SystemC Modeling Using TLM-2.0

Intermediate Level - 3 days

SystemC Modeling Using TLM-2.0 is the authoritative industry standard 3-day training class teaching the final OSCI TLM-2.0 transaction-level modeling standard, which was itself released in June 2008. This class was developed by the authors of the IEEE 1666™ SystemC® Language Reference Manual and the TLM-2.0 User Manual.

This class builds on the Doulos Comprehensive SystemC class to prepare the engineer for practical project readiness using transaction-level modeling with SystemC and TLM-2.0. The OSCI TLM-2.0 standard enables interoperability between transaction-level models from different sources while allowing the fast simulation speed necessary for virtual prototyping. This class gives delegates the opportunity to hear the features of TLM-2.0 explained by people who worked at the heart of the OSCI standard development, and to practice their skills in coding a variety of common use cases.

Unique full working examples are provided as a starting point for your own experimentation and projects.

Because Doulos is independent, delegates can usually use their choice of design tools during the workshops. Workshops are based around carefully designed exercises to reinforce and challenge the extent of learning, and comprise approximately 50% of class time.

Doulos has a world-wide lead in independent SystemC know-how having been active in SystemC-based methods since 2000. We have delivered SystemC training and support to engineers in more than 170 companies world-wide – including direct involvement with methodology and tool developers in such companies as ARM, Cadence, CoWare, Mentor Graphics and Synopsys.

Who should attend?

Hardware, software and systems engineers who have a good working knowledge of C++ and SystemC, and want to learn to use the OSCI TLM-2.0 standard.

What will you learn?

- How to apply your SystemC knowledge for effective transaction-level modeling
- How to make best use of the features of the OSCI TLM-2.0 standard
- How to use the TLM-2.0 interfaces, sockets and payload to build fast, interoperable models
- How to trade off simulation speed against timing accuracy while retaining interoperability
- How to model the features of common bus protocols

Pre-requisites

There are three necessary steps in the learning path to effective SystemC usage:

- **Step 1** - C++ skills, gained by attending Essential C++ (or equivalent)
- **Step 2** - SystemC skills, gained by attending Fundamentals of SystemC, or equivalent.
- **Step 3** - Transaction-level modeling skills, gained by attending SystemC Modeling Using TLM-2.0

For further information contact your local Doulos Sales Office.
SystemC Modeling Using TLM-2.0

Intermediate Level - 3 days

Hardware or embedded software engineers with a background in Verilog, VHDL or C, will usually need to attend both Comprehensive SystemC (which comprises Essential C++ and Fundamentals of SystemC) and SystemC Modeling using TLM-2.0 within the space of a few weeks or months.

Engineers with an excellent working knowledge of C++ (or some other object-oriented programming language) may be able to fast-track some of the SystemC learning requirement. Please contact Doulos direct to discuss and assess your specific experience against the pre-requisites.

Training materials

Doulos training materials are renowned for being the most comprehensive and user friendly available. Their style, content and coverage is unique in the EDA training world, and has made them sought after resources in their own right. Fees include:

- Fully indexed class notes creating a complete reference manual
- Workbook full of practical examples and solutions to help you apply your knowledge

Structure and Content

DAY 1

Introduction
Role of SystemC and TLM • Evolution of TLM within OSCI • TLM requirements and use cases • Coding styles • TLM-2.0 structure and architecture • Interoperability layer • Utilities • The OSCI TLM-2.0 release kit and documentation

TLM-1.0
Modeling principles • Blocking versus non-blocking interfaces • Unidirectional interfaces • Bidirectional interfaces • Request and response objects • Convenience ports • TLM FIFO interfaces • Request-response channel

TLM-2.0 Architecture
Initiators, targets, and interconnect • Initiator and target sockets • Pass-by-reference • Forward and backward paths • TLM-2 core interfaces • Blocking versus non-blocking transport • Standard socket classes • Socket binding • Introduction to the generic payload and base protocol

Blocking Transport Interface
Blocking transport interface • Timing annotation • Temporal decoupling • Quantum keeper • Global quantum • Synchronization-on-demand • Loosely-timed coding style

DAY 2

Generic Payload
Generic payload attributes • Mutability • Command, address and data attributes • Byte enables • Streaming • Response status • Generic payload memory management

Non-blocking Transport Interface
Non-blocking transport interfaces • tlm_sync_enum • Forward, backward and return paths • Phases
SystemC Modeling Using TLM-2.0
Intermediate Level - 3 days

AT timing model • Base protocol rules • Early completion • Pre-emption • AT timing annotation • Payload event queues • Approximately-timed coding style • Request and response exclusion rules • Back-pressure • AT interconnect

Convenience Sockets
Simple sockets • b/nb conversion • Tagged sockets • Multisockets • Coding interconnects and address translation • Hierarchical binding • Passthrough sockets

Direct Memory and Debug Transport Interface
Direct memory versus debug interfaces • Direct memory interface • DMI transaction type • DMI descriptor • Rules for granting and denying DMI • Generic payload DMI hint • Address translation for DMI transactions • Debug transport interface • Debug transport transaction type

DAY 3
Extensions
Kinds of extension • The extension mechanism • Generic payload extension methods • Extension base class • Low-level extension programming • deep_copy_from • update_extensions_from • Extension memory management • Auto extensions • Sticky extensions • Memory-manager-agnostic extensions • Instance-specific extensions

Endianness
TLM-2 endianness principles • Organisation of the data array • Mixed-endian systems • Address alignment issues • Part-word transfers • Width conversions • Endianness helper functions • Endianness conversion functions • Arithmetic mode • Byte order mode • Tuning for simulation speed

Protocol types
tlm_phase • Extended phases • Ignorable phases • Protocol types • tlm_base_protocol_types • Defining new traits classes • Guidelines for protocol creation • Bridges • Bus snooping using DMI extensions

Analysis ports
Analysis interface • Analysis port • Subscribers • When to deep-copy transaction objects

Other examples
Source code examples to use in your own projects • AT initiator types • AT target types • Permutations of the forward, backward and return paths • Full AT interconnect implementation • Implementing exclusion rules and transaction queuing • Mixed AT/LT components • Base protocol checker • Atomic operations and transaction locking using extensions

Related courses
- Comprehensive C++
- Comprehensive SystemC
- Expert SystemC Verification (in-house delivery only)
- Modular SystemC (in-house delivery only)

For further information contact your local Doulos Sales Office.
SystemC Modeling Using TLM-2.0
Intermediate Level - 3 days

Project services
Doulos Project Services enable clients to access our world-leading technical know-how and apply it directly to projects. **Expert-on-call**, **Expert-design** and **Expert-support** options can be flexibly packaged and delivered to provide valuable expertise and additional resource just when it is needed.

How to book a class
To make a provisional booking, or to obtain pricing information, please contact your local Doulos sales team. You will find contact details on our [website](http://www.doulos.com).