

# Expert VHDL

## Advanced Level - 5 Tage

Expert VHDL ist ein intensives 5-tägiges Fortgeschrittenentraining. Entwickler steigern mit diesem Training die Produktivität durch Ausbau von VHDL-Codierungs- und -anwendungsfähigkeiten. Der Fokus in den zwei unabhängigen Modulen liegt auf Sprache und Synthese, Wartbarkeit und Wiederverwendbarkeit des Designcodes, Testbenches und den neuesten Verifikationstechniken – incl. einer Einführung in OVL und die moderne annahmebasierte Verifikation.

- Expert VHDL Design (2 Tage) ist für Design-Ingenieure, die ihre Kenntnisse in der RTL-Synthese mit VHDL vertiefen sowie ihren VHDL-Codierungsstil mit Blick auf Wartbarkeit und Wiederverwendbarkeit des Design-Codes verbessern möchten. Design für die Verifikation wird mit einer Einführung in moderne annahmebasierte Verifikationsmethoden ebenfalls behandelt.
- **Expert VHDL Verification** (3 Tage) ist für Design- und Verifikationsingenieure, die Testbenches in VHDL oder Verhaltensmodelle für die funktionale Verifikation entwickeln.

Die Module können als Komplettkurs oder einzeln besucht werden. Sie bauen auf dem Doulos Industriestandard-Kurs „Comprehensive VHDL“ auf. Sorgfältig entwickelte praktische Übungen machen ca. 50 % des Trainings aus und verfestigen das Erlernete im Kontext der aktuellsten Tools, Verfahren und Methoden.

## Zielgruppe

- Designingenieure, die die Effizienz ihrer Hardwaredesigns verbessern und die Produktivität erhöhen möchten.
- Design- und Verifikationsingenieure, die effektive Testumgebungen zur Verifikation von komplexen Designs und Systemen strukturieren möchten.

## Kursinhalte

- Funktionen der Sprache VHDL, über das Erlernete in einem Grundlagenkurs hinaus
- Ein tieferes Verständnis der Sprache und Anwendung von VHDL zur mühelosen Fehlerbehebung bei VHDL-Simulations- und -Syntheseproblemen
- Grundsätze und Details für Ansätze bei der Designverifikation mit VHDL
- Strukturieren und Schreiben umfangreicher und komplexer VHDL-Testbenches
- Grundsätze und Details zum Schreiben von Verhaltensmodellen für Hardwarekomponenten in VHDL
- Das Produzieren kleinerer und schnellerer Hardware-Designs unter Einsatz von VHDL und RTL-Synthesetools
- Design-Techniken für arithmetische Operationen mit VHDL-Packages
- Details eines VHDL-Codierungsstils zur Wiederverwendbarkeit von Code sowie Aufbau einer Intellectual Property (IP) für die Wiederverwendbarkeit
- Eine Einführung zum IEEE 1076-2007c (VHPI) sowie einen Einblick in die vorgeschlagenen Änderungen für VHDL 2008

## Voraussetzungen

Um den größten Nutzen aus diesem Fortgeschrittenentraining ziehen zu können, ist die Teilnahme an Doulos **Comprehensive VHDL** oder vergleichbarem Training sowie mindestens 6-monatige Erfahrung mit VHDL erforderlich. Teilnehmer am Expert Design-Modul benötigen Vorkenntnisse in

# Expert VHDL

## Advanced Level - 5 Tage

RTL-Codierung und Synthese mit VHDL.

### Kursunterlagen

Die Doulos-Kursunterlagen sind für ihren umfassenden Informationsgehalt und die benutzerfreundliche Präsentation allgemein bekannt. In ihrem Aufbau, Inhalt und ihrer Themenbehandlung sind sie einzigartig im HDL-Trainingsbereich, was sie zu begehrten Nachschlagewerken hat werden lassen. In den Kursgebühren sind enthalten:

- Kursskripte mit vollständigem Stichwortverzeichnis, die ein komplettes Referenzhandbuch darstellen
- Arbeitsbuch mit nützlichen Beispielen zur praktischen Anwendung
- Doulos VHDL Golden Reference Guide für Sprache, Syntax, Semantik und Tipps
- Tour Guides (zur Unterstützung der Tools und Technologien Ihrer Wahl)

### Struktur und Inhalt

#### Expert VHDL Design (Tag 1-2)

##### RTL

Synthesising combinational and sequential logic • Using variables in clocked processes • Multiple drivers and tri-states • RTL functions and procedures • Kinds of decision-making logic • Using hierarchy to control synthesis • Timing constraints, area constraints, and optimisation options • Using generate and attributes for mapping onto FPGAs • Optimal one-hot decoding • Input hazards and metastability • Multiple clock edges • Synchronisation between clock domains • Synchronising reset signals • Synthesis methodology for large designs

##### IP and Re-use with VHDL

Using standard packages • Language level re-use • Standard component re-use • General re-use • Economic payback from re-use • Packaging IP for re-use • Impact of IP on the development cycle Writing re-usable RTL VHDL • Readability and maintainability • Seeing generaliseable properties Array attributes, cloning ranges • Arrays of arrays, unconstrained arrays, others • Creating regular structures using loops and generate • Using generics to parameterise widths and structures

##### VHDL Coding Styles for IP Block Design

Records • Using records and aliases for abstraction • Matrices • Converting between matrices and arrays • Representing register banks • Implementing destructive reads • State machine coding styles Synthesis and hardware encoding of state machines • Recursive instantiation and recursive functions

##### Design for Verification with Assertions

Reasons for designing with assertions • Properties and assertions • Examples in OVL

#### Expert VHDL Verification (Tag 3-5)

##### VHDL Language

Subprograms, parameters, assigning signals • User defined packages • User defined array types • Record types, selected names, aggregates, arrays of records • Types, subtypes and overloading,

# Expert VHDL

## Advanced Level - 5 Tage

conversion functions • Qualified expressions • Generics, string generics, array generics  
Configurations, binding and dependencies, generic and port maps

### Verification Environments and Methodology

The Verification Plan • Structure of a simple test bench • Structure of a complex test bench •  
Procedural stimulus generation • Reactive test benches • File I/O; TEXTIO and 'C' • Measuring delays  
• Monitoring internal signals • Generating random numbers • Collecting diagnostic data •  
Scoreboards • Coping with latency and Out-of-Order completion • Control files • Adding a user  
interface to a test bench • Writing behavioural models • Generic and parameterised test benches •  
How to implement functional coverage • How to implement run-time parameterisation • A re-usable  
generic approach to creating verification environments • Example code to take away

### How VHDL works

Signal assignments • Events and inertial delay • Deltas Drivers and resolution functions • Wait  
statements • NOW • Static elaboration, the network model • Dynamic elaboration, elaborating arrays  
and files in subprograms • VHDL Attributes

### Component Modelling

How to structure a behavioural model • Representing state • Example - behavioural modelling of a  
serial thermometer chip • Giving visibility of internal state • Modelling external timing relationships •  
Checking timing constraints using signal attributes • 1164 strength strippers • Handling 'X' on the  
inputs • Modelling memories • Modelling analogue blocks • Bus-functional models • Processor models  
• Foreign bodies for including C models for interfacing to emulators

## Vorangehende Trainings

- Comprehensive VHDL

## Verwandte Trainings

- Assertion Based Verification with PSL
- Xilinx TechClass
- Altera Designing with Quartus II
- Altera NIOS II SoPC
- Essential Perl
- Essential Tcl/Tk

## Weitere Informationen

Um einen Platz zu reservieren oder Preisinformationen zu erhalten, wenden Sie sich bitte an das  
Doulos Sales Team.