

Monday Afternoon Plenary Talk

Modeling Usable & Reusable Transactors in SystemVerilog

Janick Bergeron

Synopsys

Abstract

This paper describes how to properly use the object-oriented features of SystemVerilog to model transactions and transactors that execute them. Designers, used to a procedural programming model, will learn how to approach a transactor modeling problem using the object-oriented approach. Software engineers, used to a traditional object-oriented programming model, will learn how to adapt their methods to the special requirements of constrained-random verification. This paper will present clear and concise guidelines that can be followed to create effective, easy-to-use and reusable transactor models. Summary: Transactors ? also known as bus-functional models ? have traditionally been modeled using a procedural interface (API). Commands were provided to use every feature available in the transactor. Several commands were often necessary to configure the transactor or to execute the simplest transaction. If the transactor did not provide a feature required by a particular device under test (DUT) or testcase, it had to be modified or rewritten. By using an object-oriented approach for configuration and interfacing, transactors can easily be used out-of-the-box. A few lines are all that is necessary to create valid random stimulus to a DUT or report observed response to a scoreboard. The use of callback methods eliminates the need for a complex and ever-evolving API. Instead of creating yet-another programming language, it leverages the power of SystemVerilog to extend the functionality of a transactor in ways the original author may not have conceived. Without modification, a transactor can thus be used and reused effectively. Transactors can also be modeled in an extensible fashion. Extensible transactors allow users to define entirely new transactions and commands, should the ones provided by the original implementation prove insufficient. Usability and reusability of transactors does not happen by accident. With careful design and consistent use model, transactors can be written to meet the stringent demands of different testcases on different design under verification.

Since April 2003, he is a Scientist within Synopsys's Verification Group where he helps define the state-of-the-art in functional verification methodology and the tools that support it. He is also the author of the best-selling book "Writing testbenches: functional verification of HDL models" and the moderator of the Verification Guild. He is also a co-author of the upcoming "SystemVerilog Verification Methodology Manual". Prior to joining Synopsys, Janick was the CTO of Qualis Inc, where he designed the architecture of their Domain Verification Component technology - which was acquired by Synopsys in April 2003. In the eight years he has worked at Qualis, he defined the methodology used by their service consultants and helped create industry-leading training introductory and advanced classes. Before Qualis, Janick was a Member of Scientific Staff at Nortel Networks, in their first design verification group and was involved in writing their in-house logic synthesis tool before it was replaced by Design Compiler. Janick holds a MBA from the University of Oregon, a M.A.Sc in EE from the University of Waterloo and a B.Eng from the University du Quebec a Chicoutimi