2024
DESIGN AND VERIFICATION™
DVCON
CONFERENCE AND EXHIBITION
INDIA

Optimized Technique for Implementation of IOL Test-Suite

Vatsal Jain
Krishna Raman Singh
Anmol Kathuria
SIEMENS EDA

***Abstract*-UNH-IOL test suite's growing popularity and their adoption as an industry standard to test a design unit, leads to their implementation in many possible ways. The IOL Test Suites are provided by nearly all VIP providers and are often very complex to integrate with the design unit. The research highlights the optimized way of test bench creation and stimulus generation primarily by using proposed host-line model.**

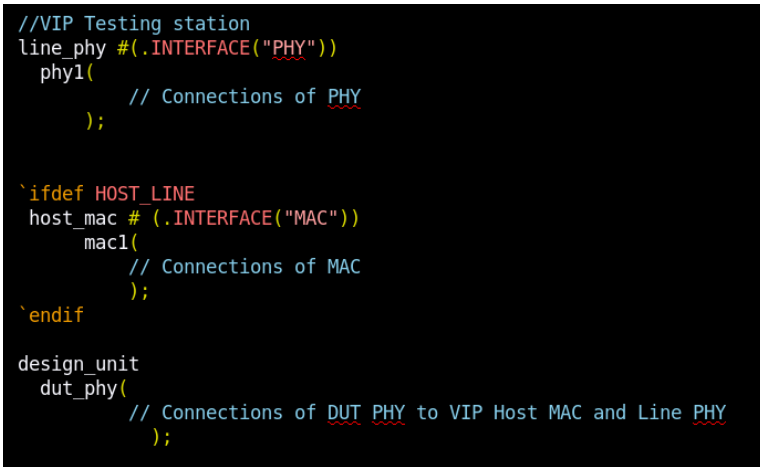
The proposed model provides easy plug and test mechanism to the design tester and ensure that the Verification Engineer focus only about the behavior of DUT and does not spend his time on stimulus generation. Proposed method also gives user freedom to modify certain parameters of the sequences using command line arguments which helps them to take care of their DUT's boundary condition.

I. Introduction

The UNH InterOperability Laboratory (UNH-IOL) is a research and testing facility that plays a significant role in advancing technology standards and interoperability. Nowadays, with growing popularity, UNH-IOL test suite’s are being adopted as an industry standard to verify the design unit. IOL Test Suites are provided by almost all Verification IP vendors and most of the times it is quite difficult to integrate it with the design unit. After successful integration Verification Engineer have to spend their time on stimulus generation, creation of API’s for traffic generation and reception and adding various Design level checks which is often very time consuming task.

This Paper describes the optimized way of test bench creation and stimulus generation mainly by using three proposed model namely:

A. *Host Line Model*: In this model, Verification IP present at Host side generates the stimulus required for design unit to drive, and Verification IP present at Line side will be used for reception of the traffic from the design unit and necessary checking condition will be taken care.



```
//VIP Testing station
line_phy #(.INTERFACE("PHY"))
  phy1(
          // Connections of PHY
      );


`ifdef HOST_LINE
 host_mac # (.INTERFACE("MAC"))
      mac1(
          // Connections of MAC
          );
`endif

design_unit
  dut_phy(
          // Connections of DUT PHY to VIP Host MAC and Line PHY
            );
```

Figure 1. Example of Command Line Argument enabling Host-Line model consisting of two instances of VIP

It ensures that the user need not worry about the stimulus generation at higher level and if users design unit supports only a particular layer in the OSI model the stimulus will be generated by the help of Host side VIP. For example [3], in case of Ethernet Protocol 10G BASER, if Design Unit supports Physical Layer only and need to send Idles at MAC Level, then it will be very cumbersome for the user to generate stimulus. The proposed method consists of plug-in test mechanisms where the user needs to provide command line argument to Transmit configured Idles at MAC Level. The automated testing will be done with the help of checks provided by the VIP provider which will be outcome/interpolation of the traffic received (Via Design unit).

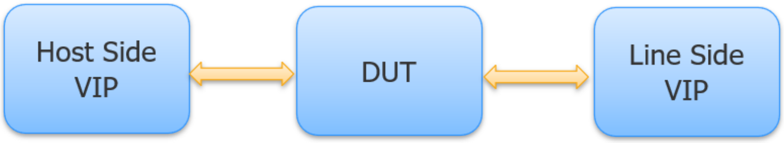
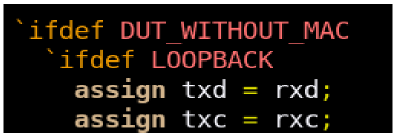


Figure 2. Host Line Model

B. *Loopback Model*: In this model, the traffic received by the design unit from the Verification IP will be loopback to the Verification IP and the behavior of the Design unit will be checked at Verification IP side taking observations defined in UNH IOL as benchmark. All stimulus generation and checks will be handled by Verification IP in this case.



```
`ifdef DUT_WITHOUT_MAC
  `ifdef LOOPBACK
    assign txd = rxd;
    assign txc = rxc;
```

Figure 3. Example of Command Line Argument enabling Loopback mode

For example [3], in case of Ethernet Protocol 10G BASER, if proper encoding/decoding of Idle control character by design unit needs to be checked, the proposed model will add checks at Verification IP end instead of asking developer to test the design unit. This is done by observing received traffic at Verification IP.

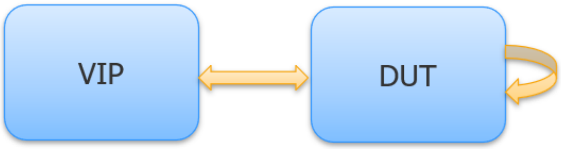


Figure 3. Loopback Model

C. *DUT-VIP Model*: In this model, two-way communication between Verification IP and Design unit takes place. It is assumed that Design Unit consists of all the necessary layers required for testing UNH IOL [1]. The proposed method emphasizes on traffic generation and checking mechanism on Verification IP side.

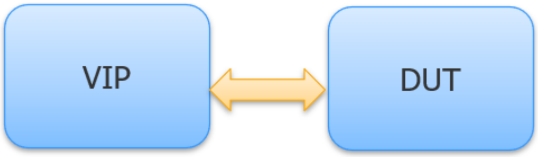
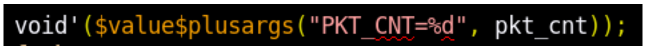


Figure 4. Design Unit-VIP Model

The Proposed models have the ability to standardize the implementation of IOL Test Suites and the Verification Engineer will have the ease of testing their design according to the proposed methodologies. The method will involve test bench creation on the basis of various command line arguments which will provide the developer freedom to choose between either of the three proposed methodologies with an added advantage of manipulating all the fixed value checks and making them user configurable so as to take into consideration various Design Unit delays/limitations which were earlier hard coded and leads to the false failing of various testing scenarios.

For Example: The command line argument described in Fig.5 denotes the no of packets received by DUT after Link-up. Sometimes due to Phy delays and Design Limitations it is possible that all the packets instructed in UNH-IOL [1] are not received thus to avoid false failing command line argument can be configured instead of hard-coding the checks.



```
void'($value$plusargs("PKT_CNT=%d", pkt_cnt));
```

Figure 5. Example Command Line Argument to take Design Unit Limitations Into Consideration

II. TESTBENCH SETUP

In the testbench top all the connections will be made in accordance to defines providing user flexibility to test their design in desired proposed methodology (i.e. Host Line Model, Loopback Model and Design Unit-VIP Model). Similarly, the environment will use the defines and based on the connections made by the user testbench will be instantiated. There will be one common file for design level checks which user can modify if he wishes to override the automatic checks provided, giving the user power to take care of the design limitations.

User will be getting switch in the proposed model and any of the testing sequence could be gated(if in case a particular sequence/scenario is not supported by Design unit) so as to make the verification process smooth and get a clear regression result.

The proposed method uses checker class:

- This class will have API's to configure/drive traffic from Design Unit side.
- Developer needs to call API's as per their requirements.
- Sequence/Test will have handle of checker class and in sequence/test API's can be call through this handle as per test requirement.

The proposed model will automate the process and will help the Verification engineer to test their design faster and in an efficient manner. It will take care of all the design boundary conditions with the help of two way communication, defines arguments and command line arguments. They will allow user to test sequence just with one command and tell if the design is acting in accordance to proposed observation or not.

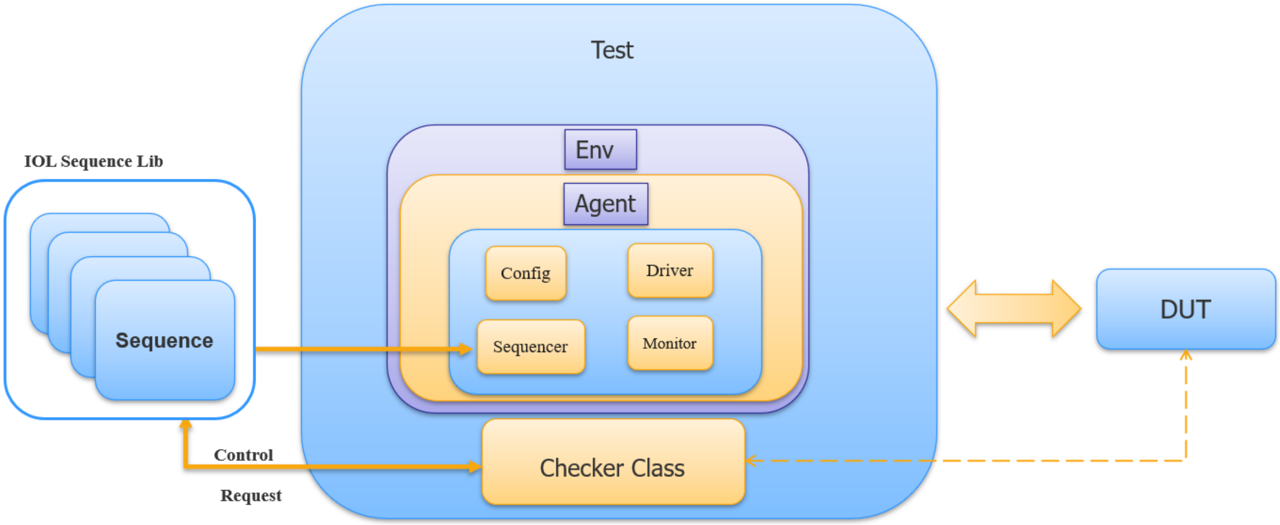


Figure 4. Test Bench Architecture

III. APPLICATION

The Proposed Methodology provides easy plug and test mechanism and provides Verification Engineer upper edge since stimulus is generated automatically and user configuration values provided with command line arguments which will take care of design limitations, skew considerations and PHY delays. This method ensure that the developer need not write Design level checks since that will be taken care at Verification IP end. This will help Verification Engineer to check there Design relatively faster.

The Proposed model gives freedom to choose between three models and the same set of sequences can be used in all three models. If the Verification Engineer wishes to write his own DUT checks and stimulus generation model he has that freedom as well and that can be done using command line arguments defined.

Siemens VIP is designed in such a way that all the proposed methodologies are integrated into our UNH-IOL provided to customers and the proposed model are evaluated for various Ethernet [2] (MAC/PHY Interfaces).

IV. SUMMARY

The Proposed research highlights mainly:

- Optimized Stimulus generation.
- Superior Testbench creation (Host Line, Loopback, DUT-VIP).
- Automatic Testing in accordance with IOL Compliance Test Suite.
- User Defined Command line arguments for design limitations.
- Standardizing IOL Compliance Test Suite testing

The method automates the IOL testing and total Design Unit testing time.

REFERENCES

[1] https://www.iol.unh.edu/

[2] IEEE Standard 802.3-2022, "IEEE Standard for Ethernet," IEEE, 2022.

[3] 10GBASE-R PCS Test Suite V1.0