V2 - Advanced VHDL for FPGA

Acquire a strong design methodology with the best of VHDL for simulation and synthesis

Objectives

- Comprehend the various possibilities offered by VHDL language
- Be able to read and test VHDL components
- Understand the logical synthesis notions
- Understand the crucial issue of implementing Finite State Machines (FSMs) in hardware
- Reusing components
- Checking Timings
- The OSVVM and UVVM VHDL verification methodologies

Prerequisites

- Basic knowledge of VHDL, V1 - VHDL Language Basicscourse level

Course environment

- Theoretical course
  - Course material (in English) in PDF and print format
  - Course dispensed using the Teams video-conferencing system
  - The trainer answers trainees' questions during the training and provides technical and pedagogical assistance
- 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
  - One PC per pair of trainees for the practical activities
  - Xilinx Vivado IDE
  - Code examples, exercises and solutions
- 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

Target Audience

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

Course Outline

First Day

Finite State Machine (1st part)

- The Finite State Machine Approach
  - Sequential Circuits and State Machines
  - State Transition Diagram
  - Transition Types
  - Moore-to-Mealy Conversion
  - Mealy-to-Moore Conversion
  - Exercises
- Hardware Fundamentals

    - Flip-Flops
    - Metastability and Synchronizers
    - Pulse Detection
    - Glitches
    - Pipelined Implementations
    - Exercises
    - Hardware Architectures for State Machines
    - Fundamental Design Technique for Moore Machines
    - Fundamental Design Technique for Mealy Machines
    - Moore versus Mealy Time Behavior
    - State Machine Categories and State-Encoding Options
    - Safe State Machines

Finite State Machine (2nd part)

- Design Steps and Classical Mistakes
    - Classical Problems and Mistakes
    - Design Steps Summary
- Regular State Machines
    - Architectures for Regular Machines
    - Number of Flip-Flops
    - Exercises
- Timed State Machines
    - Architectures for Timed Machines
    - Timer interpretation
    - Transition Types and Timer Usage
    - Timer Control Strategies
    - Time Behavior of Timed Moore and Mealy Machines
    - Examples of Timed Machines

Exercise: Designing a burstable RAM controller

Second Day

Design Methodology for Synthesis

- Designing for Synthesis
- Metastability
- Memory Synthesis
- Reset Generation
- Crossing Clock domains

Exercise: Metastability

Timing analysis and constraints

- Timing Closure challenges
- A methodology for successful Timing Closure
- Common Timing Closure Issues
- Static Timing Analysis
- Role of Timing Constraints in STA
- Common Issues in STA
- Delay Calculation versus STA
- Timing Path
- Setup and Hold
- Slack
- On-Chip Variation
- Clock
- Port Delays

- Completing Port Constraints
- False Paths
- Multi Cycle Paths
- Combinational Paths
- Xilinx Extensions

Exercise: Design closure
Exercise: Analyzing and Resolving timing violations

Third Day

Introduction to Open Source VHDL Verification Methodology (OSVVM)

- Overview
- Transaction-Level Modeling
- Constrained Random Test Generation
- Functional Coverage
- Intelligent Coverage Randomization Methodology
- Utilities for Testbench Process Synchronization
- Transcript Files
- Error Logging and Reporting: Alerts and Affirmations

Introduction to Universal VHDL Verification Methodology (UVVM)

- Utility Library
- VVC (VHDL Verification Component) Framework
- BFMs (Bus Functional Models
- OSVVM and UVVM

Vivado Debug

- Vivado Integrated Logic Analyzer (ILA)
- Adding debug nets
- Analyzing debug data
- Resources