Int. J. Engng Ed. Vol. 17, Nos. 4 and 5, pp. 367–369, 2001
Printed in Great Britain.

0949-149X/91 $3.00+0.00
© 2001 TEMPUS Publications.

A Case for Project-Based Design Education*

DAVID HARRIS
Engineering Dept., Harvey Mudd College, Claremont CA 91711, USA. E-mail: david_harris@hmc.edu

Many engineering curricula are partitioned into laboratory classes and theory classes. Theoretical lecture courses can cover the largest volume of material, but are often criticized for not engaging students, and, as a result, instilling little retention of the material. This paper contends that we teach engineering more effectively by integrating the theory with hands-on design projects. This contention, of course, is not new, yet the partition remains in all too many subjects. This paper describes an ongoing experiment of adding design laboratories to Harvey Mudd's Computer Engineering class, historically a purely theoretical subject. It then address some of the benefits and challenges that may carry over to integrating theory with laboratories in other courses.

INTRODUCTION

'I hear and I forget.
I see and I remember.
I do and I understand.' (Confucius)

MANY ENGINEERING curricula are partitioned into laboratory classes and theory classes. Theoretical lecture courses can cover the largest volume of material, but are often criticized for not engaging students, resulting in little retention of the material. I contend that we teach engineering more effectively by integrating theory with hands-on design projects. This contention, of course, is not new, yet the division remains in all too many subjects. I support my argument with an ongoing experiment of adding design laboratories to Harvey Mudd's Computer Engineering class. I then address some of the benefits and challenges that may apply to integrating theory with laboratories in other courses.

CASE STUDY: INTRODUCTION TO COMPUTER ENGINEERING

E114, Introduction to Computer Engineering, a junior level (third-year) course, covers classic computer engineering topics: digital electronics, assembly language programming, computer architecture and microarchitecture, and operating system issues. It has been taught as a purely theoretical subject, but I believe the most effective way to learn computer engineering is to build a computer. During the spring semesters of 1999 and 2000 I have experimented with adding a lab component involving the design and implementation of a 32-bit MIPS processor. The labs followed the design in Patterson and Hennessy [1] using schematic entry, simulation, and Xilinx field-programmable gate arrays (FPGAs).

Computer engineering labs are certainly not a new proposition. At many universities, students have traditionally breadboarded computers with a large number of discrete TTL devices. Such labs have become obsolete as industry has adopted higher levels of integration, building complete systems with a few components or even a single chip. Moreover, breadboarding is a very time-consuming and error-prone task involving cutting, stripping, and placing wires; modern computer-aided design tools can reduce some of the tedium and keep students efficiently focused on the essential experiences of design, specification, and debug. The CAD tools are thus used to save students time, not merely to teach current industrial practices.

The E114 class uses a sequence of twelve labs spread over the semester closely following the theoretical concepts taught in lecture while providing necessary background to use the CAD tools. The labs are listed in Table 1. The introductory labs involve design, schematic entry, and simulation of combinational logic and finite state machines using the Xilinx Foundation FPGA tools, a commercial CAD package donated by the vendor. The principles of modularity and hierarchy, which are difficult to teach in lecture, are directly applied as students reuse the full adder from Lab 1 in a 32-bit ALU for Lab 5 and save time drawing wires by hierarchically assembling the ALU from 4-bit and 1-bit blocks. The ALU and a controller from Lab 2 are again used in a simple processor running a subset of the MIPS instruction set designed in Labs 8-10. Students enjoyed building toward a larger goal and learned about modularity, hierarchy, and design reuse from these projects.

The labs are currently being modified to involve real hardware as well as the real CAD tools and real processor instruction set. A new lab based on

* Accepted 27 November 2000.

Table 1. E114 Lab assignments (Spring 2000)

Lab	Design goal	Learning objectives
1	Full adder	schematic entry, simulation, combinational logic design
2	MIPS controller	more complex combinational logic design & testing
3	Adventure game FSM	hierarchy, sequential logic design & testing, one-hot inputs and states
4	Turn signal FSM [3]	FSM design, logic implementation with FPGAs
5	32 bit ALU	arithmetic units, modularity, systematic verification
6	Fibonacci program	SPIM assembly language simulator, basic assembly language programming
7	FP addition program	assembly language programming, IEEE floating point, FP emulation, debugging
8	MIPS processor (part 1)	processor datapaths, synthesis, memories
9	MIPS processor (part 2)	processor integration and testing
10	Multicycle processor	multicycle MIPS processor, interfacing with external memory and devices, FPGA programming
11	Application	parallel interface with host workstation, embedded software design
12	Optimization contest	enrichment

Copies of the 1999 labs are available on the class web page and solutions are available to instructors from the author: http://www3.hmc.edu/~harris/class/e114

Wakerley [3] involves designing an FSM for the turn signal lights of a Ford Thunderbird and programming the design into an FPGA connected to LEDs and switches. The processor in Lab 10 is intended to be programmed into an XESS board that contains an FPGA, 32K SRAM, and a 7-segment display. Students will add a parallel port to their processor in Lab 11 to communicate with a host workstation, then run embedded software on their processor to compute a fractal and send the results to the host for display. An optional optimization contest at the end of the class will give top students a chance to delve deeper into processor microarchitecture while speeding up their MIPS processor.

TECHNICAL DIFFICULTIES

Although the labs ran relatively smoothly, we encountered some technical difficulties along the way in 1999. Our greatest problem was an unstable Windows NT lab configuration in which not all machines behaved identically. We learned that NT 4.0 is still not as robust as Unix. Upgrading the lab over the summer solved the hardware and operating system problems.

Version 1.5 of the Xilinx tools used in 1999 was not completely stable either. Like most CAD tools used in industry, these tools crash more often than would be desirable. I also found they corrupted at least four student's labs at one point or another during the semester; in these cases the student had to recreate the project or reenter a schematic. Version 2.1i used in 2000 is noticeably more robust. All in all, the computer problems from the first semester have been satisfactorily resolved.

Finally, the Xilinx Foundation Express Verilog synthesis engine is not user-friendly for Verilog beginners. The error messages are sometimes cryptic and many bogus errors are produced by a single real mistake, making debugging difficult for novices. Therefore, the later labs using Verilog in 1999 have been rewritten to use only the schematic editor. Students in the successor microprocessors course have successfully used Verilog for FPGA design, but also found the learning curve steep.

ASSESSMENT

Labs require a significant time commitment from students. This was compensated for by greatly trimming the lengths of the problem sets. To measure the time costs, students were asked on each lab and problem set to report the number of hours they spent. The averages were roughly four hours a week for labs and two hours a week for problem sets. In contrast, the professor teaching the class in past semesters estimated about five hours a week of problem sets.

Student response to the labs was assessed through two surveys. In the midterm survey in 1999 most students rated labs as the most valuable component, over lectures, problem sets, and readings. These results held both for students who wanted to be in the class and for those who were taking the class only because it is required. However, opinions of the class and especially of the lab were much better among the fraction of the students interested in professional practice in the field. In the final survey using the standard form provided by the college, students were not explicitly asked about the labs. Nevertheless, of 35 legible returned comment forms, 22 singled out the labs for positive comments and only two were negative about the labs. Student comments included:

- The labs, though long, were very informative. They really tested if you knew what was going on or forced you to figure it out.
- The labs & homework assignments are involved & fruitful when completed properly. They contribute greatly to the understanding of the course but are also enjoyable.
- The labs really helped me nail down stuff we were learning in class (even though there were some problems).
- Labs are a great idea. The learning curve for Xilinx tools was integrated into the lab progression.

It was not easy to develop exam questions that reflected material from the labs. However, students in 1999 performed better on questions about combinational logic design and microarchitecture practiced in lab than on other questions that had only been covered by problem sets.

A midterm survey in 2000 gave similar positive feedback on the labs. Students found debugging to be a time-consuming but valuable experience and preferred the combination of labs and problem sets over the prospect of doing problem sets exclusively.

Students did not complain about tool problems in 1999 as much as I would have expected. They understood that real tools come with real bugs and accepted the bugs as one more challenge in completing the labs. I also spent many hours in the lab helping students track down problems, especially when I thought the problem might relate to the tools. As mentioned earlier, the tool problems were solved in 2000. Undergraduate lab assistants also provided help on the evenings before labs were due.

The labs were successful enough that my colleague R. Wang used them without modification in the Fall of 1999. No assessment data is available.

BENEFITS AND COSTS OF PROJECT-BASED DESIGN EDUCATION

It is the author's belief that most engineering classes could be taught with better integration of theory and hands-on design labs. One of the greatest benefits is raising enthusiasm among students. For example, the excitement of building one's first computer as an undergraduate. Using industrial tools further links the design to reality in a way that students find satisfying. The labs motivate the mastery of theory; a well-designed design exercise requires application of theory rather than simple tweaking to meet specifications [2]. While such labs have conflicting goals of being tractable, yet nontrivial, the conflict can be alleviated with a sequence of labs in which modest blocks are first developed, then later assembled into a complex whole using some components supplied by the instructor. The sequence also demonstrates good design practices by example. Such practice of modularity, abstraction, hierarchy, and proper documentation is difficult to learn without exploring an existing well-designed system. Optional parts of labs and final design competitions challenge and motivate the strongest students without making the workload too great for others.

Integrating design with theory certainly has its costs. Students take longer to complete a lab than to do a problem set on an equal amount of material. Therefore, the number of concepts covered by the course must be reduced. This gap may disappear when we measure success in concepts learned rather than concepts taught. Another difficulty is that if labs build toward a larger goal, students must be able to successfully accomplish the earlier labs in order to attack the later labs. Therefore, the required parts of labs cannot be excessively difficult. In Spring 2000, the author posted solutions to previous labs so that students who missed a lab would not fall hopelessly behind. This has worked well at Harvey Mudd where the honor code is strongly respected, but might be a problem for less ethical students when labs are reused each year.

CONCLUSION

Overall, benefits of integrating design into a theoretical class outweigh the costs of covering fewer topics. Surveys show that most of the students find the labs to be the most educational part of the class despite the large amount of work they entail. Given that this is not a new idea and has been so successful, the question stands: why don't more theoretical classes introduce a design component?

Acknowledgments—The author thanks R. Wang and J. Rosenberg for fruitful discussions about E114 and digital design education at Harvey Mudd College. The Xilinx University Program donated the Xilinx Foundation tools.

REFERENCES

1. D. Patterson and J. Hennessy, *Computer Organization and Design*, Morgan Kaufmann, San Francisco, CA, 1998.
2. J. Rosenberg, Simulating appropriate uses of simulation in design, *Proc. Computing Futures in Engineering Design*, Harvey Mudd College, May 1998.
3. J. Wakerly, *Digital Design: Principles & Practices* (3rd Edition), Prentice-Hall, Upper Saddle River, NJ, 2000, pp. 584–591.

David Harris is Assistant Professor of Engineering at Harvey Mudd College. He received his Ph.D. in Electrical Engineering from Stanford University in 1999 and his SB and M.Eng. degrees from the Massachusetts Institute of Technology in 1994, where he developed a seminar on chip design for freshmen and sophomores. Dr Harris' research interests include high speed integrated circuits, microprocessors, and complex digital system design. He has consulted at Sun Microsystems and Intel Corporation and is the author of two books on high speed digital integrated circuits. When not teaching or designing chips, he can generally be found climbing mountains.