**Comprehensive VHDL for Professionals**

**Description**:

This is a comprehensive VHDL Course for FPGA / ASIC designers and students who would like to learn the most effective VHDL coding styles for simulation and synthesis. In this course, students will learn VHDL from a working environment level, to project planning and definition stages and through commands and syntax.  Much of what is taught is followed up by hands-on exercises on VHDL workstations. Simulations and synthesis are used to give students practice in VHDL programming and provide them with an understanding of the logical implementation of VHDL commands. Workshops comprise approximately 50 % of the class, and are based around carefully designed exercises to reinforce and challenge the extent of learning.

**Course** **Outline**: Each of the following topics will be covered in this course:

1. Introduction
   * Aim
     + To obtain general appreciation of what VHDL is, and how it is used in design process
   * Topics covered
     + What is VHDL?
     + History of VHDL
     + VHDL design flow and tools
     + SRAM-based FPGAs
     + Benefits of VHDL
     + VHDL resources
2. Getting started
   * Aim
     + To understand the overall flow of VHDL projects
   * Topics covered
     + Entity, ports and architecture
     + Test benches
     + Liberates and compilation
     + Toll flows
3. Design entities
   * Aim
     + To become familiar with the main features of VHDL, particularly design entities
   * Topics covered
     + Entities and architectures
     + IEEE Standard logic
     + Concurrent assignments and signals
     + Direct instantiation and port maps
     + Recommended layout
     + Names
4. Processes
   * Aim
     + To understand the process statement
   * Topics covered
     + The process statement
     + Signal assignments
     + Delta delays
     + Inertial delays
     + testbenches
5. Synthesis combinational logic
   * Aim
     + To understand the basic sequential statements, and synthesis of combinational logic
   * Topics covered
     + If statements
     + Conditional signal assignments
     + Equivalent process
     + Transparent latches
     + Case statements
     + Synthesis of combinational logic
6. Types
   * Aim
     + To understand the characteristics of INTEGER, STD\_LOGIC and STD\_LOGIC\_VECTOR types
   * Topics covered
     + VHDL types
     + Standard packages
     + INTEGER subtypes
     + Types STD\_LOGIC and STD\_LOGIC\_VECTOR
     + Slices and concatenation
     + Integer and vector values
7. Synthesis of arithmetic
   * Aim
     + How to synthesize arithmetic using vectors
   * Topics covered
     + Arithmetic operator overloading
     + Arithmetic packages – Numeric STD
     + Mixing integers and vectors
     + Resizing vectors
     + Resource Sharing
8. Synthesizing sequential logic
   * Aim
     + To understand the use of VHDL in modelling and synthesizing sequential logic, i.e. circuits containing clocks and registers
   * Topics covered
     + RISING\_EDGE
     + Asynchronous set or reset
     + Synchronous inputs and clock enables
     + Synthesizable process templates
     + Implying registers
9. FSM synthesis
   * Aim
     + To understand how to write and synthesize finite state machines using VHDL
   * Topics covered
     + Moore and Mealy machines
     + State transition diagrams
     + Enumeration types
     + Signal process state machines
     + Multiple process style
     + State encoding
     + Unreachable state and input hazards
10. Memories
    * Aim
      + To learn further useful features of VHDL for modelling and synthesizing memories
    * Topics covered
      + Array types
      + Modelling memories
      + IP generators
      + Using generated components
      + Implementing ROMs
11. TEXTIO
    * Aim
      + To understand the use of TEXTIO in reading and writing text files
    * Topics covered
      + TEXTIO
      + READ and WRITE
      + TEXTIO for Stimulus and outputs
      + STD\_LOGIC\_TEXTIO
12. More about type
    * Aim
      + To understand more about types, variables, and loops
    * Topics covered
      + Variables
      + Loops
      + Array and integer subtypes
      + Std\_logic and resolution
      + Aggregates
13. Managing hierarchical design
    * Aim
      + To understand some key points concerning VHDL components, libraries and configurations for Hierarchical Design
    * Topics covered
      + Hierarchical design flow
      + Library name mapping
      + Component declaration
      + Configurations
      + Hierarchical configurations
      + Compilation order
14. Parameterized design entities
    * Aim
      + To learn further useful features of the design entity for both synthesis and simulation in large and complex designs
    * Topics covered
      + Array attributes
      + Port and generic maps
      + Generate statement
15. Procedural testbenches
    * Aim
      + To understand the “software”-like VHDL constructs required by procedural testbenches
    * Topics covered
      + Subprograms – Procedures
      + Subprograms – Functions
      + Parameters and parameters association
      + packages
      + subprograms in packages
      + overloading and ambiguity
16. Text based testbenches
    * Aim
      + To understand the use of text files and functions for complex test benches
    * Topics covered
      + Assert
      + Opening and closing files
      + Converting between VHDL types and text files
      + Checking simulation results
      + Foreign bodies
17. Gate level simulation
    * Aim
      + To show how to configure a test bench for gate-level simulation
    * Topics covered
      + Rational for gate level simulation
      + VITAL tool flow
      + Reusing RTL test bench at gate level
      + Comparison of RTL and gate level results
      + Behavioral modelling
18. Applications portfolio
    * Communications
    * Sensors
    * Aerospace
    * Manufacturing