hamblen, Zachery C. Smith, and Winne W. Woo
School of ECE
Georgia Institute of Technology
Atlanta, Georgia USA
hamblen@ece.gatech.edu

***Abstract*— A new approach for laboratory assignments in the first C/C++ programming course is introduced. An easy-to-use low-cost ARM SOC computer module with a parts kit is used in the first C/C++ programming class taught by the School of ECE. Each student purchases a processor module along with a parts kit. A cloud based C/C++ compiler is used for software development and breadboards are used with breakout boards containing common I/O devices. Simple C++ APIs are provided for all I/O hardware interfaces along with code examples for each I/O device. Extensive wiki pages and short videos are used to provide documentation.**

Keywords- C/C++; SOC; embedded; microcontroller; microprocessor; programming;

I. INTRODUCTION

Many schools teach a second programming class at the sophomore level. The electrical and computer engineering curriculum at the Georgia Institute of Technology was recently revised to include a new four hour class, ECE 2036 Engineering Software Design, based on the C/C++ language. The first programming class is taught by the computer science department and is Mat Lab based. One of the goals of the new ECE C/C++ course was to include more engineering orientated examples and applications.

Since the embedded market represents a major portion of commercial engineering activities in ECE and the majority of embedded devices run C/C++ code [1, 2], the prevailing view of faculty was that it would be highly desirable to include several laboratory assignments in the new course demonstrating applications in embedded systems. Traditional C/C++ programming assignments with text I/O and graphics using Linux and Windows are still used for the remaining programming assignments.

Moving such examples earlier in the curriculum [3] requires careful planning and selection of devices and tools to avoid the potential pitfall of spending too much time on low-level hardware and software details since the primary purpose of the course is to be a second programming course, introduce C/C++, and not to become an embedded systems design course.

Ease of use and cost were major factors in the selection of the hardware and software tools. Since the numbers of students involved will be over four hundred students per year it was also a goal to develop a model where students purchase their own hardware similar to many of the current digital logic design courses. Breadboards and jumper wires can also be utilized from the earlier digital design course. At Georgia Tech, students are also required to purchase a laptop PC. All software development would need to be done in C/C++ and the tools would need to be extremely low-cost or free and at the same time easy to learn and use. By embracing some of the newer rapid prototyping approaches being adopted in industry, students can be more productive and able to create prototypes of embedded devices in less time. Depending on the platform selected, the same embedded hardware could potentially be used in the introductory computer architecture class and even perhaps other ECE courses.

II. HARDWARE AND SOFTWARE TOOLS

After considering of number of options, the ARM-based mbed module was selected based on the hardware and software ecosystem along with the readily available documentation provided on wiki pages. ARM processors are also used in most new embedded device designs [2].

A. Laboratory Hardware

The small low-cost mbed module seen in Fig. 1 contains an NXP LPC1768 SOC processor. The new LCP1768 SOC contains a 100Mhz 32-bit ARM Cortex M3 processor, 64K RAM memory, 512K Flash memory with a Flash accelerator, a network controller, and a wide range of I/O interfaces including USB, SPI, I2C, GPIO, ADC, DAC, RS232, and PWM.

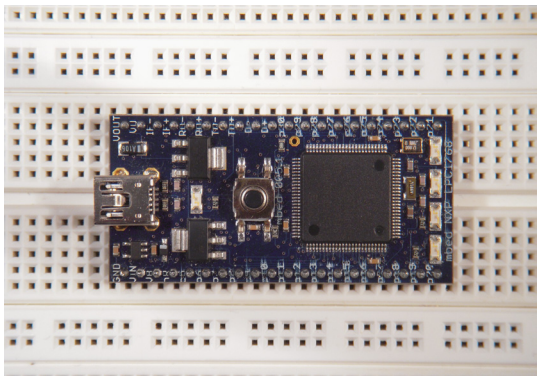

Figure 1. The low-cost mbed ARM SOC processor 40-pin DIP module plugs directly in a student breadboard and is powered via USB.

978-1-4799-0142-5/13/$31.00 ©2013 IEEE

The price of the mbed module is around half of the cost of most textbooks, so it is possible to consider the option of students purchasing their own mbed module.

An internal research project at ARM to make the embedded development process easier both in industry and at schools [4] led to the development of the mbed module. The module itself can plug into any 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 moved away from using breadboards as most new ICs were only available in surface mount packages, and not the older one tenth inch DIP style IC packages that plug directly into breadboards. A second issue with breadboarding was that a large number of wires was required to build older technology systems.

Two factors have recently combined that make breadboards an interesting option to consider once again for student laboratory work. Modern SOC processors already have sufficient internal memory and I/O interfaces on-chip for many designs. External I/O sub systems for embedded devices (i.e., sensors, displays, drivers, and networks) now typically use a serial interface that requires only a few wires. Due to a greatly increased level of hobbyist and student 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 .1 inch pins that will plug directly into a standard student breadboard.

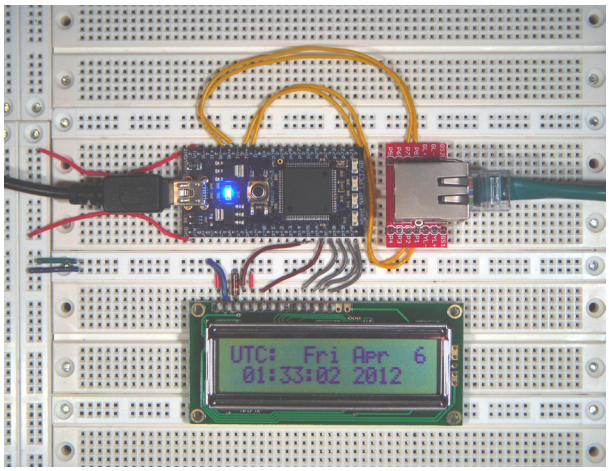

Figure 2. Internet of Things Clock project built using a breadboard with mbed and commercial breakout boards. Time is set using NTP.

B. *Laboratory Software.*

A novel cloud-based compiler for C/C++ is normally used for software development for the mbed module [5]. Traditional offline compiler tool chains can also be used and breakpoints are supported via the USB cable. For the vast majority of student projects, the cloud compiler is easier to setup and use. The cloud compiler can run in any web browser as seen in Fig 3. Macs, PCs, and even tablets can be used for software development. No complex software installation and setup is required. Student files are stored on a cloud server and development can move anywhere to any device with a web browser.

The cloud compiler is based on the widely used commercial ARM/Keil C/C++ MDK compiler. Schools can also use the more traditional offline compiler approach with a floating license server for PC labs [6, 7]. The offline compiler supports hardware breakpoints over the USB cable and also has full I/O pin emulation of the SOC on the mbed module.

A free demo version of the offline compiler is available for download, but code size is limited to 32K. Once it connects to the license server it enables the full version. We have both compiler options available, but the vast majority of students prefer the ease of use of the cloud compiler.

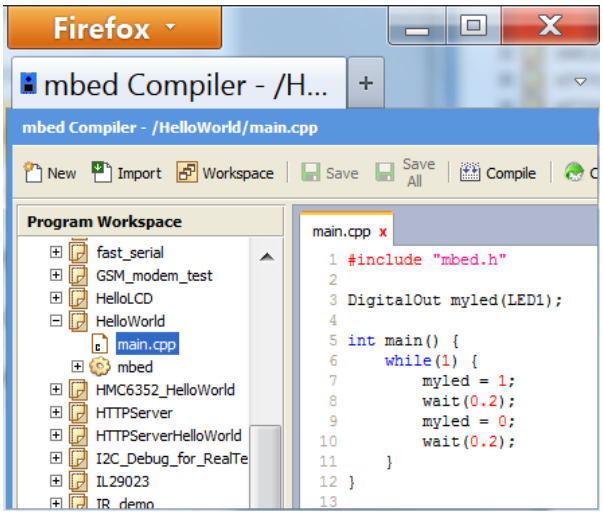

Figure 3. The mbed C/C++ cloud compiler runs in any web browser.

The C/C++ API support for the mbed module is also innovative [8]. 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 C/C++ support for networking, files, PWM, SPI, I2C, Analog I/O, timers, delays, and RS232 serial ports using simple C++ object oriented library API calls. The development for student projects can be done entirely in C/C++ and the APIs handle any low level hardware details. An optional RTOS was also recently introduced that can run eight threads and provides synchronization primitives along with message queues.

Predefined C/C++ pin names are used to specify the use of individual pins on the mbed module. I/O pins can have one of several different programmable functions on the processor. The C++ object oriented APIs 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. The extensive use of C++ classes in APIs simplifies everything for students and also reinforces many of the new ideas covered in a C++ programming course.

It is also possible to program the module using simple C only programs or ARM assembly language with the cloud compiler. While not used in a C/C++ course, this feature could be used in later classes covering assembly language and computer architecture. A second course, ECE 2035, has started using the module for this purpose.

III. MBED INVENTOR'S KIT

It is possible to do a quite a number of projects using only the basic mbed module, but an assortment of low-cost breakout boards enables a wider range of interesting student projects.

Based on previous experiences in a senior level embedded systems class [9], a selection of low-cost breakout boards was assembled in an mbed inventor's kit and made available to students. The mbed inventor's kit developed at Georgia Tech for the course is seen in Fig. 4. A description of the parts and their purpose is found in table 1.

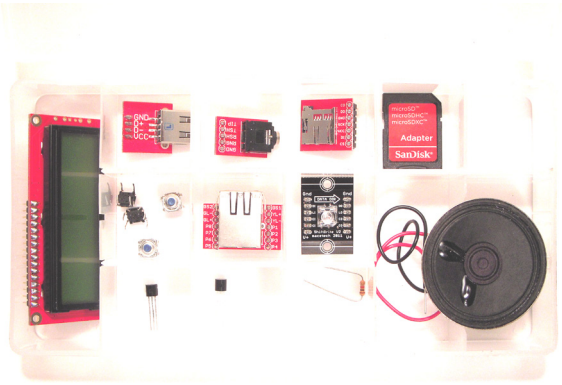

Figure 4. The mbed inventors kit was developed for ECE students.

TABLE I. MBED INVENTOR'S KIT PARTS.

Purpose	Sensor or Device
User Input	Mini Pushbuttons
Networking	Ethernet connector
2G flash file system	Micro SD card connector
USB devices	USB A connector
Display ASCII text	B&W Text LCD Display
Analog sensor data	Temperature Sensor
Display 2^{30} colors on LED	RGB LED & driver IC
Driver circuit	Transistor
Audio using D/A or PWM	Speaker and Audio Jack

Since the desired goal was for each student to be able to purchase their own mbed and parts kit at around the cost of a textbook, cost was one of the major factors in the selection of the parts. Each device in the kit also has a complete cookbook wiki page that contains the hardware connections needed for each device along with working C/C++ code examples. Any user can add mbed cookbook wiki pages with code examples, schematics, photos and YouTube videos to the mbed.org site [10]. Wiki pages with around fifty YouTube videos were developed at Georgia Tech for use in student laboratory assignments. Links to datasheets, user manuals, and application notes are also provided, but are rarely if ever needed given the quality of the existing wiki documentation along with the easy-to-use C++ I/O APIs. Laboratory assignments can use hyperlinks to refer the students to these web pages and they then know exactly how to hookup the device and use C/C++ to make it work.

IV. LABORATORY ASSIGNMENTS

In the initial offering of the course in Fall 2012, two C/C++ programming assignments utilized the new embedded hardware. Each student had the mbed module and a small parts kit. Student breadboards and jumper wires were used from the earlier required digital logic design laboratory.

A. *Thermostat Project*

The first embedded C/C++ project was a home heating and cooling thermostat. An analog temperature sensor was used to read the temperature with the A/D, pushbuttons were used for user input, and a PC terminal application window displayed messages from the thermostat. The time and temperature was also logged periodically to a file on flash memory

Laboratory assignments are provided on a web page with hyperlinks to numerous other wiki pages. These wiki pages provided wiring details along with new C++ class-based code examples based on the existing C++ I/O APIs for controlling LEDs, pushbutton debouncing, reading the temperature sensor using the A/D with the required conversion equations, reading the real-time clock, writing to a file on flash memory, and outputting messages to the PC terminal application window via USB. Wiki pages also include schematics, photos, and short YouTube video demonstrations showing how each device works individually. With this information, students were able to combine the various I/O devices to build the thermostat. In the second offering of the course in Spring 2013, a text LCD and RGB LED was added to the parts kit and is being used to display the current temperature and status.

The C/C++ topics covered in lectures at this time, basic types and variables, equations, control structures, file I/O, operator overloading, simple class definitions and member functions were all reinforced by the C/C++ code used in the embedded project.

B. *DMM, Oscilloscope, and Spectrum Analyzer Project*

The second embedded programming assignment utilized the embedded system's A/D to build a digital multimeter, an oscilloscope, and a spectrum analyzer. Skeleton code was provided with timers and interrupt code written entirely using the C/C++ APIs showing how to control the analog sample rate and take samples using the A/D.

The D/A and PWM output pins were connected to A/D inputs and used to provide analog test signals similar to a function generator. Data was stored in arrays and a C/C++

API was provided to support user menus and enable a simple graphics display on the PC.

For the spectrum analyzer, the FFT algorithm was implemented on the embedded device. Arrays, nested for loops, and more complex calculations using floating point math functions such as sine were required.

V. CONCLUSIONS

The C++ I/O APIs, wiki pages, and the cloud compiler proved extremely useful and easy for students to understand. For students, the cloud compiler approach works well since there are minimal install and setup issues. It also allows students to work anywhere with their laptop. Since the power is provided to the breadboard by the USB cable plugged into a laptop they can work on embedded projects anywhere with a Wi-Fi hot spot or network connection, even in the hallway as seen in Fig. 5.

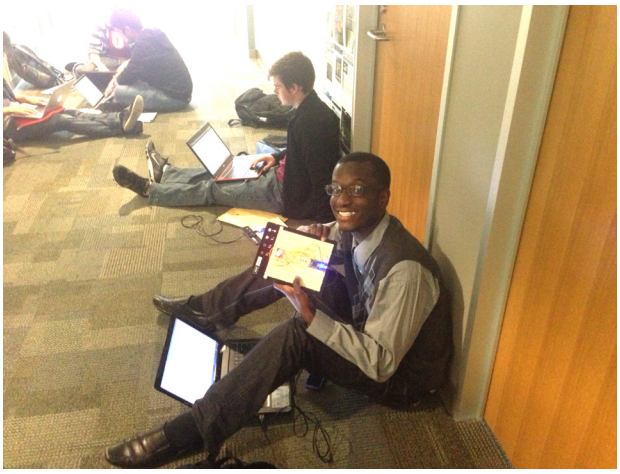

Figure 5. ECE students working on embedded C/C++ projects in the hallway using breadboards, laptops, and Wi-Fi.

Based on student surveys, a majority of students preferred the cloud compiler to a more traditional offline compiler and a large majority indicated they preferred using breadboards to a preassembled board.

The embedded laboratory projects were popular with students and faculty received numerous student requests for more laboratory assignments using mbed and fewer of the traditional C/C++ assignments using Linux and Windows platforms. Even the TAs wanted more embedded projects.

One issue we had initially was that students waiting for TA checkoffs on the due date saturated the wifi bandwidth near the TA's office. This has become a campus wide infrastructure problem especially in many of our older classrooms. Moving the TA office hours to areas in newer buildings with increased Wi-Fi and wired network connections should solve this problem.

Each term, enrollment in the courses involved increases as the new curriculum phases in. This spring, Eta Kappa Nu student volunteers assembled and sold mbed inventor's kits, breadboards, and jumper wire kits to students as seen in Figure 6. Over two hundred mbed inventor's kits were sold and wait times in the line ran over one hour. The proceeds helped fund two ECE scholarships.

Figure 6. Eta Kappa Nu mbed inventor's kit and breadboard sale.

After the first term of using the mbed module, faculty decided to continue and expand the use of the mbed module. The number of students, course sections, and faculty involved has doubled and will double again once steady state is reached in the new curriculum. Faculty developing other new ECE courses are considering using the mbed module for laboratory projects, now that all students have one available.

After gaining experience with the new technology introduced with the mbed module, an increasing number of students are also using it in their team-based senior design projects.

REFERENCES

[1] J. Turley, "The two percent solution," *Embed. Syst. Program*., vol. 16, no. 1, p. 29, Jan. 2003.

[2] "2010 Embedded Market Study," EE Times Group, New York, NY, April 19, 2010 [Online] Available: http://www.eetimes.com/electrical-engineers/education-training/webinars/4006580/2010-Embedded-Market-Study

[3] K. Ricks, D. Jackson, and W. Stapleton, "An Embedded Systems Curriculum Based on the IEEE/ACM Model Curriculum," IEEE Transactions on Education, vol. 51, no. 2, pp. 262-270, May 2008.

[4] S. Ford, "Rapid Prototyping for Microcontrollers," Arm Holdings, Cambridge, U.K. March 2012, [Online]. Available: http://www.embeddeddeveloper.com/corp/flex/mbed-IQ28.pdf

[5] J. Bungo, "Embedded Systems Programming in the Cloud: A Novel Approach for Academia," IEEE Potentials, vol.30, no.1, pp.17-23, Jan.-Feb. 2011

[6] ARM University Program, Arm Holdings, Cambridge, U.K. March 2012, [Online]. Available: http://www.arm.com/support/university/

[7] C. Styles, "Mbed Educational Program," Arm Holdings, Cambridge, U.K. March 2012, [Online]. Available: http://mbed.org/handbook/Education

[8] Mbed Handbook [Online]. Arm Holdings, Cambridge, U.K. March 2012, Available: http://mbed.org/handbook/Homepage

[9] J.O. Hamblen and G. M. E. Van Bekkum, Using a Web 2.0 Approach for Embedded Microcontroller Systems, Frontiers in Education: Computer Science & Computer Engineering, FECS 2011, pp. 277-281, Las Vegas, NV., July 2011.

[10] Mbed Cookbook Wiki [Online]. Arm Holdings, Cambridge, U.K. March 2012, Available: http://mbed.org/cookbook/Homepage