Modernization of Teaching in Embedded Systems Design—An International Collaborative Project

Saeid Nooshabadi and Jim Garside, *Member, IEEE*

***Abstract*—This paper describes the process of review, design, and delivery of a course in modern embedded systems, an international collaborative teaching project involving the University of New South Wales (Australia), Manchester University, and Imperial College, London University (United Kingdom). This project, being the first of its kind anywhere in the world, provides a learning environment that replicates the current industrial practice in embedded systems design in an easy and comprehensible setting, an environment where the processor, dedicated coprocessors, and software are all integrated to create a functional system such as used in sophisticated electronic devices, including mobile phones, Web phones, televisions, digital cameras, and personal digital assistants. Such collaborations are important in both reducing development costs in developing up-to-date, and increasingly sophisticated, courses and in addressing pedagogical issues that are common between computer and electrical engineering programs in all academic institutions. To assist students' learning experience, the course is supported with purpose built state-of-the-art programmable hardware and software development platforms, carefully planned laboratory experiments, lecture notes, weekly online quizzes, tutorials, and a companion CD-ROM as a learning tool. Since the introduction of this complete package, students' satisfaction, assessment results, and skills obtained through evaluation and assessment methods have improved markedly.**

***Index Terms*—Embedded systems teaching, international collaboration in teaching, microprocessor design teaching, programmable hardware development board.**

I. Introduction

INCREASINGLY, computer and electrical engineers are involved in the design of computer-based embedded systems to address highly-specialized and specific applications in the computer, aerospace, telecommunications, power-production, manufacturing, defense, and electronics industries. They design high-tech devices ranging from tiny microelectronic integrated-circuit chips to powerful systems that incorporate those chips with efficient telecommunications to form complex interconnected systems. Applications include consumer electronics (CD and DVD players, televisions, stereos, gaming devices), advanced microprocessors, peripheral equipment, systems for portable, desktop, and client/server computing, and communications devices (cellular phones, pagers, personal digital assistants). They also include distributed computing environments (local- and wide-area networks, wireless networks, intranets, Internet) and embedded computer systems (such as aircraft, spacecraft, and automobile control systems, in which computers are embedded to perform various functions) [1]–[4].

Education and training should reflect trends in embedded systems design [2]. Unfortunately, in many universities, the typical content in an introductory course in embedded systems and microprocessor design is similar to that of technical institutes, using an 8-bit processor to teach programming in assembly language, implementing trivial interfacing to the outside world on a prototyping board, and performing using simple control and measurement experiments [5]. The practice at the University of New South Wales (UNSW) in teaching embedded systems was similarly outdated and had not kept pace with the rapid changes in the industry practice and expectations.

Modern real-time embedded systems often employ dedicated hardware in addition to microcontroller-based software to perform tasks that are very demanding and require a great degree of design sophistication [3]–[5]. These embedded systems need to handle a mix of data-intensive and control-oriented tasks and may need architectural support for multitasking, multithreading, concurrency, polling, interrupts, and preemption, as well as user and supervisor modes of operation [5]. Until recently, these features have been the domain of desktop PCs and more powerful specialized computers. Conversely, modern real-time embedded systems in most instances are mobile in nature and need to be energy-aware to conserve battery power [4] and resource-sensitive to reduce cost [6].

Although the field of computer engineering is over four decades old, it is still changing rapidly in response to technological advances and innovation. Given the dynamic nature of the computing and telecommunication industries, designing and continually improving courses for electrical and computer engineering students is highly challenging. Nevertheless, regular updates of the learning environment in an embedded systems course are imperative. Indeed, the relevance of the course to professional practice is particularly important to electrical and computer engineering students [2], [5].

This paper details the authors' efforts in the review and redesign of an embedded systems course and in the modernization of the Digital Systems Laboratory (DSL) at UNSW. This laboratory is used by up to 500 students every year. The guiding principles in modernizing the DSL for teaching embedded systems were to emphasize professional practice and engineering design, problem solving, and critical thinking skills, communication skills, and teamwork through the design and execution

Manuscript received June 12, 2005; revised August 23, 2005.

S. Nooshabadi is with the School of Electrical Engineering and Telecommunications, University of New South Wales, Kensington, Sydney, NSW 2052, Australia (e-mail: saeid@unsw.edu.au).

J. Garside is with the School of Computer Science, University of Manchester, Manchester M13 9PL, U.K.

Digital Object Identifier 10.1109/TE.2006.872402

of well-structured and planned courses, where theory and laboratory experiences are seamlessly integrated with the latest industry practices [7].

Section II discusses the objectives of the course review and design in terms of graduate attributes and pedagogical issues. Section III describes the process of course development and laboratory modernization, including the development of hardware boards, software tools, laboratory exercises, lecture notes, and support material. Section IV describes the teaching methods employed to enhance the students' learning outcomes and achieve the desired graduate attributes. Section V describes the course assessment methodology that is based on sound pedagogical principles. Section VI describes the process of course evaluation. Section VII concludes the paper.

II. Educational Objectives

A. Graduate Attributes

The rationale for the course review and redesign was for students to acquire the following graduate attributes [2]:

- to possess the ability to design computers and computer-based systems that include both hardware and software to solve novel engineering problems, subject to tradeoffs involving a set of competing goals and constraints;
- to be able to use their breadth of knowledge in mathematics and engineering sciences that is associated with the broader scope of engineering and apply it to the narrow field of computer-based systems design;
- to have the ability to understand the issues, possibilities, and limitations of mapping and to represent a mathematical algorithm associated with an engineering theory into a format suitable for computer-based systems;
- to acquire and maintain a preparedness for the professional practice in engineering;
- to be able to design or redesign a product, a process, or a system to meet desired needs;
- to have the ability to handle the essential engineering and computing details, while always keeping the "big picture" in mind.

B. Course Development and Design Objectives

To accomplish these learning outcomes for students, the following objectives were pursued.

- Create an environment for the systematic and creative application of scientific and mathematical principles to the efficient design and operation of computer-based structures, processes, and systems [2].
- Create a setting where students can identify deficiencies in an existing solution and try novel ideas to improve it.
- Create an environment where the task of design is fundamental and central.
- Prepare educational materials that have considerable challenging content, while at same time engage students to think and discover for themselves [8].
- Create an environment in which students will feel that this particular course they have taken is the best course ever.

C. Pedagogical Issues

The process of course design and delivery was inspired by UNSW's "Guidelines on Learning that Inform Teaching" [9]. These guidelines are to enhance students' learning experiences through design and teaching an engaging, contextualized [10], and inclusive curriculum. The specific guidelines used were as follows.

- Effective learning is supported when students are actively engaged in the learning process.
- Effective learning is supported by a climate of inquiry, where students feel appropriately challenged and activities are linked to research and scholarship [8].
- Learning is more effective when students' prior experience and knowledge are recognized and used [10].
- Students become more engaged in the learning process if they can see the relevance of their studies to professional, disciplinary, and/or personal contexts [11].
- If dialogue is encouraged between students and teachers and among students (in and out of class), thus creating a community of learners, student motivation and engagement can be increased [2], [5], [12].
- Clearly articulated expectations, goals, learning outcomes, and course requirements increase student motivation and improve learning [13].
- Learning cooperatively with peers, rather than in an individualistic or competitive way, may help students to develop interpersonal, professional, and cognitive skills to a higher level [5], [7], [12].
- Effective learning is facilitated by assessment practices and other student learning activities that are designed to support the achievement of desired learning outcomes [14].

III. Modernization of the Digital Systems Laboratory

A. Search for a Suitable Platform

To address the educational objectives and pedagogical issues described in the previous section, in 2001, comprehensive research into current learning and teaching practices in the field of computer engineering and consultation with industry leaders were performed. In addition to examining curricula at other universities, the research was based on the materials published by the IEEE and the Association for Computing Machinery on computing curricula/computer engineering [2]. This research and the previous experience in teaching microprocessor design confirmed that the most effective learning environment for electrical and computer engineering students is one based heavily on laboratory experience [2], [15], [16] and one that includes use of industry-standard and modern tools [17]. The educational objective of bringing the design to the center stage inevitably means hiding some of the "grimy details" of implementation with a comprehensive modern development tool-chain.

One particular concern identified by the research and survey of existing curricula was the prevalent use of learning tools employing outdated technology [5]. One clear observation was that many courses in microprocessors and interfacing used an 8-bit

processor to teach students how to program in the assembly language and to gain experience in the interfacing on simple prototyping development boards. This practice was common because learning tools with 8-bit processors are inexpensive and easy to assemble, and until the mid-1990s, most embedded computer systems were deployed in devices with minimal processing requirements, such as washing machines and microwave ovens. However, many contemporary embedded computer systems employ a 32-bit processor and, often, coprocessors to perform demanding tasks that require a high degree of sophistication in software hardware codesign to handle a mix of data-intensive and control-oriented tasks. In this context, the learning tools used by students should categorically reflect such sophistication in the design of embedded systems [2], [4], [5].

The challenge was to find a way of creating a cost-effective learning tool incorporating a 32-bit processor and a programmable hardware support to help students attain a holistic understanding of embedded computer systems.

From the study of the current trends in the embedded systems, the evidence emerged that many contemporary embedded systems use the 32-bit advanced RISC machine (ARM) [18] processor. Due to its low power consumption, relatively high performance, and versatile instruction set, the ARM processor has become one of the industry's most popular. In fact, over 80% of all sophisticated electronic devices with 32-bit RISC core are based on the ARM processor [19], [20]. The ARM, at least in its basic incarnation, also has a reasonably simple and straightforward instruction set, free from many of the idiosyncrasies of more limited microcontrollers and relatively easy for students to use.

On the other hand, programmable hardware support (for example in building coprocessors) is now readily available in a cost-effective manner from the field programmable gate array (FPGA) vendors, such as Xilinx. Thus, providing gate arrays for students' use was essential.

B. International Collaboration and the Development of an Effective Learning Tool

To keep students engaged, instructors need to show the relevance of their studies to professional practices in the industry [11], [17]. To expose students to a state-of-the-art hardware and software development platform that is based on a processor that is widely used by the industry, in the final quarter of 2001, the UNSW development team embarked on the task of course development for modern embedded systems that was based on an ARM processor and programmable logic arrays [18].

Although the ARM processor can be emulated, a programmable development board was desired to experiment with hardware which allows students to see how things work "in the real world." In particular, learning the "ins" and "outs" of a hardware platform at the lower level gives students an appreciation of computer architecture that just programming using simulators does not [21]. Understanding the mapping between the higher level language and the hardware and its associated resource implications for embedded systems is a fundamental issue. Learning how some of the higher level concepts, such as operating systems (OSs), task scheduling, and preemption, actually relate to the underlying architecture support and instruction set is highly important for any system designer [5]. Furthermore, the use of "real metal" gives a more positive "feel" to the laboratory.

An essential feature of the development board was to include programmable logic and networking devices [4]. The inclusion of programmable logic arrays was necessary because hardware/software partitioning and codesign is an important issue in computer engineering, perhaps particularly in embedded systems [4]–[6]. For example, a mobile phone with a limited power budget, houses a general purpose processor for "housekeeping" functions and an application specific hardware to perform all the demanding real-time digital signal-processing functions [4]. Programmable logic arrays now allow students access to a significant uncommitted hardware resource in the same way that microprocessors revolutionized software teaching [16]. Thus a combination of a processor and programmable logic arrays creates opportunities for experiments in the tradeoffs of partitioning an algorithm between the hardware and software.

An approach was subsequently made to ARM Ltd. to gauge their interest in participating in the development of a learning tool. Although the company declined the offer, the liaison between UNSW and ARM helped establish a collaborative network between UNSW, the University of Manchester (UM), and Imperial College (IC), London University. The two British institutions were addressing similar learning and teaching issues for their courses on embedded systems. One quickly realized that many advantages existed in fostering a collaborative network among the three universities.

An agreement was established among the three universities collaboratively to design and manufacture an ARM-based development board for use as a learning tool in the laboratory exercises [18]. Since existing commercial development boards with the desired features were prohibitively expensive, one of the advantages of designing and manufacturing a purpose-built learning tool was achieving economies of scale by increasing the size of the production run [18]. More important, the collaborative network facilitated the sharing of results, tests and trials, ideas for the laboratory exercises, laboratory manual, software, and other support materials to accompany the new development tool that assists or would assist learning.

C. Development of Hardware Platform

During the 21-month development period of the project, the design teams at UNSW and UM went through several stages of rigorous consultation regarding the desired features of the development board, the choice of processor, FPGA chips, and other peripheral devices [18].

The outcome of the joint effort among the three institutions was the design and manufacture of an ARM-based embedded development board, augmented with two Xilinx programmable logic devices for specialized peripheral I/Os and dedicated data processing, an Ethernet chip for computer networking, and a host of other features. The main board was extended with an auxiliary extension board for easy future development and expansion.

Fig. 1 provides a systems-level view of the complete hardware development platform (called DSLMU). It comprises two

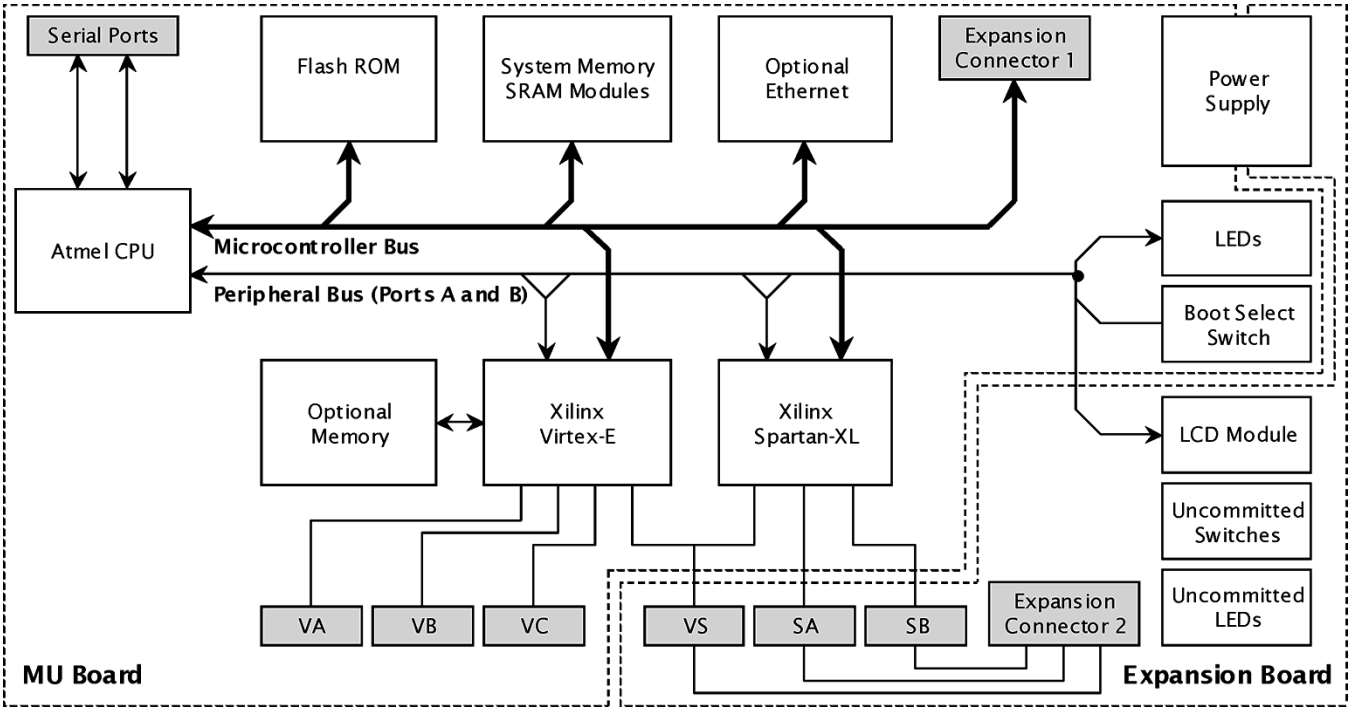

Fig. 1. Systems-level view of the complete DSLMU hardware development platform.

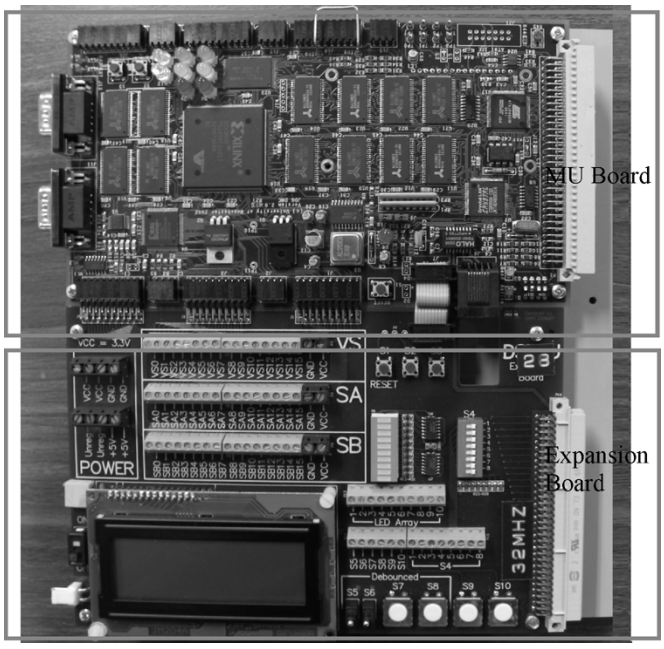

Fig. 2. DSLMU hardware development board with the main board at the top and the expansion board at the bottom.

connected boards (Fig. 2). Details of the development board including the schematics and reference manuals are publicly available in [22]. The main features of the development board are as follows.

- *Main processor:* The MU microcontroller board is based on an ARM7 processor core. The particular microcontroller used is the Atmel AT91R40008 device.
- *System memory:* Connected to the microcontroller bus is the system memory in the form of 4 MB of static RAM and 2 MB of flash ROM.
- *FPGAs:* The main peripherals connected to the microcontroller bus are the two Xilinx FPGAs: Virtex-E and Spartan-XL. These FPGAs are designed to be reprogrammed to become any sort of peripheral coprocessors that may be needed. Two FPGAs were used as a cost/capacity tradeoff, the need for which has receded since the board was designed, because of substantial reduction in the cost of FPGAs.
- *Connected Peripherals:* The DSLMU microcontroller board has a number of peripherals, as shown in the system block diagram. These peripherals include the serial ports, the light-emitting diodes (LEDs) on the MU board, the liquid crystal display module, and the Ethernet controller.
- *Uncommitted Peripherals:* The expansion board contains a number of uncommitted switches and LEDs. These I/Os are "uncommitted" in the sense that they are not connected to anything. Students can use these peripherals in their experiments, either connecting them to the FPGA input/output pins or to their own circuits.

D. Development of Software Tools

As the design of the development board was progressing, attempts were made to select a suitable suite of software development tools. Three options were considered: using ARM Ltd.'s proprietary tools, using third-party tools, or using the free and popular GNU tools sponsored by the Free Software Foundation.[1] A decision was made to use the free and far more accessible GNU development tools. Another big educational motivation for using GNU tools is that it is a single tool-chain that can

[1]http://www.fsf.org/

be used on almost any platform in existence. GNU tools can be used with any host system (Linux/Unix, Windows, Mac OS X, etc.), almost any architecture (x86, ARM, MIPS, SPARC, SH, etc.), and many target platforms (bare hardware, as DSLMU board, or simulation on the host platform). Therefore, use of GNU tools allows students to transfer their skills to another platform very easily.

The ARM microcontroller employed on the development board is a complex processor, and its proper configuration requires understanding of many low-level (and machine-specific) details that are beyond the scope of an introductory course. The system configuration is achieved by initializing the machine through boot code installed on the flash ROM. In addition, a pedagogically desirable feature was to provide a completely blank view of the ARM core where the user has full control, this objective achieved by building an emulator layer termed "kmd" on top of the bare hardware. After completion of the boot sequence, the emulator layer is executed. The kmd emulator provides the following features:

- a primitive OS, safeguarding critical system resources such as flash ROM from deletion and alteration;
- a simple debugging environment;
- a simple serial communication link with the host computer for loading user programs and passing of debugging information;
- abstraction of low-level details that are machine-specific;
- easy communication with the FPGA chips for the purpose of downloading of configuration design file and interfacing with them during program execution.

To interface with kmd at the host Linux front-end, a debugging tool termed *Komodo*[2] was designed by the UM team and later modified by the UNSW team. Features of *Komodo* include easy access to the target board via the kmd emulator layer for downloading of the user program, FPGA configuration file, and debugging.

The board's ROM also contains other software options, such as a back-end for ARM's proprietary debugging tool *Angle* (GNU tools can also use this option) and GNU real-time embedded configurable operating system (eCos). Although not used in the introductory laboratory, these software options allow access to the full speed/power of the board at the price of less user-friendly access. These options are, however, used for student projects.

E. Development of the Laboratory and Lecture Materials

As the hardware and software platforms were coming together, the development of the laboratory manual and documentation progressed in parallel. In accordance with the educational objective of Section II, simple commercial and bare-bones prototype boards were used to craft carefully a set of laboratory experiments. Laboratory exercises involve detailed study and interfacing of an ARM-based microprocessor system. Experiments explore concepts of function implementation, arithmetic computations, memory mapped I/O interfacing using polling and interrupts, and privileged and user modes of operation and their role in exception and interrupt handling using OS calls. Other important components of laboratory exercises include using interrupts and I/O timers to perform sampling, building linked list and circular buffers, and interfacing to functional units implemented on the FPGAs. The pedagogically positive aspect [13] of the laboratory exercises is that they are well written and have clear goals. Students know exactly what is required of them at any time.

Concurrently, a set of lecture slides and notes, tutorial sets, and weekly online quizzes were designed to support the practical experiments [23]. Lecture notes cover topics on a programmer model of computer organization using the assembly and machine languages, the instruction set, the process of translation from high-level language to machine instructions, number representation, computer arithmetic (including floating points and fractional arithmetic and the instruction set for support), I/O interfacing, programming I/O interrupts and exceptions (including architectural support for them), memory management and protection and architectural features to support them, the role of OS in handling exceptions, multitasking, and multithreading, the use of interrupts for sampling, linked lists, and circular buffering.

F. Development of the Companion CD-ROM

Since aspects of development were pieced together, the team at UNSW decided to publish all course-related materials on a comprehensive CD-ROM to assist student learning outside the laboratory, an important educational objective [24]. The companion CD-ROM includes everything the student would need to work from home, apart from the actual hardware, of course. Since the earlier experiments work with a software emulator only, lack of hardware is not too serious an issue. Even for the later experiments, the emulation feature provided by the software and the documentation on the CD-ROM allow students to do much of the source code development and debugging outside the laboratory. A set of scripts that students could use quickly to self-check the functional correctness of their laboratory programs was designed and placed on the CD-ROM. Using the scripts on the CD-ROM, every aspect of the laboratory platform, even the Linux desktop, can be replicated using a few simple steps on the student's home computer. The online version of the CD-ROM was subsequently placed in the public domain [22].

IV. Learning and Teaching Practices

To assist students to achieve the educational objectives of the course, the following pedagogical methods were employed.

- Well-designed teaching materials, with a high degree of integration among the hardware, software, laboratory exercises, lecture material, and other course aspects [5], are deliberately designed to convey the view of real-world embedded systems, particularly deeply embedded systems, where engineers might come across digital signal processors (such as used in voice recognition), dedicated hardware for vector processing, interrupt-driven systems, micro-OSs, and device drivers, all integrated together [3]–[5]. The key to success in any project of this kind is proper integration of the various aspects of

[2]http://www.cs.man.ac.uk/teaching/electronics/ARM_board

the course including the lecture, laboratory, and other support material.

- Students are strongly challenged and engaged when required to solve real and difficult problems in embedded systems, within the time constraints, while still keeping them focused and interested. Unfortunately, there are two conflicting requirements in any engineering experiment: the laboratory experiment must be simple enough for most of the students to complete, yet challenging and engaging enough to encourage many to think about further [7].
- Each experiment is divided into several tasks scaffolded in stages to provide the best learning results. Each task begins with a clear aim and expectation. Next a comprehensive discussion on the theoretical, algorithmic, and practical issues relating to the task is presented. This discussion also provides a reference to students' assumed skills and knowledge from other courses and disciplines. This back-referencing presents a clear context for the task at hand [10]. To make the transition between the algorithmic and design views of the task easier, an example of simple, related design is provided. The pedagogical basis of providing a simple design as the starting point is that engineers almost always design in an experiential way, by the examination of an existing design, not in a vacuum [25], [26]. Another important feature of engineering work is modification and redesign on an existing or a new platform [2]. This retrofitting requires a good understanding of the theory involved, the pros and cons of the existing implementation, and the cost/benefit analysis of retrofitted design in terms of performance and resource requirements. Tasks, whenever appropriate, expose the retrofitting view of engineering. Engineering design always relies on the utilization of features that are specific to a platform available to the designers. Similarly, in the laboratory work, students use the architectural features of the ARM processor and the other resources on the development board to create an optimized design. Tasks in some experiments provide the basis for the design partitioning between the hardware software, an important aspect of embedded systems design.
- In addition to the required set of tasks, each experiment has some (more challenging) optional tasks. These tasks build on the previous work and on an understanding of the underlying system. The provision of optional challenging tasks creates an opportunity for students to have fun as well as to enhance their learning experience [14]. Students are given extra credit (marks) for doing the optional tasks.
- Additional support and opportunity are provided to students who were willing to complete mini projects to explore the various features of the board in more depth, further exposing them to more advanced concepts and creating a more experiential learning environment [25], [26]. The scope of these mini projects is quite broad, with emphasis on projects that are multidisciplinary in nature. The broad scope of the mini projects provides the means to put this course in the context of other disciplines of electrical and computer engineering. The successful projects are published in the Project Directory on the Digital System Laboratory Web site [27]. Students are also rewarded with extra credit for doing the mini projects. Prior experience has shown that such peripheral activities have significant effects on students' motivation [14] and have been confirmed here. Some students even spend several weeks to months after the course is finished to finalize their projects. However, one should note that supervision and evaluation of such projects requires considerable effort and time commitment on the part of the teaching staff, although the time is well spent.
- The course material was developed within the context of courses taken previously and concurrently. One of the problems with the previous course, "Microprocessors and Interfacing," was that the course was offered in isolation with no regard to courses taken before, concurrently, or afterwards, therefore failing to achieve any educational objectives [10]. The first step in the new offering of this course was to move it from the second to the third year of the electrical engineering program. The change of year allowed us to present the material in this course within a framework that is familiar to nearly all students in electrical and computer engineering—that of a high-level programming language. The C programming language was assumed for the purpose because it is widely taught and is used in the prerequisite course. In addition, students had taken a course in digital electronics before this course. A previous basic digital course provides a context for the design with FPGAs on the board. To create a better context, the course was placed after an introductory course in signal processing that covered the concepts on digital sampling and processing of signals. Furthermore, whenever appropriate, examples are provided of applications using embedded systems in the other disciplines to which the students are likely to be exposed. Skills learned in this course equip students to deal better with the follow-on courses in the real-time instrumentation and control, applications of the digital signal processing, operation systems, and senior year design project.
- Material is always presented with regard to its relevance and appropriateness in the industrial and research setting [3]–[5]. For example, students are taught that the structure of underlying hardware makes more efficient the representation $(5 \times A)$ as its mathematically equivalent form of $(4 \times A + A)$. This problem is often the same problem that the Ph.D. students need in working with their designs as well. Another example is how to perform a division operation as a sequence of shifts and subtractions when the underlying hardware does not allow one to do the division directly. A further example is to illustrate that computers have limited precision by showing that $(A+1) - A$ is not same as $(A - A) + 1$ when the variable A is a very large real number. In each laboratory exercise or lecture presentation, the immediate relevance of material and its application in the latest high-tech product that has just been released onto the market, or in the final stages of development, is highlighted. This contextualization of material generates considerable enthusiasm for the course among students [10].

- Providing flexible methods of course delivery [24], assessment, and feedback is one of the essential aspects of this course. The companion CD-ROM, self-checking tool, online lecture material, weekly online quiz (followed by formal tutorial sessions to enhance the student's learning experience), and provision for asking questions online on the WebCT (the UNSW educational technology and flexible teaching and learning tool) all allow students to learn at their own pace [28], [29], solicit help when they need it, and have an enjoyable learning experience. This feature of the WebCT virtually provides for service seven days a week, 24 hours a day [28].
- Another important aspect of the course is the provision of effort, participation, and altruism (EPA). Students are highly encouraged to assist each other in the laboratory and participate in the online discussion groups on the WebCT. They are rewarded for their efforts with extra credits. This feature is especially important for non-English-speaking students who would not participate in the class otherwise. The laboratory work is organized in small groups of two students. This grouping further reinforces the value of teamwork in a collaborative environment.

V. Course Assessment Strategy

One of the good features of this new course is its assessment methodology that allows for flexibility while preserving its integrity. The course has three compulsory, assessable components: the weekly online quiz [30] (10%), the laboratory work (20%), and the final exam (70%).

- The flexible online quiz for each week covers the material presented in the lecture and the laboratory during the week. Each quiz lasts 20–30 min depending on the degree of difficulty. One of the features of the quiz is that each student sees a very similar, but not identical, set of quiz questions presented in random order. This randomization of the questions allows student to discuss the quiz questions in small cooperative groups [7], [12] without compromising its integrity. With the growth of the quiz question bank, as the course has been offered over a longer period, the danger of plagiarism in the online quiz will largely be eliminated. Students are allowed two attempts at each quiz with the average recorded as the quiz mark for the week.
- The laboratory component is perhaps the most difficult part to assess. Students are expected to do most of the preparatory work outside the laboratory, where they have considerable flexibility. Since students are provided with the tools on the CD-ROM, this expectation is very reasonable and results in greater productivity. The checking scripts allow them to check the correctness of their programs with confidence. The assessment in the laboratory is based on a system of checkpoints and on-the-spot marking by the laboratory assistants.

  To guard against plagiarism while allowing students to learn in small groups in and out of the laboratory [2], [7], other assessment strategies in addition to checking scripts are relied upon. Students are asked to explain the important concepts that they have used and learned from performing the experiments. Each experiment is appended with a question set that students are expected to answer in preparation for assessing their checkpoints.

  Active participation in student learning through group discussions [2], [12], while at the same time administering a fair and objective system of assessment in the laboratory components, requires strict discipline and careful diligence on the part of the laboratory assistants. The laboratory assistants go through informal training on a regular basis to help them with the management of the laboratory assessment task.
- The final examination is a major assessment task contributing to 70% of the total marks. However, preparing students for it through a set of sample questions and guiding students through those questions on the WebCT, without actually giving away the final solutions, is a valuable learning experience. This aspect of the course has been especially beneficial and effective in assisting student learning. Students can think through, try, and analyze the problem first and then communicate and interact with other students and instructors to reinforce their understanding from the convenience of their home [24]. This technique requires a total time commitment and dedication on the part of the course instructors.
- In addition to the compulsory tasks, the optional extra credit checkpoints for each experiment, the opportunity to undertake a minor design project, and EPA allow students to score more marks [14]. Many students attempt some of the optional extra credit checkpoints, whereas the very good students opt to do the mini projects.

VI. Peer Review, Feedback, and Evaluation

The new course on embedded systems has been in place for four semesters. To date, 750 to 800 students have taken the new course. Formal and informal student evaluations of the course have been conducted in each semester. In addition, evaluation from academic peers around the world and teaching and technical assistants has been obtained.

A. Peer-Review Evaluation

As part of the course evaluation mechanism, an international peer review of the course has been conducted. Some of the reviews have been unsolicited, and some have been invited. The main aim of the evaluation has been to check the following:

- relevance of the course materials to the industry;
- organization of the materials;
- usefulness of laboratory documentation and exercises, and their usefulness in the learning process;
- content and the organization of the CD-ROM;
- organization of the lecture materials;
- appropriate choice of tools;
- any suggestions that would make the course better.

Comments from professionals have been very positive and constructive. The UNSW DSL and the course Web sites are referenced by both ARM's[3] and Xilinx's[4] educational program

[3]http://www.arm.com/community/academy/resources.html

[4]http://www.xilinx.com/univ/xup/course.htm

sites. The DSL Web site is also quoted by several academic Web sites. The authors have received multiple requests for the use and distribution of the course materials and additional information. One of the distinguished academic authors commented:

> I am impressed with the completeness of the package of materials. The lectures cover the more "theoretical" background material, while the laboratories cover essential practical aspects of the technology. The degree of integration between the former and the latter appears to be very high. They look to be extensive and of high quality. It is a huge job organizing all these materials, and the instructors should be complimented on their commitment to pull something like this together!

Comments from the competent tutors and laboratory demonstrators have been positive as well. One of the teaching assistants for the course (a Ph.D. student), who had previously worked for Motorola in the area of embedded systems, said:

> The work carried out in the new Embedded Systems course is quite similar to the job of a software engineer working in embedded systems. In this kind of job, you're required to write a lot of software to deal with real time events, and drive real devices. This sometimes surprised students, when I would explain that, yes, what they are learning is relevant to the real world! I think the course is quite fun!

B. Student Evaluation

Tracking the student evaluation of the course on embedded systems started from its last offering in its old form until its four subsequent offerings in its new form. The overall student satisfaction ratings with the course obtained through the formal evaluation over five semesters were 2.15, 3.85, 2.95, 4.45, and 4.75 on a scale of 1 to 5. The sharp rise from 2.15 to 3.85 (1.70 points) is a result of the introduction of fundamental changes to the course as discussed in the previous sections. The further refinement of the course included the introduction of the optional tasks for each experiment and improvement in the presentation of the lectures. This refinement contributed another 0.10 improvement in the overall rating. Introduction of the weekly online quiz and better course coordination added another 0.50 points to the overall rating. Extensive participation in the online discussion groups on the WebCT, introduction of formal tutorial classes, and further refinements of online quiz, lecture presentation, and laboratory documentation improved the rating by another 0.3. Students gave full support to the course. One particular aspect of the course about which students were very positive was the level of integration and coordination between the various components of the course and the provision of online support on the WebCT.

Comments from students largely paralleled the peer review comments.

> "This was a very complete course, I really liked the labs too. I found them very helpful in understanding why everything was the way it was." "It is a very well thought out and planned course." "Lecture notes were well organized." "I have really learned a lot from the course and it has been a pleasure undertaking it, though tough, but very consistent and thorough taught course."

This new course is offered as a complete package, and since its introduction, the failure rate has steadily reduced. The failure rates for the past five offerings of the course (one in its old form and four in its new form) have been 39%, 28%, 25%, 19%, and 16%. The change in the average class mark shows a similar pattern. One important indicator of the success of this course for us was the four-fold increase in the number of students wanting to do their design projects or summer scholarship projects in the DSL. This decision demonstrates improved skills and higher confidence level among students participating in design projects in embedded systems. In the student survey conducted in June 2005, 87% of the students indicated that they would be interested in taking a further advanced course or a final year project in the embedded systems design area if they are offered. The laboratory had the highest number of summer scholarship students in 2004 (50% of students in the school).

VII. Conclusion

This paper presented a case study in international collaboration in course design and development in computer-based and embedded systems. Such collaboration is important in both reducing development costs in developing up-to-date and increasingly sophisticated courses and in fostering good relations between peer departments. The authors are not aware of similar collaborative teaching work of this scale anywhere else in the world.

The most helpful aspects of the course redesign and the Digital System Laboratory overhaul include the state-of-the-art programmable hardware platform, the companion CD-ROM, clear laboratory exercises, flexible learning environment, and online support. These aspects, coupled with industry-standard GNU development tools running under the Linux OS, give students an insight into the hardware and software aspects of real-world embedded systems.

References

[1] A. McGettrick, M. D. Theys, D. L. Soldan, and P. K. Srimani, "Computer engineering curriculum in the new millennium," *IEEE Trans. Educ.*, vol. 46, no. 4, pp. 456–462, Nov. 2003.

[2] ACM/IEEE-CS Joint Curriculum Task Force. (2004) Computing curricula—Computer engineering 2004: Report of the ACM/IEEE-CS Joint Curriculum Task Force. [Online]. Available: http://www.eng.auburn.edu/ece/CCCE/CCCE-FinalReport2004Dec12.pdf

[3] A. Bechini, T. M. Conte, and C. A. Prete, "Guest editors' introduction: Opportunities and challenges in embedded systems," *IEEE Micro*, vol. 24, no. 4, pp. 8–9, Jul.–Aug. 2004.

[4] J. Fleischmann and K. Buchenrieder, "Prototyping networked embedded systems," *IEEE Computer*, vol. 32, no. 2, pp. 116–119, Feb. 1999.

[5] W. Wolf and J. Madsen, "Embedded systems education for the future," *Proc. IEEE*, vol. 88, pp. 23–30, Jan. 2000.

[6] P. Koopman, "Embedded system design issues (the rest of the story)," in *Proc. IEEE Int. Conf. Computer Design: VLSI Computers Processors*, Austin, TX, Oct. 1996, pp. 310–317.

[7] J. W. Bruce, J. C. Harden, and R. B. Reese, "Cooperative and progressive design experience for embedded systems," *IEEE Trans. Educ.*, vol. 47, no. 1, pp. 83–92, Feb. 2004.

[8] B. Clark, "The modern integration of research activities with teaching and learning," *J. Higher Educ.*, vol. 68, no. 3, pp. 241–255, May–Jun. 1997.

[9] University of New South Wales. (2004) *UNSW Guidelines on Learning That Inform Teaching at UNSW* [Online]. Available: http://www.guidelinesonlearning.unsw.edu.au/

[10] F. Dochy, M. Segers, and M. Buehl, "The relation between assessment practices and outcomes of studies: The case of research on prior knowledge," *Rev. Educ. Res.*, vol. 69, no. 2, pp. 145–186, Summer 1999.

[11] S. R. Hatfield, Ed., *The Seven Principles in Action: Improving Undergraduate Education*. Bolton, MA: Anker, 1995.

[12] V. Tinto, "Classrooms as communities: Exploring the educational character of student persistence," *J. Higher Educ.*, vol. 68, no. 6, pp. 599–623, Nov.–Dec. 1997.

[13] J. Allan, "Learning outcomes in higher education," *Stud. Higher Educ.*, vol. 21, no. 1, pp. 93–108, Feb. 1996.

[14] A. Miller, B. Imrie, and K. Cox, *Student Assessment in Higher Education: A Handbook for Assessing Performance*. London, U.K.: Kogan Page, 1998.

[15] L. D. Feisel and A. J. Rosa, "The role of the laboratory in undergraduate engineering education," *J. Eng. Educ.*, vol. 94, no. 1, pp. 121–130, Jan. 2005.

[16] D. Bouldin, "Impacting education using FPGAs," in *Proc. 18th Int. Parallel Distributed Processing Symp. (IPDPS)*, Denver, CO, Apr. 2004, p. 142.

[17] C. H. Amon, S. Finger, D. P. Siewiorek, and A. Smailagic, "Integrating design education, research and practice at Carnegie-Mellon: A multidisciplinary course in wearable computers," *J. Eng. Educ.*, vol. 85, no. 4, pp. 279–285, Oct. 1996.

[18] S. Nooshabadi and J. Garside, "Teaching embedded systems design—An international collaborative project," in *Proc. Frontiers Education Conf. Pedagogies Technologies Emerging Global Economy*, Oct. 2005, pp. F2D25–F2D30.

[19] D. Schlesinger. (2003, Jan.) Some people missed the ARM tornado. *Software Times Essays* [Online]. Available: http://www.softwaretimes.com/files/some%20people%20missed%20the%20arm.html

[20] D. Durig. (2003, Sep.) ARM holdings, three themes for growth. Durig Capital, LLC. [Online]. Available: http://www.durig.com/arm1.html

[21] N. Chang and I. Lee, "Embedded system hardware design course track for CS students," in *Proc. IEEE Int. Conf. Microelectronic Systems Education*, Anaheim, CA, Jun. 2003, pp. 49–50.

[22] School of Electrical Engineering and Telecommunications. (2005) Microprocessor and Interfacing Companion CD-ROM. [Online]. Available: http://dsl.ee.unsw.edu.aud

[23] School of Electrical Engineering and Telecommunications. (2005) ELEC2041: Microprocessor and interfacing. [Online]. Available: http://subjects.ee.unsw.edu.au/~elec2041

[24] J. Bourne, D. Harris, and F. Mayadas, "Online engineering education: Learning anywhere, anytime," *J. Eng. Educ.*, vol. 94, no. 1, pp. 131–146, Jan. 2005.

[25] D. A. Kolb, *Experiential Learning: Experiences as the Source of Learning and Development*. Englewood Cliffs, NJ: Prentice-Hall, 1984.

[26] J. N. Harb, S. O. Durrant, and R. E. Terry, "Use of the Kolb learning cycle and the 4MAT system in engineering education," *J. Eng. Educ.*, vol. 82, no. 2, pp. 70–77, Apr. 1993.

[27] School of Electrical Engineering and Telecommunications. (2005) UNSW Projects Directory on the Microprocessor and Interfacing Companion CD-ROM. [Online]. Available: http://dsl.ee.unsw.edu.au/dsl-cdrom/unsw/projects/README.html

[28] M. S. Cohen and T. J. Ellis, "Developing criteria for an on-line learning environment: From the student and faculty perspectives," *J. Eng. Educ.*, vol. 93, no. 2, pp. 161–167, Apr. 2004.

[29] D. W. Dearholt, K. J. Alt, R. F. Halpin, and R. L. Oliver, "Foundational aspects of student-controlled learning: A paradigm for design, development, and assessment appropriate for web-based instruction," *J. Eng. Educ.*, vol. 93, no. 2, pp. 129–138, Apr. 2004.

[30] R. Dufresne, J. Mestre, and K. Rath, "The effect of web-based homework on test performance in large enrollment introductory physics course," *J. Comput. Math. Sci. Teach.*, vol. 21, no. 3, pp. 229–251, 2002.

Saeid Nooshabadi received the Ph.D. degree in electrical engineering from the India Institute of Technology, Delhi, in 1992.

Currently, he is a Senior Lecturer in microelectronics and digital system design in the School of Electrical Engineering and Telecommunications, University of New South Wales, Sydney, Australia. Prior to his current appointment, he held academic positions at the Northern Territory University and the University of Tasmania, between 1993–2000. In 1992, he was a Research Scientist with the CAD Laboratory, Indian Institute of Science, Bangalore, working on the design of VLSI chips for TV ghost cancellation in digital TV. In 1993 and 1997, he was a Visiting Faculty Member and Researcher with the Centre for Very High Speed Microelectronic Systems, Edith Cowan University, Western Australia, working on high-performance GaAs circuits; and Curtin University of Technology, Western Australia, working on the design of high-speed/high-frequency modems. His teaching and research interests include very high-speed integrated circuit and application-specified integrated circuit design for high-speed telecommunication and image-processing systems, low-power design of circuits and systems, and low-power embedded systems.

Jim Garside (M'91) received the B.Sc. degree in physics and the Ph.D. degree in computer science from the University of Manchester, U.K., in 1983 and 1987, respectively.

His doctoral work looked at digital signal-processing architectures. He spent some time working on parallel architectures with Transputers. After a brief sojourn in the software industry, working on air traffic control software, he returned to the University of Manchester as a Lecturer in 1991. His later research work has primarily been concerned with VLSI technology, particularly with the Amulet asynchronous ARM processors, spanning from circuit design to microarchitecture. At the same time he has done teaching work, upgrading undergraduate "hardware" laboratories to use computer-aided design systems and FPGAs and, more recently, processor + FPGA microcontroller systems. His current interests include low-power and power-efficient processing.