Collaborative VLSI-CAD Instruction in the Digital Sandbox*

Alex K. Jones[1] Steven Levitan[1]
[1]University of Pittsburgh
{akjones,levitan}@ece.pitt.edu

Rob A. Rutenbar[2]
[2]Carnegie Mellon University
rutenbar@ece.cmu.edu

Yuan Xie[3]
[3]Penn State University
yuanxie@cse.psu.edu

1 Introduction

This paper describes a graduate level, collaboratively taught VLSI Computer-aided Design (CAD) course concurrently offered between the University of Pittsburgh (Pitt), Carnegie Mellon University (CMU), and The Pennsylvania State University (Penn State). The course leveraged computing facilities at each of the three universities provided by The Technology Collaborative (TTC), formerly the Pittsburgh Digital Greenhouse, called the Digital Sandbox [3]. The course has been offered twice with two different educational goals.

In the first year it ran concurrently at Pitt and CMU with independent instruction between the two universities. Students at both universities completed several projects with similar specifications, however students at CMU used their choice of Java or C/C++ on Linux machines to develop their projects and students at Pitt used Objective-C and Cocoa on Apple Macintosh machines. The goal was to consider the development productivity of the different development environments. Several joint sessions were held after the completion of projects to discuss results.

The second year it ran concurrently at all three universities with lecture primarily originating from CMU that was piped into Pitt and Penn State. Projects were synchronized but no emphasis was placed on which development platform was used to complete them. Near the end, Pitt diverged from the other universities to create a collaborative project that supported a Digital Sandbox VLSI course.

2 VLSI-CAD Collaborative Curriculum

VLSI-CAD is an example of a course with content that is highly variable depending on current research in the field. As a result, it is difficult to select a text for such a course as the material can be quickly dated. The most appropriate texts by Sarrafzadeh and De Micheli were published in 1996 and 1994, respectively. Thus, the course notes in the form of a slide packet was use as the core reference text and teaching tool to support the lectures at all three sites. These slide packets were supported by fundamental papers describing the development of the core concepts of each course topic.

The course was organized into two major topics:

1. Algorithms for logic minimization and their underlying data representations including lectures on Binary Decision Diagrams (BDDs), Boolean Satisfiability (SAT), two-level minimization, multi-level minimization, and basic formal verification.
2. Algorithms for physical design and analysis including lectures on placement, routing, timing analysis, and floor-planning.

During the course two exams were conducted, one on each of the major topics, three homeworks were assigned for each of the major topics, and the students completed a programming project during each half of the course.

2.1 Presentation Materials

The use of slides for presenting material is not always a popular choice for students. In fact, based on course reviews of similar material taught using a whiteboard or with slides, many students prefer the whiteboard in spite of the advantages that slides can present much more detailed figures and slides can be reproduced without errors possible in note taking. The reason for this effect is that students are more engaged in the learning process [5]. To help solve this problem, but retain the use of slides for a fundamental reference for the course, the slides, developed by Rutenbar at CMU, were left incomplete with derivations and information to be filled in during the lecture.

Because the VLSI-CAD course was being taught at multiple sites, it was necessary to broadcast the lecture to multiple locations. Microsoft Netmeeting was used to broadcast the slides. The audio portion of the lecture was provided using a conference phone allowing students at the remote sites to ask questions about the material.

2.2 Development Productivity

CAD software emphasizes different issues of software development than typical applications. For example, VLSI-CAD tools require heavy memory usage to store large

*These collaborative course initiatives were supported by The Technology Collaborative.

circuit designs and extremely efficient data structures to manage the data. It is for this reason that commercial CAD tools typically utilize UNIX operating systems running on 64-bit machines, which can leverage large amounts of system memory and processing power. However, little emphasis is placed on the development productivity for these environments.

During the first offering, the main project from the first half of the course was to develop a SAT solver. Students from CMU developed this in the traditional Linux environment while students from Pitt developed this using Objective-C and Cocoa from Apple Computer. Objective-C and Cocoa have been shown to have several development productivity advantages over other systems [2].

During the second offering, Pitt diverged from CMU and Penn State for the second project. The project completed by CMU and Penn State was an Optical Proximity Correction (OPC) algorithm for shapes that are problematic to be reproduced during the fabrication process. The Pitt project developed a routing engine for the Micro Magic standard cell-based ASIC design flow. The reason for the divergence is that Micro Magic was being used in another Digital Sandbox course in VLSI and Micro Magic includes placement but not routing capabilities.

During the projects students from all three groups were allowed to work in teams of two using the concept of*pair programming* [1]. Use of pair programming and additionally the concept of *extreme programming* [4] were required during the router project at Pitt. Pair programming is a style of programming where two developers side-by-side, collaborating on the same code periodically switching between the *driver* and *observer* role. Extreme programming promotes small, clean, and tight blocks of code by early release of code for detection and removal of bugs.

3 Results and Conclusions

As CAD tools have heavy needs for accessing data, the performance of data structures was important. During the first offering, the students used the SAT solver project to compare Cocoa from Apple with a GNU implementation of Cocoa called GNUstep, and the Standard Template Library for C++ (STL) and results from using a hash table (HT) and linked list (LL) are shown in Table 1. While, Cocoa provided a productivity advantage, the students found that it was not scalable to large problem sizes. Figure 1 shows an example of the router project developed in the second course offering to route an arithmetic-logic unit (ALU) developed in MicroMagic from the VLSI Design course.

The result of the collaborative teaching of the VLSI-CAD course has been very positive for all three universities. Aside from minor technical difficulties from offering the course with three sites for the first time, the feedback from the students has been quite positive. The students felt that they learned a great deal of material and got a unique perspective from multiple instructors. Their hands on programming projects gave them experience thinking about CAD tool construction from an efficiency and productivity perspective.

Table 1. Summary comparison of run times of Cocoa, GNUstep, and STL.

	Average runtime (s) (% worse than best case)		
Operation	**Cocoa**	**GNUstep**	**STL**
HT Insertion	67.9 (2457%)	84.1 (3065%)	2.7 (0%)
LL Insertion	14.8 (836%)	2.0 (24%)	1.6 (0%)
HT Find	3.2 (328%)	21.9 (2802%)	0.8 (0%)
LL Find	4.7 (561%)	1.3 (78%)	0.7 (0%)
HT Traversal	0.6 (0%)	2.8 (379%)	1.0 (77%)
LL Traversal	0.6 (57%)	0.4 (0%)	0.6 (57%)
HT Copy	20.5 (1618%)	95.0 (7881%)	1.2 (0%)
LL Copy	14.8 (1189%)	1.6 (40%)	1.2 (0%)

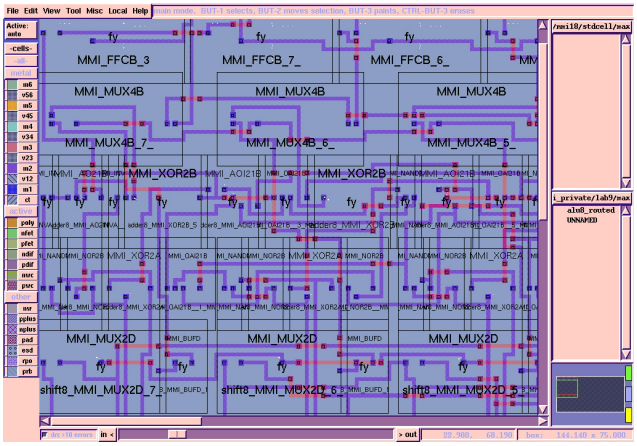

Figure 1. Example of router project in MicroMagic.

References

[1] J. Borstler, D. Carrington, G. W. Hislop, S. Lisack, K. Olson, and L. Williams. Teaching psp: Challenges and lessons learned. *IEEE Software*, 19(5):42–48, 2002.

[2] B. Brady, A. K. Jones, and I. Kourtev. Efficient cad developement for emerging technologies using objective-c and cocoa. In *Proc. of ICECS*, pages 369–372, 2004.

[3] T. Kroll, H. Schmit, and D. Landis. Cad tool support for a multi-university soc certificate program: The digital sandbox. In *MSE*, pages 47–48. IEEE Computer Society, 2003.

[4] M. Lippert, S. Roock, and H. Wolf. *eXtreme Programming in Action: Practical Experiences from Real World Projects*. John Wiley & Sons, September 2002.

[5] M. H. Mickle, L. Shuman, and M. Spring. Active learning courses on the cutting edge of technology. In *Proc. of FIE*, 2004.