

Soft-Core Embedded Processor-Based Built-In Self-Test of FPGAs: A Case Study

Bradley F. Dutton, *Graduate Student Member, IEEE*, and Charles E. Stroud, *Fellow, IEEE*

Dept. of Electrical and Computer Engineering
Auburn University, Auburn, Alabama
duttobf@auburn.edu

Abstract—This paper presents the results of a case study which investigates the use of an embedded soft-core processor to perform Built-In Self-Test (BIST) of the logic resources in Xilinx Virtex-5 Field Programmable Gate Arrays (FPGAs). We show that the approach reduces the complexity of an external BIST controller, making it particularly appealing for in-system testing of high-reliability and fault-tolerant systems with FPGAs. However, the overall test time is not improved due to an increase in the size of the required configuration files as a consequence of the inclusion of the soft-core embedded processor logic, whose relative irregularity results in less effective compression of configuration data files.

Keywords—Built-In Self-Test; Field Programmable Gate Array; System-Test; Fault Tolerance, Virtex-5.

I. INTRODUCTION

This paper presents the results of the first implementation of Built-In Self-Test (BIST) for Field Programmable Gate Arrays (FPGAs) using a soft-core embedded processor synthesized into the fabric of the FPGA under test. The approach, as originally proposed in [1], reduces the number of configuration files required for BIST by exploiting the regularity of BIST architectural structures to significantly compress and store partial configuration data in the embedded processor's program memory. The embedded processor controls and executes the BIST sequence, including retrieval and analysis (fault diagnosis) of BIST results, and reconfiguration of the FPGA for subsequent BIST configurations [1]. This embedded processor-based BIST approach is possible for two reasons: first, the growing size and complexity of FPGAs facilitates the inclusion of complex circuitry that only occupies a small percentage of the total configurable resources, leaving adequate area for BIST logic and routing; and, secondly, the ability to access the configuration memory from inside the FPGA fabric has made possible internal reconfiguration and read back of FPGA logic and routing resources.

The approach was successfully implemented in Xilinx Virtex-5 [2] but is applicable to any FPGA with internal configuration memory access. The remainder of the paper is organized as follows: Section II presents an overview of BIST for FPGAs and the previously proposed soft-core processor-based BIST technique. Section III presents the results of our implementation of soft-core embedded processor-based BIST in Virtex-5 FPGAs, including test time analysis and

comparisons with other BIST approaches for FPGAs. Section IV discussed ways in which the proposed approach might be improved, with Section V covering other potential applications of the approach. The paper is summarized in Section VI.

II. BACKGROUND

BIST for FPGAs exploits the re-programmability of FPGAs to create test circuitry in the FPGA fabric during off-line testing [3]. The only overhead is the external memory required to store the BIST configurations along with the time required to download and execute the numerous BIST configurations. No area overhead or performance penalties are incurred because the BIST logic is reconfigured with the intended system function after testing is complete. The BIST configurations are applicable to all levels of testing because they are independent of the end-user system function and require no specialized external test fixture or equipment. Over the past 15 years, a number of BIST approaches have been developed for the configurable logic and routing resources in FPGAs. Due to the programmable nature of FPGAs, all BIST approaches for FPGAs require multiple configurations of the resources under test in all of their modes of operation in order to obtain high fault coverage. Some of these BIST approaches are summarized in Table I, where the number of BIST configurations is given for each type of resource including configurable logic blocks (CLBs), input/output (I/O) tiles, random access memories (RAMs), digital signal processors (DSPs), and programmable routing resources.

TABLE I. TEST CONFIGURATIONS DEVELOPED FOR VARIOUS FPGAS

FPGA	CLBs	Routing	I/O	RAMs	DSPs	References
ORCA 2C	9	27	-	0	0	[5][6]
ORCA 2CA	14	41	-	0	0	[5][6]
Delta 39K	20	419	-	11	0	[7]
4000/Spartan	12	128	-	0	0	[8]
4000XL/XLA	12	206	-	0	0	[8]
AT40K/AT94K	4	56	27	3	0	[9] - [11]
Virtex/Spartan-2	12	283	7	5	0	[11][12]
Virtex-4	10+5	84	14+68	15	5	[13] - [17]
Virtex-5	6+5	?	15+68	16	11	[17][18]

Most research and development in BIST for FPGAs has focused on reducing the number of test configurations, reducing the size of test configuration files, and decreasing BIST execution time [4]-[18]. But the ever increasing complexity and level of integration in FPGAs has, with few

exceptions, resulted in longer test times, more downloads, and more memory required for storing BIST configurations for each new generation of FPGA. However, the increasing size and complexity of FPGAs has also created opportunities for innovation in FPGA testing. In [1], we proposed an embedded processor-based approach which exploits some of these features of current FPGAs in an attempt to improve test time and reduce the complexity of BIST. The soft-core embedded processor-based BIST approach for FPGAs incorporates additional logic in the FPGA fabric along with the BIST logic to perform tasks typically assigned to an external controller or computer. The new approach offers several advantages over the traditional external BIST approach. First, the 32-bit internal configuration access port (ICAP) is used for reconfiguration of the resources under test, eliminating the test time penalties associated with the lower speed, serial Boundary Scan interface. Secondly, the total number of configurations that are downloaded via the external configuration interface is reduced to one per test session. In addition, all control of the BIST configurations and test procedures can be implemented in the embedded processor. Finally, fault diagnosis procedures can also be performed by the embedded processor, further reducing the complexity of the external BIST controller in fault-tolerant applications and providing considerable speed-up when compared to Boundary Scan based readback and diagnosis.

The basic architecture of the embedded BIST approach for CLBs is illustrated in Fig. 1 [1]. In this particular BIST approach, one-half of the FPGA array is configured with the BIST circuitry, including multiple Test Pattern Generators (TPGs), comparison-based Output Response Analyzers (ORAs), and the Blocks Under Test (BUTs). The TPGs are constructed from CLBs or other logic resources such as DSPs, RAMs, etc. The TPGs provide identical test patterns to alternating rows or columns of identically configured BUTs whose outputs are monitored by two ORAs and compared with the outputs of two other BUTs in a circular comparison arrangement, as shown in Fig. 1. The ORAs are constructed from CLBs such that only half of the CLBs can be BUTs in a given test session and the positions of the BUTs and ORAs must be swapped during a subsequent test session in order to test all of the CLBs in half of the array.

The second half of the FPGA array is reserved for a MicroBlaze soft-core processor and any additional hardware resources associated with the processor [19]. Custom memory-mapped registers are included in the MicroBlaze VHDL model for interfacing with the BIST circuitry. One memory mapped write-only (WO) register is included for control of the BIST circuitry. The outputs of the register are connected directly to all inputs to the BIST logic. One read-only (RO) register is included at the same memory-mapped address as the output register. The inputs to this register are connected directly to the outputs of the BIST logic. Each register is general enough to be applicable to all BIST configurations that we have developed for Virtex-5. Finally, the MicroBlaze interfaces directly with the FPGAs ICAP for partial reconfiguration of the BIST array and for read back of output responses. To test all of the resources in the FPGA, a second configuration is generated with the location of the BIST logic and embedded processor

swapped. For BIST of some resources, such as input/output (I/O) tiles and cyclic redundancy check (CRC) circuits, it is possible to test all of the target resources simultaneously by placing the MicroBlaze around the BIST circuitry.

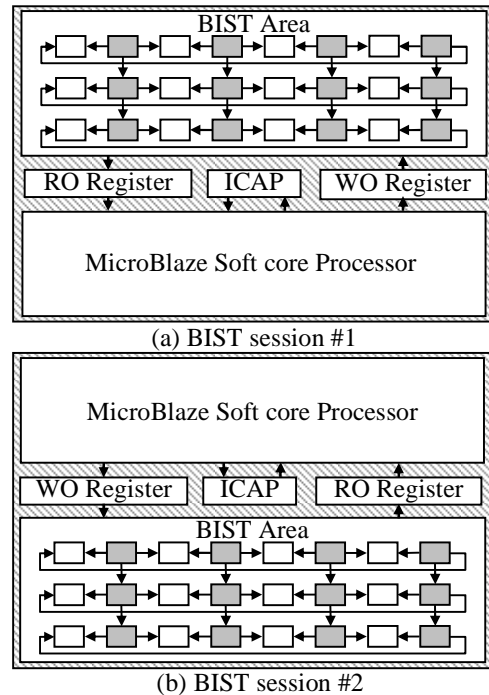


Figure 1. Simplified embedded soft processor-based BIST architecture.

III. RESULTS OF IMPLEMENTATION IN VIRTEX-5

The embedded processor-based BIST approach was implemented for BIST of Virtex-5 FPGAs using the MicroBlaze soft processor [19]. The unrouted embedded processor-based BIST configuration for the top CLBs implemented in the Virtex-5 LX30T is shown in Fig. 2. Note that two such configurations are implemented to fully test the top CLBs with the locations of the BUTs and ORAs swapped, and another two configurations are required to test the bottom half CLBs. For the purpose of embedded BIST, the MicroBlaze processor is configured with a hardware integer multiplier, five stage pipeline, and 64 kB of on-chip program and data memory (configured in Block RAMs). In Virtex-5 devices, the MicroBlaze with ICAP interface and BIST control registers occupies three DSPs, 16 block RAMs, and 400 CLBs. The percentage of utilized resources is less than 50% in the smallest Virtex-5 device such that the approach works for all FPGAs in the Virtex-5 family. Timing analysis indicates that the maximum operating frequency of the MicroBlaze processor when constrained to one-half of a device is greater than 100 MHz in all Virtex-5 devices. Therefore, all ICAP operations can be performed at the maximum clock frequency of 100 MHz.

For accurate measurements of test time to obtain experimental results with the MicroBlaze processor, an additional 32-bit hardware timer/counter was included in the MicroBlaze VHDL model. By starting the timer/counter at the

beginning of a test phase, and stopping it at the end of the test phase, the exact number of clock cycles for reconfiguration of the resources under test, test execution, and ORA read back can be determined. To extract the value in the timer/counter at the end of each test, the MicroBlaze performs a read of the timer/counter value and reports this number via a UART interface to a connected PC, where it is displayed and logged in a terminal program.

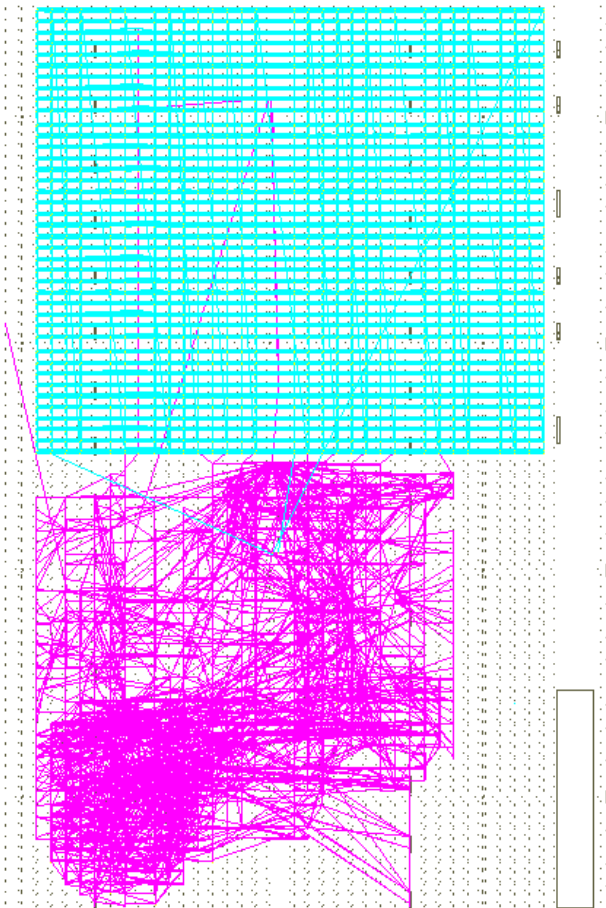


Figure 2. Unrouted embedded processor-based BIST configuration for top configurable logic blocks (CLB) in Virtex-5 LX30T viewed in FPGA Editor.

Fig. 3 shows the total test time for one session of CLB testing in several Virtex-5 devices for external configuration with full compressed configuration and partial compressed reconfiguration bitstreams downloaded and controlled via the 50 MHz Boundary Scan configuration interface and for the MicroBlaze embedded processor approach. These test times take into account all of the configurations required to achieve 100% fault coverage in the CLB in SliceL mode, as reported in [17], which used traditional external reconfiguration techniques. However, these times double to achieve 100% fault coverage in every CLB, because a second set of identically sized configurations are required with the locations of the BUTs and ORAs swapped. The optimized external reconfiguration provides the fastest overall test time when compared with the other two approaches since the entire array can be tested concurrently. This approach is approximately

twice as fast as the embedded processor approach, but is device dependent, as can be seen in Fig. 3. However, the embedded approach is significantly faster than external configuration with full or compressed bitstream download files.

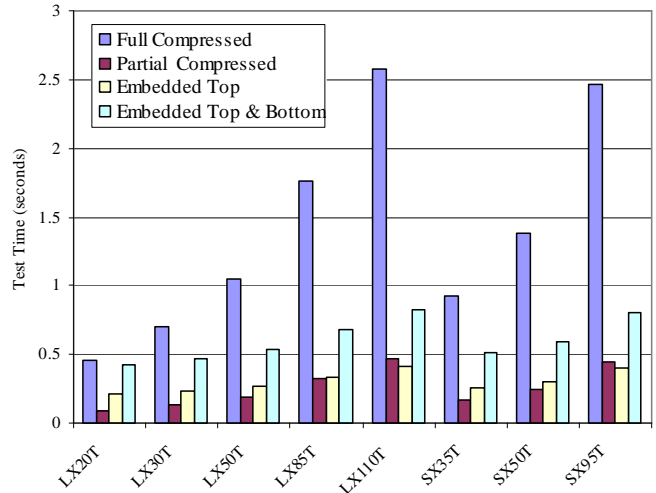


Figure 3. CLB BIST test time for external configuration (full compressed and partial compressed bitstreams) and embedded processor test time.

By studying the configuration file sizes for the two BIST approaches, the cause for the increase in test time for the embedded processor approach becomes clear. Consider Fig. 4, which shows the contributions to test time for one session of CLB BIST with the embedded processor approach. The contribution from the initial compressed full configuration download (using the 50 MHz external Boundary Scan configuration interface) is shown on bottom and the contribution from the five subsequent partial reconfigurations by the embedded processor (using the 100 MHz 32-bit ICAP) is shown on top. The overall test time is dominated by the initial download time. This is due, in part, to the slower serial Boundary Scan configuration interface; however, the main contributor to the overall test time is an increase in the size of the initial configuration file (relative to the traditional BIST approach). The cause of the size increase is due to the inclusion of the MicroBlaze configuration data in the configuration file. We observed that the inclusion of the MicroBlaze logic increased the size of the first compressed configuration file size by 2100 kB (which is approximately constant for all devices). The additional 2100 kB of configuration data is larger than the next five partial reconfiguration files combined, and, assuming a 50 MHz Boundary Scan configuration clock speed, increases the time for initial configuration by 336 ms. While it is possible to improve the timing for internal reconfiguration of the resources under test, there is no way to improve timing for the first compressed configuration download.

The potential for savings in test time does exist for systems which require fault diagnosis, and, therefore, read back of ORA contents at the end of each test phase. In this case, the embedded approach provides a speed-up of 5.4 times during read back of ORA results versus read back via Boundary Scan, as can be seen in Fig. 5.

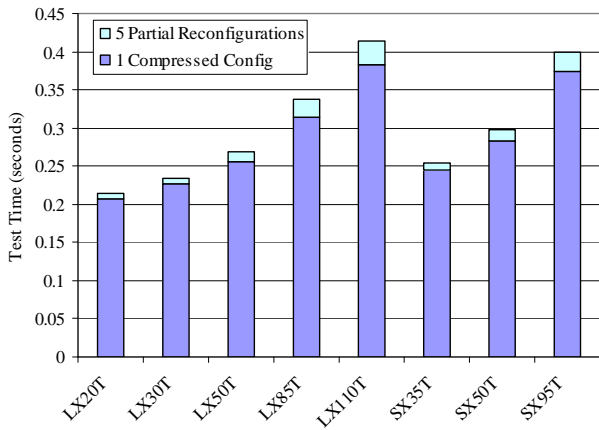


Figure 4. Contribution to embedded processor-based CLB BIST test time by initial external configuration and by five internal partial reconfigurations.

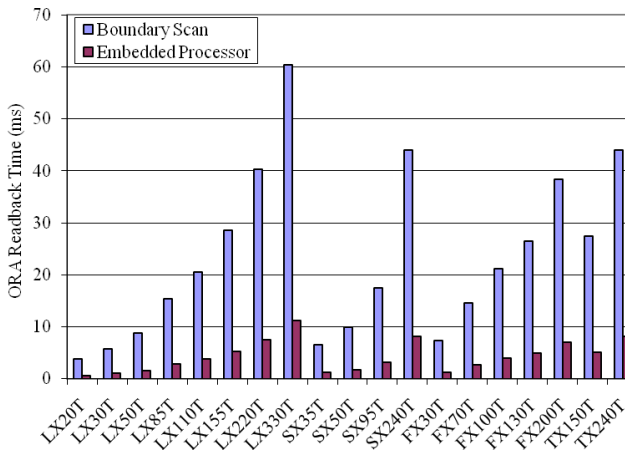


Figure 5. CLB BIST OR-A read back times for embedded processor-based approach and external boundary scan interface.

IV. FUTURE IMPROVEMENTS

The overall reconfiguration times for the embedded processor-based BIST approach can be reduced by modeling a custom processor for reconfiguration and test control. When a full custom embedded fault injection approach was compared to the MicroBlaze based fault injection presented in this work, a speed-up factor of almost 12 was observed for the FSM hardware-only approach versus the general-purpose processor-based approach. However, a hardware only implementation requires a different hardware configuration for every device and BIST session, as reported in [20]. Ultimately, with custom hardware, the reconfiguration time could approach the minimum achievable test time for the 100 MHz, 32-bit SelectMAP or ICAP configuration interface. This best case timing occurs when one word is read or written on each active edge of the clock, as is the case for configuration on a dedicated memory. The best case timing for CLB BIST east or west configurations is shown in Fig. 6 (doubling the time shown in the figure yields the total test time for all CLBs in SliceL mode). However, these times should not be directly compared to those in Fig. 4 for the embedded processor-based

approach, because Fig. 4 assumes initial configuration from the Boundary Scan interface. Another possibility is to clock the MicroBlaze at a frequency greater than 100 MHz, using a divided clock equal to 100 MHz for the ICAP and portions of the ICAP interface logic. This will, however, require the development of a custom ICAP interface. Based on timing analysis, clock frequencies around 150 MHz are attainable in the MicroBlaze processor when constrained to one-half of the FPGA. Therefore, a speed-up of approximately 1.5 times could be achieved using a multiple clock approach.

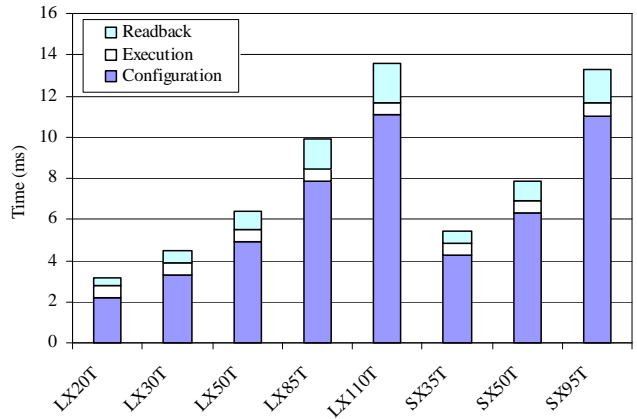


Figure 6. 32-bit, 100 MHz interface test time for full chip CLB west or east with one full compressed configuration and five partial reconfigurations.

V. OTHER APPLICATIONS

An approach similar to the embedded processor-based BIST could be applied to an external processor or microcontroller connected to the SelectMAP 32-bit configuration interface. Conceptually, the approach is similar to the approach for Atmel SoCs [9][10], except that the processor and FPGA are integrated at the board level, rather than at the chip level. The only overhead required above that for the traditional BIST approach is processor downtime for the test, additional circuit board interconnections, additional processor I/O, and a portion of the processor program memory (16,558 Bytes for one session of CLB BIST) for storing the embedded BIST software and reconfiguration data. The impact to the system could be minimized by performing tests of the FPGA as a low priority, background task, at the expense of increased test time. The approach could provide the 5.4 times speed-up of the embedded processor during reconfiguration and read back using the 32-bit, 100 MHz SelectMAP configuration interface without the penalty incurred by testing the FPGA in two sessions, one for each half. The size of the initial download is also reduced when compared to the embedded processor-based approach due to the highly optimized structure of the BIST circuitry. Furthermore, the memory required to store the BIST configurations can be reduced at the expense of some additional program memory in the hard processor.

The embedded processor-based BIST approach for Virtex-5 FPGAs is directly applicable to Virtex-4 FPGAs [21] with some modification to the BIST specific software (including device specific subroutines such as algorithmic resource under

test frame address generation) and stored configuration data. Differences between the Virtex-4 and Virtex-5 ICAP interfaces, such as byte-swapping on the Virtex-5 ICAP, are accounted for during synthesis of the MicroBlaze and associated ICAP interface circuitry based on the targeted device family. The frame address register is also arranged differently in Virtex-4 and Virtex-5, but this can be accounted for in software [22][23]. The overall test times for Virtex-4 relative to external reconfiguration closely match those results obtained in Virtex-5.

VI. CONCLUSIONS

We have presented the results of a case study which implements the first soft-core embedded processor-based BIST approach. The approach is applicable to any FPGA with write/read access to the configuration memory from within the FPGA fabric and with sufficient configurable resources to implement both the soft-core processor and the BIST circuitry. The number of external configurations of the FPGA during any BIST session is reduced to a maximum of two (one for each half of the array) and internal reconfiguration of the resources under test are performed at the maximum allowable clock frequency and data width. Read back of ORA contents and diagnosis of faulty resources under test can be performed by the embedded soft-core processor when fault diagnosis is desired, for fault-tolerant applications for example, providing a speed-up of 5.4 versus readback via the Boundary Scan interface. The approach can significantly decrease the overall test time in systems with a relatively slow external configuration interface, as was the case for the previous implementation of embedded processor-based BIST using a dedicated hard-core processor [10].

The soft-core processor approach was implemented in Virtex-5 FPGAs using the MicroBlaze processor for BIST reconfiguration, control of execution, fault injection, and fault diagnosis. Reconfiguration of the resources under test is achieved via the ICAP port in the FPGA fabric. When implemented in Virtex-5, the approach requires more testing time when compared with optimized external reconfiguration using compressed partial reconfiguration bitstreams. This is primarily due to the fact that the overall BIST approach has been architected for optimum configuration file compression. This includes orienting the BIST architecture with the configuration memory for maximizing the effectiveness of compressed download files with multi-frame write features, partial reconfiguration of the resources under test by maintaining constant placement and routing between test phases, and a single pass/fail indication to avoid partial configuration memory read back for BIST results. This is a testament to the advanced state of FPGA BIST techniques as well as the features and capabilities offered by FPGA manufacturers to decrease configuration times.

However, the soft-core processor approach is significantly faster than configuration with full or compressed configuration bitstreams alone. Only two downloads are required for each BIST session when the embedded processor-based approach is used, compared to six configurations for CLB east/west tests and nine for SerDes tests for example. BIST control, execution and fault diagnosis implemented in the embedded processor

eliminate the need for complex external test equipment for manufacturing testing and intelligent external BIST controllers for in-system testing and diagnosis in fault-tolerant applications. The architecture is applicable to any BIST for Virtex-4 and Virtex-5 FPGAs without modification of the embedded processor hardware; only the MicroBlaze program memory contents need to be changed.

REFERENCES

- [1] B. Dutton and C. Stroud, "Soft-core Embedded Processor Based Built-In Self-Test of FPGAs," *Proc. IEEE Int. Symp. On Defect and Fault Tolerance in VLSI Systems*, pp. 29-37, 2009.
- [2] *Virtex-5 FPGA User Guide*, UG190(v4.2), Xilinx, 2008.
- [3] L-T Wang, C. Stroud, and N. Touba, *System-on-Chip Test Architectures*, San Francisco, CA: Morgan Kaufmann, 2007.
- [4] S. Toutouchi and A. Lai, "FPGA Test Coverage," *Proc. IEEE Int. Test Conf.*, pp. 1248-1257, 2003.
- [5] M. Abramovici et. al., "BIST-Based Test and Diagnosis of FPGA Logic Blocks," *IEEE Trans. on VLSI Systems*, vol. 9, no. 1, pp. 159-172, 2001.
- [6] J. Nall et. al., "BIST-Based Diagnosis of FPGA Interconnect," *Proc. IEEE Int. Test Conf.*, pp. 618-627, 2002.
- [7] J. Bailey et. al., "Bridging Fault Extraction from Physical Design Data for Manufacturing Test Development," *Proc. IEEE Int. Test Conf.*, pp. 760-769, 2000.
- [8] K. Leach et. al., "BIST for Xilinx 4000 and Spartan Series FPGAs: A Case Study," *Proc. IEEE Int. Test Conf.*, pp. 1258-1267, 2003.
- [9] S. Garimella et. al., "On-Chip BIST-Based Diagnosis of Embedded Programmable Logic Cores in System-On-Chip Devices," *Proc. ISCA Int. Conf. on Computers and Their Applications*, pp. 308-313, 2005.
- [10] J. Sunwoo and C. Stroud, "Built-In Self-Test of Configurable Cores in SoCs Using Embedded Processor Dynamic Reconfiguration," *Proc. Int. SoC Design Conf.*, pp. 174-177, 2005.
- [11] S. Vemula et. al., Built-In Self-Test for Programmable I/O Buffers in FPGAs and SOCs, *Proc. IEEE Southeastern Symp. on System Theory*, pp. 534-538, 2006.
- [12] S. Dhingra et. al., "Built-In Self-Test for Virtex and Spartan II FPGAs Using Partial Reconfiguration," *Proc. IEEE North Atlantic Test Workshop*, pp. 7-14, 2005.
- [13] D. Milton et. al., "Embedded Processor Based Built-In Self-Test and Diagnosis of Logic and Memory Resources in FPGAs," *Proc. Int. Conf. on Embedded Systems and Applications*, pp. 87-93, 2006.
- [14] B. Garrison et. al., "Built-In Self-Test for Memory Resources in Virtex-4 FPGAs," *Proc. ISCA Int. Conf. on Computers and Their Applications*, pp. 63-68, 2009.
- [15] M. Pulukuri and C. Stroud, "Built-In Self-Test of Digital Signal Processors in Virtex-4 FPGAs," *Proc. IEEE Southeastern Symp. on System Theory*, pp. 34-38, 2009.
- [16] J. Yao et. al., "Built-In Self-Test of Programmable Interconnect in Virtex-4 FPGAs," *Proc. IEEE Southeastern Symp. on System Theory*, pp. 29-33, 2009.
- [17] B. Dutton and C. Stroud, "Built-In Self-Test of Configurable Logic Blocks in Virtex-5 FPGAs," *Proc. IEEE Southeastern Symp. on System Theory*, pp. 230-234, 2009.
- [18] B. Dutton and C. Stroud, "Built-In Self-Test of Programmable Input/Output Tiles in Virtex-5 FPGAs," *Proc. IEEE Southeastern Symp. on System Theory*, pp. 235-239, 2009.
- [19] *MicroBlaze Processor Reference Guide*, UG081(v.9.0), Xilinx, 2008.
- [20] B. Dutton et. al., "Embedded Processor Based Fault Injection and SEU Emulation for FPGAs," *Proc. Int. Conf. on Embedded Systems and Applications*, pp. 183-189, 2009.
- [21] *Virtex-4 FPGA User Guide*, UG070 (v2.5), Xilinx, 2008.
- [22] *Virtex-4 FPGA Configuration User Guide*, UG071(v1.1), Xilinx, 2008.
- [23] *Virtex-5 FPGA Configuration User Guide*, UG191(v2.7), Xilinx, 2008.
- [24] R. Rajsuman, "Testing a System-On-Chip with Embedded Microprocessor," *Proc. IEEE Int. Test Conf.*, pp. 499-508, 1999.