Teaching VLSI Design To Today's Students

Rama K Vedachalam and George L. Engel

Department of Electrical and Computer Engineering
Southern Illinois University at Edwardsville
Edwardsville, IL 62025-1801

Abstract

This paper describes the successful evolution of a course in VLSI (Very Large Scale Integrated Circuit) design. While a decade ago, it was acceptable for a first-semester course in VLSI design to emphasize only MOS transistor theory, process technology, physical layout, and circuit issues; this exclusively low-level approach to teaching VLSI design is no longer in the best interest of the majority of students. Some claim that since the use of hardware description languages, standard-cell libraries, and high-level synthesis are becoming so widespread in industry; emphasizing low-level circuit issues benefits only a few students. Long-time teachers of VLSI often counter with the claim that while perhaps valid, what is really being offered is a course in system design and not in traditional VLSI design where the emphasis should be on full-custom methodology.

Hence, it has been our observation that, in practice, the majority of introductory courses in VLSI focus on one methodology, while treating the other in a cursory manner. This paper will demonstrate how we successfully teach everything from MOS transistor theory and CMOS process technology through circuit and logic design, up to and including the design and synthesis of digital systems using a hardware description language in a one-semester *introductory* course in VLSI design. Topics addressed in this paper include course content, laboratory exercises, final design project, and the overall effectiveness of using state-of-the-art, industry-standard CAD tools in helping to teach VLSI design to first-time students.

I. Introduction

This paper describes the experiences of a teaching assistant and an instructor in *EE 484: Digital VLSI Design* during the Spring '98 semester at Southern Illinois University - Edwardsville (SIUE). The course is fast becoming a popular elective for graduating seniors and a core course for those graduate students wishing to pursue a career in IC design. The course was introduced four years ago and is offered once per year but may soon be offered every semester. The course has **evolved** over the past four years in hopes of keeping up with the changing nature of VLSI design. It became clear to us that it is no longer appropriate to teach VLSI the way it was taught over ten years ago, where the focus was almost exclusively on low-level circuit issues.. It has

been our observation, however, from a review of WEB course listings and discussions with colleagues that this is frequently the way VLSI courses are still taught. In other cases, however, it appears that low-level circuit issues are almost totally ignored in favor of an exclusive systems level approach to the topic. With careful planning, we feel that there is an acceptable compromise between these extremes. Much of the success of this approach relies upon the use of state-of-the-art, industrial grade software.

The class-size during the Spring '98 offering of the EE484 was twenty-one, five of of whom were undergraduates with the remainder being, for the most part, first- or second-semester master's students. A prerequisite for the course was successful completion of or concurrent enrollment in *EE483: Computer Architecture and Organization.* This pre/co-requisite was critical to the success of students enrolled in EE484. The primary textbook used throughout the course was *Principles of CMOS VLSI Design: A Systems Perspective* by Neil Weste and Kamran Eshragian. A supplementary text, *Verilog HDL: A Guide to Digital Design and Synthesis*, by Samir Palnitkar was used during the second half of the course.

In previous offerings of EE484, the University of California at Berkeley CAD (Computer Aided Design) tools had been used; most notably, Magic for physical layout, SPICE for electrical simulation, and Esim for event-level simulation. This was the first time that industrial-grade IC design tools, generously donated to the department by the *Mentor Graphics Corporation,* were used as the primary toolset. The use of the design tools during weekly, mandatory laboratory sessions were made an integral part of the course. Tutorials and laboratory exercises were prepared during the previous summer. This material, for those interested, is available via the world-wide web at http:://www.ee.siue.edu/~mentor.

II. Course Content

The course material for EE484 covered everything from MOS transistor theory and CMOS process technology through circuit and logic design, up to and including the design and synthesis of digital systems. Students were introduced to a hardware description language, Verilog, and logic synthesis using standard cells in conjunction with autoplace and autoroute techniques.

The art of digital design is changing rapidly. As circuit complexities grow at staggering rates, digital system designers turn more frequently to state-of-the-art IC design tools to help solve the problems they face. While ten years ago, it was perfectly acceptable for a first-semester course in IC design to emphasize only MOS transistor theory, process technology, and circuit design; this exclusively low-level approach to teaching IC design is no longer in the best interest of the majority of students.

While these topics still have a prominent place in a course like EE484; it is equally important to introduce students to hardware description languages (HDL), logic synthesis, and standard-cell techniques. Afterall, VLSI design is *system* design (and more!). Most successful, modern IC designs depend upon a carefully crafted balance between full-custom and standard cell approaches. The successful designer of today is the one who knows how to craft this delicate balance. It is critical that students see both sides of the mix and that they engage in meaningful exercises which introduce them to and give them a chance to practice skills in both of these

design methodologies. We believe it is importantant that students feel comfortable working at all levels and consider all design options.

In our department, we find that our students learn best when concepts discussed in class are reinforced by laboratory exercises. It is for these reasons that students in EE484 were required to spend 1 1/2 - 2 hours per week in lab working with the Mentor Graphics tools. Instruction on how to use the tools was done during the lab periods, and little or no time was spent during the scheduled lecture periods discussing their use. The laboratory component of the course was broken into two parts: preparatory exercises and a team-based design project. The exercises were designed to parallel classroom discussions and to illustrate design concepts. Mastery of skills and tool usage learned during the laboratory exercises were reinforced and evaluated during the design project phase of the course. This fact is highlighted in Table 1, shown below.

Week	Lecture Topics	Laboratory Exercises
0	Introduction, fully-complementary logic design	No lab
1	MOS transistor theory	Schematic entry using **Design Architect**
2	CMOS process technology Resistance and capacitance estimation	Gate-level simulation using **Quicksim** 2-input NAND and 3-input NOR
3	Circuit characterization - analytical models	Electrical simulation using **Accusim**
4	Propagation delay estimation	Physical layout using **ICgraph**: 2-input NAND
5	Power estimation, yield, and scaling	Physical layout using ICgraph: Inverter
6	CMOS logic design styles	Physical layout using ICgraph: 2-input NAND
7	Latches, flip-flops, and registers	LVS and parasitic extraction (**ICverify**, **ICextract**)
8	Clocking strategies, CMOS IC design options	Hierarchical design
9	CMOS subsystem design: counters, comparators Modeling and synthesis using Verilog. Assign design project.	Event-level simulation using **Lsim** Compare/contrast switch and adept (timing) mode
10	CMOS subsystem design: adders Modeling and synthesis using Verilog.	4-bit counter Verilog description(**qvlcom** and **qvlsim**) Synthesis using **Autologic** (**MDK library**) along with **AutoPlace** and **AutoRoute**
11	CMOS subsystem design: multipliers	Work on design project
12	CMOS subsystem design: high-speed multipliers	Work on design project
13	CMOS subsystem design: RAMs and ROMs	Work on design project
14	Controller design: PLAs Using Verilog to describe finite state machines.	Work on design project

Table 1. Sequence of topics and lab exercises

The first half of the semester was devoted to transistor theory, process technology, and low-level circuit issues. A decade ago the majority of the semester was devoted to such issues. The time taken to discuss these issues was shortened by de-emphasizing fabrication (is it really necessary that a first-time student be able to recite all 14 processing steps in a standard CMOS technology?), discussing only first-order effects related to transistor operation (they need only information that will make them good digital designers), and reducing the amount of time spent on technology scaling (A week's worth of lecture had to be spent and several problems had to be worked out for better understanding), yield, and packaging. Strong emphasis was placed on estimating resistances and capacitances on chip as well as a thorough discussion of propagation delay models and the derivation of expressions for the prediction of propagation delay and power consumption. In anything, lectures in these areas were enhanced. Also, a significant amount of time was devoted to discussing the various CMOS design styles, especially dynamic logic styles.

During the second half of the semester, the systems perspective of VLSI design was highlighted. Students were introduced to Verilog. This was possible because of the EE483: Computer Architecture pre/co-requisite where students become proficient users of VHDL. Most found mastering Verilog a snap after expressing designs using VHDL in EE483. Moreover, discussing the *implementation* of high-speed adders and parallel multipliers was possible because students are exposed to these concepts at the algorithmic level in EE483. While admittedly extremely important, the topics of IC testing and design for test were omitted. These topics can be addressed in a advanced course and not necessary for inclusion in an introductory course on VLSI. The capstone exercise of the course was a team-based design project which will be discussed in a later section of this paper.

III. Laboratory Exercises

During the first three weeks of lab, students were given a tutorial describing how to use Mentor Graphics' Design Architect (DA) for schematic entry, Quicksim for functional simulation, and Accusim for electrical simulation of a 2-input NAND gate. They were then asked to design a 3-input NOR gate by themselves and to simulate the design. This presented little problems for the majority of students since they were familiar with DA, Quicksim, and Accusim from previous courses and only had to deal with the fact that they were now working at a lower level of abstraction; namely, the FET as opposed to the gate level. Students were relieved that they did not have to learn another user interface or tool. This fact highlights the importance of introducing an integrated set of design tools early in a student's academic career. Design tools learned during their sophomore and junior years can be used in different ways and more effectively when a student reaches his/her senior year or begins graduate studies.

The fourth week was devoted to the physical layout of a 2-input NAND gate using the layout tool, ICgraph. This was a daunting task because students found it difficult to understand the many different mask layers. To overcome these problems, we were forced to reformat the exercise. In the new tutorial, the students were asked to design a simpler circuit; namely, an inverter and then the physical layout was explained in detail during lab, after which we returned to the original NAND gate tutorial, this time with success! This is why Table 1 indicated that layout of the NAND gate occurred during both weeks 4 and 6 with the layout of the inverter occuring during week 5. It was our observation that it was not until successfully completing

this lab that students really understood material presented in class related to CMOS process technology and IC fabrication. Since the ICgraph tool works with actual mask layers as opposed to the pseudo-layers employed by Berkeley's Magic layout editor, use of the tool is a very effective way to reinforce technology concepts introduced during lecture.

During week 7, students verified the correctness of their layouts using ICverify. This was complemented in lecture by a discussion of the utility and role of design rules in the design process. They also used ICextract to back-annotate their DA schematics. The back-annotated DA schematic was then used to perform Accusim simulations which now included the effects of parasitic capacitances. Students were asked to compare the values of the parasitic capacitances extracted by the tool with hand calculated prediction of these same parasitics using a MOSIS process parameter sheet. Students were also then asked to compare simulation results with hand calculations performed using propagation delay expressions derived in class. The importance of always verifying simulation results and performing "sanity" checks was demonstrated.

The eighth week was given to hierarchical design. Students were asked to use the layouts of the 2-input NAND and 3-input NOR gate to construct a circuit using these leaf cells. A parent cell was created which instantiated the leaf cells. Facilities in IC station such as "Auto route overflows" etc., were illustrated to make the hierarchical design exercise simpler. This exercise paralleled a lecture discussing heirarchical design based on material from the Weste textbook.

In the ninth week, event-level simulations of the various circuits generated during the semester were performed using Lsim. A comprehensive yet easy-to-follow tutorial was prepared. Students were asked to compare and contrast Lsim's "switch" and "adept" modes. In adept mode, Lsim delays agreed well (to within 20%) with earlier Accusim simulations and with analytical predictions.

The last exercise, during the tenth week, introduced students to Verilog, high-level synthesis, and use of the 2-µm MDK (MOSIS Design Kit) standard-cell library. Students were asked to enter a Verilog description for a 4-bit counter (74LS163). This design was chosen because students in the class had performed the same exercise using VHDL the previous semester (or in the case of students concurrently enrolled, earlier in the Spring semester) in EE483. Repeating the exercise in Verilog helped the class to compare and contrast the two languages. Most agreed that after learning VHDL, learning Verilog was a snap! While not unanimous, most students prefered Verilog to VHDL. This was in part due to the less verbose syntax and its similarity to the C programming language. Verilog code was compiled using Mentor's "qvlcom". The next step, after simulation of the code in "qvlsim" , was to synthesize the counter. Autologic was used for synthesis while ICstation's Autoplace and AutoRoute were used to place and route the standard cells. The class liked these exercises very much.

IV Design Project

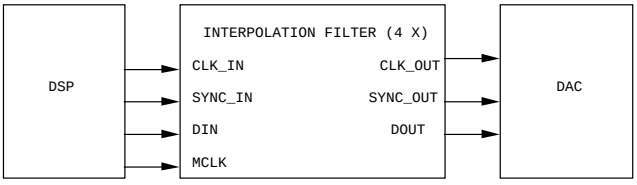
The final project was to design a linear interpolation filter. A lot of thought and time went into crafting just the "right" design problem. It was important that the problem lend itself to a solution which combined the full-custom and standard-cell design methodologies. Moreover, while reasonably complex, it had to completed in a short period of time. The design of a simple linear interpolation filter proved to be an excellent candidate for such a solution. The design specification given to the class is presented below. Students were given 6 weeks from design specification to final layout to complete the project.

FROM: George Engel, President of Technology

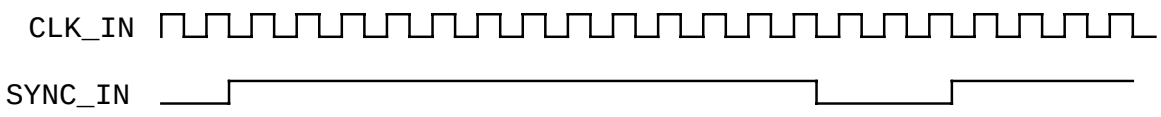
TO: EE484 Design Teams (4 people to a team)

RE: Linear Interpolation Filter

As you might already know, we are currently working on a design for a very important customer, *DSP EXPRESS, Inc.* They currently have a product where a DSP (Digital Signal Processor) sends samples (13-bits) at a rate of 44 kHz to a DAC (Digital-to-Analog Converter). The DAC output is then reconstructed using a complex, high-order continuous-time filter. The company's design engineers have realized that the current DAC is actually capable of sample rates as high as 176 kHz. Operating the DAC at 176 kHz would greatly reduce the complexity of the analog reconstruction filter. They have asked us to design a custom integrated circuit (IC) which can be inserted between the DSP and the DAC as illustrated below. I propose we design a linear interpolation filter where the output sample rate is increased by a factor of of 4 (from 44 to 176 kHz).



The DSP currently sends data to the DAC serially as illustrated below



Serial data (DIN) is valid on each rising edge of CLK_IN whenever SYNC_IN is active (HIGH), and SYNC_IN is guaranteed to be inactive (LOW) for 3 clock cycles before coming active again. SYNC_IN is active for 13 clock cycles (13 bits). For a 44 kHz sample rate, CLK_IN has a frequency of 704 kHz (16 x 44 kHz). A master clock (MCLK) is available for use. MCLK has a frequency of 11.264 MHz (16 x 704 kHz). CLK_IN is derived from and thereby synchronized to MCLK.

Given the inputs described above, you must generate CLK_OUT which should be derived from MCLK and have a frequency of 2.816 MHz (4 x 704 kHz). You should also generate data out (DOUT) and SYNC_OUT. Obviously, SYNC_OUT should also be at 4 times the frequency of SYNC_IN since we must output 4 samples for every one sample we receive from the DSP and like SYNC_IN should be HIGH for 14 clock periods and LOW for a minimum of 2 clock periods before coming HIGH again. With regard to the serial data stream, the least significant bit of the 13-bit word is transmitted first. You should maintain this protocol when the data word is re-transmitted to the DAC.

A 4X linear interpolation filter will allow the DAC to take smaller steps. For example suppose the DSP sends the digital output samples 1000 and then 2000. Your circuit which is between the DSP and the DAC should accept these values and generate the intermediate values 1250, 1500, and 1750. You see you will send 4 samples (1250, 1500, 1750, and 2000) in the same time that the DSP would have originally only sent one sample; namely, the value 2000 to the DAC. It is not a problem if the your circuit introduces a fixed delay i.e. samples arrive a little later at the DAC than they would have if your circuit were not used. Just make sure you don't drop samples.

DSP EXPRESS, Inc. has demanded that we design the smallest circuit possible. At some point they would like to integrate the DSP, the linear interpolator, the DAC, and the analog reconstruction filter on to a single die. It is important that the interpolator circuit be as small as possible. Obviously, the circuit must operate at the prescribed clock rate, but power consumption is not a major concern. You should do the following

- Develop an appropriate algorithm for linear interpolation.
- Explore possible architectures.
- For the selected architecture, develop a block diagram and corresponding RTL statements.
- Simulate the selected architecture using Verilog.
- Determine which modules are candidates for standard cell design and which are best customized.
- Synthesize the modules using Autologic and then use ICstation to AutoPlace and AutoRoute those modules.
- Prepare schematics, simulate, and layout those modules scheduled for customization. Choose appropriate logic structures, circuit design styles, and layout techniques.
- Use ICstation's AutoRoute to assemble all of the modules.
- Simulate the design using Lsim.
- Prepare and deliver an oral presentation documenting/defending your design.

The class was divided into groups of four students each. This fostered a healthy competition in the class and helped students learn how to work in a team environment. The groups began by exploring alternative architectures for the filter. Most concluded that a bit serial approach was optimal given the serial nature of the data transfers, modest data rates, and the requirement that die area be kept minimal.

After selecting an appropriate architecture, each team developed a block diagram and a corresponding RTL (Register Transfer Language) description. Some of this work was performed, in part, during lecture in order to better illustrate the approach. Students were then asked to simulate their design using Verilog. The groups decided how the workload was to be distributed among the members of the team. The instructor supplied students with some of the Verilog source code which was then used, as is, or in a modified form by the various groups. This was necessary because of the short time frame for the project, 6 weeks from design specification to final layout!

Students were then asked to determine which modules were candidates for standard-cell design and which were best customized. All agreed that the data unit and register file were candidates for full-custom layout while the finite state machine (FSM) controller leant itself to a standard cell approach. The data unit only required four leaf cells: a D-register, a full adder (sum and carry function), and a multiplexer. Each student in the team was responsible for the design, physical layout, and simulation of one leaf cell. Group members then assembled the leaf cells into larger modules using techniques they had practiced earlier in lab.

The specification and Verilog coding of the FSM controller was performed as a class exercise, in order to save time. Students were then asked to simulate the controller and correct errors until the entire design simulated correctly. The control unit was then synthesized using Autologic and the standard cells placed and routed in ICstation. Finally, the data and control unit were assembled into a working filter.

During the final exam week, each group was asked to make an oral presentation to the instructor as part of a design review in much the same way engineers working in industry are required to do. All groups successfully completed the physical layout and Lsim simulation of the data unit. Only two of the groups were able to put the control and data units together, and none of the groups had time to successfully simulate the entire linear interpolation filter using Lsim. In the future, we plan to begin the design project at least one week earlier. This should make it possible for the majority of groups to complete the event-level simulation of the filter.

V. Course Evaluation

While it is always difficult to assess the effectiveness of a course,the performance of the students in the class was encouraging. This was reflected by the final grade distribution:

A	9
B	11
C	1
D	0
E	0
I	0

Average grade: 3.3 / 4.0

Moreover, each semester, students are provided with course evaluation forms which they complete in class, collect, and then return to the department's secretary. As part of the course evaluation, students are asked to evaluate the course and the instructor in very specific ways. A summary of the responses to two of the questions is provided below.

- Was this course interesting and stimulating and does it make you want to learn more?

very much	some	not at all
16	4	0

- Please give a final letter grade to the course as a whole, based upon its interest and value to you. For this rating, consider it relative to other Electrical and Computer Engineering courses that you have taken.

A	B	C	D	E
17	3	0	0	0

Finally, several of the students have taken IC-related jobs. Feedback from both students and their employers indicated that the interview process went extremely well for these students. Employers were pleased by the students' knowledge of low-level circuit issues and suprised by their ability to describe designs using Verilog. One student was asked and was able to correctly specify a simple circuit using Verilog during his interview. While feedback is limited at this time, by all accounts the evolution of EE484 appears to be in a direction which is in the best interest of both students and employers.

VI. Summary

This paper demonstrated how we successfully taught everything from MOS transistor theory and CMOS process technology through circuit and logic design, up to and including the design and synthesis of digital systems using a hardware description language in a one-semester long *introductory* course in VLSI design. Carefully planning of the syllabus, appropriate prerequisites, and the use of industry-standard IC design tools were the reasons for the success of the course. It turned out to be possible for students to learn both full-custom and standard-cell design techniques and to engage in a meaningful design project where use of both of these design methodologies was required.

Mentor Graphics IC design tools proved very effective in helping to teach IC design to students. The tools help reinforce and illuminate concepts discussed during lecture. While some of the tools (ICgraph, Lsim, Accusim) merely replaced, albeit with added functionality and ease of use, tools from the Berkeley toolset; other programs (qvlcom, qvlsim, autologic, AutoPlace, AutoRoute, etc.) gave us an opportunity to expose students to Verilog, circuit synthesis, standard cell libraries, and automated place and route techniques not previously available to most of us in the university community. In this way, students are exposed to the latest design methods currently employed throughout the IC industry. Students truly enjoyed and appreciated the opportunity to engage in a "real-world " design with "industrial-grade" CAD tools.

VII. Acknowledgments

The authors would like to gratefully acknowledge the Mentor Graphics Corporation's Higher Education Program for their generous donation of the IC design software to the Electrical and Computer Engineering Department at SIUE. In particular, we would like to thank Julie Schroeder and Barry Carpenter who oversee the Mentor Graphic's Higher Education Program. Without their help and devotion to higher education, the success of this course would not have been possible.

References

1. Weste, N. & Eshragian, K. Principles of CMOS VLSI Design: *A Systems Perspective*. Addison Wesley (1994)
2. Palnitkar,S. Verilog HDL: *A Guide to Digital Design and Synthesis*. SunSoft Press, A Prentice Hall Title (1996)
3. Engel,G & Vedachalam R: A Paper on " Effective Use of Mentor Graphics Tools in an IC Design Course", MUG '98 proceedings.
4. URL: http://www.ee.siue.edu/~mentor/EE484/EE484.html

RAMA K. VEDACHALEM
Rama Vedachalem is a Master's degree candidate in the Department of Electrical and Computer Engineering at SIUE. In addition to serving as a teaching assistant for EE484, he is conducting research aimed at fingerprinting magnetic media with hopes of thwarting counterfeiting of passcards bearing magnetic stripes; for example, credit cards. In particular, he is helping develop a custom, integrated data acquisition unit employing a sigma-delta style converter.

GEORGE L. ENGEL

Since 1993, George L. Engel has served as an Assistant Professor in the Department of Electrical and Computer Engineering, Southern Illinois University at Edwardsville (SIUE), where he teaches courses in computer architecture, electronics, and integrated circuit design. Dr. Engel received a B.A. degree in Physics/Mathematics from Saint Louis University and then his B.S., M.S., and D.Sc. in Electrical Engineering from Washington University in Saint Louis. Dr. Engel is actively involved in joint research with industry in several areas including portable audiometry, and magnetic fingerprinting.