

# Effects of Abstraction in Stimulus Generation of Layered Protocols within OVM

Josh Rensch<sup>1</sup>, Jesse Prusi<sup>1</sup>, Mike McMahon<sup>2</sup>, Andreas Meyer<sup>2</sup>

<sup>1</sup>Lockheed Martin MS2, Eagan, MN/USA

<sup>2</sup>Design and Verification Technology/Mentor Graphics Corp., Wilsonville, OR/USA

A multi-layered protocol is a lower-layer protocol wrapped in a higher-layer protocol, for example IP over Ethernet. Multi-layer protocols are challenging because of the linkage between the layers required for stimulus generation of a design under test (DUT) that is aware of and processes both layers simultaneously. This paper will discuss the challenges of verifying a design that supports multi-layer protocols and the use of Open Verification Methodology (OVM) transaction objects to overcome them, particularly in the creation of stimuli.

When we first established our verification team, stimulus generation was done at a low level of abstraction. This approach provided a smooth transition from a design background into verification and required cycle-level control at any layer of the protocol; however it resulted in considerable time and effort to create high-level scenarios and to debug the verification environment.

In our quest to resolve these significant issues, we asked the following questions. What is the highest level of abstraction for the environment that still allows us to create the entire stimulus spectrum needed? How does the environment maintain visibility at every level of abstraction? At what level does the environment compare the outputs of the DUT with those of the reference model?

We first created an OVM transaction object. Each instance of this object stores the configuration data for one specific transaction that is created for the DUT. This object resides at the highest level of abstraction and contains information for each level of abstraction as well as all protocol layers used in the design.

Stimulus generation occurs at different layers of the protocol. For example, error injection is handled at each layer of the protocol. At the highest layer of the protocol and the highest level of abstraction is the OVM transaction that was created to contain the configuration information. At a lower layer of the protocol, there is a sequence that converts higher protocol layer objects to lower layer protocol objects for the DUT level driver. At this layer, both the configuration information from the higher layer object and hierarchical sequence constraints are used to drive both valid data and errors into the DUT.

By organizing the stimulus at the highest abstraction level, the verification effort was significantly reduced. By relinquishing some of the cycle-level control, we realized several key benefits: the verification environment was simpler and required less code; there were fewer bugs in the verification code due to reduced dependency on design specifics; it was easier to create high-level scenarios; and we were able to maintain virtual control via hierarchical constraints to guide low-level behavior from the test.

By detailing our approach, we answer the questions above and demonstrate how to use OVM sequences, at the proper abstraction levels, to generate stimulus through protocol layers to meet coverage requirements.