

concepts synopsys

Using Vera and Constrained-Random Verification to Improve DesignWare Core Quality

By Chris Rosebrugh, Group Director, IP R&D, Synopsys

Introduction

As more system-on-chip (SoC) engineers rely on re-use to cut design time and reduce risk, the demand for synthesizable cores and other forms of intellectual property (IP) continues to rise dramatically. Not surprisingly, the successful usage of IP is closely correlated to IP quality. High quality demands the use of advanced tools and methods for functional verification to ensure that the IP works in all possible usage scenarios.

As the leading provider of standards-based interface IP, Synopsys constantly evaluates and evolves our processes for achieving the high quality that our customers expect. Of course, having ready access to the best suite of products in the EDA industry allows us to freely select those tools that support advanced methodologies. We're not mandated to use specific Synopsys tools; we use the ones that improve our IP products to provide our customers the best possible design and verification reuse experience.

This article discusses our transition to the use of constrained-random stimulus generation in the development of our synthesizable interface cores, a key part of our DesignWare® IP product portfolio. We interacted with the Synopsys Verification Group much like one of their customers; we evaluated the product (Vera®) that we thought would help us, and then used it on an initial project with assistance from the same engineers that support verification customers. The result was a significant, measurable increase in the quality of our cores that directly benefited our IP customer base. Better verification also benefited our own

development engineers, since they spend more time working on new projects than helping customers deal with issues in the current products.

IP Verification Background

Although constrained-random techniques have been around as a technique for a number of years now, many design teams did not embrace it until they crossed a complexity threshold due to the size or the type of designs they created. Until they made this leap, designers relied on hand-written, directed tests that provide explicit stimulus to the design inputs, run the design in simulation, and check its outputs against expected results. This approach, while manual and somewhat error-prone, provided adequate results for small, simple designs.

Given that most IP designs are well under 100,000 gates, most IP developers have relied on directed tests, perhaps supplemented by the limited pseudo-random capabilities achievable in Verilog and VHDL. This was the case early on for the cores in the Synopsys portfolio. Our thinking changed dramatically when we faced widespread deployment of a synthesizable core that implemented the USB 2.0 Host functionality.

Prior to some initial customer engagements, the USB 2.0 Host core had been verified by traditional methods—manual directed tests backed up by some random testing in Verilog—with effectiveness measured by traditional code-coverage metrics. The suite of 450 directed tests achieved what seemed to be reasonable coverage results:

- 97.50% FSM coverage
- 95.00% FSM transition coverage
- 88.64% toggle coverage
- 84.71% condition coverage
- 98.23% line block coverage
- 98.58% line statement coverage

Despite these good results, early reports from the field indicated that our initial customers were finding some corner-case problems that had not been caught by our fairly extensive test suite. We realized that the directed test approach did not scale with design complexity: although the USB 2.0 Host core was at most a few hundred thousand gates (depending upon configuration options), it contains extremely complex control logic with many combinations of conditions that are hard to set up with directed tests. We realized that we had to achieve more effective verification before we deployed this core to our broad customer base.

Transitioning to Vera and Constrained-Random Verification

We considered a number of possible ways to improve our IP verification process and decided to adopt Vera for the USB 2.0 Host core. Because constrained-random verification can automatically generate a large number of test cases within the parameters (constraints) specified by the verification team, it can hit corner cases that neither the design nor verification engineers would have ever anticipated. Without constrained-random stimulus, the bugs lurking in these corners hide until late in the development cycle, or aren't found at all until customer usage.

In addition to its overall complexity, the USB 2.0 Host core had another aspect that made verification challenging. The functionality and performance of the core is application and configuration dependent, so even if there are versions successfully running in silicon, corner-case bugs could still emerge if the core is used in a different way in another application. By automatically generating constrained-random tests across all configurations, we could achieve much broader verification across its different operating modes.

