VHDL Logical Synthesis and Simulation for Xilinx™ FPGA design

5 days - 35 hours

OBJECTIVES

- Understand the 7-Series FPGA architecture
- Comprehend the various possibilities offered by VHDL language
- Understand the logical synthesis notions
- Know the different writing style and their impact on the quality of synthesis results
- Know the performance that can be expected from Xilinx FPGA
- Learn how to configure compilation options and implementation constraints
- Manipulate the debug tools and implementation reports

RELATED TRAININGS

- Static Timing Analysis (STA), Xilinx Design Constraints (XDC) and Advanced use of Vivado
- Designing with the Xilinx™ 7-Series Families
- Designing with the Xilinx™ UltraScale and UltraScale+ Families
- Vitis™ High Level Synthesis

PREREQUISITES

- This training is intended to electronic engineers who already have a good knowledge in designing digital electronic circuits, who are willing to acquire a strong designing methodology, and to take the best of VHDL language and the associated synthesis and simulation tools for designing Xilinx FPGA.

PARTNERS

XILINX | Authorized Training Provider

CONFIGURATIONS

- Software Configuration :
  - Vivado Design Suite 2020.2
- Hardware configuration:
  - Recent computer (i5 or i7)
  - OS Linux 64-bits (Windows 10 compatible)
  - At least 16GB RAM

- Display resolution recommended 1920x1080

CHAPTERS

DAY 1

- 7-Series FPGA architecture
  - General structure
  - CLB and slices notion
  - Dedicated RAM blocks and use modes
  - Dedicated multipliers and DSP48 blocks
  - In/Out blocks
  - Clocks distribution, MMCMs and PLLs
  - Configuration

DAY 2

- Writing rules of VHDL code in logical synthesis
  - Notion of entity / architecture
  - Concurrent and sequential instructions
  - Predefined types and objects
  - Predefined operators and of use extended by using standardized packages
  - Concurrent instructions : when, with select, for generate
  - Practical labs

DAY 3

- Writing rules of VHDL code in logical synthesis (next)
  - Process
    - Importance of the sensitivity list
    - Sequential instructions : if, case, loop
    - Use of variables
    - A few tricks to avoid
    - Potential interpretation incoherencies between the logical synthesis and the simulation : how to avoid it
- Hierarchy management for a better use
  - Organization of design by functional modules : what routing to choose ?
  - Inference and instancing notions
    - When is it important to instantiate primitives or macros ?
  - Precautions for an evolutionary and / or re-usable code
  - Importance of module's name selection and of the nets to facilitate the physical implementation, the simulation and the tuning
  - Does the hierarchy have to be preserved during the logical synthesis ?
  - Practical labs

DAY 4

- Advanced VHDL language for optimization and code re-use in logical synthesis
  - Notion of variable and example of use
  - Genericity and automatic configuration of re-usable modules
  - Useful predefined attributes in logical synthesis
  - Functions and procedures
  - Definition of packages and libraries
  - Practical labs
- Hardware designing methodology in logical synthesis
  - Asynchronous conception and classic tricks
    - Metastability and hazards of functioning
    - Limits of functional simulation and timing on asynchronous designs : how to get over them?
  - Asynchronous event management
    - Random
    - Data streams
  - Synchronous design
  - Static timing analysis : how to use it?
  - Optimization of performance irrespective of the target
  - Pipeline notion
  - Practical labs
- Implementation and tuning tools
  - Implementation flow and bitstream generation
  - Reports Analysis
  - Main implementation options
  - Implementation results analysis tools

DAY 5

- Test benches and simulation
  - A few basic rules for the writing of an efficient testbench
  - VHDL instructions specific to simulation
    - Wait and its various forms
    - Loop
    - Assertions
    - Data types
    - Others
  - Writing components models intended to make the simulation more realistic
  - Use of existing models and simulation packages
  - Practical labs
  - Writing and reading of ASCII files
    - Allocation of a data flow from a file
    - Storage of the simulation results in a file
    - Command interpreter
  - Generating information messages
  - Practical labs

TEACHING METHODS

- Classroom training:
  - Face to face
  - Presentation by video projector
  - Provision of PDF course materials
- Virtual training:
  - Onlive training
  - Presentation by Webex
  - Provision of PDF course materials

SUPPORT

- Authorized Trainer Provider XILINX : Engineer Electronics and Telecommunications ENSIL
  - Expert FPGA XILINX - Language VHDL/Verilog - RTL Design
  - Expert SoC & MPSoC XILINX - Language C/C++ - System Design
  - Expert DSP & RFSoC XILINX – HLS - Matlab - Design DSP RF
  - Expert ACAP XILINX – AI Engines – Heteregenous System Architect

METHODS OF MONITORING AND ASSESSMENT OF RESULTS

- Attendance sheet
- Evaluation questionnaire
- Evaluation sheet on:
  - Technical questionnaire
  - Result of the Practical Works
  - Validation of Objectives
- Presentation of a certificate with assessment of prior learning

CONCERNED PUBLIC

- Technicians and Engineers in Digital Electronics

CONTACT

Administratif : +33 (0)6 30 94 50 17
Formateur : +33 (0)6 74 52 37 89
info@mvd-training.com