A Mixed HDL/PLI Test Toolbox

Nastaran Nemati, Zainalabedin Navabi

Electrical and Computer Engineering Department
Faculty of Engineering – Campus #2 – University of Tehran, 14399 Tehran, IRAN
{nastaran, navabi}@cad.ut.ac.ir

Abstract

Using RT level hardware description languages (HDL) in test and DFT, helps advancing test methods to RTL, and at the same time alleviates the need for use of software languages and reformatting designs for evaluation and application of test techniques. In moving to HDLs and HDL environments for test and testability applications, HDL limitations such as inability of description of complex data structures and procedural programming must be considered. PLI (Procedural Language Interface) provides a library of C language functions that can directly access data within an instantiated Verilog HDL data structure. In this work, by means of the PLI interface, a mixed HDL/PLI test package is proposed.

1. Introduction

Utilizing existing software test tools, a design must be synthesized to gate level before undergoing the test phase. This puts a large overhead on the process of design and test of a digital core. It is desirable to bring testing in the hands of designers, which certainly requires that testing is applied at the level and with the language of the designers. This way, designers will be able to better combine design and test phases. Furthermore, using the same environment for design and test, all components in a large design can be tested independent of other components. We can also test some components while others are being designed. In a mixed level design, these advantages make it possible to test a single component described at the gate level, while leaving other components in RTL or even at the system level.

By means of the procedural language interface (PLI), a test designer can have the advantages of doing testable hardware design in an HDL, and having software tools for manipulation and evaluation of designs. PLI provides the necessary accesses to the internal data structure of the compiled design, so test methods can be performed in a mixed environment more easily and without having to mingle with the original design.

2. PLI Test Toolbox

PLI gives access to the internal data structure of a compiled design for reading and writing net and reg values, and for tracing design modules and ports. Such facilities enable us to make an efficient environment to perform test algorithms without manipulating the HDL designs. Figure 1 shows the general form of implementing and running test programs in such a mixed environment.

3. Preparation Phase

Before using this test toolbox, the HDL code of the design under test must be converted into an input format appropriate for applying test applications. We use an FPGA synthesis tool and conversion program to translate the synthesis output to our required net-list format.

4. PLI Functions

PLI utilities that are useful for developing test applications are provided in our HDL toolbox. These utilities are introduced in the following Subsections.

4.1 Fault Injection

The most important utilities for implementing test algorithms are fault injection (FI) and fault removal (FR). As noted, PLI provides mechanisms for reading and writing net and reg values. Therefore we can force and release values in the data structures corresponding to nets, which gives us capabilities for fault injection and fault removal on and from circuit lines. PLI also enables us to have control on delays and to check changes on net and reg values. Utilizing these capabilities, transient fault injection, coupling and bridging fault injection and removal are implemented in this package.

4.2 Fault Collapsing

Another process needed in many test applications is fault collapsing (FC). Reference [2] discusses a line FC method that is based on gate types that circuit lines are connected to the inputs of. Since PLI allows tracing all design hierarchies down to gate primitives, and allows us to identify primitive types that a line drives, the FC method of [2] is easily implemented by PLI routines. Primitive types, that are a decisive factor in stuck-at fault values of gates, can be looked up with PLI routines.

4.3 Signal Activity Estimator

Since PLI enables tracking changes on nets and registers, by monitoring these changes on all the nets in a design, the signal activity of different modules can be estimated and compared together. This utility can be applied to power estimation, test generation and statistical fault simulation.

4.4 Module Enabler-Disabler

Module-enabler-disabler is another PLI utility that can externally select a module to be enabled or disabled. By using this utility to enable or disable gate level or behavioral level description of modules, hierarchical fault simulation is achieved.

4.5 Test Generation and Evaluation Utilities

Providing useful utilities for deterministic test generation is one of the important features of this test toolbox. A number of these utilities are described below.

4.5.1 Walkers

As mentioned before, PLI provides some routines to trace the design hierarchy. Using such features we have developed functions that can start from an internal node and find all primary inputs feeding the nodes, or find all primary outputs that are fed by nodes. Because these functions walk toward PI or PO we refer to them as *walkers.* By invoking these walkers properly in HDL test-benches, fault activation and fault propagation - which are two important parts of deterministic test generation algorithms -, can be achieved.

4.5.2 Controllability and Observability Measurements

In order for developing efficient test generation algorithms, it is necessary to have good means of measuring the controllability and observability of internal nodes in the design under test. By means of the same routines which are used for implementing walkers, functions capable of estimating these measurements can be employed.

4.5.3 X-Path checker

Several deterministic test generation algorithms like PODEM require checking the existence of x-paths. The mechanism of realizing x-path checker using PLI is also the same as that for walkers.

5. Test Applications

PLI utilities mentioned above are used within HDL test-benches for developing test applications that apply to a design under test.

The most important feature of these test applications is that they are provided in a parametric form. In other words, the required test-benches for developing these test applications for various DUTs can be provided in a semi-automatic way by just setting a number of parameters in the template test-benches.

5.1 Fault Simulation

5.1.1 Serial fault simulation

As soon as the FI and FC means are provided, we will be able to prepare for the PLI serial fault simulation. Since the fault injection and fault removal can be done with no HDL code overhead and no changes in the design core, this environment makes it possible to have a fast serial fault simulation. The run time of serial FS for ISCAS benchmarks is measured for a stand-alone FS, VHDL, and PLI. The PLI results are close to those of the stand-alone fault simulator.

5.1.2 Hierarchical fault simulation

The simulation of behavioral description of modules is much faster than for their gate level. Therefore by including both descriptions for each module, enabling all modules in the behavioral level but the one which is supposed to be tested, the fault simulation process can become faster. As mentioned before, by applying module-enabler-disabler for selecting different abstract level descriptions of modules in a DUT, hierarchical fault simulation is implemented.

5.2 Test Generation

Test generation is one of the most critical algorithms in test domain. Two algorithms are developed in this toolbox for test generation.

5.2.1 Pseudo random Test Generation

Like serial fault simulation, just by invoking fault collapsing, fault injection and fault removal in an HDL test-bench, pseudo random test generation is implemented. The test-bench controls the timing of fault injection, fault removal and fault simulation.

5.2.2 Pseudo deterministic Test Generation

Those test generation utilities which were discussed in Subsection 4.5 are used for advancing deterministic test generation methods. The provided method is similar to the basic PODEM.

5.3 DFT Evaluation

Using PLI, a designer can develop a virtual tester for the circuit being designed with scan chains inserted in it. This virtual tester can be used for evaluation of various DFT implementations. This requires no modifications in the HDL code of the hardware under test and its scan, since the PLI functions will be able to access nodes and values in the CUT.

By means of the PLI utilities that we have developed, and the example templates that we have provided, a design and test engineer can create his or her own test and verification applications.

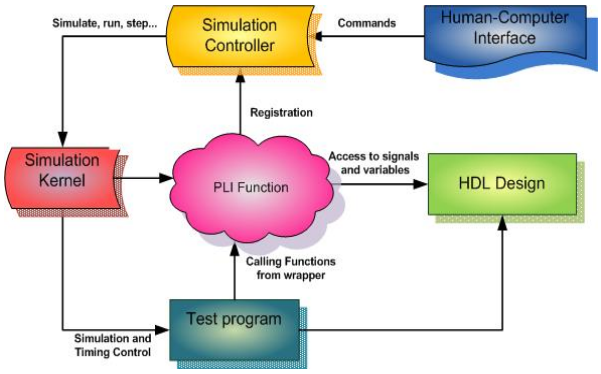

Figure 1. Implementing and running test programs in HDL/PLI

References

[1] C. J. Hesscot, D. C. Ness, and D. J. Lilja, "A Methodology for Stochastic Fault Simulation in VLSI Processor Architectures," In MoBs, 2005.

[2] M. Nadjarbashi, Z. Navabi and M. R. Movahedin, "Line Oriented Structural Equivalence Fault Collapsing," in IEEE Workshop on Model and Test, 2000.

[3] N. Farajipour, S. B. Hosseini and Z. Navabi, "Utilizing HDL Simulation Engines for Accelerating Design and Test Processes," In IEEE Int. East-West Design and Test Symposium , 2008.

[4] P. A. Riahi, Z. Navabi, and F. Lombardi, "Simulating Faults of Combinational IP Core-based SOCs in a PLI Environment," DFT, 2005.

[5] Z. Navabi, "VHDL: Modular Design and Synthesis of Cores and Systems," McGraw Hill, 1998.