# XROACH: A TOOL FOR GENERATION OF EMBEDDED ASSERTIONS

*Márcia C. Marra de Oliveira, José Augusto M. Nacif, Claudionor Nunes Coelho Jr., Antônio Otávio Fernandes*

Computer Science Department, Universidade Federal de Minas Gerais, Brazil.

**ABSTRACT**

White-box verification is a technique that reduces observability problems by locating a failure during design simulation without the need to propagate the failure to the I/O pins. White-box verification in chip level designs is implemented using assertion checkers to ensure the correct behavior of a design. With chip gate counts growing exponentially, today's verification techniques, such as white-box, cannot always ensure a bug free design. This paper presents a tool that synthesizes previous embedded assertion checkers to become part of the released chip. Extending white-box verification techniques to deployed products helps locate errors that were not found during simulation / emulation phases. We present preliminary results of XRoach processing an $I^2C$ communication core.

## 1. INTRODUCTION

Today, increased complexity of chips and reduced time to market has done the verification phase of design a hard task for designers.

Traditionally, Integrated Circuit (IC) verification is designed using white-box testing approach, which add monitors in internal points of a design, resulting in observability increase. However, because of the increasing complexity of current designs and shorter development times, exhaustive verification approach cannot be executed, resulting in the possibility of bugs in the released product.

A technique proposed in [1] extended the verification phase to product lifetime, increasing the possibilities of error detection and future correction, if we are working in the context of FPGA's (field-context programmable array).

In this paper, we present XRoach, a tool to automate Open Verification Library (OVL) [2] assertion checkers chaining. The tool compiles and chains previous instantiated Assertions in a new design. This new architecture is used together with an assertion processor, which will decide what to do with the detected error.

This paper is outlined as follows. In Section 2 the concepts of verification based on assertions are discussed, while Section 3 presents an architecture to extend OVL to support on chip run time debug. Section 4 presents XRoach Tool. In Section 5, we present the preliminary results. Finally, in Section 6 we conclude with our remarks and present future work.

## 2.VERIFICATION BASED ON ASSERTIONS

The ability to test a design correlates to the ability of controlling and observing the behavior of a design. The increase of design complexity over the past years has weakened the ability to test a design, as even if a design error can be controlled, it may be very difficult to observe the error using the design I/O pins [3].

White-Box Verification is a technique to improve observability of a design by embedding assertion monitors, which we call assertions in this paper. Using White-Box a designer can locate a failure internal to the design because assertions can trigger immediately after an error occurs.

Assertions are inserted into a design based on the knowledge about legal and illegal behavior of internal design structures [4],[5]. Usually, the assertions are inferred by a designer according to interface rules or unwanted corner cases of the design.

## 3. ON CHIP RUN TIME VERIFICATION

The paper [1], proposes an architecture for an assertion engine to be used in reconfigurable designs by extending the use of the White-Box Verification technique beyond the simulation/emulation phases of a design. The main idea was to modify the most used assertion library, OVL to support on chip run time debug. This technique was based on the Boundary-scan [6] concept along with synthesizable assertions. This architecture provides support to solve assertion routing problems although an assertion processor is needed to identify which assertion had failed and enable/disable the scan chain.

Figure 1 (a), presents a typical assertion module from the OVL. In order to use it in a scan-chain architecture extra pins were defined, four inputs and two outputs, as shown in Figure 4 (b).

The run-time version of the assertion library was extended with signals *ei, esci, eo, esco, esclk* and *escen_n*. Signal *eo* stands for the error output, which is the conjunction of all possible errors, and it signals that an internal error triggered by a failed assertion has occurred inside the chip. Signal *escen_n* disables the test expression evaluation in the assertion, enabling the scan-chain to be exercised. Signals *ei, esci, esco and esclk* are error in, normal scan input, output, and clock signals.

A chip's interface will be enlarged with signals *eo, esco, escen_n* and *esclk.* After the chip lowers the error output signal *eo*, a monitor starts investigating which assertion has triggered *eo* by first enabling the error scan chain (by asserting *escen_n*).

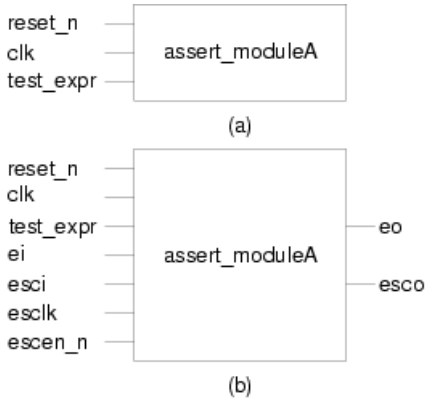


**Figure 1. (a) Typical OVL assertion; (b) OVL assertion modified for scan-chain architecture.**

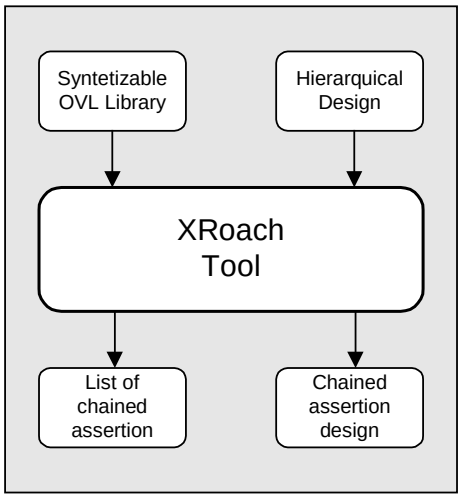


**Figure 2. XRoach functionality block diagram.**

At this time, the first assertion result appears in the error scan output (*esco*). Then, at subsequent cycles, by pulsing *esclk*, a new assertion result appears at *esco* until all assertions have been scanned. For a design with n assertions, after n cycles, all assertion violations will appear as zeros in the output of *esco*.

## 4. XROACH ARCHITECTURE

The objective of XRoach tool is to compile a design, chaining previous embedded assertions in an architecture that provides support to on chip run time debug. Its main features are mapping of any complexity level, hierarchy structure generation and insertion of chained structures.

Figure 2 shows how XRoach Tool works. XRoach processes verilog hierarchical designs with OVL assertions instantiated by development engineers. It compiles the design and links it to a version of the OVL, that has been synthesized, verified and prepared to deal with chained assertions. XRoach output files are the verilog design with chained assertions and a list of the assertions in the chained order, which is needed to debug the design when an assertion fails.

## 5. RESULTS

This section presents preliminary results of XRoach processing an $I^2C$ (Inter Integrated Circuit) verilog core.

$I^2C$ is a two-wire, bi-directional serial bus that provides a simple and efficient method of data exchange between devices. $I^2C$ standard was developed by Philips semiconductors[7]. Its applications include LCD drivers, remote I/O ports, RAM, EEPROM, data converters, digital tuning and signal processing circuits for radio and video systems, and DTMF generators.

Figure 3 depicts an example of an $I^2C$ core chaining. The total number of inserted assertions in the original design is 5. White circles are original design modules and gray circles are the chained assertions. After XRoach had mapped design's hierarchy, the assertions are chained with parent's input and output signals connected to the input and output signals of the instance. When a module definition is found, XRoach calculates the amount of additional wires needed and inserts the new pins *(ei, esci, eo, esco, esclk, escen_n*) in the module definition. If a module instantiation is found XRoach defines the instance order (first, intermediary or final) and inserts new pins.

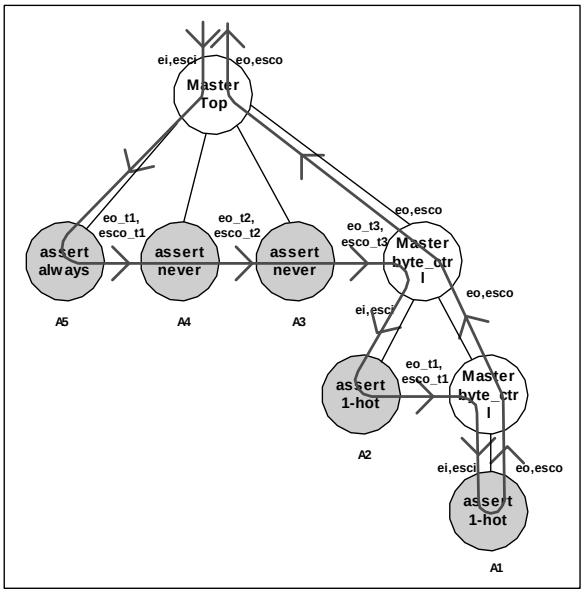


**Figure 3. $I^2C$ hierarchy after the assertion chaining.**

In case of multiple instances, as the always and never assertions instantiated by the Master_top, new wires (*eo_t1, eo_t2, esco_t1, esco_t2*) were created to accomplish the internal connections, preserving connection model with parent module. The output chaining order will be A1, A2, A3, A4 and A5.

## 6. CONCLUSIONS AND FUTURE WORK

White-Box Verification has been used as the best choice for validating designs. This paper presented XRoach tool to automatize the insertion and chaining of assertion checkers, extending the use of White-Box verification to the released product. XRoach processing results for an $I^2C$ core were presented. As future work, we are instantiating assertions in more complex cores like 32-Bit processors.

## 7. ACKNOWLEDGMENTS

This paper is partially supported by CNPq under grant 8301072002-9.

## 8.REFERENCES

[1] J. A. Nacif, F. M. de Paula, H. Foster, C. Coelho, F. C. Sica, D. C. da Silva and A. O. Fernandes. An Assertion Library for On-Chip White-Box Verification at Run-Time. In *Proceedings of Latin American Test WorkShop*, 2003.

[2] H. Foster and C. Coelho. Assertions Targeting a Diverse Set of Tools, In 10th Annual International HDL Conference Proceedings, 2001.

[3] S. Devadas and K. Keutzer. A unified approach to the synthesis of fully testable sequential machines. *IEEE Transactions on Computer-Aided Design.* 10(4):39-51, 1991.

[4] 0-In Design Automation, Inc. Assertion-Based Verification for Complex Designs. The Verification Monitor.2002.

[5] J. Bergeron. *Writing Testbenches Functional Verification of HDL Models.* Kluwer Academic Publishers. 2000. [6] IEEE. *Standard* 1149.1-2001. IEEE Press, 2001.

[7] Philips Semiconductors. The $I^2C$-Bus Specification, 2000.