



Comprehensive SystemC™ 5 jours



Présentation

Comprehensive SystemC™ est un cours de formation de 5 jours qui enseigne le langage SystemC™, une bibliothèque de classes C++ pour la modélisation au niveau Système. SystemC est habituellement utilisé pour modéliser des systèmes qui comprennent des parties matérielles et logicielles à un niveau d'abstraction transactionnel.

Ce cours couvre le cœur du langage SystemC et ses applications à la modélisation au niveau transactionnel. Le cours satisfait au standard IEEE 1666-2005 et la bibliothèque de classe SystemC 2.2.

Ce cours est divisé en deux parties. Les participants peuvent suivre soit l'intégralité du cours de 5 jours ou n'assister qu'à la partie Fundamentals of SystemC. La participation aux deux parties est recommandée. Les participants ayant une très bonne connaissance de C++ peuvent se dispenser de suivre la première partie.

Essential C++ for SystemC (jours 1-2) permet de faire évoluer les participants de la connaissance de base du langage C à une bonne connaissance en C++ qui est la base du langage SystemC. Ceci est un moyen rapide et efficace pour passer d'une connaissance du langage C au langage C++ qui est une base nécessaire pour apprendre le SystemC.

Fundamentals of SystemC (jours 3-5) bâtit à partir de la connaissance acquise dans la première partie les éléments d'utilisation pour l'application pratique de SystemC pour la modélisation transactionnelle. Ce cours décrit l'utilisation de la bibliothèque de classe SystemC V2.2 et son application à la modélisation de systèmes, de communication et de matériel et logiciel au niveau transactionnel ainsi que l'implémentation matériel/ logiciel.

Le module Fundamentals of SystemC inclus une introduction au standard SystemC TLM 2.0.

TLM 2.0 est davantage détaillé dans le cours SystemC Modelling using TLM-2.0.

Les applications pratiques utilisent des exercices sélectionnés pour renforcer l'efficacité de la formation. Ces exercices occupent 50% du temps de la formation.

Doulos a été actif dans la méthodologie SystemC depuis l'année 2000. Plus de 170 Sociétés ont été formées dans le monde entier y compris une participation directe avec les Sociétés de développement d'outils telle que ARM, Cadence, CoWare, Mentor Graphics et Synopsys.

Objectifs pédagogiques

- Utiliser SystemC pour la modélisation de circuits logiques.
- Utiliser SystemC pour la modélisation au niveau système.
- Acquérir une expérience pratique dans l'utilisation des bibliothèques de classe SystemC.

Qu'apprendrez vous?

Les caractéristiques du langage C++ pour maîtriser SystemC

Les techniques de programmation orientées objet telles qu'elles sont utilisées dans les bibliothèques de classe SystemC.

Le langage SystemC avec les types de données et « channels ».

Comment utiliser au mieux le simulateur SystemC pour déboguer et valider vos modèles

Comment aller de la modélisation RTL à la modélisation au niveau transactionnel.

Comment écrire des modèles au niveau transactionnel de structure de plateforme « System on Chip ».

Comment maîtriser les modèles SystemC entre les niveaux d'abstraction :

- Une introduction au standard SystemC TLM-2.0
 - Une vue d'ensemble de la synthèse à partir de SystemC (option)
 - Une vue d'ensemble de la bibliothèque de vérification SystemC SCV (option)
-

Connaissances requises

Essential C++ for SystemC (jours 1-2) Les participants ont besoin d'une connaissance de base du langage de programmation C en particulier des fonctions C, des variables, des types de données, des opérateurs et des états. Le cours s'applique également aux personnes n'ayant pas de connaissances préalables de C++ , ou encore celles qui souhaitent rafraîchir leurs connaissances de C++ ou pour les concepteurs de matériel connaissant VHDL ou Verilog.®

Fundamentals of SystemC (jours 3-5) Une connaissance pratique de C++ et des concepts de programmation orienté objet est une nécessité. Une connaissance de base de la conception matériel est nécessaire. Il est nécessaire d'avoir suivi le cours Essential C++ ou un équivalent. Les participants avec une expérience C++ doivent vérifier leur connaissance avec le contenu du cours Essential C++. Le cours s'adresse aux ingénieurs de conception matériel , logicielle ou systèmes mais pour avoir le meilleur résultat les participants devraient être des utilisateurs d'un langage de programmation (de préférence C++) ou d'un langage de programmation matériel (VHDL ou Verilog®).

Contactez AMBLOT directement pour vous aider à évaluer votre expérience par rapport aux connaissances requises.

Supports de cours

Les manuels de cours Doulos sont réputés pour être les plus détaillés et les plus faciles d'utilisation. Leur style, leur contenu et leur exhaustivité sont uniques dans le monde de la formation.

Sont compris dans la formation :

- Les notes de cours indexées constituant un manuel de référence complet.
 - Le cahier d'applications rempli d'exemples et d'application pratiques pour vous aider à mettre en œuvre vos connaissances.
 - Le guide de référence Doulos décrivant le langage, la syntaxe..
-

STRUCTURE AND CONTENT

Essential C++ for SystemC (2 days)

Day 1

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

Day 2

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 (3 days)

Day 3

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

Day 4

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 handler • 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

Day 5

Exploration of the application of Transaction-Level modeling

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

Doulos acknowledges trademarks and registered trademarks are the property of their respective owners