Comprehensive SystemC ONLINE

PLEASE NOTE: This is a LIVE INSTRUCTOR-LED training event delivered ONLINE.

It covers the same scope and content as a scheduled in-person class and delivers comparable learning outcomes. Daily sessions comprise 4-6 hours of class contact time.

Comprehensive SystemC is a training class introducing SystemCTM, a C++ class library for system-level modelling. SystemC is typically used to model systems that have both hardware and software content at the transaction level of abstraction.

The syllabus covers the SystemC core language and its application to transaction-level modelling. The class complies with IEEE 1666-2005 and the SystemC 2.2 class library.

Comprising 2 modules, engineers can attend either the full 5-day class or just the Fundamentals of SystemC module. Attendance of both modules is recommended unless attendees already have a good background in the use of C++.

- Essential C++ for SystemC (sessions 1-4) takes engineers who have a basic knowledge of the C programming language and gives them a fast-track way to acquire a good grounding in C++, which is an essential foundation for learning SystemC.
- Fundamentals of SystemC (sessions 58) builds on the foundation laid by Essential C++ to prepare the engineer for the practical use of SystemC for transaction-level modelling. The class describes the core SystemC v2.2 class library and its application for system modelling, virtual platforms, and hardware implementation.

Fundamentals of SystemC includes an introduction to the SystemC TLM-2.0 standard. TLM-2.0 is taught in more detail in a separate 3-day follow-on class SystemC Modeling using TLM-2.0.

The workshops are based around carefully designed exercises to reinforce and challenge the extent of learning, and comprise approximately 50% of class time. Delegates can use the tools and platform of their choice on all exercises and workshops.

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 500 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 design engineers who wish to become skilled in the practical use of SystemC for modelling digital hardware
- System engineers and architects who wish to become skilled in the practical use of SystemC for system level modelling
- Software engineers who already have good knowledge of C/C++, but who wish to acquire some practical experience in the use of the SystemC class libraries

What will you learn?

- The C++ language features necessary to master SystemC
- Object-oriented programming techniques as used by the SystemC class libraries
- The SystemC core language, data types and channels

- How to make best use of the SystemC simulator to debug and validate your models
- How to move up from RTL modelling to transaction-level modelling
- An introduction to the SystemC TLM-2.0 standard
- An overview of high-level synthesis using SystemC (optional)
- An overview of the SystemC Verification Library SCV (optional)

Pre-requisites

- Essential C++ for SystemC Delegates need basic knowledge of the C programming language, in particular familiarity with C functions, variables, data types, operators, and statements. This module is suitable for people with no previous knowledge of C++, as a refresher for those with limited knowledge of C++, or for hardware engineers who are familiar with VHDL or Verilog®.
- Fundamentals of SystemC A working knowledge of C++ and of object-oriented programming concepts is essential and basic knowledge of hardware design is recommended. Prior attendance of the Doulos Essential C++ class (or equivalent) is required. Delegates with C++ experience should check their knowledge against the SystemC C++ Pre-requisites available from Doulos, before attending. The course is suitable for electronic hardware, software or systems engineers, but in order to gain maximum benefit from this course, delegates should be active users of either a high-level software programming language (ideally C++) or a hardware description language (VHDL or Verilog®)

Course 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 course notes creating a complete reference manual
- Fully indexed class notes creating a complete reference manual
- Workbook full of practical examples and solutions to help you apply your knowledge
- Doulos SystemC Golden Reference Guide for language, syntax, semantics and tips.

Structure and Content

Essential C++ for SystemC (4 sessions)

Learn about the differences between C and C++

From C to C++

Header files • Function overloading • Operator overloading • Pass-by-reference • const reference • Default arguments • I/O streams • Namespaces • Stream manipulators • Stream operator overloading • Standard string class • Stringstreams • Static, automatic, and dynamic storage • new and delete

Classes and Objects

Learn the principles of object-based design • Classes and objects • Inline members versus separate compilation • Public and private class members • Member functions • Scope resolution

Special Member Functions
Constructors • Destructors • Copy constructors • Initialization versus assignment • Pointers versus objects • The assignment operator • this • Constant objects and members

Vectors
Learn to make the most of the built-in standard classes • The C++ standard library • Vectors versus arrays • Common vector operations • Iterators

Master the subtleties of object-oriented programming in C++

Subobjects
Class relationships • Subobjects versus pointers • Initializing members • Initializing const members

Inheritance
Learn to exploit the power of object-oriented programming • Derived classes • Inheritance • Protected members • Up- and down-casting

Virtual Functions
Delve deeper into object-oriented programming techniques • Overriding methods • Virtual functions • Polymorphism • Identifying types at run-time • Examples from SystemC • Abstract base classes

Templates and Conversions
Advanced C++ features used in the SystemC class libraries • Function templates • Class templates • Examples from SystemC • Implicit conversions • User-defined conversions

Extra Features
Friends • Static members • Order of initialization • Multiple inheritance • Exceptions

Fundamentals of SystemC (4 sessions)

Become proficient in using the features of SystemC

Introduction
Learn the background to SystemC and how SystemC fits into the system-level design flow • The architecture of the SystemC release • The benefits and risks of adopting SystemC • The objectives of transaction-level modeling

Getting Started
Learn how SystemC source code is structured and how to organise files • SystemC header files and namespaces • Compiling and executing a SystemC model

Modules and Channels
How to describe the structural connections between modules • Modules • Ports • Processes • Signals • Methods • Primitive channels • Module instantiation • Port binding

Processes and Time
Describing concurrency and the passage of time • SC_METHOD • SC_THREAD • Event finders • Static and dynamic sensitivity • Time • Events • Clocks • Dynamic processes

The Scheduler
Gain an insight into how SystemC manages the scheduling of processes and events • Starting and stopping simulation • Elaboration and simulation callbacks • The phases of simulation • Event notification • wait and next_trigger

Learn to apply SystemC to modeling data, communication and busses.

SystemC Data Types
Data types for bit-accurate and hardware modeling • Signed and unsigned integers • Limited and finite precision integers • Assignment and truncation • Bit and part selects • Bit and logic vectors • Hexadecimal numbers

Debugging and Tracing
Learn about the facilities provided by SystemC to ease debugging and diagnostics • The report hander • Customizing report actions • Writing trace (vcd) files

Interfaces and Channels
Learn how channels are used to abstract communication and create fast simulation models • Hierarchical, primitive and minimal channels • Interface method calls • SystemC interfaces • Port-less channel access • The SystemC object hierarchy • The class sc_port • How to make the most of ports, channels and interfaces • sc_export

Bus modeling
Learn the techniques required to write and use bus models in SystemC • Master and slave interfaces • The execution context of interface method calls • Blocking and non-blocking methods • Using events and dynamic sensitivity within channels • Multi-ports • Port binding policies

Exploration of the application of Transaction-Level modelling

Additional Features
sc_signal_resolved • register_port • sc_process_handle • Event finders • default_event • pos vs. posedge vs. posedge_event • sc_event_queue • request_update and update • Passing arguments to spawned processes • terminated_event • sc_set_stop_mode

Introduction to TLM-2.0
Transaction Level Modeling • Virtual platforms • The architecture of TLM-2.0 • TLM-2.0 coding styles • The interoperability layer • TLM-2.0 utilities • Initiator, target, and interconnect • Initiator and target sockets • Generic payload • Response status

Further TLM-2.0
Software execution and simulation • The time quantum • b_transport • Timing annotation • Temporal decoupling • The quantum keeper • Base protocol rules • DMI • Simple sockets • Extensions • Interoperability

Supplementary Subjects

Fixed Point Types
Fixed point word length and integer word length • Quantization modes • Overflow modes • Fixed point context • The type cast switch • Utility methods

Overview of SystemC Synthesis
RTL versus behavioural synthesis technology • The work of the OSCI synthesis working group • Synthesizable data types • Synthesis restrictions • Clocked threads and resets

Overview of the SystemC Verification Library
Introduction to and aims of SCV • Constrained random verification methodology • Extended data types to support introspection • Randomization • Transaction Recording

IEEE 1666-2011
An overview of the latest version of SystemC, that is, IEEE 1666-2011 and SystemC 2.3