

# ECE 443

## Hardware Design with VHDL

### Overview

**Course Catalog Description:** This course covers the principles of programmable logic devices (PLDs) and the synthesis and simulation features of the VHDL programming language. The architectures, programming and design flows associated with FPGAs are described. Students are introduced to several commercial CAD tools including Xilinx ISE FPGA design software, Model-Sim, and Xilinx CORE generator software. Hardware experiments on FPGAs are carried out in the laboratories and projects.

**Prerequisites:** C- or better in 438.

**Textbook:** Pong P. Chu, *FPGA Prototyping By VHDL Examples, Xilinx Spartan-3 Version*, A John Wiley & Sons, 2008.

**Semesters offered:** Fall

**Level:** Undergraduate

**Credits:** 3

**Class times:** 2.5 hours of lectures per week. Approximately 10 lectures are laboratories.

**Course Coordinator:** Prof. Jim Plusquellic

**Table 1: Objectives, Implementation and Assessment**

	Objectives	Implementation	Assessment	A	B	C	D	E	F	G	H	I	J	K
O1	Understand basic design flow process.	Students learn the design flow and simulation processes associated with creating and verifying a simple behavioral VHDL design using an FPGA. 5 hrs lecture and 5 hrs of laboratory recitation in weeks 1, 2 & 3.	Tutorial and laboratory exercise using Xilinx ISE CAD software.			X								X
O2	Understand behavioral modeling	Students write behavioral VHDL code to program an FPGA to solve a design problem by implementing a specific behavior	Laboratory hardware demonstration and write-up with simulation results.		X	X		X						X

**Table 1: Objectives, Implementation and Assessment**

	<b>Objectives</b>	<b>Implementation</b>	<b>Assessment</b>	<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>	<b>K</b>
O3	Understand validation process and its importance using simulation.	Students write and simulate behavioral VHDL code that implements a state machine for a serial communication channel between computer and an FPGA. 5 hrs lecture and 5 hrs of laboratory recitation in weeks 4, 5 & 6.	Laboratory report with schematics and simulation results.	X	X	X		X						X
O4	Understand the divide and conquer hierarchical decomposition process associated with a complex design	Students design and implement a memory interface for the serial transfer state machine in lab O3. 5 hrs lecture and 5 hrs of laboratory recitation in weeks 7 & 8.	Laboratory hardware demonstration	X	X	X		X						X
O5	Understand the design paradigm of a complete system.	Students design and implement a video interface to the data stored in the memory of lab O4. 5 hrs lecture and 5 hrs of laboratory recitation in weeks 9 & 10.	Laboratory hardware demonstration	X	X	X		X						X
O6	Gain experience working in teams on a large design problem.	Students design and implement a video game on the FPGA by reusing code from the previous laboratories. 7 hrs lecture and 7 hrs of laboratory recitation in weeks 11-16.	Laboratory hardware demonstration and oral presentation.	X	X	X		X					X	X

**Table 1: Objectives, Implementation and Assessment**

	Objectives	Implementation	Assessment	A	B	C	D	E	F	G	H	I	J	K
O7	Understand cost, complexity and time-to-market trade-offs among alternative programmable logic devices and ASICs, understand internal organization and embedded features of modern FPGAs.	12 hrs of lecture interleaved with above over the whole term.	Final Exam	X				X						X

**Table II: Expectation and Assessment Outcome**

**Fall 2008, Dr. Jim Plusquellic**

General expectations:

**Laboratories:**

The laboratories serve several major roles:

1. Develop a student’s understanding of the hardware design paradigm through a hands-on experience with actual FPGAs.
2. Develop a student’s understanding of behavioral modeling and simulation as a means of validating designs.
3. Develop a student’s coding style and debug skills.

For 1), expect that all students will be able to successfully implement and demonstrate 70% of their designs in hardware. For 2), expect that all students will be successful in designing and validating their design in simulation. For 3), expect that all students will become proficient in hierarchically partitioning their designs to reduce complexity and will develop a set of debug techniques.

**Project:**

The project builds on the code and skills developed in the laboratories to solve a complex design problem in a team setting. Groups of students work together on implementing and demonstrating their design on an FPGA and describe their strategy and experience in an oral presentation. A competition is created to inspire students to devote time and energy on the project.

**Final Exam:** Expect 75% of the students to score 70% or better.

**Table 2: Objectives, Implementation and Assessment**

	Objectives	Outcome Assessment	Evaluation
O1	Understand basic design flow process.		
O2	Understand behavioral modeling		

**Table 2: Objectives, Implementation and Assessment**

	<b>Objectives</b>	<b>Outcome Assessment</b>	<b>Evaluation</b>
O3	Understand validation process and its importance using simulation.		
O4	Understand the divide and conquer hierarchical decomposition process associated with a complex design		
O5	Understand the design paradigm of a complete system.		
O6	Gain experience working in teams on a large design problem.		
O7	Understand cost, complexity and time-to-market trade-offs among alternative programmable logic devices and ASICs, understand internal organization and embedded features of modern FPGAs.		

**Sample Course Schedule:**

Introduction (1 lecture): Focus of the course, course prerequisites, FPGA, PLD and ASIC definitions, functions served by FPGAs today, fusible link, mask programmable, PROMs, EPROMs, EEPROMs, Flash, SRAM-based technologies.

Introduction to VHDL (1 lecture): Definition of EDA and VHDL, synthesis and design flows, technology trade-offs in terms of cost, complexity, time-to-market, schematic vs. HDL as design entry points, structural VHDL, module definition, basic language constructs, similarities/differences with high-level programming languages such as C, language support for simulation, examples.

Overview of VHDL (1 lecture): Hardware modeling concepts, use of hierarchy to handle complexity, examples, primitive gates supported within VHDL, structural VHDL examples, timing constructs supported for simulation, explicit vs. implicit structural VHDL descriptions, continuous assignment, module port syntax, RTL/data flow examples, clock synchronization, behavioral VHDL language constructs and examples, VHDL for design synthesis, VHDL language conventions.

FPGA Architectures I (1 lectures): Origins of FPGAs, taxonomy of PLD technologies, programmability capabilities of PROMs, PLAs, and PALs, definition and structural overview of CPLDs, definitions, structural details and distinctions among ASICs, gate arrays, structured ASICs and full custom chips, definition of intellectual property (IP), design flows, gap filled by FPGAs, basic architecture of FPGAs and pro-

programmable logic block (PLB) elements, FPGA-ASIC hybrids, example applications.

Xilinx ISE tutorial (1 lecture): Covers project creation, assigning a specific FPGA, behavioral VHDL as a design entry point, assigning module inputs/outputs to specific pins on the FPGA, synthesis to schematic, translation, mapping and FPGA programming file generation, behavioral simulation tool.

Digilent board tutorial (1 lecture): Discuss details of the Digilent board that embeds the Xilinx FPGA. Connect Digilent boards to computer via parallel port. Run ISE tools to generate an example programming file. Program and test the FPGA on the Digilent board.

VHDL Data Types and Operators (1 lecture): basic VHDL data types, nets and registers, rules for assignment, memory declaration syntax, VHDL logic values, strings, constants, arithmetic operators, bit operators, reduction operators, logical operators, relational operators, conditional operator, concatenation operator, operator precedence.

VHDL Event Driven Simulation (1 lecture): Purpose of simulation, event driven model, examples, basic algorithm data structures, modeling of propagation delay, examples, language support for building test benches, building waveforms in test benches, special variables and control constructs, examples.

VHDL Behavioral Constructs (1 lecture): VHDL behavioral language constructs, procedural statements vs. continuous assignment statements, blocking and non-blocking versions of assignment, examples, event control operators, sensitivity list, VHDL coding of flip-flops (FF) and latches, VHDL behavioral language constructs such as **if**, **case**, **with** and **when**. Guidelines to follow to ensure VHDL code is synthesizable by CAD tools.

Chip Scope and Xilinx CORE generator tutorials (1 lecture): Chip Scope: Covers software for this hardware debugging tool. CORE generator: Covers families of cores available and tool used to customize the core.

VHDL State Machines (1 lecture): Mealy vs. Moore, state definition, next state and output logic specification, synchronous vs. asynchronous behavior, combinational vs. sequential logic synthesis, FSM styles, serial adder, and other examples using Mealy and Moore styles, FSM diagrams, algorithmic state machine representations, ISE synthesis of FSM to schematic diagrams.

FPGA Architectures II (1 lecture): SRAM-based FPGAs, IP security mechanisms, anti-fuse-based and EEPROM/Flash-based FPGAs advantages/disadvantages over SRAM-based FPGAs, fine-, medium- and course-grained architectures trade-offs of PLBs, MUX- vs. LUT-based PLB architectures, internal details of PLBs, embedded e-RAMs and multipliers.

VHDL Design Examples (1 lecture): Examples including bit counting and serial division. Synthesis of combinational and sequential language constructs.

FPGA Architectures III (1 lecture): Embedding of hard and soft cores such as microprocessors, e-RAM, clock trees and clock managers, configurable I/O standards, gigabit transceivers, hard, soft and firm IP.

Configuration Options and FPGA vs. ASIC Design Styles (1 lecture): configuration data and commands, scan-chains linking configuration cells, serial and parallel configuration modes, hardware interfaces to configuration data, differences between FPGA

and ASIC design styles including coding style, use of pipelined logic, asynchronous logic, delay chains, clock gating, and PLLs.

Schematic-based and HDL-based Design Flows (1 lecture): Design flow using schematic as design entry point, optimizations processes for mapping, packing and place and route synthesis steps for FPGAs, verification. Design flow using behavioral code and RTL as design entry point, optimization processes, Verilog vs. VHDL.

**Grading:** Typical grading:

Final	20%
Labs	35%
Project	40%
Class Participation	5%