RTL Quality for TLM Models

Preeti Sharma, Senior R&D Engineer I, Synopsys (India) Pvt. Ltd., Noida, India
(preetis@synopsys.com)

***Abstract*—Moore's law, which predicts that every two years, the number of transistors that can be placed onto a chip gets doubled, holds true for the enormous complex systems being designed today. With potential increases in chip sizes, and corresponding increase in functionality and product features, design quality and verification processes must be faster and efficient, to pace up with productivity. Bug escapes and low-quality products, not only incur considerable costs to companies, if identified at later stages, but also seriously damage a company's reputation in market.**

The need of the hour is hence debug ease, reduced number of lines of code, and robust design and verification processes. For design development, Transaction Level Modeling is emerging as a standard solution for virtual prototyping that provides faster simulation speeds.

Since RTL is a direct representation of the SoC architecture, virtual prototypes should be as close in functionality and behavior to their respective RTL, so as to yield substantial gains in hardware/software development life of a complete SoC or ASIC based system. This call for the need of a similar and standard verification flow for TLM models, as currently exists for RTL designs.

RTL designs are known to use formal tools and standard verification set-ups to verify complex designs and ensure functional correctness. One such hybrid methodology is the Universal Verification Methodology (UVM). It is highly desirable, that TLM models also go through similar high quality verification processes, to ensure that the standard ways that find bugs in RTL, can enable TLM model developers, to find bugs in their design as well, alongside measuring behavioral deviation with respect to the RTL IP. This paper explains how a similar verification approach can be developed for a TLM IP, using same standard UVM methodologies, that RTL verification engineers use worldwide.

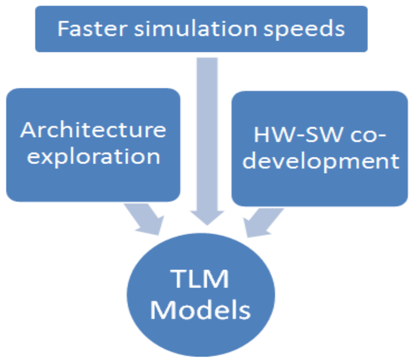
Keywords—TLM;verification;RTL;UVM;standard;simulation

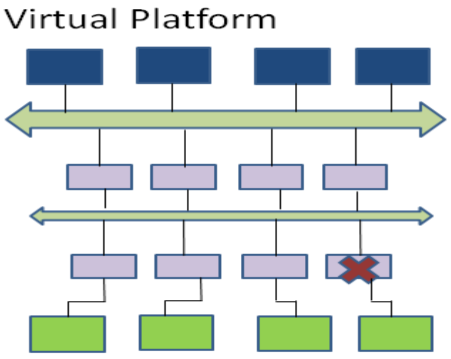
I. INTRODUCTION

The complexity of devices under test (DUTs) is continuously increasing in the semiconductor industry, primarily because of growing functional content complexity and the sheer amount of functional blocks contained in a typical SoC. Hence, the hardware engineers continue to aim for developing DUTs, with more complicated functionalities, faster speeds and with low costs. At the same time, these SoCs are expected to perform better with high quality compared to their predecessors. This phenomenon has thrown a significant challenge to test engineers to develop more complex test systems.

With such ever increasing competition, there are two prime concerns these SoCs face in the digital design world today. First and foremost is time to market for a product from the concept to the end level product. Virtual prototyping, as a method is gaining acceptance, to provide a solution for the same.

Transaction-level modeling (TLM), a high-level approach to modeling hardware designs, is a methodical and systematic virtual prototyping solution. In TLM, details of communication among modules are separated from the details of the implementation of functional units or of the communication architecture. Such shift in abstract level suggests that an order-of-magnitude improvement in designer productivity is indicated. SystemC is the best standard for describing transaction-level designs with a link to implementation and offers the best opportunities for reusability. It models hardware concurrency and describes both timed and untimed behavior for processes, pins, threads, and control logic.





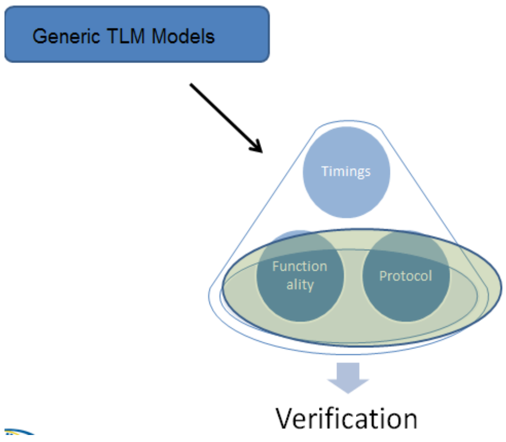
The problem now arises when the most basic intent of using TLM models, i.e. faster simulation speeds and hence easy hardware-software co-development, get compromised. This happens when a complex virtual platform consists of faulty TLM blocks. The amount of time needed to debug, analyze and fix the root-cause issue in a poor-quality TLM module in a complex virtual platform, over-shadows the benefits that it can provide, i.e. less time-to-market in developing end-level system level.

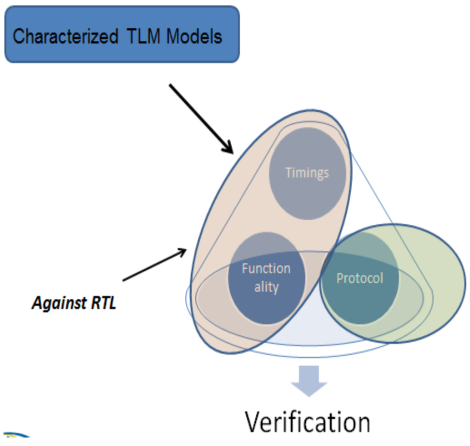
II. TLM VERIFICATION CHALLENGES

A standard SoC development flow is primarily RTL->Gates->Silicon. TLM models can now be added in this flow as equivalence to RTL, for early design and verification of system level designs. Observing this flow carefully, it can be noticed that functional verification has become a bottleneck for today's virtual platforms, due to a corresponding increase in the complexity and size of a platform. This has led to more efforts and time-consumption in the creation of a robust and comprehensive TLM-driven verification flow.

Since, a TLM model can be represented as an entry point to the SoC verification flow, an efficiently verified TLM model does not only mean much higher productivity, but also the ability to explore and verify a greater number of different implementations and ease in debugging, leading to high quality and better performance of virtual platforms.

TLM models primarily concentrate on three major areas during a verification flow: protocol compliance (for which the TLM model is developed), functionality (with or without the RTL design availability) and timings (with respect to RTL simulation results). A generic TLM model, which is not based on a specific RTL configuration, is primarily concerned with protocol compliance and functionality and less on timings. Similarly, a characterized TLM model, which is based on a specific RTL design configuration, needs to verify its functionality and timings and match them more closely with the respective RTL design, other than protocol compliance.





The most important point to note here is that currently in market, there is no established standard TLM-driven verification flow, thereby causing each developer or verification engineer to develop his own ad-hoc techniques and flows to verify his TLM models, leading to huge methodology deviations.

Besides, the need to verify, a) functional corner cases, that results from various interacting models in a platform, and b) a number of hardware and software configurations with which the TLM-IP must be tested, calls for thinking in a new direction, where a TLM model verification flow can be synchronized and standardized with an RTL IP verification flow. This means, a set of standard requirements need to be addressed and met, for verifying TLM IPs, similar to that of their corresponding RTL IPs, to detect design errors as soon as possible.
For a successful transition from TLM to RTL model usage, a dedicated TLM model verification environment is highly important, such that:

1. It specifies clear verification goals,
2. Follows standard processes of reaching these goals
3. The verification testbench fully complies with design/protocol requirements
4. Covers complete protocol interface, all possible untested functionalities and corner cases
5. Reaches for unanticipated tests through random traffic, and
6. Is robust, reusable, modular, scalable and configurable

III. A SOLUTION: VERIFICATION IPS

The UVM is an industry standard verification library and methodology developed by Accellera, developed mainly for RTL design verification. It is primarily written in System Verilog, which is a unified language for design and verification. It is effective for designing advance testbenches for both RTL and Transaction level models, since it has features like constraint randomization for stimulus generation, functional coverage, assertions, object oriented constructs (such as class, inheritance etc).
Verification IPs based on UVM, provide all sufficient components and features needed to verify SoC designs. Highly qualified and extensively used Synopsys VIPs, hence support the following:

- Support System Verilog based testbenches, including built-in methodology support for UVM, VMM and OVM
- Modular test-bench architecture that provides:
    - Programmable number of masters, slaves etc.
    - Features to simplify test-bench development
    - Protocol and test coverage
- Rapid creation of complex tests through
    - sequence library
- Constrained random verification
- Error injection

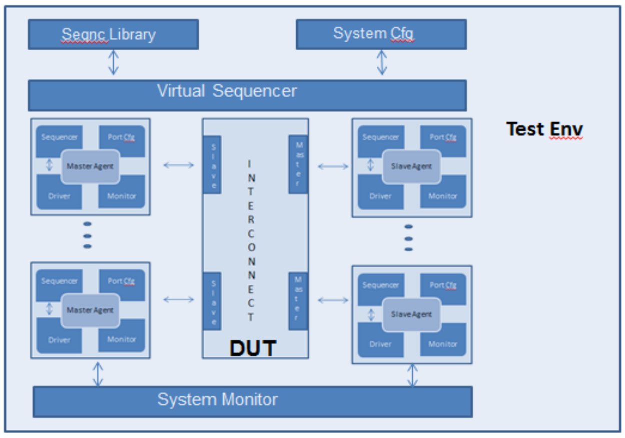
With such a vast scope of verification through UVM VIPs, a combined usage of both TLM and UVM methodologies, across both SystemC and SystemVerilog is now apparent, to leverage the advanced features of both in an integrated environment.

Looking at all the above features, it is easy to comprehend that the Universal Verification Methodology based Verification IPs provide the complete solution package for the verification challenges for TLM models.

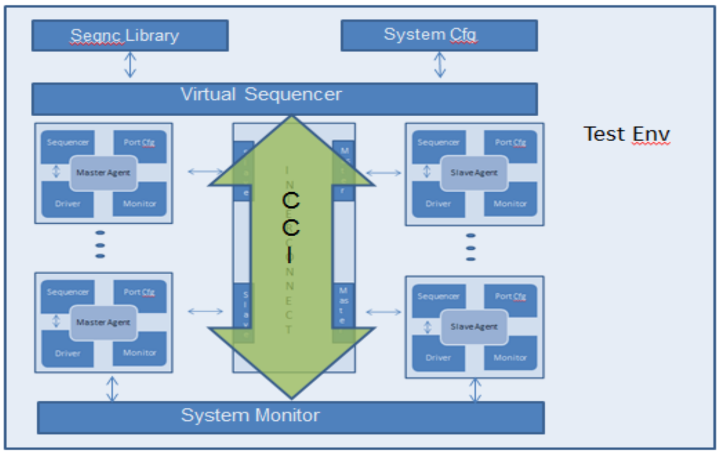
IV. THE PROPOSAL

A complete UVM verification methodology comprises of a UVM testbench that contains the following:

- UVM components (such as master/slave/interconnect agents, drivers, monitors, sequencers etc.)
- A UVM environment that creates and connects the above and
- A UVM test that becomes the start-point of firing sequences, doing checks for protocol correctness, recording coverage, and hence ensuring overall quality of the IP.



We propose a simplified and novel approach to verification of TLM models through reusing and refining the above UVM components. The creation of TLM model verification flow is accomplished by reusing RTL verification infrastructure in the form of VIP masters and slaves, and RTL tests in UVM and SystemVerilog. Necessary wrappers are created, that encapsulate a combination of different VIP agents, using Synopsys virtual platform creator tools. These modified System Verilog-UVM VIPs are integrated with SystemC-TLM IPs and then this hybrid platform is co-simulated. An objective data with pre-defined/modified VIP sequences are generated. Hence the simulation gets comprehensively verified using VIPs. This co-simulation of UVM VIPs is possible for TLM IPs that supports both cycle-accurate interfaces, and TLM generic payload interfaces.



V. CASE STUDY

As a case study, a generic cache coherent interconnect (CCI) based on AMBA-ACE protocol interface was taken as the required TLM DUT. The objective was to verify this SystemC-TLM based CCI model, using UVM based verification IPs. Since, Synopsys UVM based Verification IPs, provide structured test benches that are reusable and easily maintainable, the TLM CCI model could be completely verified for a number of pre-packaged traffic.

Using these testbenches, a combination of VIP agents was created, that can serve as a master or slave, or a combination of all, for verification of the TLM CCI model. Using advanced Synopsys tools, such as Platform Architect (PA-MCO), a combined platform is created and connected, using UVM VIPs and TLM IPs. Then, a set of UVM sequences in System Verilog, is tapped from a VIP master, and is fired on to the required CCI model that acts as a DUT. The output of the TLM interconnect model is again tapped and is injected into the corresponding VIP slave that acts as a scoreboard for pass/fail verification of fired data sequences.

Different combinations of above flow can be made where:

- Only VIP masters can be used, for TLM slave model verification
- Only VIP slaves can be used, for TLM master model verification
- A combination of VIP masters and slaves can be used, for verification of a number of TLM models.

Necessary test-bench modifications to enable this connection of a UVM-System Verilog world with a totally different TLM-SystemC world, are generally nonintrusive using PA-MCO, and hence are independent of any third-party tools/libraries.

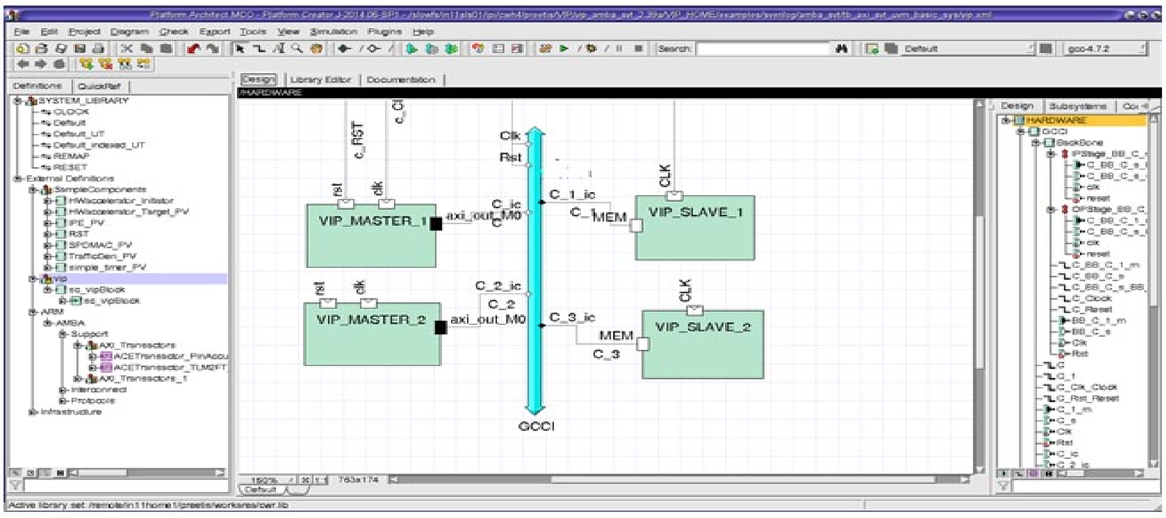
Verification of expected traffic being fired from a VIP onto the TLM interconnect model and vice versa can be easily verified through Synopsys debugging and analysis tools such as vpexplorer.

This experiment resulted in

- Running variety of directed/random tests
- Randomized constrained testing
- Rapid creation of new complex tests
- Identifying untested/unanticipated bugs in CCI model
    - AMBA-ACE protocol compliance
    - Unaligned, narrow traffic
    - Issues regarding Make Unique, make Invalid etc.

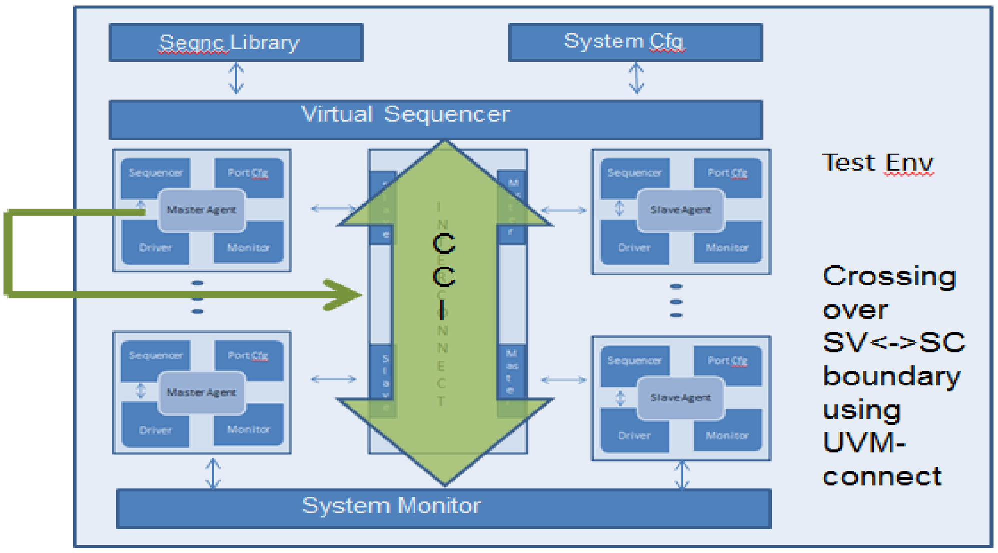
Hence, a number of untested and unanticipated bugs were identified and fixed in the cache coherent interconnect that could not be caught using non-standardized TLM SystemC based initiators/targets. Moreover, complete AMBA-ACE protocol compliance was covered using the same.

Below is a sample platform, created for verification of a TLM IP, with cycle-accurate interface:



A variant of the above is also possible. The UVM library now also supports TLM. The UVM methodology description is being extended to show how TLM and RTL models can be combined in a comprehensive regression solution. This can be achieved through UVM Connect. The UVM Connect package builds on existing standards: SystemVerilog, SystemC and UVM, allowing TLM models in each language to communicate with each other. The package also includes an API that allows SystemC to interact with, and control the execution of, UVM testbenches. The UVM Connect library, allows object passing across System Verilog and SystemC worlds using certain UVM Connect library functions.

Below is the prototype of a platform, created for verification of a TLM IP, with TLM generic payload interface, using UVM Connect feature:



VI. CONCLUSION

Since the process of verification parallels the design creation process for an RTL IP, this unified verification flow has demonstrated a near seamless transition between RTL design and verification of virtual prototypes, that are developed using standard TLM methodologies. It significantly takes advantage of verification industry standards of UVM, for developing and reusing bigger, complex and randomized tests, thereby catching more functional bugs in TLM models.

The TLM modeled platform, hence verified using standard UVM interface, can help in meaningful development of newer RTL features of high quality and better RTL upgrades to higher versions, and will carry a lot of value proposition for RTL designers who can use these TLM models for identifying earlier bugs. To summarize, this flow ensures

- Equivalence between TLM and respective RTL design
- Productivity, as an existing proven methodology is re-used, that creates modular and complex tests faster
- Efficiency as a standard and comprehensive verification methodology is used

REFERENCES/ACKNOWLEDGEMENTS

[1] Sandeep Kumar, Synopsys

[2] www.testbench.com

[3] www.vmmcentral.com