American Journal of Embedded Systems and Applications
2015; 3(1): 1-10
Published online March 25, 2015 (http://www.sciencepublishinggroup.com/j/ajesa)
doi: 10.11648/j.ajesa.20150301.11
ISSN: 2376-6069 (Print); ISSN: 2376-6085 (Online)

Embedded systems pedagogical issue: Teaching approaches, students readiness, and design challenges

Intisar Ibrahim[1], Rosmah Ali[1], Mohamad Zulkefli[2], Nazar Elfadil[3]

[1]Engineering Education School, Universiti Teknologi Malaysia, KL, Malaysia
[2]Advanced Informatics School, Universiti Teknologi Malaysia, KL, Malaysia
[3]College of Computing, Fahad Bin Sultan University, Tabuk, Saudi Arabia

Email address:

irsintisar2@live.utm.my (Intisar I.), rosmah.kl@utm.my (Rosmah A.), mzulkefli.kl@utm.my (Mohamad Z.), gazoli@lycos.com(Nazar E.)

To cite this article:

Intisar Ibrahim, Rosmah Ali, Mohamad Zulkefli, Nazar Elfadil. Embedded Systems Pedagogical Issue: Teaching Approaches, Students Readiness, and Design Challenges. *American Journal of Embedded Systems and Applications.* Vol. 3, No. 1, 2015, pp. 1-10.
doi: 10.11648/j.ajesa.20150301.11

Abstract: This paper discusses the challenges of teaching and design of embedded system course. A survey on different approaches of teaching embedded systems design course was presented. The embedded system design class/course is often the course in which students are exposed to fairly complex design problems. In this study, the authors explained the importance of teaching embedded systems courses in a way that can shrink the gap between academically taught skills and the skills sought after by the industry. The literature on embedded systems design was found to be diverse, including relatively small amounts of research data and much larger amounts of evaluation literature. Methodological rating schemes were used to compare for confounding influences in the research studies. Many studies were found but only few of them were included because of lack of methodological rigour in the research and poorly developed outcome measures. This paper suggested that by understanding the cognitive mindset of different discipline, suitable methods can be advised. These can be implemented not only in higher education institutions but also in industrial training. Comments and future work for further research are also listed.

Keywords: Embedded Systems Design, Teaching Challenges, and Readiness

1. Introduction

Today's economy demands a better educated workforce than ever before as the industry requires workers with more complex knowledge and skills This paper discusses issues related to the teaching and learning of embedded systems course, a foundational course in the area of computer engineering which requires knowledge and skills in computer science and electrical engineering.

Computer engineering has traditionally been viewed as a combination of both Computer Science (CS) and Electrical Engineering (EE). It has evolved over the past three decades as a separate, although intimately related, discipline. Computer engineering is solidly grounded in the theories and principles of computing, mathematics, science, and engineering and it applies these theories and principles to solve technical problems through the design of computing hardware, software, networks, and processes [1].

The core/backbone courses of the electrical and computer engineering disciplines include embedded system design, computer architecture and organization, circuits and signals, digital logic, electronics, computer networks, and VLSI design. Among the above mentioned courses, this paper focuses on approaches and challenges of teaching and learning an embedded system design course.

Even though embedded systems have been designed for more than three decades; the academic course of embedded systems is relatively undefined course, which is mostly regarded as an interdisciplinary field combining areas such as computer science, computer engineering, automatic control, and electrical engineering [1].

Embedded systems courses are often regarded differently among academic institutions/ universities. The course may have evolved from or within departments of computer science, electrical engineering, or Mechatronics. As a consequence, some universities treat and teach embedded systems as a specialization of computer science, whereas some departments use it to promote education and research in

computer engineering/electrical engineering [2].

This paper is organized as follows: Section I presents an introduction of the embedded system design. Section II discusses the nature of embedded system design courses. Embedded system design teaching approaches will mainly be discussed in section III. Design and teaching challenges and the conclusions will be elaborated in section IV and V respectively.

2. Embedded System

An Embedded system course generally aims to help students to undertake the design and development process for an embedded computer system and to know how to integrate an embedded hardware, software, and operating systems to meet the functional requirements of certain embedded applications.

2.1. What is an Embedded System

Institute of Electrical Engineers (IEE) defines an embedded system as devices used to control, monitor or assist the operation of equipment, machinery or plant [3]. Some researchers define an embedded system as "A microprocessor-based control system which processes a fixed set of programmed instructions to control electromechanical equipment which may be part of an even larger system" [4].

Nevertheless, [5] defines an embedded system as "one that has computer-hardware with software embedded in it as one of its most important component."

2.2. What is an Embedded System Course

This question had been asked by several researchers [5- 8] during the last couple of years. Most of them asked it in terms of defining embedded systems and how it should be taught. Some considered the embedded system course as an approach to preparing students to become industrial designers. While others considered as a rehearsal/training for students on industrial applications and concerns. Embedded system courses generally deal with the design and analysis of the software and hardware for a dedicated application. [5].

The classic approach starts with the lowest hardware level. The Central Processing Unit (CPU) registers, address bus, data bus, and memory are discussed in connection with a basic assembly language instruction set [6]. Two decades ago this was definitely the way to do it, since the microprocessors of those days were very simple. Also the limited functionality did not allow for a higher language approach.

Nowadays, the assembly language approach is by no means redundant, since this is was the only way to show the students what exactly is happening on the machine level [8].

3. Embedded System Teaching Approaches

Teaching embedded system is challenging mainly because of interdisciplinary relationship, limited time, large variations of students motivations, skills, and backgrounds. There are several teaching approaches, among them: (1) hardware oriented, (2) software oriented, and (3) hardware and software integration oriented.

The most famous learning approach taken by several instructors in teaching an embedded system design course is to learn by doing in a bottom-up fashion. One of the advantages of a bottom-up approach to learning is that the student begins by mastering simple concepts. Once the student truly understands simple concepts, he/she can then embark on the creative process of design, which involves putting the pieces together to create a more complex system.

Embedded systems afford an effective platform to teach new engineers how to program for three reasons: (1) there is no operating system. Thus, in a bottom-up fashion the student can see, write, and understand all software running on a system that actually does something, (2) embedded systems involve real input/output that is easy for the student to touch, hear, and see, and (3) Third, embedded systems are employed in many every-day products, motivating students to see firsthand, how engineering processes can be applied in the real world [9].

Given the importance of embedded systems, several universities/institutes have developed a course for undergraduate and postgraduate students studying Electrical Engineering, Mechatronics, and Computer engineering. This course is generally entitled as Embedded Systems or Embedded System Design. It is intended to introduce students to various aspects of embedded systems in the lectures and to familiarize students with the hardware/software co-design methodology. The course is built such that so students, after successfully finishing the course, must be able to design an embedded system. There are several courses that contribute as a pre/foundation to the course and some consider as post/continuation of it; as shown in Fig.1.

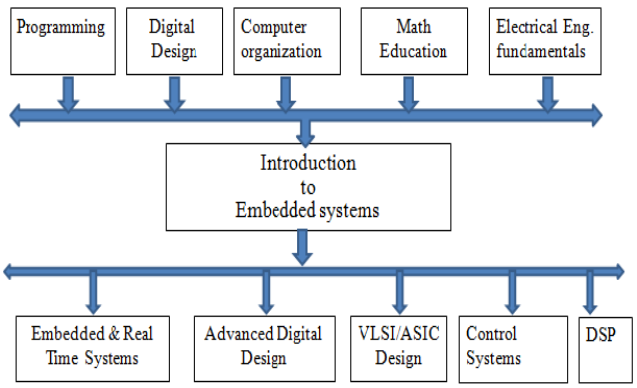

Fig. 1. *Placement of Embedded system in curriculum*

Other scholars have proposed different course curriculum for embedded systems and an embedded programming [10, 11]. In spite of the discrepancies, the goal is to teach a common set of topics and to allow the student to interact with devices in the lab at both the software and hardware levels.

The purpose of teaching embedded systems at most universities/institutes is to provide the industry with

competent staff. Apart from that, the purpose of performing research is to educate students in embedded systems, and to create new knowledge in the area of embedded systems for sake of supporting the industry.

The courses of embedded system can be categorized into three categories; namely: (1) hardware oriented, (2) software oriented, and (3) hardware and software integration design [14]. Traditional, Computer Science curricula focuses on the training for logic reasoning and programming skills. System integration is often not covered in computer science curricula. As the embedded platforms migrate from 8-bit microprocessors to 32-bit microprocessors, the engineers require different skills to design modern embedded systems. Most computer engineering programs teach programming and design skills that are appropriate for a general-purpose computer operating under control of a commercial operating system rather than for the more specialized embedded systems. Embedded systems courses are being integrated into computer science, electrical and computer engineering, software engineering programs, and cross-disciplinary courses. Current embedded systems courses span a number of topics including designing real-time systems, low power systems that contain reconfigurable hardware, embedded processors, software, and digital signal processing. For most of industry application, HW/SW integration design is required. To meet the requirement, the teacher should provide the fundamental concepts for real-time systems, embedded hardware, real-time scheduling, real-time operation systems, and power management [15].

Research into learning and teaching in higher education over the last decades has provided a variety of concepts, methods, and findings that are of both theoretical interest and practical relevance. It has revealed the relationships between students' approaches to studying, their conceptions of learning, and their perceptions of their academic context. It has revealed the relationships between teachers' approaches to teaching, their conceptions of teaching, and their perceptions of the teaching environment [47].

Interview-based research carried out in Britain and Sweden during the 1970s had identified three predominant approaches to studying in higher education: (1) a deep approach, based upon understanding the meaning of course materials; (2) a surface approach, based upon memorizing the course materials for the purposes of assessment; and (3) a strategic approach, based upon obtaining the highest grades[47].

Research into teachers' approaches to teaching in higher education was directly modeled on the concepts, methods, and findings of research into students' approaches to studying. Trigwell and Prosser [54] identified five different approaches to teaching among these teachers that were differentiated in terms of their intentions and their teaching strategies: some approaches were teacher-focused and were aimed at the transmission of information to the students; others were student-focused and were aimed at bringing about conceptual change in the students. Those approaches are: (1) Teaching as imparting information; (2) Teaching as transmitting structured knowledge; (3) Teaching as an interaction between the teacher and the student; (4) Teaching as facilitating understanding on the part of the student; and (5) Teaching as bringing about conceptual change and intellectual development in the student.

3.1. Hardware Oriented Teaching Approach

To learn embedded system, in general the course must include five parts: hardware architecture, boot loader and device driver, embedded real-time operation system, debug method for embedded system, and embedded application [16, 17]. While the Framework practices are organized around five primary themes related to improving teaching and learning [18]; namely: (1) Curriculum and Academic Goals: What do we expect all students to know and be able to do in each course, grade and subject?, (2) Staff Selection, Leadership, and Capacity Building: How do we select and develop the leaders and teachers needed to ensure every student in the system meets these expectations?, (3) Instructional Tools: Programs and Strategies: What programs, strategies, materials, and time allocation do we use to teach the necessary content and skills?, (4) Monitoring Performance and Progress: How do we know if students learned what they should?, and (5) Intervention and Adjustment: If students are not learning

Students should have an opportunity to select or buy their very own development kits right from the beginning. Upon given a certain application the student must be able to: (1) select an embedded system platform consisting of hardware and software components that is able to fulfil the requirements of the targeted application; (2) realize a software-only implementation, analyze it, and identify its performance bottlenecks; and (3) overcome the identified performance bottlenecks by moving the bottleneck functions in question towards a hardware implementation.

3.2. Software Oriented Teaching Approach

In embedded software development, students were introduced to several ideas of cross-platform development; among them: (1) host based embedded software development and embedded target environments, (2) integrated development environments, (3) interrupts and interrupt handling, and (4) embedded software architectures. Then the course introduced the existing techniques from software engineering, and briefly covered several topics including embedded software development, software development lifecycle, software development models, including the waterfall model, the spiral model, rapid application development model, and object-oriented approaches. [17].

Software systems designers, in contrast, use sequential building blocks, such as objects and threads, whose structure often changes dynamically. Designers can create, delete, or migrate blocks, which can represent instructions, subroutines, or software components. An abstract machine, also known as a virtual machine or automaton, defines a block's formal semantics operationally.

Nevertheless, some courses give students experience with three distinct levels of design of embedded software; namely:

(1) bare-iron programming, (2) programming within a real-time operating system, and (3) model-based design. In each case, students are taught to think critically about the technology, to probe deeply the mechanisms and abstractions that are provided, and to understand the consequences of chosen abstractions on overall system design.

Edward stated that [55], embedded software has traditionally been thought of as "software on small computers." In this traditional view, the principal problem is resource limitations (small memory, small data word sizes, and relatively slow clocks). Solutions emphasize efficiency; software is written at a very low level, operating systems with a rich suite of services are avoided, and specialized computer architectures such as programmable DSPs and network processors are developed to provide hardware support for common operations. These solutions have defined the practice of embedded software design and development for the last couple decades.

3.3. Hardware-Software Integration Oriented Teaching Approach

Most of the offered embedded courses, missed software and hardware co design approach. Such drawback and shortage is due to fact that co design itself is a course despite it can be incorporated into hardware optimization courses. Such an integrated teaching approach will rise up students' interest and understanding of hardware and software development and their integration aspects.

Hardware/software co-design means meeting system-level objectives by exploiting the synergism of hardware and software through their concurrent design [48]

4. Embedded System Design Approaches

Several embedded system design methodologies exist to help designers. Ferens [19] stated that system design methodologies can be classified historically into three categories; namely:

Captures and simulates methodology that last between 1960's to 1980's. During such era the hardware and software were totally separated by system gap. The system requirements were used to be written by software designers and later given to hardware designers to implement it. Both designers will not be able to know whether their design would satisfy the specifications until the gate level was fully produced, this might be considered as main disadvantages or challenges. Furthermore, specifications can be always upgraded, as can its implementation.

Describe and synthesis methodology that was around between 1980's to late 90's. In such method designers specified first what they wanted in Boolean or Finite state machines, and then the synthesis tools generated the implementation in terms of logical level net lists. In this design approach, the behavior or function comes first, and the structure or implementation comes afterwards. Today's designs are too large for checking and this is might be considered among faced challenges.

Specify, explore and Refine methodology that last since 2000's. This is approach shrink the gap between hardware and software by introduce a method that includes both hardware and software. The challenge that might face designers that they must deal with several models in order to verify the impact of design decisions on every verification/testing metric.

This is section will emphasis on specify, explore and refine design methodology. The strength and weakness of each design methodology will be explained.

4.1. Software Oriented Design Approach

The embedded system software design approach advantages over the hardware design can be classified into; but not limited to: (1) Easier to change when new hardware versions become available, (2) Programmability for complex operations, (3) Faster development time, (4) Modularity and portability, (5) Use of standard software engineering, modeling and RTOS tools, (6) Faster speed of operation of complex functions with high-speed microprocessors, and (7) Less cost for simple systems [20].

Apart from above mentioned advantages, still there are several challenges; among them: (1) Reliable software, (2) Cheap, available systems using unreliable components, (3) Electronic vs. non-electronic design tradeoffs, (4) Software- and I/O-driven hardware synthesis, and (5) Distributed system tradeoffs among analog, power, mechanical, network, and digital hardware plus software.

4.2. Hardware Oriented Design Approach

Hardware implementations provide advantage of processing speed. Some of those advantages can be listed as: (1)Reduced memory for the program, (2) Reduced number of chips but at an increased cost, (3) Simple coding for the device drivers, (4) Internally embedded codes, which are more secure than at the external ROM, and (5) Energy dissipation can be controlled by controlling the clock rate and voltage.

4.3. Co - Design Oriented Approach

Design that depends on software based functionality in today's embedded products cause's delays in project completion if you wait for the hardware prototype to begin software development and debugging. Concurrent design and verification of hardware and software is a trend that reduces time-to-market.

Embedded systems for real-time applications are implemented as a mix of hardware and software sub-systems interacting in one cohesive environment.

It is the purpose of this special issue to shed some light on the recent developments of co-design in different embedded system application domains.

Table 1. *Show the strength and weakness of each design methodology*

Approach	Pros	Cons
Software Oriented Design Approach	Easier to change when new hardware versions become available. Programmability for complex operations. Faster development time. Modularity and portability Use of standard software engineering, modelling and RTOS tools. Faster speed of operation of complex functions with high-speed microprocessors. Less cost for simple systems.	There are several challenges; among them: Reliable software. Cheap, available systems using unreliable components. Electronic vs. non-electronic design tradeoffs. Software- and I/O-driven hardware synthesis, and Distributed system tradeoffs among analog power, mechanical network, and digital hardware plus software
Hardware Oriented Design Approach	Reduced memory for the program. Reduced number of chips but at an increased cost. Simple coding for the device drivers. Internally embedded codes, which are more secure than at the external ROM. Energy dissipation can be controlled by controlling the clock rate and voltage.	Reduced number of chips but at an increased cost. Designers will not be able to know whether their design would satisfy the specifications until the gate level was fully produced.
Co - Design Oriented Approach	An integrated teaching approach will rise up students' interest and understanding of hardware and software development and their integration aspects. Exploiting the synergism of hardware and software through their concurrent design.	Drawback and shortage is due to fact that co design itself is a course despite it can be incorporated into hardware optimization courses.

5. Challenges of Teaching Embedded System Design

Embedded system design is a challenging problem that represents the future of digital system design [20]. Teaching students to handle this complexity and challenge is the central challenge as well.

5.1. Embedded System Design Challenges

Several designers place less emphasis on the mechanics of embedded system design and more on critical thinking about design technologies and on how the design of embedded software affects the behavior, safety, and reliability of systems. Such design behavior adds more challenges in addition to the above mentioned ones.

Stephen [21] stated that the more fundamental ideas saw in practical embedded system design the balance between top down and bottom up design necessary to build high performance systems.

As embedded systems consist of hardware and software, both productivity factors need to be taken into account when we aim to design entire systems. Considering that additional software productivity gap, the situation for entire systems only gets worse. In this context, the actual needs in embedded software complexity would require an estimated growth of 2x over 10 months in order to satisfy the complexity involved in building real systems [22].

Worth to mention that, designers actually face two design gaps at the same time; namely: (1) a software design gap; and (2) hardware design gap. Combining both productivity gaps, result in a large system design gap. Moreover, additional complexity is created from the close interaction and tight dependency between the software and hardware domains. In other words, the necessary interfacing of software and hardware adds yet another layer of complexity [22].

Thus, Hardware-Dependent Software (HDS) is at the core of this system design challenge, as it deals exactly with those parts of the embedded software that interact directly with the underlying hardware. Nevertheless, HDS can be defined as the software in an embedded system that closely interacts with the underlying hardware platform. Looking at HDS from the perspective of software architecture, designers can identify HDS as a layer of software modules in between the application software and the underlying hardware platform. In other words, HDS can very well be seen as low level software [22].

5.2. Embedded System Teaching Challenges

Martin [25] classified embedded system teaching challenges into three categories; namely: (1) students related challenges, (2) lecturers related challenges, and (3) course contents challenges.

The students' related challenges might be due to the following facts: (1) Lack of sufficient knowledge and skills from background related disciplines, (2) Hugh competence and motivational differences among students that makes difficult to select starting level of teaching during lectures and laboratory practice, (3) Programming embedded microcontroller is an essential topic in embedded system design. However, very considerable numbers of engineering students are very opposed to programming, and (4) More general learning mentality issues like planning skills (leaving everything for the last moment, especially during course projects).

While the lecturers' related challenges might be due to the following facts: (1) As embedded system covers several fields of electrical engineering and computer science, a lecturer is challenged to be an expert in all of above mentioned disciplines, (2) Very dynamic progress of technologies used in embedded system design adds a requirement of constantly updating the knowledge, and (3) Compatibility of many hardware and software tools and their versions puts a

requirement to adapt laboratory supplement material constantly.

Nevertheless courses contents' related challenges might be due to the following facts: (1) Too limited time for so many theoretical topics and practical skills is very evident in embedded system courses, and (2) Embedded system is relatively new and not yet well defined discipline.

Several education researchers stated that what we teach and what students actually learn can different. Therefore, this section wills mainly emphasis on current teaching challenges that affect embedded system courses in particular. Generally, teaching embedded system design is quite challenging, since such courses require a broad knowledge in many fields like computer science, mathematics, physics, and engineering disciplines. Apart from that, what most universities are teaching in embedded design is not satisfactory.

The lack of highly skilled engineers capable to develop software for embedded devices seems to be currently a common and severe problem in all industrial countries. This fact, associated with the growing body of knowledge produced in the field, is creating a growing interest in proposing curricula on embedded software and systems for university-level education. A good overview of some current issues in embedded systems education can be found in several articles of the ACM Transactions on Embedded Computing Systems [23].

A major challenge is to create courseware that is simultaneous durable and practical. If there is too much emphasis on how to achieve design goals with today's technology, then the students receive technical training with only near-term value. If too much emphasis on foundational theory, the students gain no intuition about the physical realities of embedded systems. Berkeley University has a well established series of courses in digital logic design that manages this balance very effectively. For example, EECS 150, a core undergraduate course in this subject, includes a major project component that takes students through sizable, practical design problems, and complements this with foundational analytical tools that transcend the mechanics of today's design technology. Despite the above mentioned effort, Berkeley has gain an additional challenge of working with a less mature topic. [24]

6. Students Readiness Challenges

6.1. Readiness Definition

Readiness has been variously theorized as a particular chronological age, as a stage or level of development in students, as a set of skills and competencies, as a process, and as a set of relationships. Nevertheless, each of these conceptions has different implications for the roles and responsibilities of students, lecturers, and schools.

Among advocates and policy researchers, readiness is discussed more and more as an interactive process or set of relationships in which the student, lecturer, and the university interact in ways that support, or fail to support, the student's physical, cognitive, and social-emotional development. However, in practice some researchers have pointed out that Readiness is nearly always defined in terms of students' skills or characteristics.

Until recently, students' readiness typically was considered a function of reaching a certain age or of progressing through specific stages of development that were influenced almost entirely by chronological growth and students' inherent characteristics.

However, a strong body of research has cast doubt on assumptions that students tend to progress in some lockstep fashion through specific stages of development and that they must reach a particular age or maturity before they are "ready to learn":

More recent research has led many to reinterpret the stage theorists' views; there is strong evidence that students, when they have accumulated substantial knowledge, have the ability to abstract well beyond what is ordinarily observed.

Indeed, the striking feature of modern research is that it describes unexpected competencies in young students, key features of which appear to be universal. These data focus attention on the student's exposure to learning opportunities [50-51]

6.2. How to Measure Readiness

Basically, readiness measurements/tests can be classified into several categories; among them: (1) measurements that measure developmental milestones, (2) measurements that measure academic knowledge, and (3) measurements that represent a combination of the development milestones and academic knowledge.

Worth to mention that, many researchers have found that the widely used readiness tests are relatively poor predictors of future carrier success and that typical assessment practices lack sufficient validity and reliability for making placement decisions[49].

Moreover, there is evidence that many readiness measures, no matter what their effectiveness in assessing students' skills, do not do a good job of predicting any individual student's academic performance, even in the early grades.

To measure the effectiveness of activities to improve the curriculum design, researchers use common evaluation methods such as questionnaires and surveys. For example, the curriculum design for embedded systems' course papers aims to let students be work-ready engineers in various industrial fields and to imagine relationship between final embedded-products and elementary contents of lectures that they study in college or early undergraduate studies has shown that 80% of students affirmed the effectiveness of their proposed improvement strategies [41]. Thus, it is important to develop an effective measurement instrument to evaluate students' readiness for the course.

As proactive measure, it is important to assess students' readiness for the course. This requires a valid and reliable an assessment instrument. There are many instruments to assess students' readiness for career, colleges and research [31-32]. However, there is yet to be an instrument specifically for

measuring readiness for the Embedded System Design course.

6.3. Problems Associated with Readiness Assessment

Assessment is a significant issue in discussing student readiness, both in terms of its use in research studies such as those reviewed in this synthesis and in terms of its use by carrier for diagnostic or placement purposes.

Nevertheless, assessing young students is a challenge because of students' inexperience, their sensitivity to contexts, and a range of cultural and developmental issues [50].

Furthermore, young students' growth patterns are "unstable and episodic," rather than orderly and uniform, so that comparisons of young students at any given time may not accurately reflect their developmental trajectory [50].

Social-emotional competence is important for students' school success [51]. However, most assessments of students' social-emotional competencies have failed to yield stable or predictive data [51].

Researchers have raised many questions about the validity of specific readiness assessments, including some of the measures used in research studies reviewed for this synthesis.

7. Conclusion and Discussion

The purpose of this paper is to argue for a view on embedded systems education where the survey results find that Embedded systems is a course with a thematic identity and a functional legitimacy, which means that the embedded systems companies are requesting engineers capable of conceiving, designing, implementing and operating embedded systems.

The fast developing area of embedded systems is especially affected by the lack of curriculum flexibility and well trained instructors. During the past decade the contents of many courses were extended by microcontroller related topics. However, this was done by individual teachers without an overall concept, and only within the approved curriculum structure.

Furthermore, the lack of coordination necessarily led to overlapping course contents which further reduced the teaching efficiency. There was also a traditionally disjoined relation between teaching general purpose programming languages, which are considered hardware independent, and hardware specific embedded system topics.

Authors' ambition was to significantly increase the teaching efficiency in this field and remedy all above shortcomings. This may seem rather ambitious, but due to the rigid nature of most available embedded system design curriculum structure it was now or never.

Researches in Teaching and Learning in Embedded Systems courses are mostly focused on undergraduate studies or at early stage in university studies [40]. For example, Mondragon-Torres [38] presents a new concept of the sequence of three embedded systems design courses being taught for computer engineering technology for undergraduate students. Another research [44], Zhang proposed a global curriculum design framework which has an overall view of the process. The framework forms into a cycling pattern which is able to adapt to the changes of time and society [42]. Further work, our analysis and readiness evaluation will be focused on postgraduate students, second year and final year undergraduate students.

While this document addresses student readiness for learning in an online environment, it has merely skimmed the surface.

The purpose of this paper was to explore various types of teaching embedded system design approach and their challenges, as we discussed, the Embedded System Teaching Challenges Nakutis [25] classified embedded system teaching challenges into three categories; namely: (1) students related challenges, (2) lecturers related challenges, and (3) course contents challenges.

To measure the effectiveness of activities to improve the curriculum design, researchers use common evaluation methods such as questionnaires and surveys. For example, the curriculum design for embedded systems' course papers aims to let students be work-ready engineers in various industrial fields and to imagine relationship between final embedded-products and elementary contents of lectures that they study in college or early undergraduate studies has shown that 80% of students affirmed the effectiveness of their proposed improvement strategies [41].

In this paper also we discussed the purpose of teaching embedded systems and the purpose of performing research is to educate students in embedded systems, and to create new knowledge in the area of embedded systems for sake of supporting the industry.

We are witnessing an ever faster technological development in the area of embedded systems. All these changes called for an urgent response to make the teaching of embedded systems more attractive and affordable to a wider population [12, 13].

To overcome this problem, it is important to develop an effective measurement instrument to evaluate students' readiness for the course. Authors' recommend in the further work is to develop an instrument to measure students' readiness for the embedded system course for all computer engineering and electrical engineering students.

Measuring career and life success can be difficult. However, there is a considerable body of research that has examined what factors can be said to lead to success.

7.1. Summary

In this literature the majority of the studies focus on the embedded teaching approach explore and refine design methodology, the strength and weakness of each design methodology will be explained. Embedded system software design approach should use hardware a resource in the most appropriate way that is depends not only on the application but also on the embedded system hardware implementation. Nevertheless, both software engineers and hardware engineers must design hardware dependent software and software dependent hardware as well.

Our focus in this paper has been on teaching embedded system challenges and readiness; embedded system design is a

challenging problem that represents the future of digital system design. Teaching students to handle this complexity and challenge is the central challenge as well. As embedded systems consist of hardware and software, both productivity factors need to be taken into account when we aim to design entire systems.Classified embedded system teaching challenges into three categories; namely: (1) students related challenges, (2) lecturers related challenges, and (3) course contents challenges. Several education researchers stated that what we teach and what students actually learn can different. Therefore, this section wills mainly emphasis on current teaching challenges that affect embedded system courses in particular. Generally, teaching embedded system design is quite challenging, since such courses require a broad knowledge in many fields like computer science, mathematics, physics, and engineering disciplines. Apart from that, what most universities are teaching in embedded design is not satisfactory.

The lack of highly skilled engineers capable to develop software for embedded devices seems to be currently a common and severe problem in all industrial countries. Also A major challenge is to create courseware that is simultaneous durable and practical. If there is too much emphasis on how to achieve design goals with today's technology, then the students receive technical training with only near-term value. If too much emphasis on foundational theory, the students gain no intuition about the physical realities of embedded systems.

To measure the effectiveness of activities to improve the curriculum design, researchers use common evaluation methods such as questionnaires and surveys. For example, the curriculum design for embedded systems' course papers aims to let students be work-ready engineers in various industrial fields and to imagine relationship between final embedded-products and elementary contents of lectures that they study in college or early undergraduate studies has shown that 80% of students affirmed the effectiveness of their proposed improvement strategies. Thus, it is important to develop an effective measurement instrument to evaluate students' readiness for the course.

7.2. Comment and Future work

The purpose of this paper is to argue for a view on embedded systems education where the survey results find that Embedded systems is a course with a thematic identity and a functional legitimacy, which means that the embedded systems companies are requesting engineers capable of conceiving, designing, implementing and operating embedded systems.

The fast developing area of embedded systems is especially affected by the lack of curriculum flexibility and well trained instructors. During the past decade the contents of many courses were extended by microcontroller related topics. However, this was done by individual teachers without an overall concept, and only within the approved curriculum structure.

Furthermore, the lack of coordination necessarily led to overlapping course contents which further reduced the teaching efficiency. There was also a traditionally disjoined relation between teaching general purpose programming languages, which are considered hardware independent, and hardware specific embedded system topics.

Authors' ambition was to significantly increase the teaching efficiency in this field and remedy all above shortcomings. This may seem rather ambitious, but due to the rigid nature of most available embedded system design curriculum structure it was now or never.

Researches in Teaching and Learning in Embedded Systems courses are mostly focused on undergraduate studies or at early stage in university studies [42]. For example, Mondragon-Torres [40] presents a new concept of the sequence of three embedded systems design courses being taught for computer engineering technology for undergraduate students. Another research, by Zhang [44] proposed a global curriculum design framework which has an overall view of the process. The framework forms into a cycling pattern which is able to adapt to the changes of time and society [44]. Further work, our analysis and readiness evaluation will be focused on postgraduate students, second year and final year undergraduate students.

While this document addresses student readiness for learning in an online environment, it has merely skimmed the surface.

The purpose of this paper was to explore various types of teaching embedded system design approach and their challenges, as we discussed, the Embedded System Teaching Challenges Nakutis [25] classified embedded system teaching challenges into three categories; namely: (1) students related challenges, (2) lecturers related challenges, and (3) course contents challenges.

To measure the effectiveness of activities to improve the curriculum design, researchers use common evaluation methods such as questionnaires and surveys. For example, the curriculum design for embedded systems' course papers aims to let students be work-ready engineers in various industrial fields and to imagine relationship between final embedded-products and elementary contents of lectures that they study in college or early undergraduate studies has shown that 80% of students affirmed the effectiveness of their proposed improvement strategies.

In this paper we also discussed the purpose of teaching embedded systems and the purpose of performing research is to educate students in embedded systems, and to create new knowledge in the area of embedded systems support the industry.

We are witnessing an ever faster technological development in the area of embedded systems. All these changes called for an urgent response to make the teaching of embedded systems more attractive and affordable to a wider population [12, 13].

Thus, it is important to develop an effective measurement instrument to evaluate students' readiness for the course. Authors' recommend in the further work is to develop an instrument to measure the student's readiness for the embedded system course for all computer engineering and electrical engineering students.

Measuring career and life success can be difficult. However, there is a considerable body of research that has examined what factors can be said to lead to success.

Acknowledgements

The authors thank Mr. Daoud Sadden for technical assistance and analysis of collected data. We also thank the UTM for their financial support.

References

[1] Draft. "Computer Engineering as a Discipline" [Published 2008]. http://www.ccny.cuny.edu/compeng/upload/IEEE-CpE-excerpts.pdf

[2] Martin Grimheden, Martin Törngren. "How should embedded systems be taught?: experiences and snapshots from Swedish higher engineering education". SIGBED Review 2(4): 34-39 (2005)

[3] Sebastian Fischmeister, "Introduction to Programming Embedded Systems". http://www.cis.upenn.edu/~lee/06cse480/lec-into_to_prog_embedded_systems.pdf.

[4] "Introduction to Embedded system". http://vanilla47.com/Machine%20Language%20and%20Assembly%20Language%20Programing/Introduction%20to%20Embeded%20Systems.pdf

[5] M. Grimheden and M. Törngren, "What is embedded systems and how should it be taught?—results from a didactic analysis," ACM Trans. Embed. Comput. Syst., vol. 4, pp. 633–651, August 2005. Available: http://doi.acm.org/10.1145/1086519.1086528.

[6] Peter Jamieson. "Arduino for teaching embedded system. Are computer scientists and engineering educators missing the boat?" International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS'11), 2011. Las Vegas, NV.

[7] M. J. I. N. Institute, "Industriell Programvaruutveckling," Tech. Rep., 2003.

[8] C. L. Dym, A. M. Agogino, D. D. Frey, and L. J. Leifer, "Engineering design thinking, teaching, and learning," Journal of Engineering Education, vol. 94, pp. 103–120, 2005. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.72.1593.

[9] Tadej Tuma, Iztok Fajfar. "A new curriculum for teaching embedded systems at the University of Ljubljana". Proceeding of ITHET'06. 7th International Conference on Information Technology Based Higher Education and Training, 2006. Pages 14-19

[10] Janos Sztipanovits, Gautam Biswas, Ken Frampton, Aniruddha S. Gokhale, Larry Howard, Gabor Karsai, Tak-John Koo, Xenofon D. Koutsoukos,Douglas C. Schmidt: Introducing embedded software and systems education and advanced learning technology in an engineering curriculum. ACM Trans. Embedded Computer, Syst. 4(3): 549-568 (2005)

[11] Mitali A. et.al. "A Survey on Impact of Embedded System on Teaching". MIT International Journal of Electronics and Communication Engineering Vol. 3, No. 1, Jan. 2013, pp. 36–38.

[12] Stephen A. Edwards. "Experiences teaching an FPGA-based embedded systems class" ACM SIGBED Review - Special issue: The first workshop on embedded system education (WESE) Volume 2 Issue 4, October 2005 Pages 56-62.

[13] João M. et.al. "Teaching Embedded Systems Engineering in a Software-Oriented Computing Degree". Proceeding of 37th ASEE/IEEE Frontiers in Education Conference, 2007.

[14] Embedded Systems Curriculum – A White Paper – 2/8/2006.

[15] Nakutis, Þ, Saunoris, M. "Challenges of embedded systems teaching in electronic engineering studies" Electronics and Electrical Engineering. ISSN 1392-1215. 2010, nr. 6(102), p. 83-86.

[16] Rainer Domer, et.al. "Introduction to Hardware-dependent Software Design". Proceeding of the 2009 Asia and South Pacific Design Automation Conference (ASP-DAC'09); pages 290-292.

[17] Wang CL, Yao B, Yang Y, Zhu Z, "A Survey of Embedded Operating System". http://3c.nii.org.tw/3c/silicon/embedded/2003-03/0228/Survey%20of%20Embedded%20OS.pdf

[18] Christian Obkircher. "Anytime, Everywhere? A Critical Comparison of Different Teaching Concepts in Embedded Systems Education". Term paper: Didactics in computer engineering, Vienna University of Technology, Pages 1-4, JUNE 02, 2007. https://ti.tuwien.ac.at/ecs/teaching/courses/dinf-ss08/papers/obkircher.pdf

[19] K. Ferens. "Teaching Embedded System Design". Proc. 2012 Canadian Engineering Education Association (CEEA12) Conf. pages 1-4. 2012.

[20] Peter Bertels. Et. al. "Teaching skills and concepts for embedded systems design". Journal of SIGBED Review, Vol. 1, pages 1-4. 2009.

[21] Joao M. Fernandes, R. J. Machado. "Teaching embedded systems engineering in a software oriented computing degree". Proceeding of Frontiers In Education Conference - Global Engineering: Knowledge Without Borders, Opportunities Without Passports, 2007. FIE '07. 37th Annual, pages 5-10. 2007. file:///C:/Users/nfadel/Downloads/2007-FIE-ASEE-IEEE%20(3).pdf

[22] Koopman, P., "Lessons Learned in Teaching a Complex Distributed Embedded System Project Course," CPS-Ed 2013, April 8, 2013. file:///C:/Users/nfadel/Downloads/Lessons_Learned_in_Teaching_a_Complex_Distributed_Embedded_System_Project_Course%20(5).pdf

[23] Koopman, P. & Szilagyi, C., "Integrity in Embedded Control Networks," IEEE Security & Privacy, 2013.

[24] Kai Qian, Xiaolin Hu, and Liang Hong. "Experience on Teaching Multiple CS Courses with Portable Embedded System Labware in a Box". Proceedings of the World Congress on Engineering and Computer Science 2011 Vol I, WCECS 2011, October 19-21, 2011, San Francisco, USA. http://www.iaeng.org/publication/WCECS2011/WCECS2011_pp257-259.pdf

[25] Martin T., et.al. “Experiences from large embedded systems development projects in education, involving industry and research”. Proceeding of the 12th ACM, issue (Volume 4, Number 1), January 2007. Special Issue on the Second Workshop on Embedded System Education.

[26] Martin G., and M. Törngren. “How should Embedded Systems be taught? Experiences and snapshots from Swedish higher engineering education”. ACM SIGBED Review, 2(4):34-39, Oct 2005.

[27] Martin G and M. Törngren. “What is embedded systems and how should it be taught? – Results from a didactical analysis”. In the ACM Transactions on Embedded Computing Systems (TECS); Special Issue on Education, Volume 4, Issue 3 (August 2005).

[28] Paul C. et. al. “Guidelines for a graduate curriculum on embedded software and systems”. ACM Transactions on Embedded Computing Systems (TECS). Volume 4, Issue 3 (August 2005), pages: 587 – 611.

[29] Martin T. et al. “Towards an industrial Framework for Embedded Systems Tools”. First Workshop on Hands-on Platforms and tools for model-based engineering of Embedded Systems (HoPES'10) – at the European Conference on Modelling Foundations and Applications, June 2010.

[30] D. D. Gajski, S. Abdi, A. Gerstlauer, and G. Schirner, Embedded System Design: Modeling, Synthesis, Verification, Springer, ISBN 978-1-4419-0503-1, July 2009.

[31] Yu Lo, S. Abdi, D. Gajski, "Transaction Level Model Automation for Multicore Systems" in Behavioral Modeling for Embedded Systems and Technologies (Gomes and Fernandes, eds), IGI Global, Hershey, Pennsylvania, 2009

[32] P. Marwedel, Peter Marwedel (2003). "Embedded system design", Springer.

[33] Cuellar, M.,Chung, E. and Lucido, J.A. (2012) http://www.usc.edu/programs/cerpp/docs/CERPPASHE_Paper_final.pdf

[34] Jacobs R., Marsha, M. and Prince, S. (2008) http://ace.arkansas.gov/cte/informationForms/curriculumFrameworks/Documents/Frameworks/WPR-Frameworks%2011-08.pdf

[35] ARTIST network of excellence. Guidelines for a graduate curriculum on embedded software and systems. (2003)

[36] Bastian Haetzer, Gert Schley, Rauf Salimi Khaligh, and Martin Radetzki. 2011. Practical embedded systems engineering syllabus for graduate students with multidisciplinary backgrounds. In Proceedings of the 6th Workshop on Embedded Systems Education (WESE 11), Jeff Jackson, Peter Marwedel, and Kenneth Ricks (Eds.). ACM, New York, NY, USA `

[37] University of Pennsylvania, "Master of Science Program in Embedded Systems". http://www.cis.upenn.edu/grad/embedded.shtml (visited in July 2011).

[38] Delft University of Technology, "Master of Science Program in Embedded Systems". http://home.tudelft.nl/en/study/master-of-science/master-programmes/embedded-systems

[39] Universitat Stuttgart, "Masters Program in Information Technology (INFOTECH)". http://www.uni-stuttgart.de/infotech/

[40] Mondragon-Torres, Antonio F; Christman, Jeanne W, (2013) "A comprehensive embedded systems design course and laboratory," Microelectronic Systems Education (MSE), 2013 IEEE International Conference, vol., no., pp.56,59.

[41] Peter Bertels, Michiel DHaene, Tom Degryse, and Dirk Stroobandt. 2009. Teaching skills and concepts for embedded systems design. SIGBED Rev. 6, 1, Article 4 (January 2009)

[42] Ricks, K.G.; Jackson, D.J.; Stapleton, W.A.,(2008). "An Embedded Systems Curriculum Based on the IEEE/ACM Model Curriculum," Education, IEEE Transactions on, vol.51, no.2, pp.262,270

[43] Takayama, Y.; Koga, T.; Nitta, T.; Yanagisawa, H.; Shigemura, T., (2012). "Curriculum design for engineering education on embedded system based on broad partnership with university, corporation and local school," Teaching, Assessment and Learning for Engineering (TALE), 2012 IEEE International Conference on, vol., no., pp.T1D-1, T1D-7.

[44] Yu Zhang; Zhaoqing Wang; Licheng Xu, (2010) "A global curriculum design framework for embedded system education," Mechatronics and Embedded Systems and Applications (MESA), 2010 IEEE/ASME International Conference on , vol., no., pp.65,69.

[45] Winsler, Adam. School readiness: The need for a paradigm shift. Carlton, Martha P.; School Psychology Review, Vol. 28(3), 1999. Special issue: Beginning school ready to learn: Parental involvement and effective educational programs. pp. 338-352.

[46] The Condition of College & Career Readiness ,2013

[47] ACT’s College Readiness System Meeting the Challenge of a Changing World , 2014 www.act.org/research/policymakers/pdf/NCSPolicyBrief.pdf.

[48] College and career readiness , A review and analysis conducted for Generation Next ,2013

[49] John T. E. Richardson. Students’ Approaches to Learning and Teachers’ Approaches to Teaching in Higher Education. Educational Psychology. Vol. 25, No. 6, December 2005, pp. 673–680

[50] GIOVANNI DE MICHELI. Hardware/Software Co-Design. PROCEEDINGS OF THE IEEE, VOL. 85, NO. 3, MARCH 1997.

[51] Carlton, Martha P.,Winsler, Adam. School readiness: The need for a paradigm shift. School Psychology Review, Vol. 28(3), 1999. Special issue: Beginning school ready to learn: Parental involvement and effective educational programs. pp. 338-352.

[52] Barbara T. Bowman, M. Suzanne Donovan, and M. Susan Burns, Editors; Committee on Early Childhood Pedagogy; National Research Council. ISBN: 0-309-50389-2, 468 pages, 6 x 9, (2000)

[53] Meisels. Readiness for School-Educators’ Perceptions and the Australian early Development Index. JOURNAL OF AUSTRALIAN RESEARCH IN EARLY CHILDHOOD EDUCATION. VORUME 15 ISSUE 2- 2008.

[54] Trigwell, K., & Prosser, M. (1993). Approaches adopted by teachers of first year university science courses. Research and Development in Higher Education, 14, 223–228.

[55] Edward A. Lee Guest Editorial in System Design Frontier, Volume 2, Number 1, January 2005 Shanghai Hometown Microsystems Inc http://www.hwswworld.com/