Developing Project Based Course Using a Hardware Description Language

Ghassan E. Shahine[1], Nizar Al-Holou[2]

[1] *Falcons Consulting, Ann Arbor, MI: gshahine@yahoo.com*
[2] *University of Detroit Mercy, Detroit, MI: alholoun@udmercy.edu*

ABSTRACT

This paper describes a course about very high speed integrated circuit hardware description language (VHDL) that was designed and taught to seniors and graduate students at the University of Detroit Mercy. The course motivation is to bridge the gap that students encounter when getting involved in the design of digital circuitry in the workplace. This course prepares them adequately to learn the hardware description language VHDL and be able to use state-of-the-art tools to design, simulate and build real-world circuits and systems. This saves them around six months in effective work time, and provides them with maturity to solve real world problems. The course uses development boards based on field programmable gate arrays (FPGAs), which enable students to perform rapid prototyping and verification of their projects. When application specific integrated circuits (ASICs) are the ultimate goal, the replacement of any code that includes an FPGA primitive instantiation becomes necessary. On the other hand, current FPGAs speed of 500 MHz and high density at a continually lower cost, make it less compelling to use ASICs in many situations.

1. INTRODUCTION

Students have different learning styles and may be classified as active, visual, inductive, sensory, and sequential (Felder et al. 1993; Felder 1988). Many studies have shown that traditional classroom instruction is not necessarily the best approach to teaching college students (McKeachie, 1986; Polio, 1984; Rigden et al., 1993) and therefore, an innovative new teaching pedagogy is needed. As a result, educational institutions have started new approaches to enhance student learning (Brodersen et al, 1993; Oaklay II, 1996; Swafford et al, 1996; Yoder, 1993). This course was designed to equip students with the ability to design and specify real-world digital systems using the VHDL hardware description language at the behavioral, register-transfer level (RTL), and structural levels. This means that students should be able to write code that represents a specific design, and then simulate the design. After successful simulation; students synthesize, implement, and download the bit-stream file of their design to development boards and verify its proper functioning. Lecture notes are provided to the students. Students are given assignments, many of which require simulations, as well as board verification. Student groups of two are required to come up with a project idea and then realize it. The process involves an abstract, progress report, final report, simulations, verification, and class presentation. Section two of the paper elaborates the course topics where we emphasize language basics and concepts. Homework and laboratory assignments are introduced in section three, and here we stress real-world applications. Section four covers the projects that have been accomplished. Our conclusions and new ideas are presented in section five, and acknowledgements in section 6.

2. COURSE TOPICS

In this course, students start by learning the basics of FPGAs, the configurable logic block (CLB), the input output block (IOB), and the look-up table (LUT), as well as mastering the design tool that we use which is the integrated software environment (ISE) of Xilinx.

VHDL basics are taught, including the major and most used constructs of the VHDL language. Several textbooks are suggested for reference, and we selected (Navabi, 1998) as the assigned textbook. Students learn the main thrusts of VHDL, which are code execution or simulation, and synthesis.
In simulation, students learn how VHDL provides users with the ability to simulate abstract behavioral models that may not be synthesizable. Such simulation provides many benefits, such as:
a) Proof of concept.
b) Proper documentation of any design for future revision.
c) Abstract models called test-benches that allow test vectors to be applied to a VHDL unit. These test-benches may be applied to the purely behavioral models as well as to their synthesizable counterparts. Behavioral models are simpler to code and hence provide us with the expected outputs of the design. Consequently, we can compare these expected outputs with those of the synthesizable design to verify that the design is working as expected.

Board-level verification involves synthesizing, implementing, and then downloading designs to a development board (Digilent, 2005) that has a Xilinx Spartan3 FPGA. A design could be a combinational circuit such as a sprinkler system, a simple calculator, or a registered circuit that may be modeled as a finite state machine (FSM), such as a vending machine, or a traffic light controller. Students also learn how to build a proper keyboard as well as a VGA (Video Graphic Adapter) controller. Students finish the course by working on projects. The ability to simulate the VHDL code as well as the ability to download designs to the development board and verify proper functioning of the targeted application provides students with unparalleled troubleshooting capability as well as self-confidence in their own ability to achieve real designs.

Lectures cover 1) the basic design units in VHDL, 2) VHDL data types, 3) the equivalence between a piece of logic and a process, 4) the difference between concurrent and sequential statements, 5) hierarchical design, 6) coding of combinational as well as registered circuits, 7) test-benches, 8) control loop mechanisms, 9) the concepts of delta cycles, transactions, events, and event scheduling, 10) how and when to use constants and generics, 11) the differences between signals and variables, 12) functions and procedures as well as the differences between them, 13) the importance of a process sensitivity list, especially for proper simulation, and 14) finite state machine coding including Mealy and Moore machines. Advanced topics are covered, and issues pertinent to the readability, scalability, and reusability of VHDL code are emphasized.

3. COORDINATION OF HOMEWORK AND LABORATORY ASSIGNMENTS

Homework and laboratory assignments were synchronized with the lectures. Some assignments were extensions of the examples used in the lectures. Students start by learning how to use the ISE tool and its different capabilities. This includes design entry and editing, syntax checking, simulation, synthesis, RTL viewing, implementation, and bit-stream file generation. In the early home work assignments students design multiplexers, flip flops, shifters, counters, adders, serial adders, etc. In subsequent assignments students learn how to use the generate statement, and they build hierarchical designs. To achieve their goal, students have to check the syntax of their code and simulate the design; hence they learn about test-benches early on. They also download the design to the development board, which enables them to learn about the different processes applied to the code. They learn about the synthesis process, which transforms the VHDL code to Boolean equations, i.e., the *netlist*. They also learn about the implementation process which merges the different netlists of the design, maps the Boolean equations to look-up tables and flip flops, and places and routes the design into the FPGA. In addition, they learn about the user-constraints file (UCF), which assigns the input and output pins to the FPGA, and applies timing constraints to the placement and routing process. Students connect the design inputs to the switches and push-buttons, and the outputs to the board's light emitting diodes (LEDs) and seven segment displays. Students also spend time designing real-world combinational and sequential devices. They start with the design of a sprinkler system, which takes into consideration many parameters including time of day, air humidity, soil dryness, water level etc. In another laboratory exercise students apply their knowledge of finite-state machines to implement a vending machine which (1) keeps track of the amount of money that has been deposited, (2) recognizes the product selection, (3) checks that enough money was deposited before dispensing the selected item, and (4) returns any excess money after a transaction.

4. PROJECTS

Students are given some project ideas, and they decide which project they want to work on. They may also modify the project idea or come up with an idea of their own. It is probably the most exciting and intense learning experience throughout the course. Students spend many hours researching, writing, and modifying VHDL code. They write test-benches to simulate their designs, then download their implemented design's bit stream file to the development board several times a day, until the design works adequately. Below are last year's projects that the students attempted with great success:

4.1 Stepper Motor Control via an RS232 Link

In this project, students used two development boards. An RS232 transmit circuit is implemented on one board, and an RS232 receive circuit is implemented on the other board. The number of steps is set via switches on one board and is transmitted via the RS232 link connecting the 2 boards. The second board is connected to a stepper motor as well as to a VGA display port. The stepper motor rotates according to the specified number of steps, and the VGA displays this information graphically. In this project students gain the knowledge of 1) how to convert from parallel to serial data and vice versa, 2) how to implement the RS232 serial link, 3) how to display data graphically on a VGA display, and 4) how to output data to a stepper motor.

4.2 Serial Peripheral Interface (SPI) Controller

In this project students use two development boards. A keyboard is connected to one of them, and a VGA display to the other. Both boards are connected via 4 wires. A SPI transmit controller is implemented on one board, and a SPI receive controller is implemented on the other. Data inputted by the keyboard to the transmit development board is sent via the SPI link to the receive development board. Received data is displayed on the VGA display, as well as the seven-segment display of the second board. In this project the student gains the knowledge of 1) how to decode the scan keys of the keyboard, 2) how to implement the SPI interface controller on two boards, 3) how to save an image of the display data in FPGA memory efficiently, and 4) how to build a character generator for the display characters.

4.3 Text Editor

In this project students use a development board, a keyboard, and a VGA display. The system operates as a simple text editor with basic editing capabilities, such as backspace, character delete, line delete, tab, and line wraparound. FPGA memory is used to store the text. In this project students learn to 1) design a keyboard controller and decode the scan keys, 2) build a VGA display controller, 3) gain some insight into the different possibilities of text editing, and 4) modify the display and the memory simultaneously to maintain data integrity.

4.3 Basic Animation

Here students started with building a simple image of a rabbit and displaying it on the screen. Then they used keyboard keys to give the impression that the rabbit can respond to keystrokes. The mode of operation may be changed to show a succession of images, to give the impression that the rabbit is running. The speed of motion may be increased or decreased using a counter whose speed is controlled by the keyboard. In this project students learn 1) how to design a keyboard controller and decode the scan keys, 2) how to build a VGA display controller, and 3) basic issues in animation.

4.4 Temperature Measurement

In this project students connect a temperature sensor to the development board. The sensor is a MAX6666 that converts the ambient temperature into a ratio-metric PWM output. The temperature information is contained in the duty cycle of the output square wave. The PWM signal generated by the sensor will be read by the timer-counter function of the FPGA. The extracted temperature information is then graphically displayed on a VGA monitor, thus indicating the room temperature. In this project students learn 1) how to build a VGA display controller, 2) how to display on-line data graphically on the display, 3) how to interface with simple sensors, and 4) how to decode the data appropriately.

4.5 Microprocessor/ Microcontroller Design

In this project students may choose an existing microcontroller or a microprocessor, such as the 8251 priority-interrupt controller (PIC), or the Xilinx picoblaze 8-bit processor, and then re-implement their chosen device in VHDL. They also try to implement an adequate instruction set that would run a simple program that the instructor may suggest to check the proper operation of the processor. The simulator has been extremely important to check the proper sequencing through the different instructions before hardware verification is attempted. In this project students learn 1) how to build a simple controller, and 2) many details about the simulation tool.

5. CONCLUSIONS

Starting from a basic logic design course, this course takes the student to a high level of achievement and competence. This course enables students to design with the language and the tools of the workplace and teaches them the practical techniques and tools to realize real-world designs. We have found that students are extremely excited about these projects, and are willing to put in the time needed to accomplish the design goals. In the coming offering of the course we will spend more time discussing basic blocks such as the keyboard, the VGA controllers, the character generator, and the UART interface. These topics will be elaborated in the lectures during the second half of the course, and will be included in their homework and laboratory exercises. This would allow students to spend more time exploring more complex projects. We also intend to introduce more Xilinx tools, as the System Generator (SYSGEN), that enable the user to build designs at an abstract level using Simulink blocksets, and the Embedded Development Kit (EDK), that enable designers to work efficiently with the microblaze processor softcore, as well as the PowerPC hardcore that is available on the Xilinx Virtex 2P and Virtex 4 F series of

FPGAs These tools are rapidly becoming standard tools in many research and development laboratories.

6. ACKNOWLEDGEMENT

We wish to acknowledge the continuing support of the Xilinx University Program (XUP) that includes both the development boards and the software tools. We also wish to acknowledge the suggestions and help of Professor Bernard Galler.

REFERENCES

1) J. Brodersen, *et al*, "The ELF Project: Creating the Future Laboratory," Vanderbilt University, the IEEE FIE 1993 Proceedings, pp. 227-279.
2) Felder, R.M., "Reaching the Second Tier-Learning and Teaching in College Science Education," Journal College Science Teaching, Vol. 23, no. 5, April 1993, pp. 286-290.
3) Felder, R.M., and L.K. Silverman, "Learning and Teaching Styles In Engineering Education," Journal of Engineering Education, April 1988, pp. 574-581.
4) McKeachie, W. Teaching tips: A Guide for the beginning College Teacher, Heath, 1986.
5) J.S. Rigden, D.F. Holcomb, and R. Di Stefano, " The Introductory Physics Project," Physics Today, Vol. 46, April 1993, pp. 32-37.
6) Pollio, H. R. " What Students think About and Do in College Lecture Classes," Teaching-Learning Issues, No. 53, University of Tennessee, Learning Research Center, 1984.
7) Burks Oaklay II, " Helping Faculty Develop New Asynchronous Learning Enviroments," 1996 FIE Conference Proceedings.
8) Michael L. Swafford, C. Graham, D. J. Brown and T. N. Trick, "Mallard: Asynchronous Learning in Two Engineering Courses," 1996 FIE Conference Proceedings.
9) Mark A. Yoder, "Sparking Life in a Circuits Classroom," Rose-Hulman Institute of Technology, the IEEE FIE 1993 Proceedings, pp. 128-130.
10) Mark A. Yoder, "Sparking Life in a Circuits Classroom," Rose-Hulman Institute of Technology, the IEEE FIE 1993 Proceedings, pp. 128-130.
11) Zainalabedin Navabi, "VHDL Analysis and Modeling of Digital Systems," Second Edition, McGrawHill, 1998.
12) C. H. Roth, "Digital Systems Design Using VHDL," PWS Publishing Company, 1998.
13) L. Dong, and M. Z. Atashbar,"An FPGA Experience in ASIC Design, Proceedings of the 2005 ASEE North Central Conference, 2005.
14) Digilent, "Spartan-3 Starter Kit Board User Guide, 2005.