Comprehensive SystemC

Formation de 5 jours

Advanced Logic Synthesis for Electronics
www.ALSE.fr

DOULOS

Présentation

Comprehensive SystemC™ est un cours de formation complet de cinq jours consacré au Langage SystemC™ : une bibliothèque de classes C++ destinée à la modélisation hardware au niveau Système.
SystemC est souvent utilisé pour modéliser des systèmes qui comprennent à la fois des parties matérielles et une dimension logicielle, avec des communications abstraites au niveau transactionnel. On utilise également souvent SystemC pour la construction de bancs de test sophistiqués.

Ce cours est donc consacré au cœur du langage SystemC et à ses applications pour la modélisation transactionnelle. Il est aligné sur le standard IEEE 1666-2005 et sur la bibliothèque de classes SystemC v2.2.

La formation est divisée en deux parties. Les participants peuvent suivre l'intégralité du cours de cinq jours (fortement recommandé) ou n'assister qu'à la deuxième partie *Fundamentals of SystemC*, mais uniquement s'ils ont une *très bonne* connaissance préalable de C++ (classes et partie objet).

Essential C++ for SystemC (jours 1-2) permet efficacement d'ajouter à une connaissance de base du langage C une bonne connaissance C++ (OOP & classes), indispensable à la compréhension du langage SystemC.

Fundamentals of SystemC (jours 3-5) s'appuie sur les connaissances acquises dans la première partie et décrit pas-à-pas l'utilisation de la bibliothèque de classes SystemC v2.2 et son application à la modélisation de systèmes et de la communication matériel / logiciel au niveau transactionnel.

Ce module inclut une Introduction au standard SystemC TLM 2.0. Cette extension du langage SystemC fait l'objet d'une formation complémentaire « *SystemC Modelling using TLM-2.0* ».

Compléments indispensables à la complexité de l'enseignement théorique, les exercices pratiques qui concluent chaque chapitre assurent la mise en œuvre pratique des concepts et ils sont essentiels à la qualité de l'enseignement. Ces exercices occupent typiquement environ 50% du temps de la formation.

Doulos a été actif dans la Méthodologie SystemC depuis l'année 2000 et plus de 170 Sociétés ont été formées dans le monde entier. Doulos a participé directement avec les Sociétés de développement d'outils telle que ARM, Cadence, CoWare, Mentor Graphics et Synopsys.

Objectifs pédagogiques

Maîtriser :

- Les éléments du langage C++ indispensables pour appréhender SystemC
- Les techniques de Programmation Orientées Objet (utilisées dans les bibliothèques de classes SystemC).
- Le langage SystemC avec les types de données et les « channels ».
- La bonne utilisation du simulateur SystemC pour mettre au point et valider vos modèles
- Aller de la modélisation RTL à la modélisation au niveau transactionnel.
- Écriture des modèles au niveau transactionnel pour des « Systems On Chip ».
- Les modèles SystemC de différents niveaux d'abstraction.
- Une introduction au standard SystemC TLM-2.0.
- Vue d'ensemble de la synthèse à partir de SystemC (option).
- 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 les fonctions C, les variables, les types de données, les opérateurs etc.
Par contre, la connaissance préalable de la partie objet (C++) du langage n'est pas exigée.
Ces eux jours sont particulièrement utiles aux utilisateurs du lanage C, à ceux qui souhaitent rafraîchir leurs connaissances en C**++** et pour les concepteurs hardware utilisant plutôt VHDL ou Verilog®.

Fundamentals of SystemC (jours 3 à 5). Une c*onnaissance profonde et complète* du langage C**++** et des concepts de Programmation Orientée Objet est une nécessité. Une connaissance de base de la conception matérielle est nécessaire. Il est nécessaire d'avoir suivi le cours *Essential C++* ou équivalent.
Cette formation s'adresse aux ingénieurs de conception Matériel (HW), Logiciel (SW) ou Système.

Supports de cours

Les manuels de cours Doulos sont réputés pour être détaillés, précis et faciles d'utilisation. Leur style, leur contenu et leur exhaustivité sont uniques dans le monde de la formation HDL. Ils sont souvent utilisés ensuite comme référence.

Sont compris dans la formation :

- Le Classeur du cours théorique, avec un Index.
  Il constitue un Manuel de Référence concis.
- Le Cahier des Exercices pratiques, qui permet de mettre en œuvre les connaissances théoriques.
- Le « Doulos Golden Reference Guide » : Aide-mémoire SystemC très exhaustif et pratique (syntaxe, sémantique, trucs et astuces). Il est destiné à devenir pendant longtemps le compagnon de votre clavier...

Programme de la Formation

Essential C++ for SystemC (Jours 1 & 2)

Jour 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

Jour 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 (jours 3, 4 et 5)

Jour 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

Jour 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 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
◆ Unifying ideas: transaction ports, transaction FIFO, design pattern for publisher/subscriber, callbacks for snooping and error injection
◆ Base classes for transaction, transactor, environment ◆ Automating the activity of testbench components

Coverage and planning

◆ Coverage-driven TBA methodology ◆ Coverage planning as the first step in verification process
◆ SystemVerilog coverage constructs in details ◆ Analyzing and interpreting coverage data

Jour 5

Additional Features and Introduction to 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