SynthWorks

Expert VHDL Training for Hardware Design and Verification

PLC2
PROGRAMMABLE LOGIC
COMPETENCE CENTER

PowerWorkshop Professional VHDL Testbenches and Verification with OSVVM

Heutige FPGA und ASIC Designs haben sich in Größe und Komplexität drastisch seit den Anfängen des digitalen Schaltungsentwurfs weiterentwickelt. Diese aufwendigen Schaltungen werden daher mit Hardware-beschreibungssprachen, wie VHDL, als Hierarchie von Teilsystemen erstellt. Die Teilsysteme werden dabei meist durch standardisierte Businfrastrukturen wie AXI, PLB, Avalon oder WishBone verbunden, sowie mit einem Soft-CPU-IP-Core oder einem eingebettetem ARM Prozessor kombiniert. Solch ein Entwurf ist bei weitem zu komplex als das man ihn mit simplen Assertion-basierten VHDL Testbenches verifizieren könnte.

Mit der Open Source VHDL Verification Methodology (OSVVM) wird eine strukturierte Vorgehensweise aufgezeigt, welche eine hohes Maß an Wiederbenutzbarkeit im Testbench Code ermöglicht. OSVVM ist eine freie und als Open-Source verfügbare VHDL Library, die Pakete, Datentypen und Unterprogramme, sowie Algorithmen anbietet, welche in fast jeder Testbench benötigt werden. Es besteht kein Grund das Rad ständig neu zu erfinden. Als neuestes Feature bietet OSVVM nun auch vordefinierte Verifikations-IPs, sodass ein breites Spektrum an Standardbussen abgedeckt wird.

OSVVM bietet eine Methodik, welche die folgenden Themenschwerpunkte abdeckt: Transaction-Based Modeling (TBM), Self-Checking, Scoreboards, Memory Modeling, Functional Coverage, Directed, Algorithmic and Constrained Random, sowie Intelligent Testbench Test Generation. Eine VHDL Testbench-Umgebung basierend auf OSVVM ist dabei so mächtig wie andere vergleichbare Verifikationssprachen, wie beispielsweise SystemVerilog oder ‚e'.

Die Schulung startet mit einfachen Testbenches und steigert den Abstraktionsgrad kontinuierlich. Die Teilnehmer lernen die Verwendung von Unterprogrammen und Bibliotheken, Lesen und Schreiben von Dateien, Herausforderungen bei der Modellierung, Transactions-Based Testbenches, Bus Functional Models (BFM), Transaction Basen Models (TBM), Record Datentypen, Resolution Functions, Abstraktionen für Interface Connectivity, Methoden zur Modellsynchronisierung, Protected Types, Access Types (Pointer), unterschiedliche Datenstrukturen (z.B. Scoreboards), Directed, Algorithmic, Constrained Random und Coverage Driven Random Testerzeugung, Self-Checking (Ergebnisse, Timing, Protokoll-Tests und Error Injection), Functional Coverage, Darstellung von Analogwerten und periodischen Signalverläufen, Timing und Ausführung des Codes, Testpläne und Configurations kennen.
Dieser Kurs enthält mehrere Beispiele, die direkt als Vorlage bei der Entwicklung eigener Testbenches benutzt werden können. Als Ergebnis erhalten Sie eine Testumgebung auf Systemebene, die transaktionsgesteuert und selbsttestend ist. Praktische Übungen bieten die Möglichkeit das Gelernte anzuwenden.

Anwendbare Technologien

Keine

Voraussetzungen

Fundierte Kenntnisse in VHDL und digitalem Schaltungsentwurf (z.B. Professional VHDL)

Dauer und Kosten

5 Tage, € 3.100,– netto pro Teilnehmer inklusive ausführlichen Schulungsunterlagen sowie Pausengetränken und Mittagessen

Agenda

- Testbench overview
- From basics to subprograms
- Transactions and subprograms
- Modeling for verification
- VHDL I/O
- **Lab Review:** Testing w/ subprograms
- Transaction-based models (TBM / BFM)
- Elements of a transaction-based model
- Data structures for serification
- **Lab Review:** UartTx BFM
- Creating sests
- Constrained random testing
- Functional coverage
- Execution and timing
- Configurations and simulation management
- Advanced coverage
- Advanced randomization
- **Lab Review:** Scoreboards, randomization and coverage
- Test plans
- Modeling RAM
- Transaction-based BFM part 2