Industrially Inspired Just-in-Time (JIT) Teaching

Alex K. Jones, *Senior Member, IEEE* and Steven P. Levitan, *Senior Member, IEEE*

***Abstract*—This paper describes a curriculum that relies on leaving material for instruction flexible for as long as possible prior to class meetings, termed "just-in-time (JIT) teaching." As such, we describe a project-oriented course sequence that integrates several related disciplines in microelectronic-system design with an industrial inspired teaching model. By keeping the material of the course updated JIT to be relevant to students' conceptual discoveries and roadblocks as well as changes in the technology and design tools the students receive a more "real" design experience. The industrial inspired organizational model supports. Our initial results show that the students gain faster insight and retention of material because they are focused on solving the problems rather than just completing the assignment to achieve a grade.**

I. Introduction

Just-in-time (JIT) teaching is thought to be a model of teaching for instructors who are too busy or too lazy to create lesson plans and course outlines well in advance of embarking on the course. However, many instructors find themselves all too often in a situation where a lesson or assignment must be modified, updated, or even redesigned from scratch due to some change in material. In the semiconductor industry, these changes occur quite frequently. Thus, an instructor who has their course material set prior to the beginning of a teaching term may find themselves teaching obsolete material. However, using a form of JIT teaching, the material can constantly be adapted to ensure that the students are always being exposed to the state of the art information.

This paper describes some of the benefits of a course which leverages a controlled form of JIT teaching and several additional teaching innovations in order to provide an adaptive course curriculum that promotes student exposure to state of the art content and better retention of material while pushing students to use problem solving skills. To accomplish this, we use an innovative classroom teaching technique inspired by the industrial research and development model. Students are not only exposed to deadlines and due dates as experienced in the classroom model, but form interdependent teams with an established leadership hierarchy. Milestones are formed and developed by the students and they must react to available technology, problems with tools, or other interdependent projects, etc. Additionally, students are evaluated both in terms of their technical contributions and their function in the industrial project roles (e.g. developer, project lead, manager, etc.). The success of this technique can be assessed both by traditional metrics, such as command of material and progress toward a common class goal, but also by non-traditional metrics, such as project role evaluations.

In particular, we examine the case study of combining two University of Pittsburgh project-oriented courses in VLSI and System-on-a-Chip Design (SoC) into a team, JIT teaching style course with a project of designing a network-on-chip (NoC). Students enrolled in the SoC course are typically responsible for specification design and evaluation, intellectual property development, and system integration/testing tasks. Students enrolled in the VLSI course are typically responsible for back-end synthesis, testing, performance evaluation, and testing. Students enrolled in both courses are responsible for system-design concepts such as developing an interface for an ARM processor soft-core into the system, as well as back-end synthesis, layout, and optimization of the processor and related elements (e.g. memory, bus, etc.).

As this course is currently being offered we are similarly experimenting with JIT paper development. For example, the submitted form of this paper will include some initial thoughts and studies that will be updated upon acceptance with more information from the course as it has progressed to be included in the camera-ready version. We expect final results, including assessment, to be included upon completion of the course for presentation at the conference. Our final step will be to integrate these results into a journal version for submission to the IEEE Transactions on Education.

The remainder of this paper is organized as follows: Section II provides some background on the relevant course sequences offered and previous educational innovations. Our industrial inspired classroom model is described in some detail in Section III. Section IV contains a discussion of the application and some of the benefits of JIT teaching both in general and in context of the NoC project occuring at the University of Pittsburgh. We discuss our proposed assessment techniques in Section V. Finally, in Section VI we describe our initial results and conclusions from our current offering of the course.

II. Background

The JIT teaching described in this paper is separate from the just-in-time-teaching strategy, abbreviated as JiTT, developed in conjunction with Indiana University Purdue University Indianpolis (IUPUI) and the US Air Force Academy [1, 2]. This technique, which has been applied effectively to the science disciplines of biology, chemistry, and physics [3], asks students to complete web-based "warm-up" assignments just prior to class to spark discussion within the actual course meeting. In contrast, the JIT teaching presented in this paper takes into account environmental factors such as available technology, strengths and weaknesses of the students' retention previous material and details of the actual course project to direct the material stressed in each course meeting.

The authors have a history of classroom innovation, particularly related to collaborative teaching and developing shared resources for innovative teaching [4–6]. The pedigogical issues presented in this paper extend these principles in the creation of industrial inspired, JIT teaching concept realized at the University of Pittsburgh in a course to do a complete design of a SoC from high level design down to timing closure of placed and routed layout. Thus the project spans the topics

S. Levitan is with the Department of Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, PA, 15261 USA e-mail: levitan@pitt.edu.

A.K. Jones is with the Department of Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, PA, 15261 USA e-mail: akjones@ece.pitt.edu.

normally covered in a SoC course [7] and a VLSI design project course [8]. This semester we have chosen an NoC system with processor/memory nodes and routers as the two main components in the system.

III. Industrially Inspired Classroom Model

As described in Section II, the project-oriented course case-study is the second term of a course sequence providing the practical component to complement the theoretical background provided by the first term. Traditionally at the University of Pittsburgh, the course sequences of VLSI-Design I/II and Hardware Design/SoC have been taught separately. Part of the innovation of this described project-oriented course is the combined/team teaching of two complementary project courses in what is effectively a single course with a single project while retaining a focus on the appropriate topic area for the students enrolled in either course (e.g. SoC vs. VLSI).

In part, to help accomplish this, we have developed the course using an industrially inspired teaching model as opposed to a standard classroom paradigm of lecture with separate laboratory work. The industrial model includes both the establishment of an industrial style hierarchy as well as utilization of state-of-the-art development practices often employed in industry to spark productivity. In some sense the class turns into a "start-up" style company with its first product as the class term project. The instructors act as the chief executive officer (CEO) and chief technology officer (CTO) with graduate teaching assistants filling the roles such as chief operating officer (COO) and chief financial officer (CFO), etc. As this course contains graduate and undergraduate students, the graduate students are expected to take on more leadership roles such as manager and project leads, although, undergraduates are also given an opportunity to fill these roles as their demonstrated capabilities allow.

During the term, students work in groups of two and four people. One group is assigned to overall project integration and testing, while all other groups are assigned projects that compose individual pieces of the system. Typically, groups are created that include two students from the SoC course to develop the specification and hardware IP for a particular component. This IP is passed to two VLSI students who take the IP to hardware realization and optimize the ultimate design. This creates a feedback loop between the two teams to optimize the design and adjust the specifications as appropriate.

Each class meeting contains time for new material relevant to the current or future projects to be presented initially by the instructors and then eventually by the project teams in the form of status reports. Upon completion of the presentation portion of the class meeting, the remaining time is provided for students to interact to discuss specifications, milestones, deliverables to hand off between teams and to the customer, etc. It is during this time that more formal project team meetings often occur say between a manager or CEO/CTO and various project leads, etc. It is expected that teams will also use this time for more informal information passing.

A motivation for the two person teams is that we promote the use of industry oriented design practices for rapid development. For example, teams use techniques such as *pair programming* [9–14] and *extreme programming* [15–20], where each of the developers can both independently and collaboratively develop a solution to the assigned problem.

A. Pair Programming

Pair programming is a style of programming where two developers work side-by-side, collaborating on the same design, algorithm, code, or test. One programmer, the *driver*, has control of the keyboard/mouse, and actively implements the program. The other programmer, the *observer*, continuously observes the work of the driver to identify *tactical* defects, such as syntactic, spelling, and so on, and also thinks *strategically* about the direction of the work. Because the two programmers periodically switch roles, they work together as equals to develop software. In this configuration, each subgroup of two will utilize pair programming.

B. Extreme Programming

Extreme programming is a methodology that emphasizes many small clean blocks of code to create a larger whole combined with an "early and often" style of code release to other project teams and ultimately the customer. These two principles provide a mechanism for developing tight code, as the developer is focusing on a relatively small task, and for early detection and removal of bugs. Thus, the course project is decomposed into these bite size chunks, typically in the form of IP blocks, for pairs of programmers to develop. These will be integrated on a weekly basis and can be "released to the customer" through testing. During the next week, the customer will report bugs and request additional features to allow the project to evolve over time.

C. Hierarchy

During each term, every student has the opportunity to participate in the management team and/or as a project leader. A single group of students will be assigned the task of management for the current project as a whole under the guidance of the instructors. They also double as the "customers" and will evaluate each code release. The instructors guide the development of specifications for the project but the students and design teams actually create the specifications and coordinate the interaction of development groups. The project leaders are held accountable to senior management for failures to meet deadlines or to satisfy initial project goals.

IV. Impact of Just-In-Time (JIT) Teaching

JIT teaching is a teaching style that comes from the observation that students are most highly motivated to learn the answers to their own questions rather than a scheduled set of fixed concepts. This is supported by researchers from the VaNTH Engineering Research Center [21], who have identified challenge-based instruction as a viable technique to satisfy the "How People Learn" principles [22–24]. HPL is based on a framework containing four concepts that inspire learning and knowledge retention: (1) learner-centered, (2) knowledge-centered, (3) assessment-centered, and (4) community-centered [25].

Challenge-based instruction formulates the material to be taught to the students into a series of challenges. Once

a challenge is issued, the instructor provides or references previously taught information about different methods or tools that can be used to solve the challenge problem. The students are then required to combine their tools to create a final solution to the problem. This requires the students to not only learn the material, but also utilize reasoning skills. JIT teaching, combined with the industrial model of classroom organization, extends challenge-based learning a step further by creating an environment that includes challenges into the scope of a larger project, but also requires flexibility to adapt to changes in environment. These environmental changes might include changes of tools used and intellectual property availability, both developed externally and internally to the class. Additionally, the course promotes interactivity amongst the students in the course to solve problems.

Having students' needs guide the presentation of material is both challenging for the teachers, and more exciting for the class. However, there are several prerequisites for a successful class using JIT. First, JIT works best in a project class environment where there is a project-goal driving the learning process. Our experience shows that the actual project is not important, but rather a project should be chosen that spans the domains that need to be covered in the class. From a practical point of view, the project should ideally be chosen by the class instructors. Second, the students need a basic understanding of the concepts that will be explored in the class. Therefore, JIT should be used in a class that follows prerequisites or classes in the relevant material. Third, the teacher–student relationship in the class needs to be a cooperative, "coaching" relationship rather than a lecture/test relationship. It is important that student evaluation be done in this context so that working for a grade is not the primary motivator for the students.

In terms of implementation, the class is driven by the progress of the students on the project. Students give presentations every class period. These presentations start as casual discussions of the requirements of the project, and early design explorations, but evolve over the semester into formal engineering review sessions. The presentations highlight progress, problems, and also understanding/lack of understanding of both key concepts and practical issues.

Examples of key concepts in our System Design course are synchronization in multiclock systems, metastability, and asynchronous handshake circuits. An example of practical issues would be details in using synthesis tool flow, or writing efficient behavioral hardware code for synthesis. Both of these kinds of problems/issues can be best addressed with JIT teaching. We find that giving a lecture/tutorial tuned to the current problem on the students' mind, what they need to know in order to complete their project, keeps students very focused on the material.

Another aspect of JIT is that in industry technology changes, requirements change, and schedules change. This is very different than in the normal classroom where a homework assignment is fixed (often in the syllabus at the beginning of the term). There is one approach, and one due date, and once the work is completed it is forgotten. In industry, designs are done and redone multiple times, both to explore the design space for a fixed set of requirements as well as to perform re-design under changing requirements. Thus, in the class we emphasize the design process (both the human process and the tool flow) rather than the design itself. For instance we teach the importance of using scripting for driving the synthesis tools vs. the GUI interfaces. A common activity is to first push the design through the flow, top to bottom, then adjust the scripts to optimize the design, and finally to throw it out, and keep the scripts for the next iteration of the design. Students are very attached to their first design but need to learn that it can take many many runs of the tools to achieve good optimization and timing closure.

In our course we initially started with 45nm technology but changed midflight to 130nm due to the availability of a memory compiler compatible with that technology. We have also recently identified the processor core we will be using for our network nodes. Neither of these decisions has changed the project at the high level, but each has implementation implications. For example these decisions have meant presentation of new material on the memory compiler tool flow, development of interfaces between the synthesized memory and the ARM Cortex processor core, and test benches that load software into the synthesized memory and then drive the hardware of the system.

A. Wiki and Documentation

One of the most important components to enabling this course is the use of a central repository for students to interact and share everything from documentation to source code. To accomplish this we utilize a wiki. Students have write and file upload permissions so that they can post design documents and presentations. Class news and updates are posted on the wiki as well as pointers to relevant tutorial information.

Students are required to post their presentations prior to class time, to ensure that the material is captured in a timely fashion. The wiki ensures that student edits do not overwrite any important information and all files are stamped with upload times and author's names.

However, vendor proprietary documents are kept on the Linux platforms that run the tool suites. These sit behind a firewall and the files are in a restricted group access directory.

V. Proposed Assessment

We are conducting several forms of assessment to evaluate the effectiveness of the methods of this pedagogy. First, in our standard course survey we are adding questions inquiring specifically about the unusual style of the course to gauge the students' feelings about the effectiveness of this model. Second, we are conducting "employee reviews" throughout the course. These reviews are based on web-based surveys completed with a separate set of questions filled out by the project lead and the standard developer. The manager questionaire asks for self-evaluation, evaluation of the project, and the developer they worked with. The questions require the student to articulate the purpose of the project, the team's role, the skills utilized, how the work was partitioned, and the strengths and weaknesses the developer they worked with. The developer is asked similar questions and also to evaluate the direction given by the project lead.

Finally, we are evaluating the amount of material (in terms of progress on the project) that can be accomplished through JIT teaching compared with a more traditional classroom organization. For example, the SoC course was offered in isolation using a similar project in which a NoC was developed. The difference was that without VLSI support, the design was targeted to an FPGA using vendor IP for microprocessors, bus interfaces, etc. As the current offering is incomplete, we cannot draw any conclusions at this point. However the initial indications are that the current teaching method and student interaction is motivating productive direction. We expect to provide more information on this during a conference presentation.

VI. Initial Results and Discussion

It is always challenging to convince engineering students that the ground rules of a new class are different than previous classes. In fact several lecture periods were devoted to defining the ground rules for the class. Several students insisted they would rather "just do their own project" than be involved in a single class project, with grading based on participation, cooperation, and dependence on others for success. However, as the class culture has evolved this attitude is changing. One adjustment has been to make it clear that sub-group membership and leadership will not be constant throughout the course. Teams and team leaders will change. It has also been challenging to convince the students that learning the capabilities of the tools is part of their job responsibility. The tutorials provided by the instructors provide only the simplest path through the tools. Many hours of "manual shredding" are required to master the tools. In the same vein, we have discovered that students do not like to read the feedback reports that come from the tools. Error messages are often ignored and warnings are not even noticed. Some very small and fast designs were delivered as "working" when in fact 90% of the design was optimized away due to a synthesis error. Therefore we have made it a requirement that all error messages must be addressed and all warnings understood before a synthesis step is considered completed.

Even with the use of state-of-the-art commercial tools in the classroom, the design process in industry is fundamentally different than the normal student experience. Several important design trade-offs in industry revolve around available resources (designers' time, number of available tool licenses, number of workstations available for simulation, etc.). Even the firmest deadlines in the classroom are often ignored. On the other hand, in industry these deadlines are taken quite seriously. Therefore resource allocation is an important set of skills we are trying to teach. Another set of trade-offs are about build vs. buy IP. Access to Mentor Graphics moduleware and Synopsys designware libraries are often ignored by students who want to do everything themselves at the expense of completing a design on time. The final, and perhaps most important, difference between industry designs and academia is in verification of designs. In the current class we emphasize the need for verification, and have a sign-off process between the SoC teams and the VLSI teams for each IP block. However, we have not imposed standards for test coverage at this point.

The University of Pittsburgh offers two additional courses at the Senior/Graduate level, Hardware Verification and Embedded Systems, that we hope can be incorporated as background courses to this project-course, enabling students having taken those courses to consider those disciplines within the larger group project.

Acknowledgment

Special thanks to Cadence, Mentor Graphics, Synopsys (in no particular order) for the tools, ARM for the processor core and associated programming tools, and MOSIS for supporting JIT teaching.

References

[1] G. Novak *et al.*, *Just-In-Time Teaching: Blending Active Learning with Web Technology*. Prentice Hall, 1999.

[2] E. T. Patterson, "Just-in-time teaching: Technology transforming learning," in *Invention and Impact: Building Excellence in Undergraduate Sceince, Technology, Engeering and Mathematics (STEM) Education*. AAAS, 2003, ch. Successful Pedogogies, pp. 49–54.

[3] K. A. Marrs *et al.*, "Use of warm up exercises in just in time teaching: determining students' prior knowledge and misconceptions in biology, chemistry, and physics," *J. Coll. Sci. Teach.*, 2003.

[4] A. K. Jones, S. Levitan, R. A. Rutenbar, and Y. Xie, "Collaborative vlsi-cad instruction in the digital sandbox," in *Proc. of MSE*, 2007.

[5] A. K. Jones *et al.*, "Exploring rfid prototyping in the virtual laboratory," in *Proc. of MSE*, 2007.

[6] I. Kourtev, R. Hoare, S. Levitan, T. Cain, B. Childers, D. Chiarulli, and D. Landis, "Short courses in system-on-a-chip (soc) design," in *Proc. of MSE*, 2003.

[7] W. Wolf, *Computers as Components: Principles of Embedded Computing System Design*, 2nd ed. Morgan Kaufmann, 2008.

[8] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits: A Design Perspective*. Second Edition: Prentice Hall, 2003.

[9] J. Borstler, D. Carrington, G. W. Hislop, S. Lisack, K. Olson, and L. Williams, "Teaching psp: Challenges and lessons learned," *IEEE Software*, vol. 19, no. 5, pp. 42–48, September-October 2002.

[10] L. Williams *et al.*, "Strengthening the case for pair programming," *IEEE Software*, vol. 17, no. 4, pp. 19–25, July-August 2000.

[11] L. A. Williams and R. R. Kessler, "All i really need to know about pair programming i learned in kindergarten," *Communications of the ACM*, vol. 43, no. 5, pp. 108–114, May 2000.

[12] L. Williams and R. Kessler, *Pair Programming Illuminated*. Addison-Wesley Publishing Company, June 2002.

[13] G. Canfora *et al.*, "Lessons learned about distributed pair programming: what are the knowledge needs to address," pp. 314–319, June 2003.

[14] J. Haungs, "Pair programming on the c3 project," *IEEE Computer*, vol. 34, no. 2, pp. 118–119, February 2001.

[15] M. Lippert *et al.*, *eXtreme Programming in Action: Practical Experiences from Real World Projects*. John Wiley & Sons, September 2002.

[16] S. McConnell, *Rapid Development*. Microsoft Press, July 1996.

[17] K. Beck, *Extreme Programming Explained: Embrace Change*. Addison-Wesley Publishing Company, October 1999.

[18] K. Beck and M. Fowler, *Planning Extreme Programming*. Addison-Wesley Publishing Company, October 2000.

[19] R. Jeffries, *Extreme Programming Installed*. Addison-Wesley Publishing Company, October 2000.

[20] M. Marchesi *et al.*, *Extreme Programming Perspectives*. Addison-Wesley Publishing Company, August 2002.

[21] T. R. Harris, "Vanth erc annual report," Vanderbilt-Northwestern-Texas-Harvard/MIT Engineering Research Center, Tech. Rep., 2005.

[22] A. McKenna *et al.*, "Cross-disciplinary approach to developing challenge-based instruction," in *Proc. of the American Society for Engineering Education Annual Conference & Exposition*, 2003.

[23] E. D. Jansen *et al.*, "Implementation and assessment of challenge-based instruction in a biomedical optics course," in *Proc. of the American Society for Engineering Education Annual Conference & Exposition*, 2003.

[24] R. J. Roselli, "Challenge-based instruction in biotransport," in *Proc. of the American Society for Engineering Education Annual Conference & Exposition*, 2004.

[25] J. D. Bransford *et al.*, Eds., *How people learn: Brain, mind, experience, and school*. National Academy Press, 2000.