**Novateur Publication's**
**International Journal of Innovation in Engineering, Research and Technology [IJIERT]**
**ICITDCEME'15 Conference Proceedings**
**ISSN No - 2394-3696**

# VERIFICATION APPROACH USING UVM

Pankaj Vitankar
Department of E & TC (VLSI and Embedded System),
University of Pune, VACOEA,Ahmednagar, India

Dr. A.K.Kureshi
Department of E & TC (VLSI and Embedded System),
University of Pune, VACOEA,Ahmednagar, India

## ABSTRACT

Verification is one of the most important activity in the flow of ASIC/VLSI design. Verification consumes large amount of design flow cycle & efforts to ensure design is bug free. Hence it becomes intense requirement for powerful and reusable methodology for verification. The Universal Verification Methodology (UVM) is a powerful verification methodology that was architected to be able to verify a wide range of design sizes and design types.
UVM is derived from other methodology like VMM, OVM, eRM. It is useful to verify designs in any language like verilog, VHDL, System Verilog. Reusable verification environment is possible using UVM & hence saving considerable time in Verification cycle. This paper talks about the architecture of environment using UVM. It also focuses on terms & ways used in Verification using UVM.

## INTRODUCTION

The Universal Verification Methodology, commonly referred to as UVM, is designed to have verification of ASIC, full custom. It is also helpful to verify FPGA based designs.
All the constructs and capabilities of UVM may not be useful to one project rather it depends on the project, hence the name Universal. UVM base class libraries provide the common platform for verification engineer to develop complex test bench. There are often multiple classes, methods or macros those are basic constructs to develop test bench.
UVM Class Reference Manual documents all of these classes but the manual does not make it clear which classes are intended for end users of UVM to use in a UVM testbench, and which classes are intended for internal use within the UVM methodology.
The goal of this paper is learning and adopting UVM. Paper also suggests architecture of test bench using preferred classes for developer.

## LITERATURE SURVEY

Verification is important part of VLSI Design. History shows failures of chips due to lack of proper verification strategy. One such example is Nokia. After the bug found in one of their model, the complete product was out of use. The statistics also shows that around 70% of total time for chip specs to manufacture process is required for verification. Due to increasing need of reducing time to market, time required for verification need to reduce.
Reduction in time for verification can happen through reusability. That means the stuffs already developed for verification should be reused for current scope of verification. Maximum is the reusability minimum time required for verification.
There are many methods suggested by different EDA vendors. Some of such well known methodologies include eRM (e Reusable Methodology), introduced by Verisity & later adopted by Cadence. eRM requires knowledge of e language & Spec man tool. Another methodologies like OVM (Open Verification Methodology) and VMM are introduced by Synopsis.
There become need to have tool independent methodology & should be universal across. This gives the UVM (Universal Verification Methodology) which supported by Cadence, Mentor Graphics as well as Synopsis. The UVM becomes interesting due to tool independent. The base code of UVM is System Verilog.

## SCOPE & SYSTEM DESIGN

The scope of this paper is approach to develop verification environment using UVM. The required components definitions, there use & to reach to test bench development.

## UVM TEST BENCH ARCHITECTURE

UVM libraries are huge set of classes, macros. It is the first step for verification engineer to select proper classes for his/her verification components development. Once selected proper classes, verification engineer need to implement those components as per the requirement.
Figure 1 show typical architecture and suggests UVM classes to be used for the development of those classes.

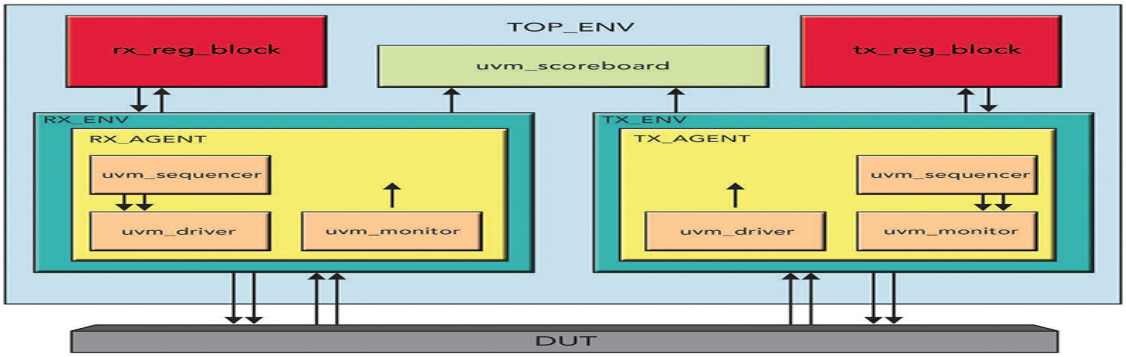


**Figure 1 Typical UVM test bench**

All complex test benches may be architected as shown in the figure with little or more modification depending on project.
As shown in the figure the environment should be instantiated in the test case. This makes easy for configuration of environment with respect to test case.
The environment may consist of one or more agents depending on interface protocol supported by DUT. All agents have similar architecture and consist of one or all of monitor, driver, sequencer and sequences.
UVM scoreboard is used for data checking. *reg block is used for SFRs programming.
This paper does not details of reg block but provides reference for the same.
Paper is intended to cover details of architecture, classes to be used.

## RESULT

With UVM test bench, functional coverage can be achieved. Typical UVM test bench with coverage definition & test cases are the main outcome of this effort. The showcase of waveform & functional coverage numbers to indicate its importance in verification field

## REFERENCES

[1] Universal Verification Methodology (UVM) 1.1 User's by accellera
[2] Universal Verification Methodology (UVM) 1.2 Class by accellera
[3] Rich Edelman Mentor Graphics Fremont, CA Shashi Bhutada Mentor Graphics, Los Angeles, CA
"An approach to automating UVM test bench writing".
[4] Mark Litterick Jason Sprott Jonathan Bromley (Vanessa Cooper) "Advanced UVM in the real world"