

Built-In Self-Test for Programmable I/O Buffers in FPGAs and SoCs

Sudheer Vemula, *Student Member, IEEE*, and Charles Stroud, *Fellow, IEEE*

Abstract—The first Built-In Self-Test (BIST) approach for the programmable Input/Output (I/O) buffers in Field Programmable Gate Arrays (FPGAs) and configurable System-on-Chip (SoC) implementations is presented. The I/O buffers are tested for their various modes of operation along with their associated programmable routing sources. A general BIST architecture, applicable to any FPGA or SoC with embedded FPGA core, is presented along with the capabilities and limitations of the approach. Experimental results are presented for BIST configurations developed for the I/O buffers in FPGAs including Atmel AT40K and Xilinx Virtex series as well as SoCs including Atmel AT94K series.

I. INTRODUCTION

All Field Programmable Gate Arrays (FPGAs) include programmable Input/Output (I/O) buffers to communicate with the other components present in the system. Configurable System-on-Chips (SoCs) include embedded FPGA cores with associated programmable I/O buffers which communicate not only with external components but also with other embedded cores. Typical programmable I/O buffers have several architectural features and associated routing resources to/from of the core of the FPGA. Since defects are possible in any resource of an FPGA, they may also occur in the I/O buffers and/or their associated routing making information exchange with other components impossible or unreliable. This implies that the I/O buffers should be tested to ensure the fault-free operation in the same manner as the programmable logic and interconnect resources in the FPGA core are being tested. While a number of Built-In Self-Test (BIST) approaches have been developed for the programmable logic and routing resources in the FPGA core [1]-[7], they have neglected testing the I/O buffers and their associated routing resources.

We present a BIST approach that can be used to test the programmable I/O buffers of any FPGA or FPGA core in a SoC. We begin with an overview of relevant prior work in BIST for FPGAs and FPGA I/O buffer testing in Section II followed in Section III by a discussion of the typical architectural features of programmable I/O buffers in FPGAs. A detailed description of the BIST approach is presented in Section IV and experi-

mental results are given in Section V for fault simulations as well as actual implementations in several different devices. The paper concludes in Section VI with a discussion of the capabilities and limitations of the I/O buffer BIST approach.

II. BACKGROUND AND MOTIVATION

A number of BIST approaches have been developed to test the programmable logic and routing resources in FPGAs and embedded FPGA cores in SoCs [1]-[7]. The basic technique is to configure some of the programmable logic blocks (PLBs) in the FPGA core as Test Pattern Generators (TPGs) and Output Response Analyzers (ORAs). These TPGs and ORAs are used to detect faults in PLBs under test (BUTs) in logic BIST or wires under test in routing BIST.

In logic BIST, the BUTs and ORAs are arranged in alternating columns (or rows) and two or more identical TPGs are used to drive the alternating columns (or rows) of BUTs as illustrated in Figure 1 [1]-[3][7]. The output responses of identically programmed BUTs are compared by ORAs in neighboring columns (or rows). The basic design of the comparison-based ORA (also used to shift out BIST results) is illustrated in Figure 2. During a given test session (Figure 1a), the BUTs are repeatedly reconfigured in their various modes of operation until they are completely tested. While the logic resources of the PLBs are completely tested by most logic BIST approaches, there is also logic in the I/O buffers, such as multiplexers, flip-flops and latches which also should also be tested to ensure proper operation when the intended system function is programmed in the FPGA. In fact, the logic resources in unbonded I/O buffers are sometimes used by FPGA synthesis tools to implement the system function.

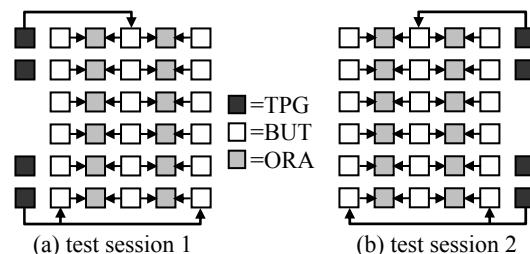


Fig. 1. FPGA logic BIST architecture.

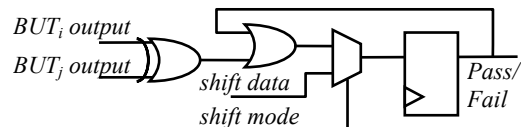


Fig. 2. ORA design.

This work was supported in part by the National Security Agency under contract H98230-04-C-1177.

Sudheer Vemula is with Dept of Electrical & Computer Engineering, 200 Broun Hall, Auburn University, AL 36849-5201 USA (e-mail: vemulsu@auburn.edu).

Charles Stroud is with Dept. of Electrical & Computer Engineering, 200 Broun Hall, Auburn University, AL 36849-5201 USA (phone: 334-844-1806; fax: 334-844-1809; e-mail: strouce@auburn.edu).

Two types of routing BIST approaches, comparison-based and parity-based, have been used to test the programmable interconnect resources in FPGAs [3]-[6]. While both approaches have been proven to be effective in detecting faults, they have been used exclusively to test the routing resources in the FPGA core, leaving the routing resources associated with I/O buffers untested.

BIST has been used to test the speed of I/O buffers in [8]. Additional circuitry, such as delay locked loop, test registers and comparators, was included for each buffer under test. This BIST circuit was developed for implementation in an Application Specific Integrated Circuit (ASIC) to test the set-up and hold time of the registers associated with the I/O buffer. But other resources present in the I/O buffer are not tested by this method [8]. An IDDQ testing approach for I/O buffers in an FPGA was proposed in [9]. The test signals were generated both internally and externally, but the results were analyzed only by externally monitoring the signal values [9]. As a result, this technique is not a BIST approach to test the I/O buffers of an FPGA since internal logic resources were used only for generation of test patterns to the I/O buffers due to the limited external access, particularly in the case of unbonded I/O buffers.

The Boundary Scan external test (EXTEST) mode can be used to test the input and output buffers, tri-state control, and pads [10]. However, the EXTEST capability cannot be used to test the flip-flops used for registered inputs and outputs or the programmable routing resources connecting the I/O buffer to the FPGA core. While the Boundary Scan internal test (INTEST) could be used to test these flip-flops and routing resources, the INTEST feature is not supported in many FPGAs; this includes Atmel AT40K and AT94K series devices [11][12]. Therefore, a BIST approach for I/O buffers would provide the ability to test the flip-flops used for registered inputs and outputs, dedicated programmable routing resources, and other programmable resources (as will be discussed in the next section) typically associated with the programmable I/O buffers regardless of what Boundary Scan features are supported or not supported for a given family of FPGAs or SoCs.

III. OVERVIEW OF I/O BUFFERS IN FPGAs AND SoCs

A simple model of the typical programmable I/O buffer found in FPGAs is illustrated in Figure 3. It consists of a bi-directional buffer with tri-state control (TC). Multiplexers controlled by configuration memory bits are used to select registered or non-registered signals to (TC and OUT) or from (IN) the pad. The registers are used for critical timing parameters such as set-up and hold time and can be flip-flops, latches, or may be programmable as either a flip-flop or latch. Programmable clock enable and/or programmable set/reset, under the control of configuration memory bits, may also be associated with each register. In addition, the

output buffer typically has programmable drive capability along with programmable pull-up, pull-down, and keeper features. The input buffer typically has programmable I/O voltage, filter and delay characteristics.

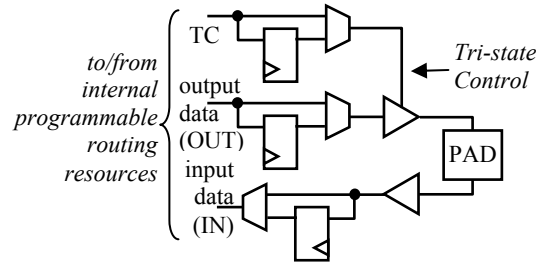


Fig. 3. Typical programmable I/O buffer architecture.

IV. I/O BUFFER BIST ARCHITECTURE

The basic idea of the I/O buffer BIST architecture is to configure some of the PLBs as TPGs and ORAs while the I/O buffers under test are configured as bi-directional buffers, as illustrated in Figure 4. In this way, test patterns produced by the TPG can be applied to the output portion of the I/O buffer while comparison-based ORAs monitor the input portion of two or more identically configured I/O. As a result, the pad provides a loop-back path of the TPG test patterns to the ORAs such that testing of the complete I/O buffer is accomplished.

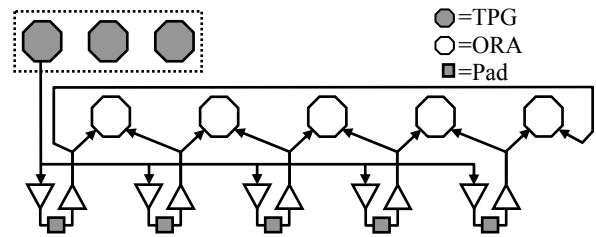


Fig. 4. I/O buffer BIST architecture.

This BIST architecture facilitates the testing of all bonded and un-bonded I/O buffers and is, therefore, package independent. The I/O buffer BIST architecture is similar in some respects to the logic BIST architecture of Figure 1. Here the I/O buffers replace the BUTs while the PLBs act as the TPGs and ORAs. The output response of each buffer is monitored by two ORAs and compared with the output responses of two other I/O buffers; this greatly reduces the chances of equivalent faults in I/O buffers escaping detection as is the case in logic BIST [2]. The comparison-based ORAs use the same construction as the logic BIST ORA illustrated in Figure 2. One important difference in the I/O buffer BIST architecture is that only one TPG is required since the core PLBs and routing resources in the core of the FPGA are assumed to have been previously tested with traditional logic and routing BIST approaches developed for FPGAs.

Another important difference is that an additional ORA is incorporated, as shown in Figure 4, to enable a circular comparison. This circular comparison-based BIST architecture has been shown to provide higher diagnostic resolution and better fault detect capabilities than the logic BIST architecture shown in Figure 1, where the BUTs along the edge of the array are monitored by only one ORA [13]. Furthermore, this I/O buffer BIST architecture facilitates the use of the diagnostic procedures previously developed for identifying faulty cores in any set of multiple identical cores or structures (like I/O buffers) in FPGAs and SoCs [13].

The TPG for the I/O buffer BIST can be an N -bit Linear Feedback Shift Register (LFSR) or an N -bit counter where the Most Significant Bit (MSB) of the counter is used to drive the set/reset signal to the flip-flops/latches of the I/O buffer while the remaining bits of the counter are applied to other inputs of the I/O buffer such as clock enable, OUT, TC and other unselected inputs to the multiplexers that source TC and OUT. The value of N depends on the number of inputs present to the largest multiplexer in the I/O buffer.

The I/O buffers are repeatedly reconfigured in their various modes of operation such as activated pull-up or pull-down, selection of different multiplexer inputs, registered versus non-registered inputs and outputs, programmable active levels of clock enable and set/reset, active edges of clock, etc. Similarly, the routing resources associated with the I/O buffers are also tested via repeated reconfiguration of the connections made from the TPGs to the I/O buffers and from the I/O buffers to the ORAs. The minimum number of BIST configurations needed to test the I/O buffers and their associated routing resources is usually a function of the number of possible signal paths to the output portion of the I/O buffer.

All programmable features and resources of the I/O buffers are can be tested with this BIST architecture. However, in some FPGAs, the I/O buffers have transmission gates that allow the input and output portions of the I/O buffer to share programmable routing resources. A slightly different BIST approach can be used to test these transmission gates for stuck-off faults to reduce the complexity of repeated reconfiguration of routing resources from the output of the input buffer to the ORA. (Note that stuck-on faults in these transmission gates are detected by the BIST configurations using the architecture shown in Figure 4). To test stuck-off faults, opposite logic values are stored in the two flip-flops for the input and output data portions of the I/O buffer, as shown in Figure 5. This can be done while the test is performed at the end of the previous BIST configuration. Next, one of the transmission gates is activated by dynamic partial reconfiguration to create a feed-back loop from the output of the input buffer to the input of the output buffer. As the flip-flops are clocked, the tog-

gling logic value on the output is monitored by the ORAs for a few clock cycles to observe whether the transmission gates are able to pass both logic values. The same procedure is performed, in turn, by activating other transmission gates, if more than one transmission gate is present in the I/O buffer.

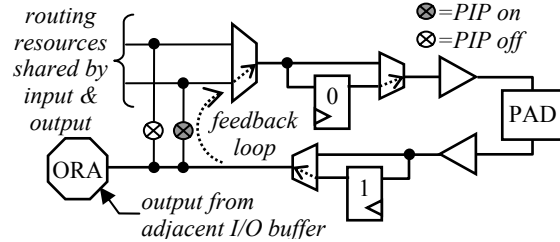


Fig. 5. Transmission gate BIST configurations.

Some FPGAs have the ability for any I/O buffer to drive a global reset. This feature requires a slightly different type of BIST architecture since only one I/O buffer can drive the global reset at any given time. As a result, a separate BIST configuration is needed for each I/O buffer in order to test its ability to drive the global reset. All of the previously discussed BIST approaches are independent of the array size or, in this case, the number of I/O buffers. As a result, the BIST test time (excluding download time) is constant and independent of the array size. However, the number of configurations required to test the global reset input connection on all of the I/O buffers for stuck-off faults is equal to the number of programmable I/O buffers associated with the FPGA core. Fortunately, by developing a regular BIST architecture (shown in Figure 6) and using partial dynamic reconfiguration, this feature can be tested with one download, without a large increase in test time.

During this BIST sequence, the TPG generates the expected output response of a flip-flop to be reset by the global reset for comparison in the ORA. Whenever, the MSB (C1) of the 2-bit counter is a logic 1, the flip-flop is asynchronously reset, otherwise the LSB (C0) value is clocked through the flip-flop to the ORA.

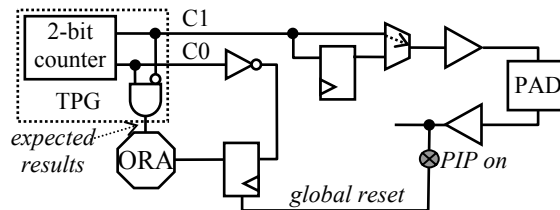


Fig. 6. Global reset PIP stuck-on test.

Stuck-on faults in the PIP connection from each I/O buffer to the global reset are detected in all I/O buffers in parallel using the BIST architecture illustrated in Figure 4. A stuck-on fault in the PIP to global reset can be observed by inserting a special ORA, illustrated in Figure 7, during one BIST configuration. In this BIST configuration, the tri-state buffer is always enabled while

the flip-flops in both input and output portions of the buffer are activated and reset prior to the BIST sequence. Note that both flip-flops can be reset by the global reset. In the example shown in Figure 7, we assume the global reset is active high. During the BIST sequence, a logic 1 from the TPG is clocked into the flip-flop of the output buffer such that if the PIP to the global reset in any I/O buffer is stuck-on, both flip-flops will be reset, and a logic 1 will never be observed at the output of the input buffer flip-flop. Therefore, a logic 0 in this ORA at the end of the BIST sequence would indicate a stuck-on fault in at least one of the I/O buffers.

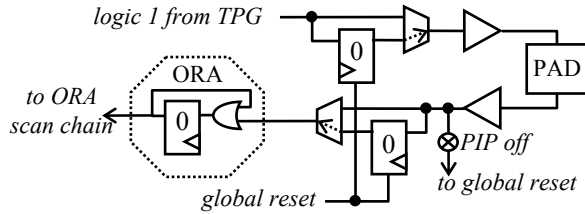


Fig. 7. Global reset PIP stuck-on test.

V. IMPLEMENTATION RESULTS

We have implemented the proposed I/O buffer BIST in several FPGAs and configurable SoCs from Atmel and Xilinx. Table I summarizes an overview of the basic programmable features of the I/O buffers typically found in Atmel and Xilinx devices. The following subsections describe other unique features of their I/O buffers along with the results of our BIST development and implementation in the various devices.

TABLE I
I/O BUFFER FEATURES

Feature	Atmel	Xilinx
Flip-flop/latch	flip-flop only	both
Set/reset	asynchronous reset	programmable
Clock enable	no	programmable
Pull-up/pull-down	yes	yes plus keeper
I/O standards	3	16
Output drive levels	3	7
Multiplexer inputs	8	20
PIP to global reset	yes	no

A. Atmel FPGAs and SoCs

The I/O buffers in Atmel devices are divided into two types, referred to as primary and secondary I/O buffers, depending on their position with respect to the peripheral PLBs. The secondary I/O buffer has one less input to the tri-state control and output multiplexers and two fewer transmission gates at the output of the input buffer compared to the primary I/O buffer. Due to restrictions in the dedicated routing resources between the I/O buffers and the periphery PLBs in the array, primary and secondary I/O buffers are tested in separate sets of BIST configurations.

A total of 14 BIST configurations were developed to completely test the logic and routing resources in the primary I/O buffers while a total of 11 BIST configura-

tions were developed for the secondary I/O buffers. A similar set of 23 of BIST configurations are used to test the I/O buffers of the AT40K series FPGAs; in these I/O buffers the multiplexers to TC and OUT are slightly smaller and there are no flip-flops for registered input and outputs. The BIST configurations are summarized in Table 2 in terms of the number of configurations using the general, transmission gate, and global reset BIST approaches. With the exception of the global reset test, this set of BIST configurations is independent of the array size or the number of I/O buffers associated with the FPGA core in any Atmel AT94K series SoC. The fault detection capabilities were verified through fault simulation (see Figure 8 for individual and cumulative fault coverage) as well as physical fault insertion in actual devices by manipulating the configuration memory bits to emulate shorts, opens, and stuck-at faults in the logic and routing resources associated with the buffers.

TABLE II
ATMEL I/O BUFFER BIST CONFIGURATIONS

BIST Configurations	AT94K SoC		AT40K FPGA	
	Pri- mary	Sec- ondary	Pri- mary	Sec- ondary
General (Fig 4)	9	8	8	7
Transmission Gate (Fig 5)	4	2	4	2
Global Reset (Fig 6)	1	1	1	1
Total	25		23	

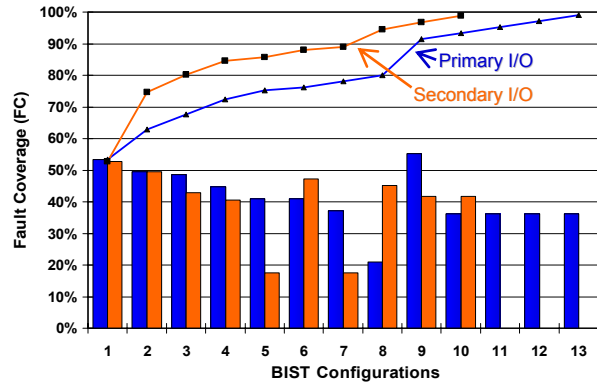


Fig. 8. Individual and cumulative fault coverage for Atmel AT94K SoC I/O buffer BIST configurations.

To generate the complete set of BIST configurations, three master BIST configurations were developed (one for primary I/O buffers, one for secondary I/O buffers, and one for global reset tests for both primary and secondary) using Atmel's Macro Generation Language [11][12]. The remaining BIST configurations were then automatically generated from the three master configurations by manipulating the bits in the configuration bit streams to be downloaded into the FPGA to run the BIST sequence. Further improvements in test time were obtained for the AT94K series SoC by using dynamic partial reconfiguration from the embedded processor core to reconfigure the I/O buffers for subsequent BIST configurations after the initial download such that all testing is performed with three downloads.

B. Xilinx FPGAs and SoCs

All of the I/O buffers in Xilinx Virtex I [14] and Spartan II [15] devices are identical with groups of four buffers connected to the PLB array through the same type of global routing matrices as associated with the PLBs themselves. In addition, there is no global reset PIP in each I/O buffer. As a result, all I/O buffers can be tested in parallel. The number of inputs to the multiplexers for tri-state control, output, set/reset, clock and clock enable inputs is much larger than Atmel. There are 20 inputs for the clock and 16 for the other inputs. There are 27 transmission gates on the output of the input buffer to routing, but the transmission gate BIST architecture (Figure 5) can only be used to test a subset of these. Therefore, we use the general I/O buffer BIST architecture (Figure 4) for all I/O buffer testing with a set of 27 BIST configurations required to completely test the I/O buffers. When the global routing matrix connections to the I/O buffers are ignored, the logic and routing resources of the I/O buffers alone only require ten BIST configurations for complete testing.

VI. SUMMARY AND CONCLUSIONS

A general BIST approach was presented to test the programmable I/O buffers and their associated routing resources in an FPGA or the FPGA core of an SoC. This BIST approach is applicable to any FPGA or SoC but, we have also presented two alternative BIST architectures to test specific features, including transmission gates and global reset connections, sometimes found in programmable I/O buffers.

BIST configurations were developed, downloaded and verified in actual devices including Atmel AT40K series FPGAs and AT94K series SoCs as well as Xilinx Virtex I and Spartan II series FPGAs. Fault detection capabilities were verified using fault simulation, where 100% stuck-at gate-level fault coverage was obtained, and physical fault injection emulation via manipulation of the configuration memory bits. These results indicate that the I/O buffer BIST approach can detect all the faults present in the logic and routing resources associated with each I/O buffer. While the BIST can also detect major defects that affect the analog programmable features (such as pull-up, pull-down, tri-state, etc.), it cannot, however, detect all parametric faults such as V_{OL} , V_{OH} , V_{IL} , V_{IH} , current sink and source capabilities, delay, etc. It should be noted that the BIST approach can detect faults in the configuration bits that control the programmable analog parametric features such as drive capability, delay, and voltage levels. Therefore, the BIST approach provides a good sanity test of the I/O buffers.

The I/O buffer BIST approach described in this paper can be used in manufacturing testing at wafer-level and device-level testing. Since it can test all unbonded as well as all bonded I/O buffers, it can be used to test

packaged devices without additional test development or increase in testing complexity due to limited access to unbonded pads. The I/O buffer BIST approach can be used for system-level testing only if all other connecting devices can be tri-stated since all I/O buffers are configured as bi-directional buffers during the BIST sequence. An alternative approach for system-level testing is to develop BIST configurations that only test those bonded pads that serve as outputs and bi-directional buffers in the target system in conjunction with using the traditional Boundary Scan EXTEST for testing the remaining input buffers (since the output portion of those buffers is not used by the system).

In conclusion, a BIST approach for programmable I/O buffers is particularly important since logic and routing resources in unbonded I/O buffers are frequently used by synthesis tools to implement the system function. It should be noted that this is the first BIST approach developed for FPGA I/O buffers. When used in conjunction with other logic and routing BIST approaches for FPGAs, we can achieve complete testing through BIST for the entire FPGA.

REFERENCES

- [1] C. Stroud, *A Designer's Guide to Built-In Self-Test*. Kluwer Academic Publishers, Boston MA, 2002.
- [2] M. Abramovici and C. Stroud, "BIST-Based Test and Diagnosis of FPGA Logic Blocks," *IEEE Trans. on VLSI Systems*, Vol. 9, No. 1, pp. 159-172, 2001.
- [3] C. Stroud, K. Leach, and T. Slaughter, "BIST for Xilinx 4000 and Spartan Series FPGAs: A Case Study," *Proc. IEEE Int'l Test Conf.*, pp. 1258-1267, 2003.
- [4] D. Fernandes and I. Harris, "Application of Built-In Self-Test for Interconnect Testing of FPGAs," *Proc. IEEE Int'l Test Conf.*, pp. 1248-1257, 2003.
- [5] C. Stroud, J. Nall, M. Lashinsky and M. Abramovici, "BIST-Based Diagnosis of FPGA Interconnect," *Proc. IEEE Int'l Test Conf.*, pp. 618-627, 2002.
- [6] X. Sun, J. Xu, B. Chan and P. Trouborst, "Novel Technique for BIST of FPGA Interconnects," *Proc. IEEE Int'l Test Conf.*, pp. 795-803, 2000.
- [7] C. Stroud, J. Sunwoo, S. Garimella and J. Harris, "Built-In Self-Test for System-on-Chip: A Case Study," *Proc. IEEE Int'l Test Conf.*, pp. 837-846, 2004.
- [8] Cheng Jia and L. Milor, "A BIST Solution for the Test of I/O Speed," *Proc. IEEE Int'l Test Conf.*, pp. 1023-1030, 2003.
- [9] L. Zhao, D. Walker and F. Lombardi, "IDDQ Testing of Input/Output Resources of SRAM-Based FPGAs," *Proc. Asian Test Symp.*, pp. 375-380, 1999.
- [10] C. Maunder and R. Tulloss, *The Test Access Port and Boundary Scan Architecture*, IEEE Computer Society Press, 1990.
- [11] ____, "AT40K Series Field Programmable Gate Array," Data Sheet, Atmel Corporation, 2002, available at www.atmel.com.
- [12] ____, "AT94K Series Field Programmable System Level Integrated Circuit," Data Sheet, Atmel Corp., 2001, available at www.atmel.com.
- [13] C. Stroud and S. Garimella, "Built-In Self-Test and Diagnosis of Multiple Embedded Cores in SoCs," *Proc. Int'l Conf. on Embedded Systems and Applications*, pp. 130-136, 2005.
- [14] ____, "Virtex Field Programmable Gate Arrays," Product Specification DS003-1, Xilinx, Inc., 2001.
- [15] ____, "Spartan II 2.5V FPGA Family: Complete Data Sheet," Product Specification DS001, Xilinx, Inc., 2001.