iJETRM
International Journal of Engineering Technology Research & Management

A REVIEW OF EVOLUTION OF MODERN TESTBENCH ENVIRONMENT FOR VERIFICATION USING SV/UVM

Amulya M S[*1]
Sujatha Hiremath[2]
*[1]First Author (Department of Electronics and Communication Engineering, RV College of Engineering)
[2]Second Author (Department of Electronics and Communication Engineering, RV College of Engineering)

ABSTRACT

In the recent years, there has been an exponential growth in design and complexity. The time taken to develop products and deploy them in the market has become a major parameter which gives a competitive edge to any developer. In the VLSI Design flow, Verification takes up almost 70% of the time and is the key to reducing time to market and improving products. Testbench environment established for verification purpose has been significantly improved over the past few years. The emergence of System Verilog / Universal Verification Methodology (SV/UVM) has played a major role in this process. UVM ensures well defined standard protocols including parallel bus schemes such as AMBA, memory and communication interfaces such as Ethernet can be implemented with ease and interoperability. This paper discusses the evolution of the testbench environment with the rise of UVM. The modern testbench is designed to ensure ease of use for the designers and inter communication between UVM components in a standard framework.

Keywords:
Universal Verification Methodology, System Verilog, SoC Verification, IP Verification, Device Under Test

INTRODUCTION

Verification has become a significant bottleneck in any modern VLSI complex design flow. Improving the verification efficiency in terms of time , quality and cost has become of paramount importance. The time to market is also mainly driven by the verification cycle System-on-Chip(SoC) is a major level for verification. [1]
It is typically divided into smaller pieces with respect to design hierarchy to make the process of verification easier and more light. These pieces constitute the Intellectual Properties(IPs)[3][7] . The SoC Verification provide a lot of challenges like:

- Finding all Bugs early in design life cycle
- Increasing complexity in design
- Limited Time
- Limited Person and Compute Resources

A tremendous effort is spent on SoC verification and learning details inside IP design, creating complex testbench and testcase for every IP, and debugging the internal behavior of IP.
Analyzing the testbench environment setup in SoC verification is an ideal feature in principle however challenging in the implementation in reality. This paper discusses the evolution of the SV/UVM concepts which have emerged to support the testbench environment setup for the modern VLSI design

iJETRM
International Journal of Engineering Technology Research & Management

SYSTEM VERILOG AND UNIVERSAL VERIFICATION METHODOLOGY

Widespread adoption of best practice verification techniques is not supported by System Verilog alone. Hence the need to develop a suitable methodology to support and ensure interoperability and reusability emerged. This was satisfied by the rise of Universal Verification Methodology. [2] SV/UVM has gradually dominated the verification landscape.

UVM BASICS

A joint effort by various vendors to develop a Base Class Library (BCL) and methodology has led to the development of UVM for interoperability. [2] It is based on System Verilog hardware verification language and is a well established verification approach for IP and SoC.[3][6,7,8]

It uses System Verilog for creating components and TLM (Transaction Level Modeling) for interconnects between components.

Levels of UVM Verification can extend from large to small systems as shown in Fig 1.[5]

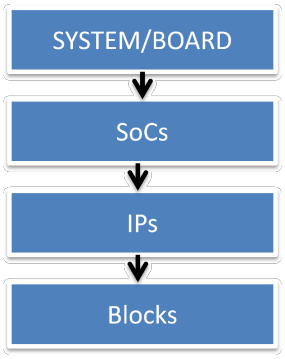

Fig 1: Levels of UVM Verification

Advantages

- Provides a comprehensive BCL supporting construction and deployment of VCs and testbench
- Reduces user's coding effort dramatically. The verification cycle is shorted with minimum people resources and efforts
- Reusability is maximized ensuring the reduction in SoC verification complexity and test debugging problems.
- Separates tests from testbench , making multi-master and multi-slave systems verification
- Enforces certain aspects of interoperability automatically.

Base Class Libraries are ideally specified to ensure:[9-12]

- Users and third parties can develop VCs to provide a common set of architecture and interconnect standards.
- Extensive documentation of the architecture conventions and important aspects of best practice enable domain-specific skills to be ported from one project to another for the engineers exists.
- Faster sharing of expertise and proven techniques has been catalyzed among user base can be enabled.
- A testable set of minimum requirements are established to be met for any practically useful implementation.

The development of UVM methodology and BCLs has led to widespread adoption of System Verilog.

iJETRM
International Journal of Engineering Technology Research & Management

SYSTEM VERILOG BASICS

The main aim of System Verilog is to be sufficiently expressive enough to model digital systems and different levels of abstraction from functional model level to netlist level .It provides Object-Oriented Programming capabilities to support various needs of verification and modeling. [2] [5][13-14]

Advantages

- It supports reuse of code across users and projects.
- It also interfaces well with existing infrastructure since it is developed and maintained by more than one vendor. An active market exists for Verification Intellectual Property to support this valuable feature. The reusable components are called Verification Components (VCs). These VCs must be interoperable in a user's environment.
- It also has a rich library set of implementable code for common recurring problems such as I/O operations, math functions etc.
- It has C-like control constructs but VHDL like package and import features.

UVM Testbench Specifications

There are two main levels of UVM verification, namely IP-level and SoC-level.
It enables reuse of complex IPs verification environment in even more complex SoC design.
SoC complexity grows by integration of multiple IPs and inter-IP-communication among IPs on chip from other vendors/ development teams.[3] It requires diverse and extensive verification due to factors such as large design size, variety of verification views.
The main issues faced currently[3] are:

- Larger verification scope: Increase in SOC verification scope has been done to verify integration of IPs and the inter-IP-communication among the different IPs besides verification of the RTL design module inside the SoC.
- Complexity of low power verification: It becomes complicated to cover every possible power mode transitions, shutoff conditions and isolation control based on power intent in all IPs and SoC.
- Difficult simulations with respect to HW-SW co-simulation and Analog Mixed Signal designs within the IP at the system level: It is more difficult when SoC design contains multiple processors.
- IP and SoC requires different knowledge specifications. It is difficult to support debugging test scenarios for IPs in SoCs for IP engineers since they have limited knowledge about the verification environment and top level SoC design. Each IP block is developed by its own team focusing on a particular requirement alone. Hence it is imperative for all features of each IP to be fully verified at IP level so a bug-free IP can be delivered to SoC team.

At Register Transfer level, there are two famous methods to help reduce verification time and accelerate simulation process - hardware acceleration and simulation

A common UVM-based testbench is divided into three major parts[2]:

1. Instantiation of DUT and interfaces for intercommunication between components in main testbench is done by top module
2. The UVM Verification components (UVCs) , virtual sequencer and the register model are all included in the main testbench.
3. The input stimulus based on UVM virtual sequencer is defined in the test scenarios.

iJETRM
International Journal of Engineering Technology Research & Management

UVM testbench constitutes reusable verification components, including a full set of elements for simulating, checking and collecting coverage information for a specific design or protocol [15-21]
Different components such as stimulus, scoreboard, driver, coverage, sequencer and monitors constitute the UVM Verification Environment as shown in Fig.3. The data generated by the master sequencer is sent to the Device Under Test (DUT) by the driver. The data received in the DUT is compared using a scoreboard with the expected data. Sampling of this data and the responses is done by the monitor. Configuration parameters are used to configure these components. All components can be reused. [2][5][7]
Assertions acts as constraints which define the expected and legal behavior when interaction between blocks occur. System Verilog assertions are used to implement complex protocol checks.
Interfaces are established between the testbench and DUT as shown in Fig.2 to provide a strong connection between the two with multiple features.
The DUT is verified by applying these verification components to it [5]

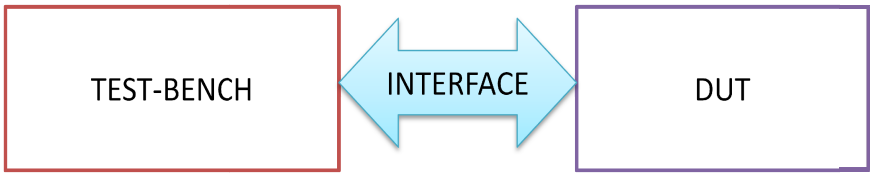

Fig 2: Interface connection between testbench and DUT

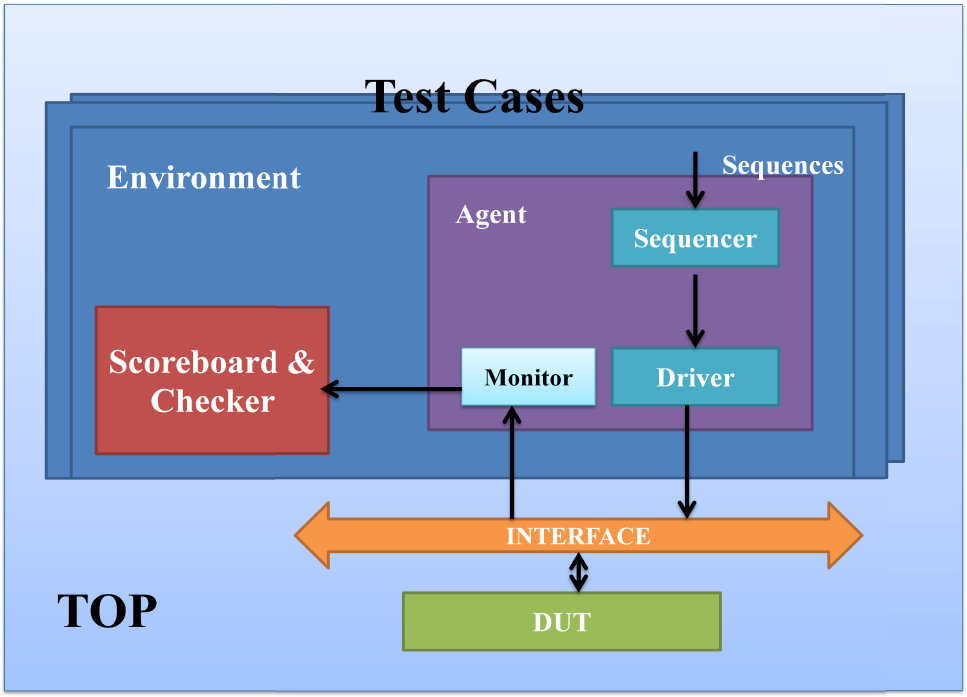

Fig 3: UVM Testbench Block Diagram

iJETRM
International Journal of Engineering Technology Research & Management

METHODOLOGY

The methodology adopted in this paper is an intensive literature survey on the evolution of SV/UVM and their advantages.

CONCLUSIONS

UVM has been developed as rich and capable library which can be utilized for real verification projects, whether large or small with System Verilog as a strong base. It offers powerful features for verification experts but is not fully understood and can be daunting for VHDL or Verilog designers who want to work on Verification. This paper provides an overall picture on the advantages of SV/UVM and how a testbench environment can be brought up with specific focus on the terms used to make it easier to understand the framework. The paper's survey provides a basis for any SV/UVM developer to start with and build upon a foundation.

ACKNOWLEDGEMENT

This research was supported by R V College Of Engineering, Bangalore.

REFERENCES

[1] Salah, Khaled, and Mohamed AbdelSalam, *Smart auto-correction methodology using assertions and dynamic partial reconfiguration,* 2017 24th IEEE International Conference on Electronics, Circuits and Systems (ICECS), 50-53. 2017.

[2] Ni, Wei, and Jichun Zhang, *Research of reusability based on UVM verification*, 2015 IEEE 11th International Conference on ASIC (ASICON), IEEE, 2015.

[3] Salah, Khaled, and Hassan Mostafa, *Constructing Effective UVM Testbench for DRAM Memory Controllers*, 2018 New Generation of CAS (NGCAS),IEEE, 178-181., 2018.

[4] Zhaohui, Hu, Arnaud Pierres, Hu Shiqing, Chen Fang, Philippe Royannez, Eng Pek See, and Yean Ling Hoon, *Practical and efficient SOC verification flow by reusing IP testcase and testbench*,2012 International SoC Design Conference (ISOCC), IEEE, 175-178., 2012.

[5] Shekhar Padhee, Subhranshu, *Verification of Analog and Mixed Signal IPS using SV-UVM Verification Methodology*, PhD diss., 2019.

[6] Salah, Khaled, *A UVM-based smart functional verification platform: Concepts, pros, cons, and opportunities,* 2014 9th International Design and Test Symposium (IDT), IEEE,94-99, 2014.

[7] Raghuvanshi, Siddharth, and Viswajeet Singh, *Review on universal verification methodology (uvm) concepts for functional verification,* International Journal of Electrical, Electronics and Data Communication 2(3) ,2014.

[8] Salah, Khaled.,A Unified UVM Architecture for Flash-Based Memory. *2017 18th International Workshop on Microprocessor and SOC Test and Verification (MTV)*, IEEE, 1-4., 2017.

iJETRM
International Journal of Engineering Technology Research & Management

[9] Sutherland, Stuart, and Don Mills, *Synthesizing systemverilog busting the myth that systemverilog is only for verification,* SNUG Silicon Valley : 24,2013

[10] IEEE Std 1800-2009 ,*IEEE Standard for System Verilog- Unified Hardware Design, Specification, and Verification Language,* http://dx.doi.org/10.1109/IEEESTD.2009.5354441.

[11] UVM 1.0 EA Class Reference, http://www.accellera.org/activities/vip

[12] Bromley, Jonathan, *If SystemVerilog is so good, why do we need the UVM? Sharing responsibilities between libraries and the core language,* Proceedings of the 2013 Forum on specification and Design Languages (FDL),IEEE, 1-7, 2013.

[13] Yun, Young-Nam, Jae-Beom Kim, Nam-Do Kim, and Byeong Min, *Beyond UVM for practical SoC verification*, 2011 International SoC Design Conference, IEEE,158-162., 2011.

[14] Kastelan, Ivan, and Zoran Krajacevic, *Synthesizable SystemVerilog Assertions as a Methodology for SoC*, 2009 First IEEE Eastern European Conference on the Engineering of Computer Based Systems, IEEE, pp. 120-127., 2009.

[15] Mehta, Ashok B., IEEE-1800-2009 Features, *SystemVerilog Assertions and Functional Coverage,* . Springer, New York, NY, 215-253, 2014.

[16] Francesconi, Juan, J. Agustin Rodriguez, and Pedro M. Julian, *UVM based testbench architecture for unit verification,* 2014 Argentine Conference on Micro-Nanoelectronics, Technology and Applications (EAMTA), IEEE,. 89-94 , 2014

[17] Kotha, Srivasthav, Rajin Ravimony, and N. Mohankumar, *Automated UVM Based Verification of Device Life Cycle Management IP,* International Conference on Intelligent Computing, Information and Control Systems, Springer, Cham,574-583, 2019.

[18] EV, Jaideep Varier, and Karthigha Balamurugan, *Design of Generic Verification Procedure for IIC Protocol in UVM,* 2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA), IEEE, 1146-1150, 2019.

[19] Chakravarthi, Veena S, SOC Design Verification, *A Practical Approach to VLSI System on Chip (SoC) Design*,. Springer, Cham, 141-172, 2020.

[20] Mefenza, Michael, Franck Yonga, and Christophe Bobda., *Automatic UVM environment generation for assertion-based and functional verification of systemc designs,* 2014 15th International Microprocessor Test and Verification Workshop, IEEE, 16-21, 2014.

[21] Vijayan, U., C. A. Anjo, B. Vignesh Raja, and N. Arun Kumar, *Development of basic template environment for functional verification of VLSI design using UVM,* International Journal of Emerging Technology and Advanced Engineering 3(12) ,2013.

[22] Yashas, D., P. Sri Hari Babu, and N. Shylashree., *UVM-based Logic Verification of Input Output Interface*, 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT), IEEE, 420-423, 2019.