Integration of CAD Tools and Structured Design Principles in an Undergraduate Computer Engineering Curriculum

D.L. Andrews, Mitch Thornton
Department of Computer Systems Engineering
University of Arkansas
dla@engr.uark.edu,mat1@engr.uark.edu

Introduction

The Computer Systems Engineering undergraduate curriculum at the University of Arkansas is incorporating computer aided engineering and design (CAE/CAD) packages into undergraduate courses. The intent of augmenting the curriculum with these packages is to enhance the students theoretical understanding of the material with hands on design and analysis experience. The University of Arkansas is not the first university to recognize the benefits of CAE and CAD packages in the classroom. Many other universities have reported on their efforts of augmenting curricula with these packages [1][2]. The benefits of using these packages in a university setting is also confirmed by the number of new textbooks, and revisions to previously printed textbooks incorporating new exercsizes and problems based on these packages.

Integrating CAD and CAE packages into courses also allows the students to focus their efforts on developing a clean, efficient design instead of spending a large portion of their time engaged in drafting. Although design is emphasized in all digital design and computer architecture classes, we have integrated a junior level class that combines subystem design with the teaching of structured design approaches. The structured design approach is explicitly stated to the students, and represents an intentional attempt to teach top down structured design. Mentor graphics [ref] provides a convenient teaching environment in support of this approach, allowing the students to design, debug and verify small subsystems, and then combine these subsystems into a larger system.

General Advantages

The main advantage of using CAD/CAE packages in the undergraduate curriculum is the reinforcement of student understanding of theoretical principles based on practical analysis. Another equally important advantage is the preparation for designing and developing complex systems, much like students will see in industry. With the incorporation of these tools, instructors can assign fairly complex designs that otherwise, would be unrealistic. The schematic entry capabilities and libraries of built in components normally associated with these packages allow the rapid prototyping of complex designs. This is a key advantage that helps students apply the theoretical principles learned in the classroom to the real world problems associated with following a design cycle through completion.

Student response concerning the use of these packages are generally favorable. One interesting response received from the students is an increased interest in the subject material due to the use of these packages. Student interest is heightened by the prospect of actually building a design that without these packages, they would only have studied theoretically. Students also are interested in using these packages during their undergraduate education as an enhancement to their professional development. Students who would have otherwise received limited experience in using design packages in their particular specialty gain expertise and familiarity with the CAE/CAD packages that will benefit them during their engineering careers.

These packages also provide augmented material for instructors to present in the classroom. Simulation results can be used effectively to reinforce the characteristics of circuits and show the similarities and differences that exist between the actual and theoretical circuit operations.

General Disadvantages

Three disadvantages of using these packages are the extra work required by students (and instructors), the maintenance and operation of these packages on an accessible computer system, and assuring that the

packages are inserted in the baseline curriculum as part of the required course material. Assigning the use of these packages on homework assignments generally implies 24 hour student access to computer facilities. These packages tend to require large amounts of memory, and can exhibit unusually slow response times as the system load increases. In one class at the University of Arkansas, approximately 80% of the students in a class of 35 waited until early evening the night before a particularly long and tedious two week assignment was due to log into the network and complete their assignments. Some students were unable to obtain licenses to run the software, others experienced network difficulties due to the large transient load of students, and still others left frustrated by the very slow response times observed.

The second disadvantage is the additional time requirements for both students and instructors. The additional time required for students to run the simulations must be acknowledged and accounted for when assigning homework. The additional time required for the instructor includes not only the time it takes to develop and run through each simulation, but also the time required to generate handouts detailing the steps required to log into the system, use the particular package, and obtain results. Tutorials for most of the packages in use at the University of Arkansas have been written by instructors and graduate students to compensate for the general lack of textbooks available on the use of these packages.

The third disadvantage is defining a baseline in the curriculum for teaching these software packages. Different instructors rotate through undergraduate teaching responsibilities, and some are unwilling to incorporate these packages into their classes. This creates students with different knowledge levels in follow on classes that assume all students are familiar with the package. This can result in frustration for the students who did not have prior exposure to this material if assignments are made based on previous familiarity with the package.

Design Course Sequence

Figure 1 shows a flowchart of required courses in the computer systems engineering department that utilize digital circuit CAD tools. Table 1 lists the packages and content that are covered using these packages in these courses. Approximately 75% of the laboratory experiments in the introductory digital design class (CSEG 2513) require student construction of circuits. This decreases to 50% in the second digital design class CSEG 2523. The subsystems organization (CSEG 3533) and computer architecture (CSEG 4983) utilize CAD tools almost exclusively. The capstone senior design course (CSEG 457v) allows more flexibility for students to choose the tools they feel best suited for their projects. Typically, students will perform design entry and preliminary simulations using the CAD tools, followed by the construction of a prototype in the laboratory.

Design Entry Using CAD Systems

Design entry has undergone a fundamental paradigm shift in the past decade. Previously, the predominate method for design entry involved the use of graphical user interfaces and schematic entry. The most common method in use to day is the use of hardware description languages (HDLs). These languages differ from traditional imperative languages based on the notion of sequential program counters. Rather, HDLs are utilized as circuit descriptions for discrete-event based simulators. Therefore, the HDL description of the circuit can serve as a medium for specification documentation, simulation description, and as input for automated synthesis tools. By emphasizing the concepts of discrete event-based simulation and design specification, introduction to the use of HDLs may be presented in the classroom in the context of performing engineering design functions rather than as the presentation of the syntax of yet another programming language.

The two predominate HDLs in use today are VHDL and Verilog, both of which are standardized by the IEEE. These languages have the desirable property of allowing the designer to specify purely functional behavior as well as a structural description. This is one of the primary advantages of using HDLs as compared to schematic entry. Since the HDL is utilized as a description to a discrete-event driven simulation engine, a functional specification of a circuit may actually be simulated before the tedious process of developing a structural representation occurs.

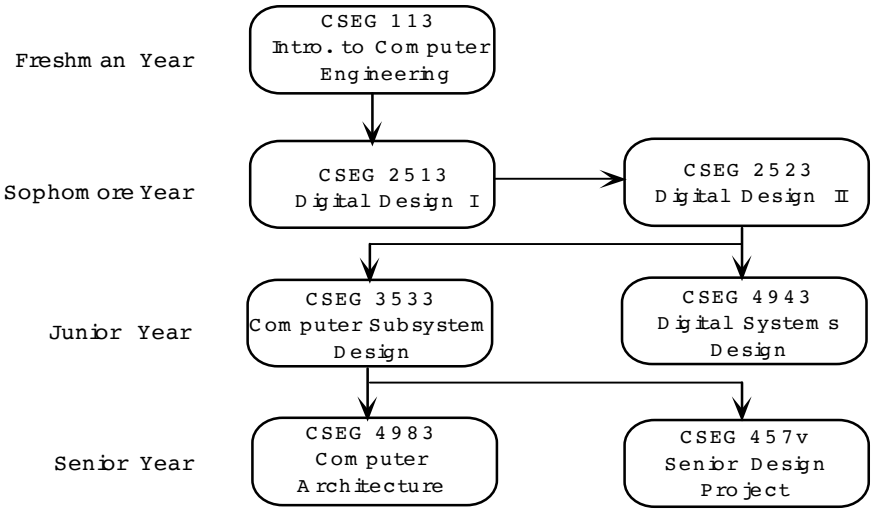

Figure 1. Courses That Utilize CAD Systems

Course Level	Course Content	Packages
CSEG 1113	Introduction to Design Process	Simple FPGA excersize using Altera/Xilinx FPGA Tools
CSEG 2513	Combinational and Sequential Circuit Design	Schematic Capture and Simulation with Mentor
CSEG 2523	Design with LSI, PLD devices and Introduction to FPGA	PALASM, Xilinx, Altera FPGA tools, VHDL using Mentor
CSEG 3533	Microprocessor, LSI Support Chips and Interfacing	Schematic Capture, Simulation and VHDL with Mentor Graphics
CSEG 4943	System Design With FPGAs	Mentor Interface to Xilinx XACT
CSEG 4983	Computer Organization and Design	Schematic Capture, Simulation using Mentor Graphics
CSEG 457V	Capstone Senior Design Project	Student Choice

Table 1. CAD/CAE Packages In Use At University of Arkansas

Many universities now utilize HDLs in course-work, in the past 5 years several new digital design texts have emerged that focus on the use of HDLs. A common approach to HDL instruction is to introduce the language syntax with supporting design examples. We first introduce HDL usage in the CSEG 2523 course as shown in Table 2. Our approach differs from other universities in the following two aspects:

1. A simple text based design entry language for PLDs is presented before more complex HDLs such as VHDL and Verilog are discussed.

2. Instead of discussing HDL syntax first with supporting examples, the concepts of discrete simulation algorithms and techniques are presented with examples written in HDL.

In the CSEG 2523 course, students are exposed to the architecture of simple programmable PLA, PAL, and PROM devices as a first introduction to programmable logic devices (PLDs). After the discussion of the internal architecture, problems are assigned requiring the students to generate fuse map information manually for simple designs as motivation for the use of a basic PLD based design system such as PALASM. This is a convenient means to introduce textual circuit description since mostly of the early PLD based languages were merely formatted Boolean and state equations; concepts that the students are

already familiar and comfortable with. Again, it is fairly easy for the students to generate the test vector files and simulation directives since they used simulators in conjunction with the schematic entry tools that were used in the previous prerequisite course.

The second aspect of our approach involves describing simulation algorithms and techniques instead of presenting HDL syntax an examples. This approach seems to better retain the interest of the students since going over language syntax can be a very dry experience and require more mental memory effort than conceptual understanding. By surveying the techniques of discrete event circuit simulation, concepts regarding the analysis and physical characteristics are reinforced with a secondary benefit of introducing the use of a particular HDL.

Integration of Design Methodology

The principles of a structured design methodology are formalized in the junior level computer subsystem design course. This is the first level where the students are exposed to the design and analysis of moderately complex systems. The course is presented analogous to a structured top down complete design cycle. The course uses a series of laboratories, first starting with a requirements assessment, a top level design, and than the detailed design of the various subsystems of a fictitious single board computer flight control system. Each laboratory combines subsystems designed and tested in previous laboratories, in a structured top down design fashion.

Models of the chips used for this course are obtained from our SmartModels [ref] library, and various Mentor libraries. Models are used for the Intel 8086, 8284A clock generator, RAM, ROM, 8255 Programmable Interface Chip, 8251 UART, and other I/O devices. These models include a complete behavioral specification for each chip. Simulation of these behavioral models allow complete system simulations to be run. The behavioral models for the memory chips include the capability of specifying small machine code programs for execution during the simulation. The behavioral model for the 8086 actually executes these programs during simulation, generating the correct handshake signals for data transfers, interrupt signals for external interrupts, etc.

The first laboratory focuses on a simple clocking and reset circuit for the 8086 microprocessor. This circuit is simple enough to allow the students to gain the confidence of entering a completely new design, taking the design from schematic entry through simulation. In the next laboratory, the students build up a memory system comprised of both RAM and ROM. Mentor allows the students to develop and implement the decode circuitry for the memory system, based on the actual timing requirements of the 8086. The simulations performed on their decode circuits and memories allows the students to blend the theoretical aspects of memory organization with the practical aspects of designing a fairly complex realizable system in accordance with actual system timing requirements. Subsequent laboratories continue to build on these laboratories, adding complexity to the design in a structured Socratic fashion.

The use of Mentor in this course reinforces the students understanding of the complexity and timing effects of propagation delays through decode circuitry. The schematic entry portion of Mentor using predefined library models allows the rapid development of a complete system comprised of fairly complex circuitry. Students are able to focus their efforts on developing a clean, efficient design instead of spending a large portion of their time engaged in drafting. The simulation portion of Mentor allows the students to quickly see the real effects of their design. This environment provides the students the ability to iterate their designs, much as is done in industry.

Conclusions

An approach for the integration of modern CAD tools and structured design approaches into a computer engineering curriculum was described. By closely coordinating the introduction of CAD tool usage throughout the course sequence, unnecessary repetition of instruction regarding the mechanics of tool usage was described. Furthermore, we feel that emphasizing design methodology using the tools rather than focusing solely on their use allows students to learn how to use these tools without losing valuable engineering design instruction time. Finally, by introducing the use of HDLs by teaching simulation,

specification generation, and automated synthesis techniques allows the instruction to retain more design content versus the more common approach of simply describing the language syntax and analyzing a number of design examples.

Bibliography

[1] Haggard, Roger L., *"Classroom Experiences and Student Attitudes toward Electronic Design Automation*", Proceedings of the 25th Southeastern Symposium on System Theory, Los Alamitos Ca., IEEE Computer Society Press, 1993, pp. 411-415

[2] Andrews, D., A. Azemi, S. Charlton, and E. Yaz, *" Computer Simulation in Electrical Engineering Education*" Proceedings of the American Society of Engineering Education Conference , Baton Rouge, LA. March 24-25, 1994

[3] Nixon, M.S., "*On a Programmable Approach to Introducing Digital Design*", IEEE Trans. Educ., vol. 40, pp. 195-206, 1997