A VHDL COURSE FOR ELECTRONICS ENGINEERING TECHNOLOGY

Robert W. Nowlin and Raji Sundararajan
Electronics and Computer Engineering Technology
Arizona State University East
Mesa, AZ 85206

ABSTRACT

Hardware Description Languages, VHDL and Verilog HDL, are being used extensively in industry to describe digital systems from very abstract levels down to gate levels and are in greater and greater use every day. Students who are trained in either of these languages have an advantage in the job market. The Electronics and Computer Engineering Technology Department at Arizona State University has offered a course in VHDL to both undergraduate and graduate students for the past three years. Students learn the basics of the language in addition to many advanced techniques such as generate statements, guarded signals and block statements. An attractive serendipity of a course in VHDL is that students are forced to learn (or re-learn) more accurately how hardware elements operate because they can cannot model the hardware operation unless they understand it. Students learn to describe hardware operations using several VHDL modeling techniques such as behavioral and structural. The techniques of VHDL are re-enforced through a series of mini-projects and a major final project in which students learn to apply their VHDL knowledge on a commercial grade VHDL simulator. Another benefit that this course has engendered is that several of the graduate students have used their knowledge of the language to incorporate it as either a major or supplemental portion of their masters' projects.

INTRODUCTION

The microprocessor and with it the PC has invaded every business and many homes. Many designs today are incorporating embedded processors or their first cousins, the microcontroller. Even the old standby analog world is becoming more and more digital. The pressure on designers in industry today is to design products and bring them to market at an ever accelerating pace. Designs that lag behind, even though sometimes technically better, never see the market. Many software packages have been developed to help designers achieve and produce their designs more efficiently. One of these classes of software packages is the hardware description languages or HDLs.

VHDL, a hardware description language for Very High Speed Integrated Circuits (VHSIC) was developed in the early 80's to help the government standardize methods of describing hardware designs. This later became IEEE standard 1076 [1] and was modified and updated in 1993. Most digital designs today use the hardware description techniques provided

by VHDL or VERILOG. Students who possess a working knowledge of VHDL or VERILOG have a real advantage at job search time over students who have not been exposed to these concepts [2].

The Electronics and Computer Engineering Technology Department at Arizona State University has been teaching a senior level undergraduate and graduate course in VHDL since Spring 1993. This course integrates the students' knowledge of high level languages, digital design concepts, and microprocessors. Students are challenged by the course because they are required to actually understand how simple devices such as flip-flops work in order to design a good digital model using the techniques of VHDL. Students are given the basics of the VHDL language and are required to supplement their knowledge by modeling various well-designed course assignments. They are also required to produce a project that is completed with a team of two or three other students. Two commercially available VHDL software packages are available for students to use.

COURSE DESCRIPTION

Table 1 outlines the major topics covered in this course. The basic constructs of VHDL, such as Entity declarations, Architectures, Configurations, and Packages, are introduced and their relationship described. The basic concepts of modeling using Behavioral, Dataflow, and Structural models are also quickly introduced. This is done in an effort to give the students an overall appreciation for concepts and techniques so that they can get on the computer and begin to apply these ideas as quickly as possible. Since their knowledge is quite limited initially and their understanding has not had time to mature by the time the first assignment is made, the basic code is provided to them. The real task of the first assignment is to learn the software: how to enter the VHDL code, how to compile it and how to run simulations. Two software packages are available for student use, IKOS Voyager [3] and VIEWlogic Workview [4]. IKOS Voyager runs on a UNIX platform but Workview runs on a PC.

After the basic concepts have been introduced, each is explored in greater detail as the course develops. The first basic concept explored in greater detail is that of *basic language elements*. Identifiers and the data objects of constants, variables, and signals and their declarations are discussed. Next data types and subtypes along with the scalar types of enumeration, integer, floating point, and physical are discussed. The concept of the VHDL operators is also introduced.

Several additional specifics of entity declarations are next discussed in greater detail but the full discussion of entity declarations is saved for still later. The behavioral architecture model is also expanded upon at this time in the course. This includes a discussion of the process statement and the sequential statements: variable and signal assignment, wait, if, case, null, loop, exit, next, and assertion statements. The concept of the delta delay is introduced as are the ideas of signal drivers, and the various signal assignment models.

Table 1: Course Outline

- Introduction and Tutorial
- Basic Language Elements
- Behavioral Modeling

- Dataflow Modeling
- Structural Modeling
- Generics
- Configurations
- Subprograms and Overloading
- Packages
- Libraries
- Advanced Features

Although the dataflow model is not emphasized, it is introduced at this time in the course. This brings up the concepts of concurrent and sequential signal assignments. The concept of delta delays is discussed in greater detail and the idea of a signal having multiple drivers. Block statements are also discussed in greater detail at this time.

Next the most concrete, lowest level of modeling, that is, structural, is discussed in detail. To be able to model at this level, the student must know what components, e.g., flip-flops and gates, make up the design. Component declarations and instantiations are illustrated with numerous examples. Example 1 shows a component declaration and instantiation for a simple 2-input NAND gate.

Example 1: Component Declaration & Instantiation
for 2-Input NAND Gate

```
component NAND2  -- Component declaration
        port(A, B: in std_logic; C: out std_logic);
end component;

-- Component instantiation:
N1: NAND2 port map(S1, S2, S3);
```

The next topics covered in the course are the generic statements, and the whys and hows of configuration specifications and configuration declarations.

The concepts of subprograms in VHDL are next discussed. The differences between functions and procedures and their uses are illustrated through examples of each. Subprogram and operator overloading are also discussed.

Finally, package declarations, package bodies, libraries are discussed. Many advanced features are also covered. These advanced features include generate statements, aliases, qualified expressions, guarded signals, and attributes.

Throughout the course, digital systems models and modeling examples are given and discussed. Example 2 illustrates a simple 4*1 multiplexer modeled in behavioral style.

Example 2: 4*1 Multiplexer Model

```
entity MUX is
  port(A, B, C, D: in std_logic;
```

```
        CTRL: in std_logic_vector(1 downto 0);
        Z: out std_logic);
end MUX;

architecture MUX_BEHAVIOR of MUX is
        constant MUX_DELAY: TIME := 10ns;
begin
PMUX: process(A, B, C, D, CTRL)
        variable TEMP: std_logic;
begin
        case CTRL is
                when "00" => TEMP:= A;
                when "01" => TEMP:= B;
                when "10" => TEMP:= C;
                when "11" => TEMP:= D;
                when others => TEMP:= Z;
        end case;
        Z <= TEMP after MUX_DELAY;
end process PMUX;
end MUX_BEHAVIOR;
```

COURSE ASSIGNMENTS

To complement the lecture and enhance the student's learning experience, a lab (or assignment) manual has been developed. The students are also required to select a more complicated system than required by the labs to model as a final project (There is not a lab associated with the course. These "labs" are really just assignments, but are being referred to as labs.). The labs that have been developed are listed in Table 2 and some of the projects that the students have modeled are given in Table 3.

Table 2: Laboratory (Course) Assignments

Lab 1: Registers and Combinatorial Logic

Code for an entity and architecture pair using the structural style of coding is developed for an 8-bit register with read, write and reset logic. Library use clauses are introduced. The students are also required to develop the code for another architecture using Register Transfer Language.

Lab 2: Testing the Reg8 Design

The students are required to create a test bench and a free running clock process with a period of 50ns. Using a Configuration, the students create a library to contain their files, compile and simulate the Reg8 coded in Lab 1.

Lab 3: Counters and Math Operators

In this assignment, the students design and encode a 4-bit synchronous up/down counter. They simulate the design using their own stimuli to make sure they cover all count conditions. The students are also introduced to several built-in functions that convert vector types to integers and vice versa.

Lab 4: Address Decoder

In this laboratory assignment, students create an entity and architecture pair that will decode an 8-bit address bus and generate six strobes based on the address value. Again, they compile, simulate and verify the operation of their designs and code.

Lab 5: Shift Register

In this lab the students code an 8-bit shift register, converting a serial stream of bits into 8-bit parallel data.

Lab 6: Moore State Machine

For this lab, the students code a state machine that acts like a car wash. The user inserts a quarter into the machine and selects the type of spray to wash the car. A timer runs for 15 minutes and stops the water spray when the timer finishes. The user can select different wash types while the timer is running.

Lab 7: Mealy State Machine

This assignment is the same as the previous lab; however, the code for the state machine is based on the Mealy model rather than the Moore model.

Lab 8: Preloadable Counter

The 4-bit up/down counter from Lab 3 is used in this lab but a preload condition is added.

Table 3: Sample Projects

- Phase Shift Key (PSK) Modem
- DMA Controller
- Morphological Filter
- Ping-Pong Machine
- 8085 Microprocessor
- ALU
- Memory Refresh Controller
- Programmable Traffic Light Controller
- DLX RISC Machine
- SRAM with Built-in Test Facility
- 10 Bit Asynchronous Serial Data Transmitter
- 4-Bit Bi-Directional Shift Register (74194)
- 256 Bit SRAM
- I486 Processor Bus Modeling & Simulation
- FIFO Data-Rate Buffer
- I8251 USART

The first three times this course was taught two projects were required because the lab manual had not been developed. The first project was used as an introduction to code writing and as a vehicle for learning the software. The second project was expected to require considerably more code and time to develop since the students no longer had to learn how to use the software. The projects were developed using teams of from two to five members. The

team members divide the work into manageable sections with each member assigned a section. The code for each section is then developed by the team member responsible for that section. Once all sections of the code are working independently, the team assembles all the code together and makes it work as a single entity. Reports are written which detail the design process, the code development, and which give the results of the simulation usually with elaborate wave forms. The quality of the reports surprised the instructor. They are generally very well written, some were as many as 100 or more pages in length, and many were bound with spiral binders.

Now that the lab (assignment) manual has been developed, several of the labs are done instead of the first project.

STUDENT PERFORMANCE

The first couple of times this course was offered it was immensely popular; many students were denied entry mainly due to classroom size and the computer load it generated. It was filled with students from the Electrical Engineering and Computer Science Departments as well as student from our own department. Student reaction and performance have generally been very good. Students in the course were asked to rate the quality of the textbook and supplemental material, with 47% responding “very good," 43% responding “good," and 10% responding “fair.” When the students were surveyed about the value of the homework in support of course topics, 40% rated the homework “very good,” 48% rated it “good,” and 6% each responded “fair,” or “poor.” Students also rated the value of the laboratory experiments and projects, with 57% responding “very good,” 38% responding “good,” and 5% responding “fair.” When quizzed about the reasonableness of exams and quizzes in covering course material, 56% of the students rated them as “very good,” 37% rated them “good,” and only 7% rated them as only “fair.”

The instructor gave weekly pop quizzes and when asked about the value of this practice, the overwhelming majority of the students were in favor of its continuation. They gave comments such as, “It kept us on our toes” and “It forces us to keep up in class” (This was a very informal oral survey taken in class.). The students were also asked to rate the overall quality of the course and instruction. Thirty-six percent (36%) said the quality was “excellent,” 51% said it was “very good,” and 13% said it was “good.” Even though this class was usually offered late afternoon (4:40p.m.-5:55p.m.), the students were generally attentive in class and worked very hard to learn the various software packages and to complete the lab assignments and projects.

CONCLUSIONS

The hardware description languages and their use in digital design are gaining more and more ascendancy. The use of VHDL and VERILOG in producing designs speeds their completion and insures a higher quality product. Even though many designs are still being done through the schematic entry technique, students and others who know the modern techniques of hardware description languages have a distinct advantage [5]. Many of the students who have taken the HDL course taught by the Electronics and Computer Technology Department at Arizona State University East have used it as a stepping stone to employment.

Many employers are very eager to interview these students once they know that they have had a course such as this one.

REFERENCES

[1] J. Bhasker, A VHDL Primer, Revised Edition, Prentice Hall PTR, Englewood Cliffs, NJ 07632, 1995 (ISBN 0-13-181447-8).

[2] Nowlin, R. W. Private conversations with students - 1994, 1995, and 1996.

[3] Voyager Series User's Guide - Vols. I - IV, IKOS Systems, Inc., Cupertino, CA, 1994.

[4] Workview PLUS on Windows - Vols. 1 - 6, VIEWlogic Systems, Inc., Marlboro, MA, 1993.

[5] O'Dell, J., "We Ask You", *Graduating Engineer*, March, 1996, p.18.

BIOGRAPHICAL SKETCH

ROBERT W. NOWLIN is the Chair of and an Associate Professor in the Department of Electronics & Computer Engineering Technology in the College of Technology and Applied Sciences at Arizona State University East. He received a BSEE from the University of Washington in 1963, an MSEE from San Diego State University in 1969 and the PhDEE from Texas Tech University in 1975. He taught at several other universities before coming to ASU and has extensive industrial experience. His current research interests are microcontrollers and VHDL. He is a member of ASEE and a senior member of IEEE.

RAJI SUNDARARAJAN is an Assistant professor in the Department of Electronics & Computer Engineering Technology in the College of Technology and Applied Sciences at Arizona State University East. She received a BSEE from the University of Madras, India in 1981, an MSEE from Indian Institute of Science in 1988 and the PhDEE from Arizona State University in 1993. Her teaching interests are electrical power and digital systems and research interests are high voltage engineering, insulation and modeling. She is a member of ASEE and IEEE.