

A Consortium-based Model for the Development of a Concentration Track in Embedded Systems¹

**Gerald C. Gannod², Forouzan Golshani, Ben Huey, Yann-Hang Lee,
Sethuraman Panchanathan and David Pheanis**

**Department of Computer Science & Engineering
Arizona State University**

Box 875406

Tempe, AZ 85287-5406

{gannod, golshani, huey, yhlee, panch, pheanis}@asu.edu

Abstract

The Department of Computer Science and Engineering at Arizona State University has deployed a novel infrastructure for a concentration track in embedded systems that combines important aspects of academic content with the latest in research and industrial practices. The concentration track emphasizes fundamental issues such as the balance between hardware and software and the respective trade-offs of building embedded systems. It is realized through the use of formal course work and hands-on experience that is channeled through a capstone project implemented as internships.

1 Introduction

Rapid proliferation of embedded systems in a wide range of consumer and professional applications is expected to continue as is evident in the increased number of Internet applications with links to home appliances and industrial controllers. This trend is prompting a move from dedicated chip-set systems towards powerful processors that can be programmed to achieve a wide variety of functions.

At Arizona State University, we are completing the implementation of a novel infrastructure for a concentration track in embedded systems that combines important aspects of academic content with the latest in research and industrial practices. The concentration track emphasizes fundamental issues such as the balance between hardware and software and the respective trade-offs of building embedded systems.

Our curricular project spans the entire spectrum of activities related to the design and delivery of educational and research efforts and is characterized by three main innovative components namely, 1) a new industry-university collaborative model for integrating basic and applied research into a degree program, 2) creation and delivery of state-of-the-art course content and appropriate laboratories, and 3) creation of capstone projects that are implemented through internships.

The curricular project involves the synthesis of the core of an embedded systems program based on the latest research and close cooperation with industry. The content of the program draws heavily upon advanced research and development in industry and academia and are reinforced by

¹ This research supported by NSF Educational Innovation Grant EIA-0122600.

² This author supported in part by NSF CAREER Grant CCR-0133956.

specially structured internship activities that have been developed as part of this effort. The core material, which is not currently found in traditional computer engineering programs, provides the content that industry consultants have specifically identified as critical for engineers to function productively in the area of embedded systems.

Research incorporated into the core focuses on reliability and availability, safety, performance, and management of integrated hardware and software designs. In recent years, there have been a number of efforts to extend research results developed in either hardware design or software engineering into the integrated hardware / software co-design required for embedded systems. To date, these extensions have remained either in the research domain or a few isolated graduate topics courses. In this project, we are incorporating the latest research into the core content for an undergraduate concentration in embedded systems while providing students with real-world industrial experiences.

The design and development of this program has been catapulted by the recently formed *Consortium for Embedded and Internetworking Technologies* (CEIT). This consortium involves a partnership between ASU, Intel, and Motorola and represents a new paradigm for establishing a program where the needs of industry trigger program design and development. In this model, collaboration facilitates high quality education via the integration of the latest academic and industrial research with practical expertise and experience.

The remainder of this paper is organized as follows. Section 2 discusses background material and motivation in the area of embedded systems and reviews related work. Section 3 identifies some of the driving forces that have led to the establishment of the CEIT. The curriculum project being performed at ASU is described in Section 4, including the structure of the program as well as planned courses. Section 5 reports on the work completed to date, draws conclusions and outlines future work.

2 Background and Related Work

Certain characteristics of embedded systems distinguish them from other systems. Specifically, embedded systems typically interact in real time with their environment. As a result, they operate effectively on unbounded input sequences and similarly produce output data streams that are unbounded in length. Frequently, embedded systems have safety-critical constraints or are required to deliver ultra-high availability. Because they function without human intervention and often in hostile or remote environments, they also must be reliable, fault tolerant, robust, deterministic, and must consume small amounts of power and while sustaining long life cycles with negligible maintenance.

The fundamental problem facing the design of embedded systems is heterogeneity. That is, the developer is faced with many styles of potential behavior that may involve diverse algorithms and design principles. In addition, they must deal with many programming languages, operating systems, and a variety of technologies (e.g., digital signal processing, microcontrollers, field-programmable gate arrays (FPGAs), and application-specific integrated circuits). A simple example is the domain of signal processing. Many electronic systems, such as modems, digital cellular phones, multimedia systems, and video conferencing systems perform a combination of functions including signal processing, communications, and control algorithms. For instance, signal processing algorithms might be mapped into software for digital signal processors, or onto

dedicated or configurable hardware. The communication and control algorithms might also be mapped into software for microcontrollers, or onto dedicated or configurable hardware.

In general, the design of modern embedded systems is growing increasingly complex. This is due to:

- The need to determine the software/hardware partition of the application
- Complex communication interfaces and protocols
- Complex real-time applications.
- Increased functionality driven by low cost silicon and connectivity
- Time-to-market and return on investment (ROI) demands
- Design and cost constraints for total cost of ownership

In addition, industry is constantly driven to shorten design cycles, improve quality, and reduce the cost of their systems. When systems are heterogeneous, no one single design method is applicable to the entire system. If a group designs subsystems using their own methods and implementation technologies, the group cannot evaluate the full impact of the subsystem on the system behavior and cost. Successful design approaches integrate application-specific design methods and implementation technologies in a formal, consistent and extensible framework. This type of a framework enables the entire system to be specified, simulated, and synthesized, which in turn improves product quality. To reduce the design cycle, the framework can take advantage of application-specific design methods to validate specifications, perform fast simulations, and generate efficient implementations.

The importance of bringing embedded systems into undergraduate programs has been recognized by a number of other institutions. The VESL project [1] at Michigan State University, for instance, is aimed at introducing embedded systems to undergraduates via cooperative learning and project work. While their program is focused on embedded systems, the individual courses are loosely coupled in the sense that the overall program lacks the continuity needed to provide graduates with an effective “big-picture” of embedded systems engineering.

Several computer science and engineering programs offer embedded systems courses including Carnegie-Mellon University, University of Texas, University of California at Berkeley, and University of Colorado at Boulder. Finally, the University of California at Irvine [2] offers an embedded systems concentration at the graduate level. While each program carries a common theme of introducing embedded systems in both undergraduate and graduate courses, none of the programs offer a comprehensive coverage of the area to merit a degree concentration. In addition, each lacks the inclusion of a significant internship program, which is crucial to complement instruction with practice.

The embedded systems program at ASU is unique and differs from the above programs in the sense that it brings together all of the facets required for the creation of a successful embedded systems undergraduate program; namely, research, curriculum, development experience and internships.

3 The Consortium Model

Based on our recent interactions with industry, we have defined a set of goals for a new embedded systems program in response to a clear message indicating there is a gap between the

knowledge that is needed and the knowledge that is provided to graduates [3]. Critical topics that need to be addressed include:

- Knowledge of the hardware/software interaction and dependencies
- Hardware and device driver programming; such as DMA, communications processors, and DSP processors
- Programming parallel tasks in a multitasking/multiprocessor environment
- Bandwidth (throughput)
- Effects of caching, hardware processing, and bus operations
- Understanding and using board and chip specifications
- Dealing with customers in a field-test environment
- Knowledge of the tools used in hardware (board layout, FPGA programming) and software development (integrated development environments, compilers, debuggers, etc.)
- The design process, including specification, analysis, documentation, work scheduling, etc.

From the standpoint of technical knowledge, the needs of partner companies are very similar. Computer Engineering graduates need to understand hardware programming in order to be proficient as embedded system developers. They need to be conversant in utilizing operating systems and have the ability to read and write specifications and board schematics. In addition, they must have knowledge of the many communications protocols that are used in most of the applications while understanding real-time processing fundamentals. Performance analysis and fundamental understanding of operational characteristics of programs being developed on a given platform are essential. Finally, they must work within a complete design process from organizational, software and system engineering standpoints.

Clearly, the integration of the above requirements into the development of a core for embedded systems program requires a concerted effort in bringing the latest research results in the academic and industry realms. In addition, the practical aspects of the exposure cannot be gained by classroom experience alone but requires the students to work at a production environment under the mentorship of industrial and academic supervisors.

The Consortium for Embedded and Internetworking Technologies (CEIT) at Arizona State University, established in April 2001, is a partnership that formed between Arizona State University, the Intel Corporation, and Motorola to address the need for high quality education and research in the area of embedded systems. The consortium is composed of research, education, and executive committees that are responsible for defining short and long-term research, education, and consortium missions, respectively. One of the primary goals of this consortium is to emphasize high quality education by bringing in the latest research and practical expertise into the classroom. That is, the specific mandate of the consortium is to design an undergraduate program that is founded on a core derived from research results. By involving the faculty in research projects funded by the consortium, the focus of research faculty attention is drawn towards certain thematic areas of interest of importance to ASU, Intel, and Motorola. A centerpiece of the consortium is the inclusion of internships funded by the industry members which expose students to applied projects mentored by industrial supervisors and faculty fellows.

4 Proposed Program

4.1 Overview

The long-term goal of the curriculum project described in this paper is to introduce a new program in Embedded Systems Engineering as shown in Figure 1. The long-term goal of the curriculum project described in this paper is to introduce a new program in Embedded Systems Engineering as shown in Figure 1. The first step in achieving the ultimate goal of a full-fledged program in Embedded Systems Engineering is to successfully design a concentration track in Embedded Systems under the umbrella of the existing core of Computer Engineering. This concentration track relies upon three established premises. Namely, the use of an industry-university collaborative model (as stated in Section 3), the development of state-of-the-art course content and laboratories, and the development of internship-based capstone projects. The curriculum specific aspects of this project are summarized in Figure 2, where formal coursework (e.g., Embedded Systems Development and Embedded Systems Engineering) is coupled with the internship-based capstone projects.

The industry-university collaborative model:

In many areas of computing science, much of the research innovations originate from academic circles. In the area of embedded systems development and practice, industry has a clear lead. With the consortium of industry members committed to making the ASU program a definite success, we have the unique opportunity for our educational program to be targeted towards training students and engaging faculty in leading edge technological breakthroughs in the area of embedded systems.

Creation and delivery of state-of-the-art course content and appropriate laboratories

As presented in Section 4.2, an embedded systems core that addresses the depth and breadth of research will be designed. Students will benefit from the state-of-the-art laboratory experiences in addition to benefiting from the latest content. Consortium teams, made up of faculty and engineers from member companies, will design the laboratories, which will be mostly equipped with cutting-edge technologies.

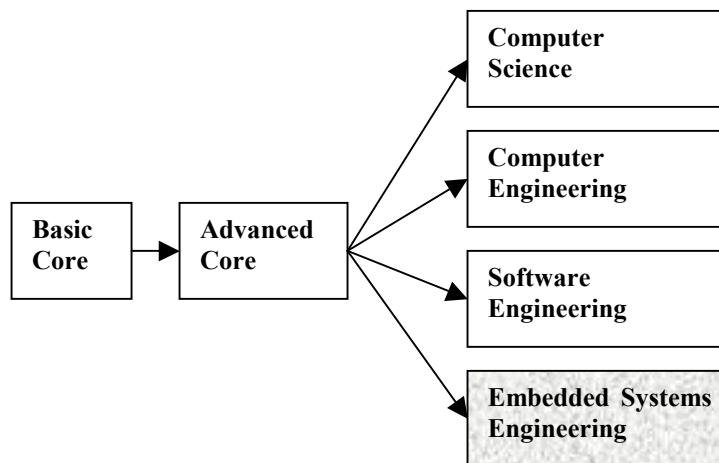


Figure 1: Degree Programs and Concentrations

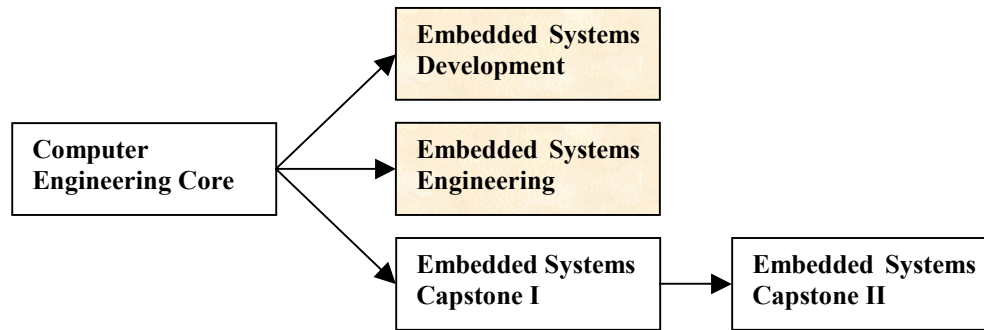


Figure 2: Embedded Systems Required Courses

Capstone project through internships

Also included in the program are mandatory internships for implementation of the capstone project sequence, which augment students' academic experience. A formal program is under development, details of which are presented in Section 4.2.3. Another important aspect of the collaboration is the mentor development program, which enables industry engineers and team leaders to properly assist in achieving student learning objectives, particularly, during the implementation of capstone projects. These mentor development programs pave the way for a meaningful and effective participation of practicing engineers to be involved in the training and education of students enrolled in this program.

4.2 Curriculum Design

The innovation of our educational approach is the integration of the latest research from academia (including our own work) and industry (through focused research projects) in the areas of real-time systems and software engineering, such as partition based system integration [4,5,6]. In this section, we discuss the details of the research integration leading to the creation of modules organized in the form of the core courses that would constitute the foundation of a concentration track in embedded systems. From the embedded systems perspective, there are many concerns that impact the construction of applications and systems. These concerns include the identification of the desired system behavior and constraints (e.g., quality attributes), design and specification of a system architecture, development of the components of the system architecture (consisting of both hardware and software), integration of the developed components into a system solution, and testing of the end product. In addition, a good understanding of, and experience with, embedded systems development projects and tools is necessary to couple knowledge of the latest research with current practices.

4.2.1 Design of Embedded Systems

As the design of embedded systems shifts to integration and component-based development, multiple applications of different critical levels and performance requirements will co-exist in a single hardware or software platform. This brings up the need for spatial and temporal partitioning [4,5,6]. The partition-based approach for embedded systems design will result in a well-defined interface to ensure service qualities and protection among various software components. It will have a profound impact on system upgrades and maintenance. This course is being developed to provide a learning environment for software and system development of

embedded systems. The course will provide opportunity for students to learn fundamental issues as well as practical development in the area of embedded computing. Various disciplines of computer sciences, such as system specification, programming languages, operating systems, architecture and hardware will be discussed with an emphasis on the integrated design of embedded systems. In addition, the course will cover design principles, abstraction, primitives, examples of existing programming languages, real-time kernels, and architectures. Laboratory projects on data acquisition and embedded controlling systems, such as real-time kernel enhancement, real-time clock server design, embedded systems software development, instrumentation, and hardware/software co-design will be included.

The intended outcomes for this course are as follows:

- The students will be able to comprehend the organization of various components of an embedded system.
- The students will be able to develop embedded applications using off-the-shelf software and hardware components and production tools.
- The students will be able to assess the cost effectiveness of various design alternatives.

From the research perspective, the course will examine the micro-kernel design for spatial and temporal partitioning in order to eliminate any interference across partition boundaries [6]. The scheduling approach for partitioned systems [4,5] and under energy constraints [7] will be addressed accordingly in the course. Software engineering topics to be covered include component-based modeling, integration and mediation of object interactions via software frameworks, verification and validation approaches for timing constraints, and support for application infrastructures using middleware.

The classroom material will be supplemented with significant hands-on experience and will involve the construction of a laboratory development platform that includes PowerPC and StrongARM-base single processor boards, Wind River's Tornado development system, National Instruments' Labview, and other additional peripheral devices. The equipment and software will allow us to establish a simulated embedded systems testbed, as shown in the Figure 3. As such, the laboratory will be flexible enough to facilitate the creation of test cases and experimental scenarios. Using host systems and target controllers, experiments that demonstrate real-time data acquisition, sensing, and actuator control can be accomplished while various software and system components for industrial automation and computer-based instruments are developed.

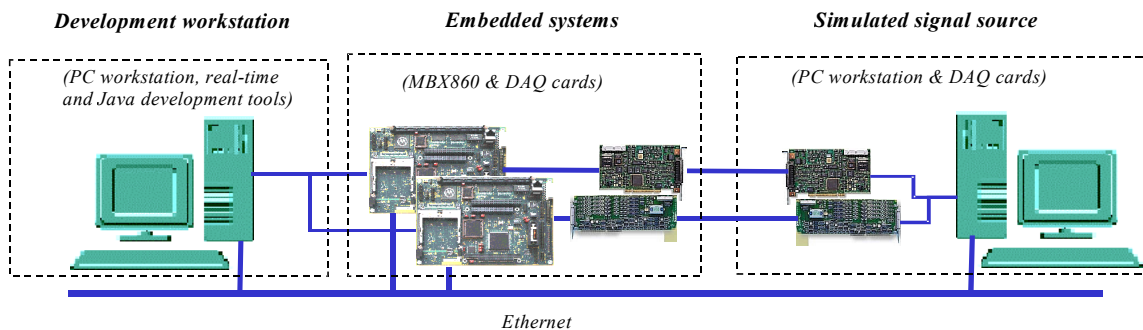


Figure 3: Proposed Laboratory Development Environment

4.2.2 Embedded Systems Engineering

The second course in the sequence focuses on embedded systems engineering. Specifically, the course will provide students with knowledge and experience on how current research in the areas of software and system architecture can be applied to the development of embedded systems. In addition, the course will stress the importance of constructing systems using component-based methodologies for integrating both hardware and software. This course, which will draw from the knowledge and skills gained from the first course, will focus on end-product construction and testing while examining critical properties often associated with embedded systems. The intended outcomes for this course are as follows:

- The students will be able to understand and address (within a software architecture) the critical issues most often associated with embedded software including high availability, performance, integrity, survivability, reliability safety, and predictability.
- The students will be able to model and analyze a system in order to observe system characteristics and the system architecture level.
- The students will be able to develop systems that utilize both hardware and software components.
- The students will be able to perform testing at different levels such as the component, integration, and system levels.

The course is divided into three topic areas. Each topic area addresses a critical need in embedded systems development including 1) the introduction and incorporation of multiple quality attributes in a system design, 2) specification and modeling of system architectures, and 3) system integration and testing of the end products.

Quality Attributes

A quality attribute is defined as any system feature that can be measured [8]. Within the context of embedded systems, the quality attributes of highest concern include availability, fault-tolerance, survivability, safety, and predictability as well as traditional attributes such as behavior, maintainability, and reusability. Many techniques have been developed to address non-behavioral quality attributes including approaches for safety analysis [9,10] as well as statistical models for measuring availability within system and software architectures [11]. Recently, an approach for identifying tradeoffs in software architectures has been developed in order to cope with the need to incorporate several quality attributes that may be both complementary and conflicting [12]. In typical computer science degree programs, with respect to system development, the focus has centered primarily upon the required behavior of various components, both hardware and software, with some attention paid to optimization. However, for computer science graduates to effectively develop systems that are, for instance, available, safe, and fault-tolerant, a paradigm shift is required that focuses upon systems as a whole with the philosophy that non-standard quality attributes are equivalent in importance to behavior and functional requirements.

In this course, we intend to introduce and cover topics that emphasize the fact that embedded systems solutions must address many different concerns at the system architecture level. To this end, we wish to incorporate the use of analysis techniques for high availability, fault-tolerance, survivability, safety, and predictability along with the traditional approaches for ensuring

correctness. Use of these techniques will provide students with the critical and analytical abilities needed to weigh tradeoffs that exist at the system level.

System Architecture

One of the most important activities in the context of embedded system development is the analysis and specification of a system architecture that incorporates both hardware and software components. In a system architecture of this sort, the overall solution to a given problem is seamless in its depiction of interfaces between software and hardware. The partition between the two is often identified either based upon the available technology or upon the decisions that best optimize the concerns of the overall system. Accordingly, the ability to design an architecture that focuses on abstraction while facilitating analysis of many quality attributes, including behavior and availability, is paramount. One approach for architecture specification and analysis that has recently gained attention is model checking. Model checking is a “lightweight” formal method that relies upon the use of formal modeling to express the behavior of a system. The enabling feature of model checking approaches is the fact that the models that are constructed are conducive to automated analysis whereby exhaustive state space search can be performed in order to verify several different properties that may be exhibited by a system. Such an approach is contrasted to traditional formal methods techniques that are dominated by user-intensive proofs. Model checking has been used for modeling and analyzing systems in several different domains including the aerospace industry [13,14] as well as for specifying and verifying hardware systems [15] and software systems [16].

In our previous research, we have developed an approach for specifying and analyzing the software architecture for embedded software used to control a space-based interferometry system [14,17]. The approach, based on the use of both manual and automated analysis of software architectures, stresses the use of state-based specification to describe behaviors. In addition, the approach uses model checking to determine reachability and safety assertions to verify support of various quality attributes. In the proposed course, we intend to introduce students to techniques for modeling the state-based behavior of embedded systems. Some of the possible projects include the state-based specification of embedded controllers for systems such as home heating systems, elevator control systems, and steam boiler controllers. For these projects, students will specify the behavior using by identifying states, events, and transitions between states. Furthermore, students will identify and specify properties as well as verify those properties using automated analysis tools that perform exhaustive state exploration to determine reachability and system safety. In the context of these projects, students will learn topics such as linear temporal logics for describing real-time properties of systems.

In addition we will incorporate the use of tools that facilitate the simulation and analysis of the models. The value of these tools and techniques is that they provide support for verifying several properties through the use of safety assertions as well as state-space search. In addition, the ability to simulate the models provides a mechanism for observing system behavior without incurring the cost of producing the system.

System Integration and Testing

Once software and hardware components are constructed, the main development effort is shifted from design and implementation to system-level integration and test. The primary concerns

include the preservation of consistency between component interfaces as well as correct behavior. System-level integration and test is an enormous task since it evolves the entire system, external interfaces and the environment. To address these issues in the curriculum, we will place an emphasis on the following two approaches: incremental integration with progressive and systemic testing, and white-box testing of temporal properties.

Various integration strategies have been investigated since 1970's with the most effective technique being incremental integration, either bottom-up or top-down approaches. To perform incremental integration and progressive test at each integration stage, we plan to use the proposed laboratory development environment to practice the construction of event simulators, drivers, and stubs. In addition, we will cover the construction of test dependency graphs in order to assist integration and regression testing. In terms of system level scheduling, even if the target system is not ready, suitable resource utilization requirements and timing constraints of each partition can be addressed.

In addition to the general approaches of software testing, this course will cover issues related to the temporal dependencies of program behavior. These dependencies occur due to asynchronous operations and events. In the proposed course, we will first illustrate this dependency through program dependency analysis with the intent of revealing race conditions in feasible execution paths. Using a white box testing strategy and program instrumentation, different arrival instances of external events will be analyzed and the coverage information measured.

4.2.3 Embedded Systems Capstone

It is our intention to expose our students to techniques that are gaining increased use in the embedded systems industry. It is our belief that there is no better way to infuse industry research and practice other than having the students placed directly in an environment that creates and fosters applied research. There is no better way for our students to gain insight into the production environment other than participating in development projects that are similar to those that they will experience when they graduate.

To address these goals we will establish the embedded systems capstone courses based on an internship program that allows our students to participate as team members on research and development projects with professional developers at local industry. By interacting with qualified and dedicated industrial partners who fully support our concern for making the internships educationally valuable, we plan to develop opportunities that go beyond offering part-time jobs.

The primary outcome of each capstone project will be to provide students with practical experience that serves as a reinforcement of the topics and techniques provided by the two core courses described above. In addition, the internship experience will help the student develop team coordination and communication skills that are difficult to attain in a typical academic environment. Each internship will include team-based product development assignments on the design and development of new, state-of-the-art applications of embedded microprocessors and microcontrollers, supervised by faculty and industry mentors.

As such, we view the capstone projects through internships as an activity that is complementary to the formal course offerings for the embedded systems program. Specifically, the capstone

projects serve the role of providing experience that supplements the formal education in the fundamentals of embedded systems engineering.

5 Conclusion

To date, development of the internship-based capstone program has resulted in a piloting of the program with twenty students in the Summer and Fall Semesters of 2001. The capstone program relies on a partnership between faculty and industry mentors whereby a clearly defined statement of work that emphasizes both academic and industrial goals is used to define the boundaries of student activity. Each semester culminates in a live presentation by students in a mini-conference held at ASU. To prepare both industrial and faculty mentors, training sessions are held that provide the knowledge required to manage the interactions between students, faculty mentors, and industrial mentors.

Formal coursework development is occurring in stages with a pilot offering of the Embedded Systems Development course planned for Fall 2002 and a pilot offering of the Embedded Systems Engineering course planned for Spring 2003. In addition, a number of other curricular activities related to the CEIT are being funded and delivered [18].

The next generation of embedded communications systems will connect and interact through a futuristic, lightning-fast and intelligent network infrastructure. The need for graduates with the ability to analyze and develop software and hardware solutions for these infrastructures is recognized to critical for companies such as Intel and Motorola. The curriculum development activities described in this paper are intended to addresses these critical needs while also exposing undergraduates to the latest research for embedded systems. Future investigations, in addition to offering the courses and launching the concentration and degree programs include the dissemination of developed materials to potential university partners in order to determine the applicability of the curriculum model in other academic environments.

Acknowledgements

This work is supported by NSF Educational Innovation Grant EIA-0122600. The authors would like to thank the industry consultants that provided the initial feedback and suggestions regarding the establishment of this work . In addition, the authors would like to acknowledge Shlomo Pri-Tal, David Dannenberg, John Robertson, and Charles Lipari for their contributions to the work.

References

- [1] Department of Computer Science and Engineering, and Department of Electrical and Computer Engineering, Michigan State University, "Visions of Embedded Systems Laboratories", online available - <http://www.egr.msu.edu/vesl/>
- [2] Department of Information and Computer Science, University of California at Irvine, Graduate Degree Programs, MS Concentration in Embedded Systems, online available - <http://www.editor.uci.edu/00-01/ics/ics.2.html>
- [3] B. Huey, S. Pri-Tal, D. Dannenberg, J. Robertson, "An Arizona Ecosystem for Embedded Systems", in Proceedings of the IEEE International Performance, Computing and Communication Conference, April 2001.
- [4] Yann-Hang Lee, Daeyoung Kim, Mohammed Younis, and Jeff Zhao, "Scheduling Tool and Algorithm for Integrated Modular Avionics Systems," *International Conference on Dependable Systems and Networks (FTCS-30 and DCCA-8)*, June 2000.
- [5] Daeyoung Kim and Yann-Hang Lee, "DC² Scheduling for Aperiodic Tasks in Strongly Partitioned Real-Time Systems," *The Seventh IEEE International Conference on Real-Time Computing Systems and Applications*, Dec. 2000.
- [6] Daeyoung Kim, Yann-Hang Lee, and Mohamed F. Younis, "SPIRIT- μ Kernel for Strongly Partitioned Real-Time Systems," *The Seventh IEEE International Conference on Real-Time Computing Systems and Applications*, Dec. 2000.
- [7] C. M. Krishna and Yann-Hang Lee, "Voltage-switching scheduling algorithms for low power in hard real-time systems," *IEEE Real-Time Technology and Application Symposium*, May 2000.
- [8] Len Bass, Paul Clemens, and Rick Kazman, "Software Architecture in Practice", Addison-Wesley, 1998.
- [9] Nancy G. Leveson, "Safeware: System Safety and Computers", Addison-Wesley, 1995.
- [10] Robyn R. Lutz, "Software Engineering for Safety: A Roadmap", in A. Finkelstein, ed., *The Future of Software Engineering*, ACM Press, 2000.
- [11] Steps in an Architecture Tradeoff Analysis Method: Quality Attribute Models and Analysis. Barbacci et al, Software Engineering Institute Technical Report CMU/SEI-97-TR-029.
- [12] R. Kazman, M. Klein, M. Barbacci, H. Lipson, T. Longstaff, and S. Carriere, "The Architecture Tradeoff Analysis Method", In Proceedings of ICECCS, 1998.
- [13] K. Havelund, M. Lowry, and J. Penix, "Formal Analysis of a Space Craft Controller using Spin", In proceedings of the 4th International SPIN Workshop, 1998.
- [14] Gerald C. Gannod, Robyn R. Lutz, and Marian Cantu, "Embedded Software for a Space Interferometry System: Automated Analysis of a Software Product Line", in Proceedings of the 20th IEEE International Performance, Computing, and Communications Conference (IPCCC 2001), April 2001
- [15] Peter van Eijk, Verifying Relay Circuits using State Machines, In proceedings of the 3rd International SPIN Workshop, 1997.
- [16] Moataz Kamel and Stefan Leue, Validation of Remote Object Invocation and Object Migration in CORBA GIOP using Promela/Spin. In proceedings of the 4th International SPIN Workshop, 1998.
- [17] Gerald C. Gannod and Robyn R. Lutz, "An Approach to Architectural Analysis of Product Lines," Proceedings of the 22nd International Conference on Software Engineering, June 2000.
- [18] Consortium for Embedded and Internetworking Technologies, online available, <http://www.eas.asu.edu/embedded/>.

Biographical Information

GERALD C. GANNOD is an assistant professor in the Department of Computer Science and Engineering at Arizona State University. He received the MS ('94) and PhD ('98) degrees in Computer Science from Michigan State University. His research interests include software product lines, software reverse engineering, formal methods for software development, software architecture, and software for embedded systems. He is a recipient of a 2002 NSF CAREER AWARD.

FOROUZAN GOLSHANI is a Professor of Computer Science and Engineering at Arizona State University. His areas of expertise include multimedia computing and communication, particularly video content extraction and representation, and multimedia information integration. He is currently the Editor-in-Chief of IEEE Multimedia and serves on the editorial boards of a number of other journals. He has published more than 120 technical articles in books, journals, and conference proceedings. He received a PhD in Computer Science from the University of Warwick in England.

BEN HUEY is an Associate Professor and Associate Dean for Planning and Administration, College of Engineering and Applied Science. He received a PhD from the University of Arizona. His research interests include language-based models for architecture, silicon compilation, design verification, and automatic test generation.

YANN-HANG LEE is a Professor of Computer Science and Engineering at Arizona State University. He received a PhD from the University of Michigan. His research interests include real-time systems, computer communication, computer architecture, fault-tolerant computing, distributed/parallel systems, and performance evaluation.

SETHURAMAN PANCHANATHAN is a professor of Computer Science and Engineering at Arizona State University and a Fellow of the IEEE and SPIE. Dr. Panchanathan is the Director of the Research Center for Ubiquitous Computing, the Chair of the Embedded Systems Research Committee at ASU and the principal investigator for the project described in this paper. He has published over 200 papers in refereed journals and conferences.

DAVID PHEANIS is an Associate Professor in the Department of Computer Science and Engineering at Arizona State University. He has won the ASU College of Engineering Award for Outstanding Undergraduate Teaching, and he has also won the Burlington Award for Outstanding Faculty Achievement. He was the general chairman of the International Conference on Computers and their Applications in 1997.