SystemC

SystemC is a set of C++ classes and macros which provide an event-driven simulation interface (see also discrete event simulation). These facilities enable a designer to *simulate* concurrent processes, each described using plain C++ syntax. SystemC processes can communicate in a *simulated* real-time environment, using signals of all the datatypes offered by C++, some additional ones offered by the SystemC library, as well as user defined. In certain respects, SystemC deliberately mimics the hardware description languages VHDL and Verilog, but is more aptly described as a *system-level modeling language*.

SystemC is applied to system-level modeling, architectural exploration, performance modeling, software development, functional verification, and high-level synthesis. SystemC is often associated with electronic system-level (ESL) design, and with transaction-level modeling (TLM).

Contents

Language specification

SystemC is defined and promoted by the Open SystemC Initiative (OSCI — now Accellera), and has been approved by the IEEE Standards Association as IEEE 1666-2011[1] - the SystemC Language Reference Manual (LRM). The LRM provides the definitive statement of the semantics of SystemC. OSCI also provide an open-source proof-of-concept simulator

(sometimes incorrectly referred to as the reference simulator), which can be downloaded from the OSCI website.[2] Although it was the intent of OSCI that commercial vendors and academia could create original software compliant to IEEE 1666, in practice most SystemC implementations have been at least partly based on the OSCI proof-of-concept simulator.

SystemC has semantic similarities to VHDL and Verilog, but may be said to have a syntactical overhead compared to these when used as a hardware description language. On the other hand, it offers a greater range of expression, similar to object-oriented design partitioning and template classes. Although strictly a C++ class library, SystemC is sometimes viewed as being a language in its own right. Source code can be compiled with the SystemC library (which includes a simulation kernel) to give an executable. The performance of the OSCI open-source implementation is typically less optimal than commercial VHDL/Verilog simulators when used for register transfer level simulation.

SystemC version 1 included common hardware-description language features such as structural hierarchy and connectivity, clock-cycle accuracy, delta cycles, four-valued logic (0, 1, X, Z), and bus-resolution functions. From version 2 onward, the focus of SystemC has moved to communication abstraction, transaction-level modeling, and virtual-platform modeling. SystemC version 2 added abstract ports, dynamic processes, and timed event notifications.

Language features

Modules

SystemC has a notion of a container class called a module. This is a hierarchical entity that can have other modules or processes contained in it.

Modules are the basic building blocks of a SystemC design hierarchy. A SystemC model usually consists of several modules which communicate via ports. The modules can be thought of as a building block of SystemC.

Ports

Ports allow communication from inside a module to the outside (usually to other modules) via channels.

Signals

SystemC supports resolved and unresolved signals. Resolved signals can have more than one driver (a bus) while unresolved signals can have only one driver.

Exports

Modules have ports through which they connect to other modules. SystemC supports single-direction and bidirectional ports.

Exports incorporate channels and allow communication from inside a module to the outside (usually to other modules).

Processes

Processes are used to describe functionality. Processes are contained inside modules. SystemC provides three different process abstractions[*which?*] to be used by hardware and software designers. Processes are the main computation elements. They are concurrent.

Channels

Channels are the communication elements of SystemC. They can be either simple wires or complex communication mechanisms like FIFOs or bus channels.

Elementary channels:

- signal: the equivalent of a wire
- buffer
- fifo
- mutex
- semaphore

Interfaces

Ports use interfaces to communicate with channels.

Events

Events allow synchronization between processes and must be defined during initialization.

Data types

SystemC introduces several data types which support the modeling of hardware.

Extended standard types:

- sc_int<*n*> *n*-bit signed integer
- sc_uint<*n*> *n*-bit unsigned integer
- sc_bigint<*n*> *n*-bit signed integer for $n > 64$
- sc_biguint<*n*> *n*-bit unsigned integer for $n > 64$

Logic types:

- sc_bit 2-valued single bit
- sc_logic 4-valued single bit
- sc_bv<*n*> vector of length *n* of sc_bit

- sc_lv<*n*> vector of length *n* of sc_logic

Fixed point types:

- sc_fixed<> templated signed fixed point
- sc_ufixed<> templated unsigned fixed point
- sc_fix untemplated signed fixed point
- sc_ufix untemplated unsigned fixed point

History

- 1999-09-27 Open SystemC Initiative announced
- 2000-03-01 SystemC V0.91 released
- 2000-03-28 SystemC V1.0 released
- 2001-02-01 SystemC V2.0 specification and V1.2 Beta source code released
- 2003-06-03 SystemC 2.0.1 LRM (language reference manual) released
- 2005-06-06 SystemC 2.1 LRM and TLM 1.0 transaction-level modeling standard released
- 2005-12-12 IEEE approves the IEEE 1666–2005 standard for SystemC
- 2007-04-13 SystemC v2.2 released
- 2008-06-09 TLM-2.0.0 library released
- 2009-07-27 TLM-2.0 LRM released, accompanied by TLM-2.0.1 library
- 2010-03-08 SystemC AMS extensions 1.0 LRM released
- 2011-11-10 IEEE approves the IEEE 1666–2011 standard for SystemC[3]
- 2016-04-06 IEEE approves the IEEE 1666.1–2016 standard for SystemC AMS

SystemC traces its origins to work on Scenic programming language described in a DAC 1997 paper.[4]

ARM Ltd., CoWare, Synopsys and CynApps teamed up to develop SystemC (CynApps later became Forte Design Systems) to launch it first draft version in 1999.[5][6] The chief competitor at the time was SpecC another C based open source package developed by UC Irvine personnel and some Japanese companies.

In June 2000, a standards group known as the Open SystemC Initiative was formed to provide an industry neutral organization to host SystemC activities and to allow Synopsys' largest competitors, Cadence and Mentor Graphics, democratic representation in SystemC development.

Example code

Example code of an adder:

```
#include "systemc.h"

SC_MODULE(adder)           // module (class) declaration
{
  sc_in<int> a, b;         // ports
  sc_out<int> sum;

  void do_add()            // process
  {
    sum.write(a.read() + b.read()); //or just sum = a + b
  }

  SC_CTOR(adder)           // constructor
  {
    SC_METHOD(do_add);     // register do_add to kernel
    sensitive << a << b;   // sensitivity list of do_add
  }
};
```

Power/Energy estimation in SystemC

The Power/Energy estimation can be accomplished in SystemC by means of simulations. Powersim[7] is a SystemC class library aimed to the calculation of power and energy consumption of hardware described at system level. To this end, C++ operators are monitored and different energy models can be used for each SystemC data type. Simulations with Powersim do not require any change in the application source code.

See also

- Accellera
- SpecC
- SystemVerilog
- SystemRDL

Notes

1.

- *"Browse Standards"*. *standards.ieee.org.*
- • www.systemc.org, the Open SystemC Initiative website Archived 2008-10-06 at the Wayback Machine

- IEEE Approves Revised IEEE 1666™ “SystemC Language” Standard for Electronic System-Level Design, Adding Support for Transaction-level Modeling -- http://www.businesswire.com/news/home/20111109006054/en/IEEE-Approves-Revised-IEEE-1666%E2%84%A2-%E2%80%9CSystemC-Language%E2%80%9D

- *"ScenicDAC1997". CiteSeerX 10.1.1.56.6483.*

- Synopsys and Co-Ware Inc., which did much of the work behind the SystemC -- http://www.electronicsweekly.com/Articles/1999/12/07/13906/stm-synopsys-in-3-year-rampd-deal.htm

- "ARM is pleased that Synopsys, CoWare and other companies have come together on SystemC, because if it is taken up by the industry, it simplifies our world," said Tudor Brown, chief technology officer of ARM Ltd" in Babel of languages competing for role in SoC - http://www.eetimes.com/ip99/ip99story1.html

- http://sourceforge.net/projects/powersim/