

SystemC: the SoC system-level modeling language

By Jerry Gipper

An SoC is literally a system on a chip consisting of both silicon and embedded software. Its design involves complex algorithm and architecture development and analysis similar to that performed in system design – a trade-off process that determines critical metrics such as SoC performance, functionality, and power consumption. Design and verification complexity is clearly on the rise.

Consequently, design tools must deliver orders-of-magnitude improvement in productivity at both architectural and implementation (RTL and physical) levels. Moreover, tools must support a methodology that enables early embedded application and system software development, long before the RTL design or silicon prototype are available. Failure to achieve the requisite improvements in design productivity would result in missed market windows and exploding design costs.

The world of Electronic Design Automation (EDA) is rapidly moving forward. As processor, board, and system designs continue to become overwhelmingly complex, it is more critical than ever that new tools emerge to make the design process more manageable. Fortunately, the industry is making progress and picking up the pace in this sector.

Increasing design capacity

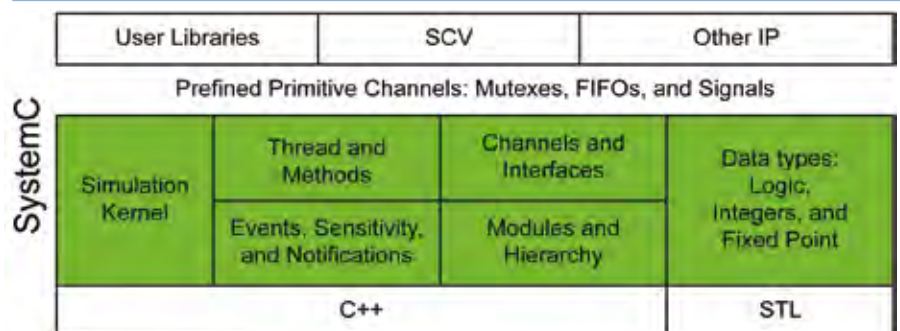
Two modeling languages in particular interest SoC designers. One of the newer additions to the tool set is SystemC, which IEEE ratified under IEEE Std. 1666-2005 in 2005. The Open SystemC Initiative (OSCI, www.systemc.org) is currently driving and supporting this effort. The second tool is SystemVerilog, ratified as IEEE Std. 1800-2005 at about the same time. The Accellera organization (www.accellera.org) spearheads SystemVerilog

efforts. The SystemC and SystemVerilog design and verification languages were developed to deliver exponential increases in design productivity in response to the industry's perennial challenge—continuing exponential increases in design and verification complexity. Knowing that these languages are increasingly used to complement SoC designs, what specific role does each of these play in SoC design?

SystemC was developed in response to demands for a standard Electronic System-Level (ESL) language that SoC designers using C/C++ could utilize. It is a single, unified design and verification language that expresses architectural and other system-level attributes in the form of open source C++ classes. It enables design and verification at the system level, independent of any detailed hardware and software implementation, and facilitates coverification with RTL design. This higher-level abstraction enables considerably faster, more productive architectural trade-off analysis, design, and redesign than is possible at the more detailed RT level. Furthermore, verification of system architecture and other system-level attributes is orders of magnitude faster than at the pin-accurate, timing-accurate RT level.

SystemC is purely a version of C++ that OSCI adapted and standardized for system-level design. The IEEE 1666 Language Reference Manual provides an unambiguous definition of the SystemC language, allowing EDA tool and IP developers to offer simulators, models, and system-level design tools that conform to the SystemC standard. Figure 1 shows the SystemC language architecture.

SystemVerilog, the first hardware description and verification language, is a major extension of the established IEEE 1364 Verilog-2001 language. Developed to improve productivity in developing large gate count, bus-intensive designs, SystemVerilog is targeted primarily at the behavioral-to-GDSII part of the SoC design flow. SystemVerilog also supports transaction-level modeling at the *transaction* abstraction level. This verification overlap between SystemC and SystemVerilog constitutes an invaluable design and verification link from the system level to chip implementation. SystemVerilog's Direct Programming Interface allows it to “call” C/C++/SystemC functions and vice versa, making SystemVerilog the first Verilog-based language to enable efficient SystemVerilog and SystemC block cosimulation.



Reprinted with permission from *SystemC: From the Ground Up*

Figure 1

Despite the two languages' complementary nature, comparative language analyses are fueling some debate that suggests designers must choose between them. But design and verification challenges have pushed mixed use of SystemC and SystemVerilog. To put this in perspective, consider that the "real" designers who work in RTL today worked exclusively at the gate level 10 years ago. SystemC does not replace RTL, but rather provides a new language to accomplish the tasks developers cannot complete in RTL. Figure 2 compares languages and demonstrates SystemC's broad usage across the tasks associated with electronic system design.

"Just like RTL is a level of abstraction that eliminates the need to worry about gates, SystemC eliminates the need to worry about RTL," said Mitch Dale, Calypto Design Systems' director of product marketing. "Each generation of tools increases design capacity." People are looking for ways to capture system-level descriptions, which include three components: models, a way to capture hardware intent, and interface with the software content.

Challenges in the hardware design community

New tools and technologies often struggle with gaining acceptance in the user community, and SystemC is no exception. For experienced hardware designers, SystemC is a new language that requires further education. Hardware designers must start thinking at a new and higher abstraction level and must learn to avoid concentrating on the details. Unlike the RTL space, where EDA tools are prevalent and readily available, the ESL design space is still being defined. System design engineers in school are more familiar with C and C++

than experienced engineers, so they might consider working with SystemC a pleasant experience.

Since the 40th Design Automation Conference in 2003, the industry has undergone a massive transition to system-level design. Now Transaction-Level Modeling (TLM), the next abstraction level above RTL, is becoming an essential area of focus. SystemC shines and system architecture and software converge at TLM and higher levels.

Design complexity is increasing so rapidly that design creation must move to higher-level abstraction. Dataquest has identified system-level design and low power as the most important design technologies, which is no coincidence considering that the available real estate on silicon is rapidly approaching 100 million gates.

OSCI to the rescue

Given the need to bring software and hardware design closer together and the numerous dialects different companies use, it was no surprise that the industry rallied behind CoWare and Synopsys when the two companies joined forces to codevelop SystemC. Most system, semiconductor, and IP companies were thrilled to see a drive toward a unified language for system-level design. Design was becoming more software-centric and IP reuse had to become a reality. A unifying SystemC would enable IP sharing at the system level and eliminate all incompatible C dialects. OSCI claims that SystemC has completely and clearly succeeded in this respect.

OSCI was launched in 1999 as an independent nonprofit organization dedicated to supporting and advancing SystemC as

an industry-standard language for ESL design. Its membership grew by 30 percent in the past year, reports the organization's executive director, Pat Sheridan of CoWare. "Clearly we are seeing a surge in SystemC since it became an international IEEE standard," Sheridan said. "For large companies and individual users alike, SystemC is now a major and mainstream feature in ESL design that provides the foundation for a worldwide ecosystem that would otherwise not be possible without a standard language."

SystemC is also gaining momentum around the world, with thriving user groups in North America, Europe, and Japan and new groups that have formed in India and Latin America. The European SystemC Users Group, for example, has grown from 40 members to more than 1,000 in the past seven years, said cochair Wolfgang Rosenstiel of the University of Tybingen.

In Japan, SystemC use continues to grow. Nearly 20 percent of respondents at EDSF now call SystemC their primary design language, doubling the number reported in 2003. "SystemC is widely used in Japan as the ESL language both for modeling and verification," said Fujitsu's Takashi Hasegawa, chair of Japan's SystemC Task Group. "With the enlargement and complexity of LSI, further improvement of design productivity and quality is required, and we believe SystemC is the solution."

In addition to increased use, SystemC and OSCI have notched a series of technological achievements, including approval of the IEEE 1666-2005 standard for SystemC and the impending release of the TLM 2.0 standard later this year.

The IEEE Std. 1666-2005 Standard SystemC Language Reference Manual is available for free at <http://standards.ieee.org/getieee/1666/index.html>. OSCI also recommends several books to help developers become more familiar with SystemC and TLM. Expect to see major advances as SystemC gains acceptance and more IP models appear. As Dale says, "It is larger than the language. It is part of the EDA revolution!" **ECD**

For additional reading:

- [1] David Black and Jack Donovan, *SystemC: From the Ground Up* (Kluwer-Academic Publishers, ISBN: 1402079885)
- [2] Ghenassia, Frank (Ed.), *Transaction-Level Modeling with SystemC: TLM Concepts and Applications for Embedded Systems* (ISBN: 978-0-387-26232-1)

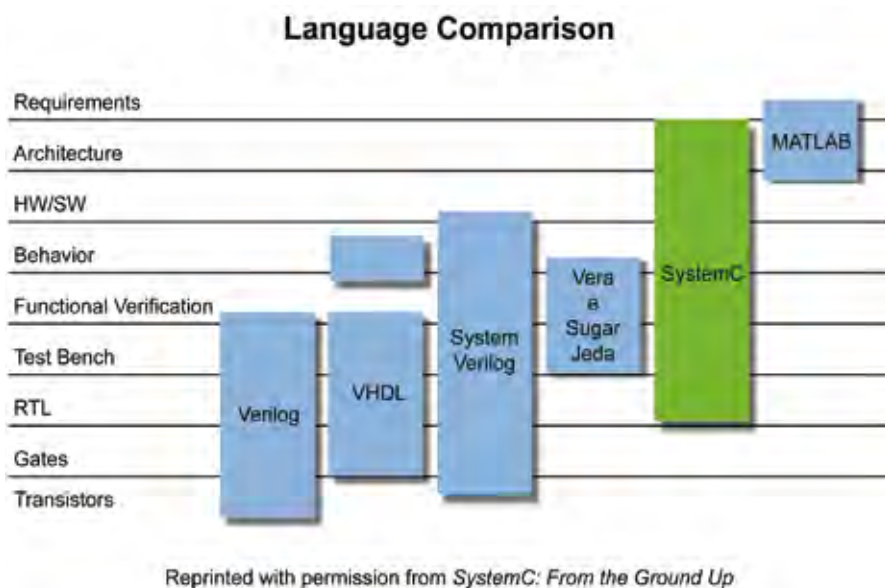


Figure 2