Including Embedded Systems in CS: Why? When? and How?

Bill Siever
Roger D. Chamberlain
Elliott Forbes
Ingrid Russell

Bill Siever, Roger D. Chamberlain, Elliott Forbes, and Ingrid Russell, “Including Embedded Systems in CS: Why? When? and How?,” in *Proc. of 50th ACM Technical Symp. on Computer Science Education* (SIGCSE), February/March 2019. DOI: 10.1145/3287324.3287327

Washington University in St. Louis
University of Wisconsin-La Crosse
University of Hartford

Including Embedded Systems in CS: Why? When? and How?

Bill Siever (Moderator)
Washington University in St. Louis
St. Louis, Missouri
bsiever@gmail.com

Roger D. Chamberlain
Washington University in St. Louis
St. Louis, Missouri
roger@wustl.edu

Elliott Forbes
University of Wisconsin-La Crosse
La Crosse, WI
eforbes@uwlax.edu

Ingrid Russell
University of Hartford
West Hartford, CT
irussell@hartford.edu

ABSTRACT

Embedded systems pervade nearly every aspect of modern life. Moreover, the emergence of both mobile platforms and Internet of Things (IoT) is furthering their reach. Although embedded systems are one of the bodies of knowledge in the ACM/IEEE-CS *Computer Engineering* Curricula, they have only passing mention in the ACM/IEEE-CS *Computer Science* Curricula. Inclusion of embedded systems concepts in undergraduate computer science can facilitate many objectives: a) they are an example of Platform-Based Development, a prominent theme in the ACM 2013 CS Curricula, b) they are often a more suitable level of complexity for educational needs than other "real world" platforms (e.g., Arduinos may be used to introduce many AP CS Principles in a single course), c) they offer a novel form of engagement, which may enhance diversity, and d) emerging areas, like IoT, are increasing demand for professionals that understand the full span of systems, from low-level firmware, to middleware and cloud computing.

This panel represents three methods of including embedded systems concepts in undergraduate computer science: 1) use of embedded systems to improve engagement in a non-major computing course, 2) a required course covering core content for both computer science and computer engineering majors, and 3) a degree program offering a formal emphasis in embedded systems via a complementary set of courses. The panelists will share their motivations for including embedded systems concepts in their programs, their approaches to integrating the content into their curricula, the teaching methods they use, the challenges they faced, and challenges that remain.

CCS CONCEPTS

• **Social and professional topics → Computer science education**; **Computer engineering education**; **Computational science and engineering education**; *Professional topics*; *Computing education*; • **Hardware → Analysis and design of emerging devices and systems**; • **Software and its engineering →** *Software system structures*;

KEYWORDS

Embedded Systems Education

ACM Reference Format:
Bill Siever (Moderator), Roger D. Chamberlain, Elliott Forbes, and Ingrid Russell. 2019. Including Embedded Systems in CS: Why? When? and How?. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19), February 27-March 2, 2019, Minneapolis, MN, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3287324.3287327

1 SUMMARY

The goal of this panel is to share interest in and challenges of including embedded systems concepts in *undergraduate computer science*. The panelists were selected to represent three different approaches currently in practice. They also represent a diverse set of both objectives and institutions. The audience may include educators who are either already using or are considering an approach comparable to one of the panelists as well as those who haven't previously considered the benefits of embedded systems exposure. Significant potential outcomes of the panel include: a) sharing tips that help bootstrap inclusion of embedded systems concepts in other programs and b) cross-pollination of ideas between the approaches already in practice.

2 PANEL STRUCTURE

The discussion will:

(1) Start with a brief introduction
(2) Give each panel member 10-12 minutes to summarize their work, which will include:
    - a description of their institution, program, and students;
    - motivations for including embedded systems concepts;
    - the approach they use for content integration;
    - the pedagogical techniques they use; and
    - the challenges they've encountered (scaling, motivating students, costs, equipment, etc.)
(3) Open the floor to guided discussion, including comparison and contrast of the different approaches. The moderator will have a prepared list of questions if discussion lags.

3 INGRID RUSSELL

As part of the University of Hartford's efforts to integrate High Impact Practices into the undergraduate curriculum, the Computer Science Department developed a model for revitalizing an introductory computer science course for non-majors. The course provides a broad introduction to the use of computers as tools for creativity, problem solving, communications, and information organization. It is taken by many students in the College of Arts and Sciences to meet the technology component of the general education requirement and by students in other colleges to meet a requirement for their major. The overarching goal of our efforts was to enhance the learning experiences in the course by applying and relating fundamental computational thinking concepts of algorithmic thinking, data representation, and computational efficiency to real-world problems in the context of a highly visual embedded system, the Arduino Nano. The Arduino platform, which is cross-platform and easy-to-use for both non-science and non-engineering students, has great promise in teaching course concepts in both hardware and software within a visual context. The platform provides a rich opportunity to engage students by showing broad applications of computing in domains that are part of their daily life, thus introducing computing in a way that may broaden participation.

We have developed and tested adaptable curricular modules and associated hands-on laboratories that can be used in introductory courses in both computer science and engineering. Features of our materials include: a) emphasis on application of ideas through implementation, b) a practical approach to real-world applications, c) hands-on learning that fosters close faculty-student interaction, and d) engaging, team-based active learning. The approach has been effective. Students obtained a greater appreciation for technology and how different technologies work together. In addition to improving student learning, we expect this model to improve student retention rates and encourage broader participation in computer science and engineering.

The model was first implemented during the 2016–2017 academic year in one section of the course. During fall 2017 and spring 2018 semesters two other instructors implemented the model and curricula in their offerings of the course. Additional sections will follow in the successive semesters and we will continue to collect assessment data of the model's impact.

4 ROGER D. CHAMBERLAIN

At Washington University in St. Louis embedded systems principles are introduced in a first-year course that is required for both computer science and computer engineering majors. The focus of the course is computing in the physical world, recognizing that the majority of computing systems today don't reside on desktops or in server rooms, nor is their I/O limited to a keyboard, mouse, screen, and network connection. The course content, detailed in [1], is motivated by core concepts that are foundational to to both computer science and computer engineering. This includes information representation, timing (*when* something happens as a functional property), automata models (finite-state machines), physical I/O (both analog and digital), computer communications, and a brief introduction to computer architecture and assembly language.

Unlike many introductory embedded systems courses, which are upper-division offerings, this is intentionally designed to be a first-year course, accessible after only one semester of introductory computer science. The course is lecture-free, high enrollment, and meets twice a week in the instructional laboratories. One meeting is devoted to studio sessions, where students perform guided exercises through the material they are expected to master. The other session provides opportunities for both assistance on assignments and assessment purposes. This instructional approach is consistent with the department's decade-long investment in active learning [2].

5 ELLIOTT FORBES

Within the last 10-15 years the barriers to entry in embedded systems development have dramatically decreased. Development platforms of the past were often vendor-locked, required expensive hardware and software developer kits (SDKs), were difficult to use, and lacked a community that was accessible to newcomers. Now, however, the open-source hardware and software communities have rallied, largely solving these issues.

Recognizing this trend, the Computer Science Department of the University of Wisconsin-La Crosse gauged interest in embedded systems in spring of 2014 by offering a special topics course on the Internet of Things. This course was popular with students and prompted another offering. While these classes were successful, we found that the most compelling use for these systems was in their integration with sensors and actuators. Unfortunately students had very little experience with the electrical issues that arise when interfacing with a broad variety of systems. To solve this we introduced two more classes — the first is a class that covers DC analog circuit analysis and digital logic and the second explores microcontrollers, common ICs, communications protocols, and prototyping issues. These classes are now a prerequisite to the Internet of Things course. This three-course sequence is distinct enough from the typical computer science curriculum that students completing all three courses can add a special designation to their computer science degree indicating an emphasis in embedded systems.

REFERENCES

[1] Roger D. Chamberlain and Ron K. Cytron. 2017. *Computing in the Physical World* (pre-release ed.). http://www.ccrc.wustl.edu/~roger/cse132/cc_v0_06.pdf

[2] Ross Sowell, Yixin Chen, Jeremy Buhler, Sally A. Goldman, Cindy Grimm, and Kenneth J. Goldman. 2010. Experiences with active learning in CS3. *Journal of Computing Sciences in Colleges* 25, 5 (May 2010), 173–179.