Towards Teaching Embedded Parallel Computing: An Analytical Approach

Zain Ul-Abdin and Bertil Svensson
Centre for Research on Embedded Systems,
Halmstad University, Halmstad, Sweden
e-mail: {zain-ul-abdin, bertil.svensson}@hh.se

Abstract— **Embedded electronic systems are finding increased applications in our daily life. In order to meet the application demands in embedded systems, parallel computing is used. This paper emphasizes teaching of the specific issues of parallel computing that are critical to embedded systems. We propose an analytical approach to deliver declarative and functioning knowledge for learning in the field of computer science and engineering with a special focus on Embedded Parallel Computing (EPC). We describe the teaching of a course with a focus on how parallel computing can be used to enhance performance and improve energy efficiency of embedded systems. The teaching methods include interactive lectures with web-based course literature, seminars, and lab exercises and home-assigned practical tasks. Further, the course is intended to give a general insight into current research and development in regard to parallel architectures and computation models. Since the course is an advanced level course, the students are expected to have a basic knowledge about the fundamentals of computer architecture and their common programming methodologies. The course puts emphasis on hands-on experience with embedded parallel computing. Therefore it includes an extensive laboratory and project part, in which a state of the art manycore embedded computing system is used. We believe that undertaking these methods in succession will prepare the students for both research as well as professional career.**

Keywords-component; embedded, multicore, manycore; university course

I. INTRODUCTION

Parallel computers used to be very big – or very small. The development of the big ones was driven by the need for high-performance scientific and industrial computing in large data centers. The small ones were typically needed for high-performance signal processing in embedded systems with volume and power constraints. Today – and in the future – almost all computers are parallel, and knowledge of parallel computing is necessary for all computer engineers and computer scientists. However, when it comes to engineering of computer-based systems, it is within the embedded systems area that you find the closest connection between different design decisions and their effects on the performance of the targeted system. For designers of advanced embedded systems, deep and up-to-date knowledge of parallel computing is absolutely necessary. This is the background of the course "Embedded Parallel Computing", given at Halmstad University. However, developing methods for teaching advanced courses at the university level requires an approach that takes into account the merging trends and is well integrated with the rest of the program.

We present an approach for teaching and learning intended for advanced courses to impart declarative and functioning knowledge in the area of computer science and engineering with emphasis on Embedded Parallel Computing (EPC). The primary aspect considered in using these methods is to create a relationship between the teaching and research activities, which also helps motivate the students to pursue research in their future. The intended learning outcomes considered for the said course are to invoke critical thinking among the students about the methods employed to design next generation computers and about the methodologies needed to program these devices efficiently to achieve better performance and energy conservation. Thus, the intended learning outcomes contribute to developing the intellectual capabilities both from the perspective of theory and practice.

The course is given at, what in Europe is called, the advanced level, i.e., as part of a Master's program. It is given during the third quarter of the first year of the two-year Master's program "Embedded and Intelligent Systems", a Computer Systems Engineering program at Halmstad University. Students are expected to know computer organization and standard computer architecture, as well as common programming methodologies. They are also expected to have programming experience and computer design experience at a level that you expect after a Bachelor's degree in Computer Engineering or Computer Science.

The size of the course is 7.5 ECTS credits, i.e., one eighth of an academic year. During the same semester as the EPC course, the students also study advanced courses on Cyber-Physical Systems, Distributed Real-Time Systems, and Real-Time Networking. During the following semester, the students are applying their knowledge in this and other subjects in an interdisciplinary project course where students following three different specializations work together. (The three specializations are: embedded systems; intelligent systems; and communication systems). Then, in the final semester, the Master thesis project is pursued, typically a research related task within the chosen specialization.

The course is intended to provide knowledge of how parallel computing can be used as a way to meet application demands in embedded systems, such as performance and power efficiency. Further, it is intended to give a general insight into current research and development in regard to parallel architectures and computation models.

Upon completion of the course, the student shall be able to:

• describe and explain the most important parallel architecture models, as well as parallel programming models, and discuss their respective pros, cons, and application opportunities,

• practically demonstrate her understanding of parallel architectures and programming models by programming parallel computer systems intended for embedded applications,

• judge, evaluate and discuss how the choice of programming model and method influences, e.g., execution time and required resources,

• relate the state of the art in the area to the current research and development, in particular such research and development that is important for the design of embedded systems,

• read and understand scientific articles in the area, to review and discuss them and to make summaries and presentations, and

• find relevant research publications and research groups in the area and have acquired an ability to continuously follow the development through journals and conference publications.

The course has been given for about 15 years, first under the name "Parallel Computer Architecture" and during the last four years under its current name, "Embedded Parallel Computing". Although embedded computing has been in focus for the course during all the 15 years, the change of name emphasized this orientation and was accompanied by a change of laboratory and project equipment to encompass a state-of-the-art manycore processor for high-end embedded systems.

II. Teaching the Embedded Aspects

Nowadays we are witnessing an emergence of embedded systems in our daily life. They are found in cell phones, broadband modems, game consoles, digital cameras, cars, health care equipment, home appliances, robots, etc. The features offered by these embedded devices increase at a fast rate, which creates a need for more powerful embedded computing resources. In recent years many such embedded systems include multiple cores, making them embedded parallel computers.

The embedded nature of these computing devices makes them stand out compared to more traditional parallel computers like general-purpose computers with many cores or super and cloud computers. The typical characteristics of an embedded system include:

- Timing guarantees and real-time operation
- Tough reliability and security requirements: working in safety-critical and/or mission-critical situations
- Tight integration of software and hardware components, which consequently often need to be developed together
- Often single chip solutions (still with many cores) leading to constraints for memory, I/O bottlenecks and on chip communication solutions like network-on-chip.
- Energy efficiency and battery life-time issues are paramount for operation of the system
- Often specific hardware augmentation is added and heterogeneous cores are used

Typically the development process adopted for the embedded systems involve using vendor-specific proprietary tools and it is quite difficult to fulfill the above-mentioned demands using these tools. It has also been observed that a large part of applications for embedded parallel computers, like multimedia and sensor signal processing, lend themselves to be described in streaming and dataflow programming languages. This opens the possibility for developing more specialized tools, supporting domain-specific languages, which in turn enable more widespread adoption of embedded parallel computers in academia. Teaching the students to make effective use of these tools in their professional career is of utmost importance for their professional development in the industry. We aim in the EPC course to teach the students to understand the effects of the different constraints imposed in an embedded system so that they are able to deal with the consequences of such limitations in their professional career.

III. The Analytical approach

The proposed bottom-up approach has been inspired by the ideas of Patt and Patel [1] and we use the dialogic concept map shown in Figure 1 to illustrate the threshold concepts related to embedded parallel computing [2]. The basic concept is that learning in engineering subjects closely resembles the process of manufacturing or designing [3], where it is important to break down the overall problem into smaller pieces or tasks and then integrate these pieces to build the complete product. Therefore the students must begin with learning the fundamental components or the building blocks and thereby learn by understanding, before they can apply the gained knowledge to solve real-world problems.

This approach is in many ways consistent with the active learning models suggested by Harasim [4], according to which knowledge is sought out and becomes relevant because of its usefulness to discipline-specific action and induces problem-solving skills. To some educationalists this may not seem to be an ideal model but it is pragmatic with respect to the knowledge delivery mechanisms and the interactions between knowledge, study habits, and the teaching arrangements based on the 3P model of teaching and learning proposed by Biggs [5]. Such approaches, based on teaching strategies and tools and designed to support learning through utilizing digital techniques effectively, will be preferable in the long run as highlighted by Säljö [6]. The 'learning by doing' metaphor emphasizes the creative element in the interpretive activities of learners; which makes it much better than the traditional methods of behaviorist and cognitivist pedagogy [4].

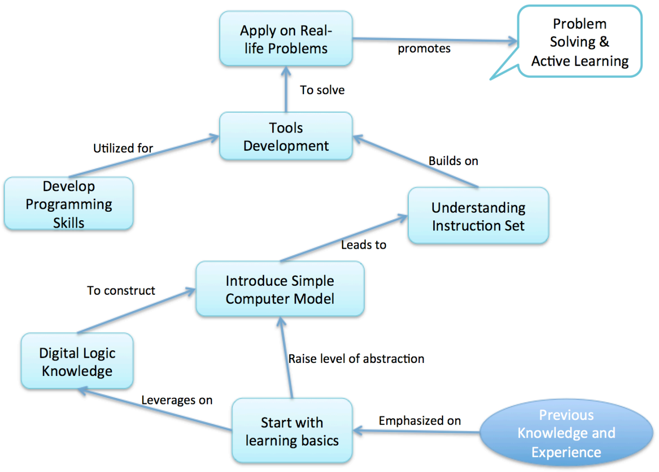
Apply on Real-life Problems
promotes
Problem Solving & Active Learning
To solve
Utilized for
Tools Development
Builds on
Develop Programming Skills
Understanding Instruction Set
Leads to
To construct
Introduce Simple Computer Model
Digital Logic Knowledge
Raise level of abstraction
Leverages on
Start with learning basics
Emphasized on
Previous Knowledge and Experience

Figure 1 Dialogic concept map leading to the intended learning outcomes.

IV. Applying the Proposed Approach to the EPC Course

Considering the instructional value and time and cost for development, we have decided to use the following three methods as the ways and methods for teaching and learning in the EPC course. The said teaching methods have been designed in such a way that they are based on the constructivist learning theory [4], where the students are engaged in building the knowledge and analysis of real-world case studies.

The course is divided into a lecture part, a laboratory part including a small project, and a seminar part. The students are expected to have previously studied courses including calculus, linear algebra, digital logic design, and computer organization. By leveraging the pre-requisite knowledge of the students, the students are taught a very simple but still rich enough model of a parallel computer system. The programming skills of the students are then developed along with the use of advanced tools such as compilers, simulators, and emulators to utilize the studied parallel computer for realizing real-world applications and perform complex analysis.

A. Lectures

Conducting lectures is found to be one of the traditional methods of delivering declarative knowledge. However to achieve better constructive alignment between the intended learning outcomes and the teaching methods in advanced courses, one has to use some innovative techniques in delivering lectures. We have adopted the reflective teaching technique when developing the tasks undertaken during the lectures, thereby collaborating with other peers during the development of lecture contents. This whole process is repetitive as it leads to continuous improvement in collaboration such that it exposes the opportunities for further analysis and highlighting areas requiring pedagogical enhancement. The lecture part initially gives a motivation for parallelism, based on demands on embedded computing (such as performance and power efficiency) and applications that require parallelism. Then it presents the fundamentals of parallel architectures (forms of parallelism, SIMD, MIMD, dataflow, reconfigurable architectures, interconnection networks, etc.) and parallel programming models (shared memory, message passing, stream programming, communicating sequential processes, process networks, etc.). Example architectures and programming techniques are presented and discussed.

As a textbook for the basics of parallel processing technologies we use the book by Hennessy and Patterson, "Computer Architecture: A Quantitative Approach" [7]. One advantage with this book is that it is electronically available in full-text form through our University library. The book needs to be complemented with other texts; descriptions of some topics particularly important from the embedded systems point of view can be found in the books by Kornaros [8] and Wolf [9]. We also refer the students to important articles in journals, like articles by Borkar [10] and articles that exemplify architectures as found in Burges [11]. We furthermore include relevant articles from our own research [12].

Undertaking interactive tasks during the course of lecture keeps the students active and at the same time, it enables the teacher to take corrective measures such as providing more explanation of a particular concept or provide suitable illustrations to make it clearer. This allows the students to become active learners rather than passive listeners. It also helps in initiating the thought-provoking process among the students and they are encouraged to ask questions or clarifications regarding the contents covered in the lecture. The collaborative approach adopted in our course can be considered a variant of the scaffolding concept proposed by Vygotsky [4], where emphasis is shifted from instruction to collaboration as part of the process of learning. However, we are not proposing the use of any specialized computer-assisted learning environment [13].

B. Laboratory and Project

In contrast to courses in social and basic sciences where the intended learning outcomes mainly deal with declarative knowledge, computer science and engineering have an emphasis towards applied science, and thus it requires imparting both declarative and functioning knowledge. We have considered following the fill-up-the-tanks model [14], where the declarative knowledge is first built up in the lecture part of the course, which is followed by a number of practical exercise sessions to apply the knowledge gained during the lectures. Carrying out such practical tasks is inspired by the Papert's constructionism [15] and it will enhance the problem solving capabilities of the students and will develop their creative abilities. The laboratory part provides hands-on experience of embedded parallel computing, primarily based on manycore processors on a chip and their available programming tools. The architectures of choice during the last few years have been the Ambric architecture [16] and the Epiphany Manycore [12].

For the Ambric architecture, we are using the aDesigner tool-suite from Nethra Imaging, which includes the eclipse based Integrated Development Environment (IDE), compiler, assembler, code generator, debugger, simulator, and performance profiler. For the actual implementation on hardware we use the GT-Board consisting of an Am2045 chipset that contains 336 RISC processors. For the Epiphany architecture, we are using the standard C programming environment based on GNU tool-chain (gcc, binutils, gdb) which allows the execution of regular ANSI-C programs on the Epiphany cores. We have been using the Parallella board [17] equipped with a 16 cores E16G301 device, for realizing the actual implementation.

In the laboratory part the tasks involve implementation of a color-space conversion algorithm, which is a common step in very many embedded image-processing tasks. During the first laboratory session the students familiarize themselves with the aDesigner IDE and Epiphany SDK while implementing a sequential version of the said algorithm. A parallel version of the same algorithm is implemented in the second laboratory session. We have also recently introduced a laboratory session based on the Open Virtual Platform (OVP) [18], in which the students get the opportunity to model the hardware manycore platform in a virtual environment and perform early design simulations.

In the project part the students get the opportunity to implement a scalable version of a fractal image calculation algorithm which they can then execute on different numbers of processors and thereby analyze the speedup and other performance parameters. The project task is performed in groups of two students, which promotes the culture of collaborative learning through teamwork, where every individual student will be responsible for a particular task and the combination of their individual modules will lead to the completion of the home-assigned project.

C. *Seminars*

Seminars have been considered to be one of the effective ways to develop knowledge-building discourse [19] in the class room as it allows the students to explore the past research related to a particular field and to be able to witness the latest developments and trends being followed in academia and industry in the given area. In the seminar part of the course, course participants make detailed studies of various sub-areas or specific architectures and lead seminars in these. The university's research projects are included in these special studies. Typical seminar topics include: array processors on a chip; parallel processing with graphics processors; mobile supercomputers; energy efficiency in parallel computing; networks on chip; dataflow programming; and reconfigurable computer architectures. The seminar sessions are moderated by the teacher in order to stimulate discussion among the students, whereby the participants can agree to disagree and reach a level of intellectual convergence, understanding and consensus. Taking the responsibility of conducting one seminar session each helps develop the following abilities of the student:

- The student is able to identify the sources of important literature related to a particular topic and reproduce the existing scientific knowledge in the course perspective.
- It enables the student to present the collected information to other fellows in the form of a collaborative discourse.
- It helps to develop the presentation skills of the student and enables her/him to get formative feedback from peers.
- It also allows the students to get a first-hand experience of methods to conduct research in a particular field such as to be able to hypothesize and reach convergence through intellectual synthesis.

D. *Instruction*

Instruction is in the form of lectures, laboratory sessions, project tutoring and seminars. The latter are given by the students based on literature studies and discussions on subjects determined in consultation with the course instructor. All participants in a seminar (i.e., all students) prepare themselves by reading an introductory text for the topic, while those responsible for the seminar search additional information and make presentation of this as a background for further discussion.

The project and the seminars shall be documented in short reports. The laboratory sessions are mandatory, and also leading at least one seminar.

V. Experiences

We have only positive experiences from several years of teaching the EPC course. Possibly the most important effect of the course is that the way it is organized and taught stimulates the students' interest in the topic and consequently makes them anxious to continue to follow the development in this fast-moving field. It is more or less always the case that students, when searching material for the seminars, find interesting material that the instructors were not aware of; thus, indirectly, the students' work even influences the direction of the research at the department.

The close connection to the research frontier also serves as an inspiration for the student to contribute to research by performing Master's thesis research or taking on PhD studies. Through the work with the seminars, the students have come to know where in the world important research is taking place, and this stimulates the students to explore the possibilities of research studies at these locations. We are satisfied to see our graduates as PhD students in several prominent research groups all over the world. Of course, we also recruit students to our own research on topics such as high-level programming methods, mapping to parallel architectures, and development of application development tools. The topics studied in the course provide the alignment between scientific development and industrial development and are also of vital importance to the industry; this serves as a breeding ground for the students to become familiar with the challenges faced in industry.

VI. Future Developments

A. Laboratory in Embedded Environment

The goal of the laboratory part of the course is that the students gain hands-on experience on real embedded hardware. The students learn the effects of using different programming techniques to get better performance and energy efficiency while dealing with the limitations imposed by the embedded environment.

Currently we are using the vendor's proprietary tools for programming, simulation, and performance evaluation of the embedded hardware. In the future, we plan to use high-level programming languages such as `occam-pi` and `CAL` for programming such massively parallel processing architectures, thus making use of the in-house developed compiler framework to compile the high-level application description to the native languages of the target architecture [20][21]. The development methodology based on high-level languages and tools supporting such languages allows the students to check the functional correctness of the application software before dealing with the low-level issues of any particular hardware. The use of higher-level domain-specific languages facilitates the application development in the form of abstractions. The knowledge gained by the use of such languages will prepare the students for building their own tools later on in their professional career, fulfilling the demands of the industry.

We also envision incorporating tools and developing methods to allow distance-learning students to participate in the course and perform laboratory exercises. We have already introduced the open virtual platform in the laboratory exercises, which is available online and can be used by students remotely. As a next steps towards a totally online delivery model, we will be adopting blended course model, where lectures and seminars will be delivered in the form of webinars using online distance-learning platforms such as edX [22] or coursera [23].

B. Wiki for developing course contents

For the seminar part of the course we would introduce a wiki to the students, in which all the literature used for the seminars including student group presentations will be placed, eventually resulting in a rich collection of material that can be used by other institutes for teaching related courses. This collected literature will also be used in evolving the course contents.

C. Relate to the NSF/TCPP Initiative

Although we are already addressing some of the key concerns raised in the NSF/TCPP report [24], such as introducing the concepts of parallelism and distribution and allowing the students to make use of these concepts in their laboratory implementations, there are still some parts of the course that could be improved in the future based on the proposals in the report.

For instance we would like to strengthen the laboratory part by introducing streaming signal processing algorithms that will develop the student's skills to exploit the implicit parallelism of the application while realizing it on the massively parallel hardware. The students will also get the opportunity of experimenting with different algorithmic approaches and high-level language constructs that enable them to perform design-space exploration for achieving better performance and energy efficiency.

We would also like the students to learn the tradeoffs in the mapping process to cover the diverse platforms with specific interconnect structure, memory organization, and computational nodes, while producing efficient implementations at the same time.

VII. Summary

An analytical approach towards imparting both declarative and functioning knowledge in the field of computer science and engineering has been presented and discussed in relation to the learning strategies introduced in the last century. The proposed approach sub-divides the overall contents of the course into several modules, which integrates teaching activities such as, delivering interactive lectures, conducting seminars, and holding lab sessions together with self-directed project. Interactive lectures allow the students to become active learners rather than passive listeners during the lectures, and it also enables the teacher to reflect upon the effectiveness of the knowledge delivery mechanisms incorporated in the course. Conducting seminars helps in developing the reasoning process among the students and enhances their abilities to practice research related activities in a given area. Designing and implementing projects lead to developing the student's creativity and allows them to apply their knowledge practically.

Acknowledgment

We are grateful to the hundreds of students who enthusiastically have contributed to the course through active work on seminars and projects. We also acknowledge the support from Nethra Imaging and Adapteva, which has made it possible to give the important hands-on experience of embedded parallel computing.

References

[1] Y. N. Patt and S. J. Patel, *Introduction to computing systems: From bits and gates to C and beyond*, MacGraw-Hill, 2001.

[2] N. Entwistle, *Teaching for understanding at university: Deep approaches and distinctive ways of thinking*, Palgrave Macmillan 2009.

[3] D. H. Jonasen, "Technology as Cognitive Tools: Learners as Designers", *Instructional Technology Forum*, 1994.

[4] L. Harasim, "Learning Theory and Online Technologies", Routeledge, 2012.

[5] J. B. Biggs, “From theory to practice: A cognitive systems approach”, Higher Education Research and Development, 12, 73-86, 1993.

[6] R. Säljö, “Digital tools and challenges to institutional traditions of learning: technologies, social memory and the performative nature of learning”, *Journal of Computer Assisted Learning*, 26, 53–64, 2010.

[7] J. L. Hennessy and D. A. Patterson, *Computer Architecture : A Quantitative Approach*, (5th Edition). Morgan Kaufmann, 2011.

[8] G. Kornaros, *Multi-core Embedded Systems (Embedded Multi-core Systems),* 2010.

[9] W. Wolf, *High-performance Embedded Computing: Architectures, Applications, and Methodologies*, Morgan Kaufmann, 2007.

[10] S. Borkar and A. A. Chien,"The future of microprocessors", *Communications of the ACM* 54.5 (2011): 67-77.

[11] D. Burgess et al., "e6500: Freescale's Low-Power, High-Performance Multithreaded Embedded Processor", *IEEE Micro* 32.5 (2012): 26-36.

[12] A. Olofsson, T. Nordström, Z. Ul-Abdin, “Kickstarting High-performance Energy-efficient Manycore Architectures with Epiphany”, in *Proceedings of 48th Asilomar Conference on Signals, Systems, and Computers*, CA, USA, November 2014.

[13] C. Bereiter and M. Scardamalia, “Learning to work creatively with knowledge”, Unraveling basic components and dimensions of powerful learning environments, *EARLI Advances in Learning and Instruction Series*, 2003.

[14] J. Biggs and C. Tang, *Teaching for quality learning at university*, Oxford University Press, 3rd Edition, 2007.

[15] S. Papert, “What is Logo? Who needs it? Logo philosophy and implementation”, Logo Computers Inc. [Online accessed] 31st January 2010. www.microworlds.com/support/logo-philosophy-papert.html

[16] M. Butts, A. M. Jones, and P. Wasson, “A structural object programming model, architecture, chip and tools for reconfigurable computing”, in *Proceedings of 15th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*. (2007).

[17] Adapteva, “Parallella – Supercomputing for Everyone”, 2014. [Online accessed] Nov. 25, 2014. http://www.parallella.org

[18] Imperas, “Open Virtual Platforms”, [Online accessed] Nov. 25, 2014. http://www.ovpworld.org/

[19] M. Scardamalia and C. Bereiter, “Technologies for knowledge-building discourse”, *Communications of the ACM*, 36(5), 37-41, 2003.

[20] Z. Ul-Abdin and B. Svensson, “Occam-pi for Programming of Massively Parallel Reconfigurable Architectures”, *International Journal of Reconfigurable Computing*. Research Article 504815, 2012.

[21] E. Gebrewahid, M. Yang, G. Cedersjö, Z. Ul-Abdin, V. Gaspes, J. W Janneck, B. Svensson, “Realizing Efficient Execution of Dataflow Actors on Manycores”, in *Proceedings of the 12th IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, Milan, Italy, August 26-28, 2014.

[22] edX online learning platform, [Online accessed] Apr. 23, 2015. https://www.edx.org/

[23] Coursera education platform, [Online accessed] Apr. 23, 2015. https://www.coursera.org/

[24] Chtchelkanova et al., "NSF/TCPP Curriculum Intiative on Parallel and Distributed Computing – Core Topics for Undergraduates", Technical Report, 23rd December, 2010. [Online accessed] Nov. 25, 2014. http://www.cs.gsu.edu/~tcpp/curriculum/