

V1 - VHDL Language Basics

FPGA Programming and Simulation with VHDL

Objectives

- Comprehend the various possibilities offered by VHDL language
- Discover the complete design flow
- Understand the logical synthesis notions
- Implementing combinational and sequential logic
- Developing Finite State Machines
- Learning how to write efficient test benches for simulation
- Checking Timings
- Reusing and configuring components

Course environment

- Theoretical course
 - Course material (in English) in PDF and printed format
- Practical activities
 - Practical activities represent from 40% to 50% of course duration
 - They allow to validate or deepen the knowledge obtained during the theoretical course
 - A PC in pairs
 - For Xilinx:
 - ▶ Xilinx Vivado IDE
 - ▶ Nexys-3 (based on Xilinx Spartan6) or Nexys-4 (based on Xilinx Artix7) board
 - For Lattice :
 - ▶ Diamond with Synplify Pro and Active-HDL
 - ▶ ECP2 or MachXO board
 - For Actel :
 - ▶ Libero with Synplify Pro
 - ▶ ModelSim and Actel Fusion
- At the start of each session, a period is devoted to an interaction between the trainees and the trainer to ensure the course fit their expectations and adapt it if needed

Prerequisites

- Knowledge of digital technology
- Concepts of Boolean algebra
- Some programming concepts are desirable (whatever language)

Target Audience

- Any embedded systems engineer or technician with the above prerequisites.

Course Outline

First Day

From the logic gate to the FPGAs

- Reminder on digital electronic
 - Combinational Logic
 - Sequential (Synchronous) Logic
- Schematics / Hierarchical representation
- Structure of an Integrated Circuit
 - SSI (small scale integration), TTL
 - MSI (medium scale integration), PALs, GALs, PLDs
 - LSI (large scale integration), CPLDs
 - VLSI (very large scale integration), ASICs, ASSPs, FPGAs
- Development of logical architectures
- Technology constraints
 - Interconnection methods (SRAM, Fuse, AntiFuse, Flash)
 - Clock distribution
 - Logic element types
 - Look Up Table
 - Basic logic cell
 - I/O modules
 - Timing issues
- VHDL Contributions
 - Benefits of VHDL programming
 - The VHDL Design Flow
 - Programming
 - Simulation
 - Synthesis
 - Mapping
 - Place and Route
 - Timing Analysis
 - Bitstream generation

VHDL Basic concepts

- The Entity / architecture concept
 - Entity declaration
 - Ports
 - Different styles of architecture
- Libraries and context
- Component instantiation
 - Port map
- Simulation flow and environment
 - The Testbench
- Getting started with the IDE
- Creating a project from scratch
- Synthesis / Translate / Map / Place and Route (PAR) /BitGen
- Report Analysis
- Assigning I/O locations using PlanAhead (editing constraint file)
- Schematics Views
- Analyzing the placement
- Flashing with Impact

Exercise: Understanding the steps of design and programming

Exercise: Getting started with the simulator, waveform generation and analysis

Second Day

VHDL Syntax

- Lexical items
 - Comments
 - Identifiers and keywords
 - Characters, Strings, Numbers, Bit strings
- Constants
- Signals
- Variables and aliases
- Data types
 - Scalar types:
 - Integer
 - Real
 - Enumerated type
 - Physical types
 - Composite types:
 - Array
 - Record
 - Special types
- Library and Packages
 - Standard package
 - IEEE packages
 - Std_logic_1164 package
 - Multi-valued types
 - Multi-driver and resolved types
 - Numeric types
- Type conversion
- Aggregates
- Attributes
 - Type attributes
 - Signal attributes

Exercise: Importing a predefined hardware definition in the project, instantiating a component

Combinational logic in VHDL - 1st part

- Concurrent instructions
 - Component instantiation
 - Signal affectation
 - Simple affectation
 - With... Select... When statement
 - When... Else statement
 - Unaffected keyword
 - Variable aggregates
 - Relational operators
 - Arithmetic operators
 - Concatenation / Slicing

Exercise: Coding, simulating and synthesizing a bounds enforcer

Exercise: Designing a 7-segment decoder

Exercise: Designing a 4-bit adder

Third Day

Combinational logic in VHDL - 2nd part

- Sequential instructions
 - Processes
 - Sensitivity list, Wait statement
 - Potential interpretation incoherencies between logical synthesis and simulation
 - Signal affectation
 - Transparent Latch
 - Use of variables
 - If... Then... Else statement
 - Case... When statement
 - Null statement
 - Iterative statements:
 - For loop
 - While loop
 - Conditional Iteration
- Numeric_std / Numeric_bit packages
 - Defined Types and Operators
 - Conversion functions
- Ambiguity about the types and the « use » clause

Exercise: Coding, simulating and synthesizing a bounds enforcer

Exercise: Designing a 7-segment decoder

Exercise: Designing a 4-bit adder

Synchronous logic in VHDL

- Limits of asynchronous designs
- Synchronous Design, Registers and Timing
- Pipeline notion
- D Flip-flop description
- Use of Variable for synchronous process
 - Variable Synthesis
- Reset and Set management
- Clock Enable
- Tri-state buffers description
- Synchronous design methodology
- Memory Synthesis
 - Asynchronous RAM
 - Synchronous RAM
- Single port
- Double port
- Pipelined
 - ROM
 - IP generator introduction

Exercise: Designing a counter/decounter

Exercise: Designing a FIFO

Fourth Day

Hierarchical Design

- Hierarchical division
- Analysis and Elaboration
- Components and Configurations

- Components
- Configuring components instances
- Direct instantiation
- Basic configurations
- Configuration declaration
- Default binding
- Configuration specification
- Port map and Generic map
 - Genericity and automatic configuration of re-usable modules
- Packages
 - Package Declarations
 - Package Bodies
 - Using package
- Libraries

Exercise: Designing a generic 4-digits BCD-counter and displaying it on a 7-segment display

Exercise: Enhancing a 4-bit BCD-counter/decounter to create a generic one

Exercise: Working with configurations

Exercise: Designing a n digits BCD-counter/decounter and displaying it on a 7-segment display

Synthesis and Testbenches

- Synthesis
 - Syntactic and Semantic Restriction
 - Creating synthesizable Designs
 - Inferring Hardware elements
 - Initialization and Reset
 - Pragmas
- Testbenches
 - A few basic rules for the writing of an efficient test bench
 - Potential incoherencies between logical synthesis and simulation: how to avoid it
 - VHDL instructions specific to simulation
 - Testbench with File I/O

Exercise: Designing and testing a logical address decoder

Exercise: Simulation of sequential processes

Exercise: Advanced simulation techniques Text files