

# Comprehensive VHDL Introduction

---

**4 Days: 50% Lecture, 50% Lab**

**Basic Level**

## **Overview**

An in-depth introduction to VHDL and its application to design and verification of digital hardware (FPGAs and ASIC). Provides a foundation in RTL and testbench coding styles required by design and verification engineers who are new to VHDL.

Through a series of lectures, exercises, and labs, students will gain a strong foundation in VHDL RTL and testbench coding techniques. Lectures contain numerous examples that show both syntax and coding style guidelines. Exercises provide immediate reinforcement of lecture materials. Labs give students hands-on experience writing RTL code, writing testbenches, running your simulator, running your synthesis tool, and programming our FPGA based lab board.

The comprehensive lecture guide provides an excellent after-class reference. Syntax and code that is not synthesizable is specifically noted. Labs account for approximately 50% of class time. Lab projects range from simple simulation and synthesis coding problems to small design projects. The design projects utilize all techniques learned in lecture and demonstrate how VHDL is used in a project environment.

## **Intended Audience**

Recommended as a first course in VHDL for design and verification engineers who need an in-depth understanding of VHDL and a solid foundation in RTL and testbench coding techniques.

## **Prerequisites**

None. Offered as a first course in VHDL. It is recommended that students are familiar with digital design.

## **Course Objectives**

Upon completion of this course, students will be able to:

- Understand VHDL syntax and coding styles relevant to logic design
- Write VHDL RTL hardware designs using good coding practices
- Understand the synthesizable subset of VHDL
- Understand problematic issues in coding hardware
- Use types, overloading, and conversion functions from standard VHDL packages (std\_logic\_1164, numeric\_std, and std\_logic\_arith)
- Print messages in testbenches using TEXTIO
- Write simple transaction-based testbenches using subprograms
- Use your VHDL simulation and synthesis tools

# Comprehensive VHDL Introduction

## Course Outline

### **Day 1**

A Quick Introduction  
Lab 1: Simple RTL and Testbench  
Data Types  
Operators  
Concurrent Statements  
Sequential Statements  
Lab 2: Clock and Reset  
Lab 3: RTL and Testbench

### **Day 2**

RTL Essentials  
Statemachine Coding Techniques  
Lab 4: RTL Code  
Data Objects  
Design Hierarchy  
Lab 5: Coding an FSM  
Lab 6: Creating Hierarchy

### **Day 3**

Testbench Essentials  
Subprograms  
Lab 7: Brute Force Testbenches  
Testbenches and Timing  
VHDL IO (TextIO)  
Lab 8: Transaction-based Testbench  
Lab 9: Creating an FPGA (Synthesis and device programming).

### **Day 4**

RTL Code  
Numeric Types  
Design Organization  
Lab 10: UART Transmit Data Path  
Lab 11: UART Transmit Statemachine  
Lab 12: Multiplier Accumulator

### **Take Home Labs**

Digital Clock

## Follow-On Courses

Students wishing to go beyond what they learned in this course should take either or both of the following courses:

**VHDL Coding Styles for Synthesis - 4 days**

**VHDL Testbenches and Verification - 4 days**

## Customization

All of our courses can be customized to meet your specific needs. Either see our website or contact us for details.

## Training Approach

This hands-on, how-to course is taught by experienced hardware designers using a computer driven projector. We prefer and encourage student and instructor interaction. Questions are welcome. Bring problematic code.

## Contact

To schedule a class or for more information, contact SynthWorks at 800-505-8435 or jim@synthworks.com

**Learn VHDL from a designer's perspective with SynthWorks.**