APPLYING FUNCTIONAL VERIFICATION TECHINQUES ON THE DESIGN OF AN IP-CORE FOR AN AUTOMOTIVE COMMUNICATION PROTOCOL

R. V. M. Pereira[a], W. S. Gutstein[a], G. G. Freitas[a], E. Recalcatti[a], J. Guerra[b] and C. A. Zeferino[a]

[a] University of Vale of Itajaí - UNIVALI
Master Program on Applied Computer Science
São Jose, Santa Catarina, Brazil

[b] Federal University of Campina Grande - UFCG
Dedicated Architectures Laboratory
Campina Grande, Paraíba, Brazil

ABSTRACT

This paper discusses the developing of a testbench for the functional verification of an IP core specified to implement slave nodes for the automotive LIN bus. The approach used to implement this testbench differs from the other ones described in the literature because it uses an asynchronous approach. The results show that the adopted strategy allowed the functional verification of the modules of an IP core which implements the link and network layers of a LIN slave node.

1. INTRODUCTION

The design of IP (Intellectual Property) cores for electronic systems is a hard and time costly task. In order to ensure the quality of the final product, which have to meet the design specification, a lot of time and efforts are spent in the verification phase. For instance, in 2003, it was estimated that 50% of the project time was spent on the verification phase [1]. In 2005, about 70% of the entire project time was dedicated to verification tasks [2][3]. These efforts on verification are necessaries because 71% of the re-spins on Systems-on-Chip (SoCs) occurs due to logic bugs (47% of them are due to incorrect or incomplete specifications), and, furthermore, 14% of failing SoCs have bugs in reused components or IPs [2]. Moreover, the verification efforts increase exponentially with the complexity and size of the designed system.

There are two approaches for verification: formal and functional [3]. The industry makes extensive use of functional verification, so that methods that use it and the tools involved in the verification process must be able to detect functional errors in the design of IC in order to solve functional problems before the final stage of manufacturing integrated circuit (IC).

This work reports the use of functional verification techniques in the development of an IP core specified to implement the interface for LIN-bus compliant nodes for the automotive systems. This IP integrates the functionalities of link and network layers of LIN protocol.

The following text is organized into three sections. Section 2 presents an overview about LIN. Section 3 presents the verification methodology used in this work and how it was applied in the validation of the IP core. Concluding, Section 4 presents the final remarks.

2. LIN BUS

The LIN bus (or SAE J2602) [4] is an inexpensive self-synchronized, single-wired and single-master interconnection architecture, mainly used in the communication among smart devices (e.g. sensors and actuators) in automotive systems. Its bandwidth is limited in the range from 1 kbps to 20 kbps, which allows its use in several applications, such as window lift, mirror switch, door lock, seat control switch and several others.

The communication among nodes in a LIN bus is scheduled by the master node. It injects a message header in the bus with an identifier that defines the message type (a Request or a Response) and the nodes involved in the transaction (the master and a slave or two slaves).

Figure 1 depicts the format of the LIN message (frame). It begins with a header including three fields: Break, which marks de begin of the message; Synch, used to synchronize the nodes; and PID, the message identifier. In the case of a Response message, the header is followed by up to 8 bytes of data and a byte of checksum.

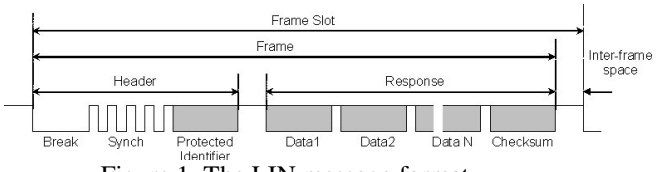

Figure 1. The LIN message format

The major benefits of LIN include its low cost and the fact that its specification is open. Aspects taken into consideration in the vehicles developed in

emerging markets like Brazil [5]. The LIN is also widely used in non-critical systems which require an inexpensive serial communication protocol. Furthermore, LIN provides a high degree of reliability due to predictability of the messages that travel on the bus, this feature due to the scheduling of messages made by the master [4].

The LIN Specification 2.1 [4] defines four layers based on OSI model: (*i*) physical; (*ii*) data link, (*iii*) transport; and (*iv*) application. Typically, the physical layer is implemented by using a transceiver IC (the PHY), while the other layers can be implemented by software or hardware. For instance, there are some commercial synthesizable IP cores which implement the data link and the transport layers. Examples include the IPs provided by CAST [6], by Intelliga [7], and by Bosch [8], which target FPGA and ASIC technologies.

3. USING FUNCTIONAL VERIFICATION ON THE DEVELOPMENT OF A LIN SLAVE IP

In this work, it was developed a verification model used in the validation of an IP core which implements the link and the transport layers for LIN slave nodes.

The verification model was developed based on BVM (Brazil IP Verification Methodology) [9]. As Figure 2 depicts, in BVM, the testbench used in the verification of a design (or DUV – Design under Verification) is composed by the following blocks:

- *Source*: generates the stimuli;
- *Reference Model*: models the DUV's functionality in a higher abstraction level (*e.g.* SystemVerilog, C, C++);
- *Driver*: converts transaction packets into signals and sends it to the DUV;
- *Monitor*: converts the DUV's output signals into transaction packets and activates the communication protocol of the DUV
- *Actor*: performs the communication protocol handshake, ensuring the integrity and synchronicity of transmission between the Driver and the Monitor; and
- *Checker*: compares the outputs of the Reference Model with the ones of the DUV.

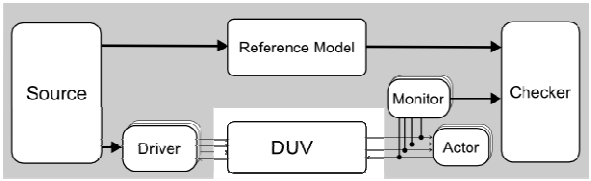

Figure 2. The architecture of a BVM testbench

The IP core designed in this work includes two bidirectional interfaces: an AMBA AXI port to communicate with the application; and a LIN port to attach the node to the LIN bus.

Figure 3 presents the architecture of the testbench developed in this work. For each communication interface of the DUV, there exists a set of Driver and Monitor blocks. Since the LIN bus is self-synchronized and does not include signals for handshaking, its Monitor dismisses the Actor block, which would be responsible for the handshake functionality. This feature added an additional difficult in the development of the testbench because it is not foreseen in the BVM architecture.

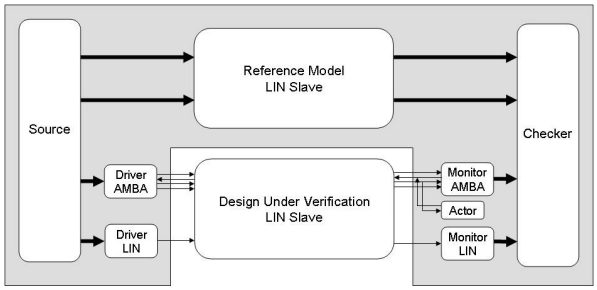

Figure 3. The architecture of the testbench for the LIN slave IP core

Both the testbench and the DUV have an internal block, not shown in Figure 3, which is responsible to generate a signal that defines the period to transfer a bit (the bit time or T_{bit}), ensuring that both works at the same bit rate (*e.g.* 9.600 bps).

The testbench and the DUV are compliant with the integrity requirements of LIN Specification 2.1 [4]. Both verify if the time constraints are respected and each if byte and message are properly transferred.

A message scheduling table was included in the Source block in order to emulate the scheduling functionality performed by the LIN master.

The testbench was implemented using SystemVerilog, while the DUV was described using Verilog. The design flow used in this project included the following development tools: Cadence IUS, Cadence SimVision, Cadence ICCR, Altera Quartus II, and Mentor Graphics ModelSim.

The IP core was validated by applying this design flow in a bottom-up approach. Firstly, they were validated the lower level blocks. After that, the six building blocks of the IP core were integrated and the testbench was applied.

Figure 4 presents the screenshot of a Cadence SimVision window that shows a waveform with the result of the execution of the testbench validating the DUV. They are shown the signals of the LIN and AMBA AXI interfaces of the DUV.

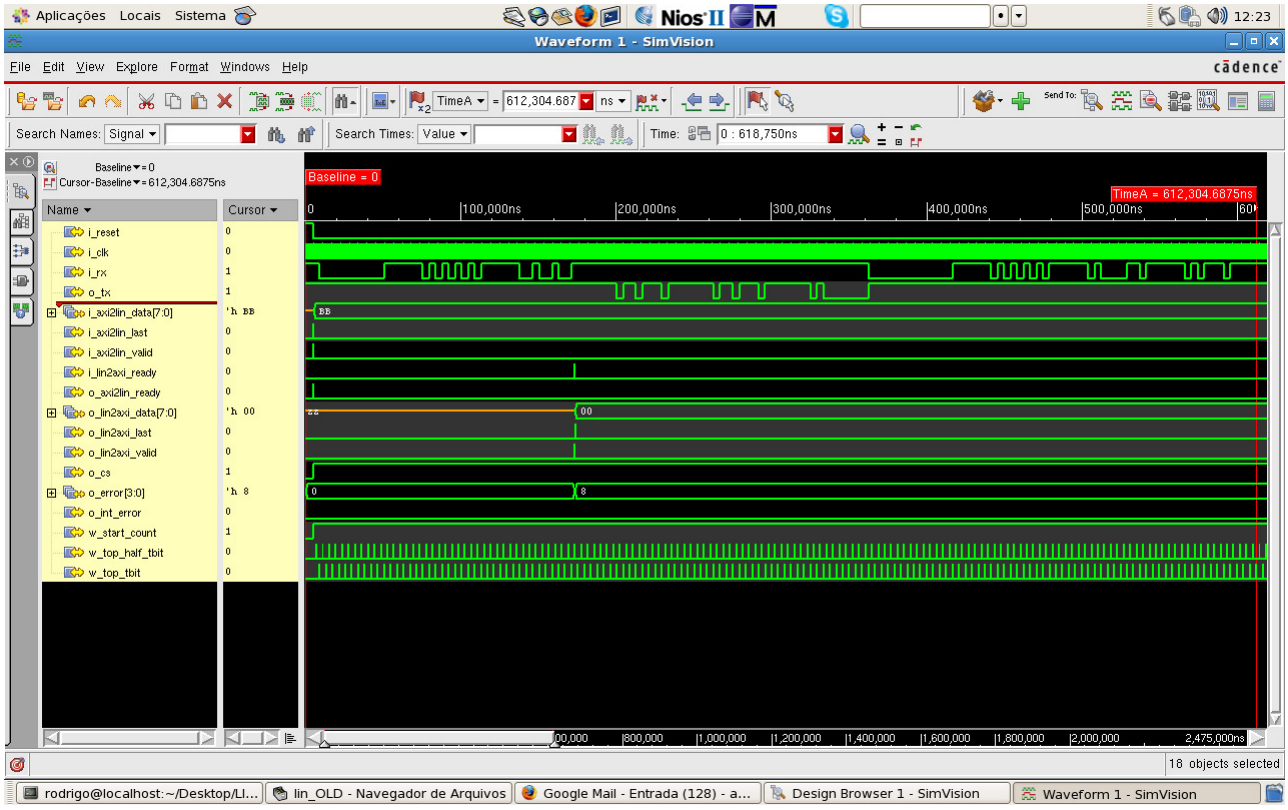

Figure 4. Waveform of an experiment which shows the execution of the testbench

4. CONCLUSIONS

This paper presented the development of a testbench to perform the functional verification of an IP core that implements the link and network layers of the LIN bus protocol. The testbench was developed using the BVM verification methodology. In order to deal with the self-synchronization of LIN bus, the BVM architecture needed to be adapted by excluding the Actor block. The testbench was applied and allowed to verify the functionality of the designed IP core.

Future works include improvements in the scheduling tables in order to enlarge the test case for validation, the physical validation of the IP core in FPGA and its implementation targeting an ASIC technology.

5. ACKNOWLEDGMENTS

This project was developed with the support of the University of Vale of Itajaí (Univali) and the Federal University of Campina Grande (UFCG), and financial support of CNPq (Concelho Nacional de Desenvolvimento Científico Tecnológico), in the context of Brazil IP program.

6. REFERENCES

[1] J. Bergeron, *Writing Testbenches: Functional Verification of HDL Models*, Second Edition. Kluwer Academic Publishers, Norwell, MA, USA, 2003.

[2] R. Mariani, “Breaking the Verification Barriers”, *Automotive Design Line*, Feb. 2005.

[3] J. Bergeron, *Writing Testbenches Using SystemVerilog*, Springer, New York, USA, 2006.

[4] LIN Consortium, *LIN specification package – Revision 2.1*, Sttutgart: LIN Consortium, 2006.

[5] A. A. Guimarães, *Eletrônica Embarcada Automotiva*, 1ª ed., São Paulo: Érica, 2007. (*only in Portuguese*)

[6] Cast Inc, *LIN Controler Core*, Woodcliff Lake: Cast Inc., USA,2007

[7] Intelliga, *Local Interconnect Network Controller Core – iLIN v1.3*, Essex: Intelliga. 2003.

[8] Bosch, *C_LIN module - for low cost LIN slave designs*. Reutlingen, Germany, Robert Bosch GmbH. 2007.

[9] K. R. G. da Silva, *Uma Metodologia de Verificação Funcional para Circuitos Digitais*, PhD Thesis in Electrical Engineering, Federal University of Campina Grande, Paraíba, Brasil, 2007. (*only in portuguese*)