International Journal of Electrical and Computer Engineering (IJECE)
Vol. 7, No. 6, December 2017, pp. 3318~3322
ISSN: 2088-8708, DOI: 10.11591/ijece.v7i6.pp3318-3322

A Unique Test Bench for Various System-on-a-Chip

Sridevi Chitti[1], P. Chandrasekhar[2], M. Asharani[3]
[1]Department of Electronics and Communication Engineering, SR Engineering College, India
[2]Department of Electronics and Communication Engineering, OU-Hyderabad, India
[3]Department of Electronics and Communication Engineering, JNTU-Hyderabad, India

Article Info

Article history:

Received Mar 25, 2017
Revised Jun 23, 2017
Accepted Jul 15, 2017

Keyword:

Environmen
IP
Reuse
SOC
Testbench
Verification methodology
VIP

ABSTRACT

This paper discusses a standard flow on how an automated test bench environment which is randomized with constraints can verify a SOC efficiently for its functionality and coverage. Today, in the time of multimillion gate ASICs, reusable intellectual property (IP), and system-on-a-chip (SoC) designs, verification consumes about 70 % of the design effort. Automation means a machine completes a task autonomously, quicker and with predictable results. Automation requires standard processes with well-defined inputs and outputs. By using this efficient methodology it is possible to provide a general purpose automation solution for verification, given today's technology. Tools automating various portions of the verification process are being introduced. Here, we have Communication based SOC The content of the paper discusses about the methodology used to verify such a SOC-based environment. Cadence Efficient Verification Methodology libraries are explored for the solution of this problem. We can take this as a state of art approach in verifying SOC environments. The goal of this paper is to emphasize the unique testbench for different SOC using Efficient Verification Constructs implemented in system verilog for SOC verification.

Corresponding Author:

Sridevi Chitti,
Department of Electronics and Communication Engineering,
SR Engineering College,
Anathasagar, Hasanparthy, Warangal, India.
Email: srideviredddy.arram@gmail.com

1. INTRODUCTION

This methodology is an open source SystemVerilog library allowing establishment of flexible, reusable verification components and assembling great test environments utilizing constrained random stimulus generation and functional coverage methodologies.. Its main promise is to progress testbench reuse, make verification code handier and create new market for universal, high-quality Verification IP (Intellectual Property). Major advantages of efficient Methodology: Other better aspects are end-of-test objection handling using phases, updated phase methods, command line processor, and config_db changes.

a. Callbacks were updated with the following additional functionality:
   - a callback iterator class type wide callback support (instead of just instance specific) callback type registration for type checking.
   - added add and delete callback by name.
   - more callback tracing was added.
b. Objections were restructured with the following additional functionality:
   - added a string description for raise/drop.
   - added ability to add external callbacks to objections.

A report catcher callback was added to allow users to manage messages via an external callback. The catcher uses the standard callback mechanism on uvm_report_objects so that callbacks can be added

type wide or to a specific report object. Using these advantages constructed a reusable testbench for verification of various SoC's [1]. Anyone looking at the code will be able to understand how reports are controlled and generated, and it will be the same from project to project.

2. DEVELOPMENT OF TESTBENCH

2.1. Verification Environment

The normal process for developing a verification environment is bottom-up. Blocks are initially verified in block-level environments, and then the integration of the blocks into SoC is verified by chip-level testbench [2]. Refers to this methodology as IP-centric methodology because the blocks are considered IPs [3]. The focus of block-level verification is to verify the blocks systematically, while the chip-level verification is for checking the integration of the blocks and the correctness of application scenarios. A bottom-up verification approach has several benefits:

- Localization of bugs: finding bugs without difficulty
- Easier to analysis all the block modes at the block-level
- Confidence in the block-level allowing them to be reused in a number of projects.

In this section we depict the recommended ordering for development of verification environment elements. Such ordering must be in wits when developing executable verification plans.

- Interfaces
- Agents: Driver, Monitor, Sequencer, and sequences
- Block level: configuration, Virtual sequencer, Virtual sequences, coverage model, constrained random sequences, scoreboard.

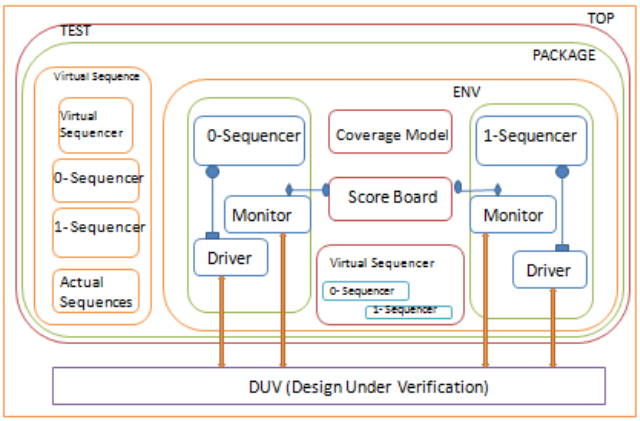

Figure 1. A unique Test Bench for SoC

2.2. Qualify VIP's

Qualify the VIP throughout development and before releasing them. First, several tools can conduct static checking on VIP components for conformance to coding styles and common errors. They can also give statistics about the size of code, cover groups and checks. Secondly, typically a simulator can provide information about memory consumption and performance bottlenecks of VIP [4]. Third, VIPs are robust to user mistakes whether in connections or proper use. Developed sanity checks that can flag early a user error. Finally, peer review is still helpful to point-out issues that are missed in the other steps.

2.3. Better Regression Management

The usual scripts that compile and run testcases come short when running multifaceted UVM SoC verification environment. Typical desires on run management is to keep track of seeds, log files of different tests, flexibility of running different groups of tests, execution time, and running on local machine or grid. Once a regression is run we finish up with data that desires to be processed to come out for useful information such as which tests passed/failed, frequent failure messages, which tests were more efficient and which seeds produced improved coverage.

2.4. Environment Reuse

Environments should be self-reliant having only familiarity about its components and universal elements and can communicate only through configuration mechanism, global events or TLM connections such as reset event. Following these rules, an environment at the block-level can be reused at the chip-level building the chip-level environment the integration of block-level environments [5]-[7].

2.5. Sequence Reuse

It is important to write down sequences with study on reusing them. In this verification methodology, there are two types of sequences: sequence which sends transactions and sequences that starts sequences on sequencers. The latter is called a virtual sequence [8].

Although goals are dissimilar between block and chip level testing, some virtual sequences from block-level can be reused at chip-level as integration tests. Interfaces that become internal at the chip-level can be regularly stimulated through some external interface. In order to construct the last type of virtual sequences reusable at chip-level, it is better to plan ahead to abstract the data from the protocol. Using this efficient verification test bench, the sequences can be reusable. Use functional abstraction by defining functions in the virtual sequence that can be overridden like:

write(register_name, value);
read(register_name, value);

2.6. Scoreboard

A critical component of self-checking testbenches is the scoreboard that is accountable for checking data integrity from input to output. A scoreboard is a TLM component [9], care should be taken not make active on a cycle by cycle basis but rather at the transaction level. In efficient verification methodology, the scoreboard is usually connected to at least 2 analysis ports one from the monitors on the input(s) side and the another on the output(s) Figure 2 depicts these connections. A Scoreboard operation can be summarized in the following equations:

Expected = TF(Input Transaction);
Compare(Actual , Expected);
TF : Transfer function representing the DUT functionality from inputs to outputs.

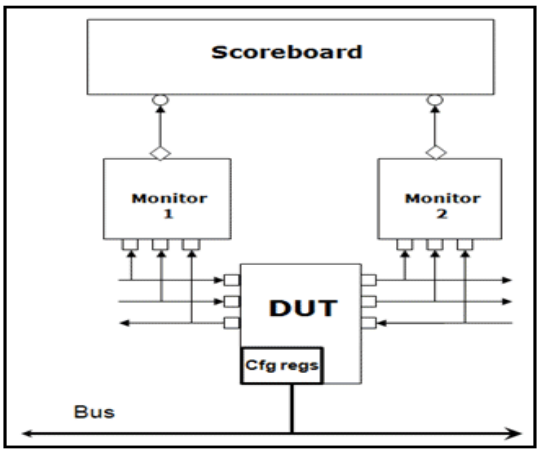

Figure 2. Diagram of Scoreboard

Sometimes the operation is described as comparator-predictor. Where the comparator checks the actual versus predicted (compare function) and the predictor computes the next output (transfer function). Usually the transfer function is not static but can modify depending on the configuration of the devices.

3. A BASIC COMMUNICATION SOC

3.1. Blocks description

In the below figure The I2C-bus is a Two-wire, half-duplex data link invented and specified by Philips (now NXP). The two lines of the I2C-bus, SCL and SDA, are bi-directional and open-drain, pulled up

by resistors. SDA is a Serial Data line and SCL is a Serial Clock line. Devices on the bus drag a line to ground to send a logical zero and release a line to send a logical one.

Serial peripheral interfaces (SPI) are commonly used to provide economical board level interfaces between various devices such as Digital to Analog Converter's, microcontrollers, Analog to Digital Converter's and other. Many IC's manufacturers manufacture components that are compatible with SPI. Serial communication is the process of transfer data one bit at a time, sequentially, over a link. A serial connection requires smaller amount of interconnecting cables (e.g., wires/fibers) and hence occupies less space. For high performance systems, FPGAs also uses SPI to interface as a output to a host, as a input to sensors.

The Universal Serial Bus is a serial bus standard to interface devices. First designed to allow connections to the PC without expansion cards, the USB became a actual communication standard for approximately all electronic devices. The USB communication is for all time initiated by a Host and responded by a Device. Universal Serial hubs act as switches to expand the number of devices per Host. Special On the- Go devices can act as either Device or host and can change roles while connected to other OTG Devices

The Master and Slave AMBA® AXI VIP (Advanced eXtensible Interface) is a extremely flexible and configurable verification IP that can be simply integrated into any SOC verification The I2S bus (Inter-IC Sound bus) is a 3-wire, half-duplex serial link for linking digital audio devices in an electronic system. The bus handles audio data and clocks independently to minimize jitter that may cause data distortion in the digital analog system. I2S bus is widely used by equipment and IC manufacturers. This document describes the implementation of I2S Controller. The Universal Asynchronous Receiver/Transmitter is a highly flexible and configurable verification IP that can be simply integrated into any SOC verification environment [10]-[13].

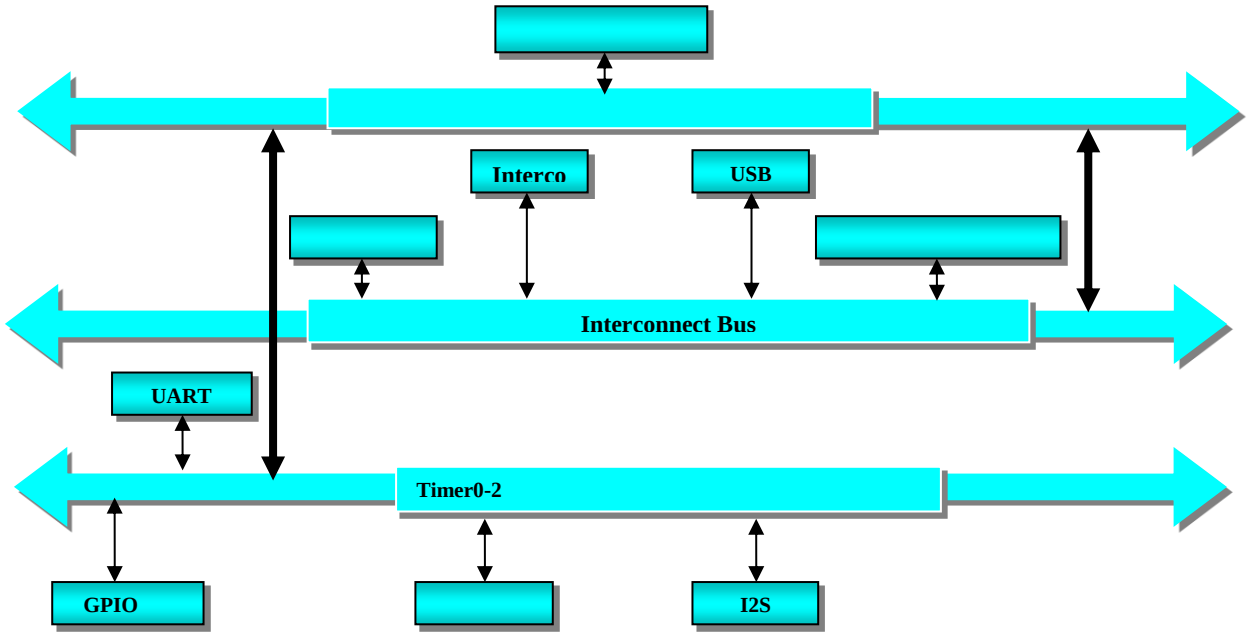

Figure 3. Basic communication SOC

4. RESULTS

Figure 4 and 5 shows the comparison between a Testbench using Open Verification Methodology and a Unique Testbench using Efficient Verification Methodology from the figure Unique testbench takes the minimum time for simulation with comparison to the other one. Unique testbench using Efficient Verification Methodology is more competent to complete verification within a short time. Figure 6 shows the simulation waveform of SOC has been carried out using Efficient Verification Methodology. From this waveform, conclude transmission and reception of data through the design under verification (DUV).

```
All tests completed successfully
$finish at simulation time 4.68 s

          NC-SIM   S i m u l a t i o n   R e p o r t
Time: 4.68 s
CPU Time:      4.12 seconds;       Data structure size:   0.0Mb
Fri Dec 16 10:54:24 2016
```

Figure 4. Simulation Report using Normal Testbench

```
All tests completed successfully
$finish at simulation time 1.02 s

          NC-SIM   S i m u l a t i o n   R e p o r t
Time: 1.02 s
CPU Time:      4.01 seconds;       Data structure size:   0.0Mb
Fri Dec 16 11:28:34 2016
```

Figure 5. Simulation Report using Unique Testbench

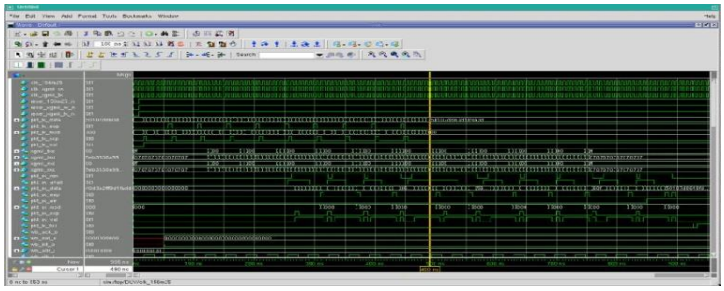

Figure 6. Simulation Results

5. CONCLUSION

The specifications of System on Chip (SoC) are verified successfully using Efficient methodology on Cadence simulator. The verification methodology helped in performance improvement and simulation time reduction in SoC verification time from previous methodology (i.e 4.68 sec to 1.08sec). Verification is very crucial to find bugs which only appear with random stimulus. The constrained random approach is more time-efficient for reaching coverage goal compared to directed test method.

REFERENCES

[1] S. K. Mohanty, *et al.*, "Test bench Automation to overcome verification challenge of SOC Interconnect," *2015 International Conference on Man and Machine Interfacing (MAMI)*, Bhubaneswar, pp. 1-4, 2015.

[2] V. S. Rashmi, *et al.*, "A methodology to reuse random IP stimuli in an SoC functional verification environment," *2015 19th International Symposium on VLSI Design and Test*, Ahmedabad, pp. 1-5, 2015.

[3] M. Mefenza, *et al.*, "Automatic UVM Environment Generation for Assertion-Based and Functional Verification of SystemC Designs," *2014 15th International Microprocessor Test and Verification Workshop*, Austin, TX, pp.16-21, 2014.

[4] F. Haedicke, *et al.*, "CRAVE: An advanced constrained random verification environment for SystemC," *2012 International Symposium on System on Chip (SoC)*, Tampere, pp. 1-7, 2012.

[5] J. Bergeron, "Writing Testbenches Using SystemVerilog," *Springer*, Business Media, 2006.

[6] www.design-reuse.com/articles/15351/complex-soc-verification-using-arm-processor.html

[7] Accellera, UVM 1.1 Reference Manual, 2011

[8] S. Rosenberg and K. A. Meade, "A Practical Guide to Adopting the Universal Verification Methodology (UVM)," Cadence Desgin Systems, 2010.

[9] UVM 1.1 kit - includes UVM base class libraries - Free downloads -www.uvmworld.org

[10] Accellera Organization Inc Verification Intellectual Property Technical Subcommittee.

[11] http://www.accellera.org/activities/committees/vip

[12] Y. N. Yun, "Beyond UVM for Practical SoC Verification," *ISOCC*, 2011.

[13] "Generic System verilog UVM based reusable verification environment for efficient verification of Image signal processing IP/SOCs," *International journal of VLSI design &communication systems(VLSICS)*, vol/issue: 3(6), 2012.