Implementing a Scheme for External Deterministic Self-Test

*Abdul Wahid Hakmi, Hans-Joachim Wunderlich, Valentin Gherman
**Michael Garbers, Jürgen Schlöffel

**Universität Stuttgart, Pfaffenwaldring 47, D-70569 Stuttgart, Germany*
***Philips Semiconductors GmbH, Georg-Heyken-Strasse 1, D-21147 Hamburg, Germany*
**(abdul.hakmi, wu, ghermanv)@informatik.uni-stuttgart.de*
***(michael.garbers, juergen.schloeffel)@philips.com*

Abstract

A method for test resource partitioning is introduced which keeps the design-for-test logic test set independent and moves the test pattern dependent information to an external, programmable chip. The scheme includes a new decompression scheme for a fast and efficient communication between the external test chip and the circuit under test. The hardware costs on chip are significantly lower compared with a deterministic BIST scheme while the test application time is still in the same range. The proposed scheme is fully programmable, flexible and can be reused at board level for testing in the field.

Keywords: *Deterministic self-test, external BIST, test resource partitioning, test data compression.*

1. Introduction

Test trade-offs include test coverage, test application time, costs of automatic test equipment (ATE) and design-for-test costs. While certain built-in self-test (BIST) schemes reduce ATE costs significantly, they may affect fault coverage, test time and hardware overhead. Test resource partitioning schemes like compression and decompression look for a compromise between hardware overhead, test time and ATE costs by using low-cost testers [3, 14, 15, 17]. Unlike BIST, these schemes do not support the reuse of the test hardware in the field.

So-called built-out self-test (BOST) schemes move large parts of the test hardware to an external chip, which generates test patterns and evaluates the responses of the circuit under test (figure 1). If the test chip is programmable, it may be put onto a board for testing several devices and by following the IEEE 1149.1 boundary scan standard, the board test is rather straightforward (figure 2).

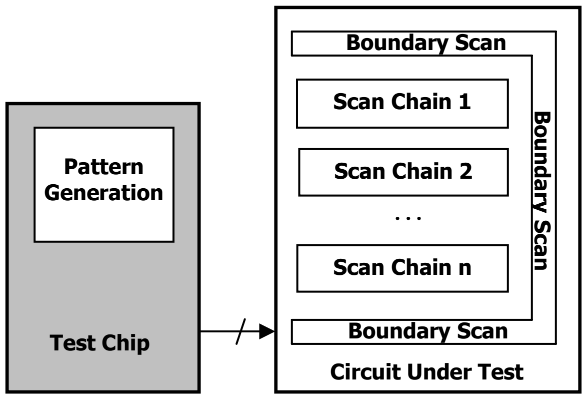

Figure 1. The BOST scheme

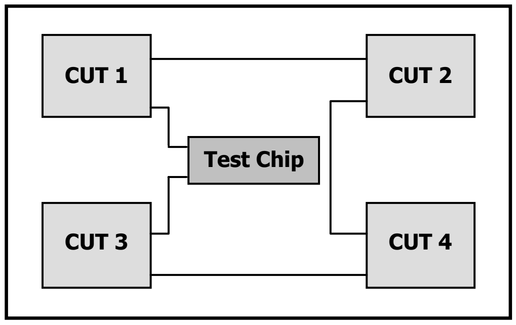

Figure 2. BOST at board level

This type of test chips has been proposed for the last two decades. The first approach supported random pattern testing [2, 6] which requires random pattern testable CUTs and very often rather long test application times, too. Test application times were reduced by weighted random pattern testing [16], but in some cases a rather large number of weights had to be stored on the test chip and put limits to the feasibility.

Less memory and shorter test length are required by test chips supporting pseudo-exhaustive testing [9], and they may have benefits concerning defect coverage, but they put additional constraints on the CUTs as their cones must not exceed certain limits and hardware segmentation may be required.

A test chip producing precomputed deterministic patterns has not been considered feasible up to now as the amount of test patterns and consequently the memory requirements increase with the CUT size. In this paper it is shown that a mixed mode BIST scheme combining random patterns and deterministic patterns is also suited for external testing.

The goal of this paper is an external deterministic BIST strategy which does not touch the mission logic by segmentation cells or test points but requires just scan design. Since it is too costly to store a complete deterministic test set on the test chip, a mixed-test is implemented which is based on bit-flipping [18] and encoded bit-flip information stored off-chip [12]. Appropriate hardware development is required for both sides i.e. off-chip and on-chip and will be described in section 2.

In order to minimize the transfer of information from off-chip to on-chip the bit-flipping information is stored in a compressed form. The use of conventional decompression methods may lead to long test times, consequently new decompression architecture is proposed that is independent of the test set and has the ability to generate a complete encoded block in a single clock cycle. The compression and decompression methods will be described in section 3. In section 4, we will describe the communication protocol and the testing procedure, while section 5 will discuss the experimental results.

2. Target structure

The self-test circuitry is partitioned between off-chip and on-chip (see figure 3) and a communication protocol is defined. The selection of the resources to be built externally has a direct impact on the hardware overhead and the testing time.

The bit-flipping logic (BFL) changes a few bits of the random vectors generated by the LFSR in order to produce precomputed deterministic patterns. In [18] it is shown, that random vectors can be found which leave most of the bits of the BFL not specified, the remaining bits are mostly "0" as they match with the random vectors, and only very few bits must be "1".

The main idea of the external deterministic BIST is to compress and store the BFL outputs in an external-chip memory (3(a)) and to generate the BFL information on-chip by a small but fast decoder while other test circuitry is still on-chip (3(b)). This idea is based on several observations:

- It is impractical to transfer completely specified deterministic patterns from the external chip.
- Only the bit-flipping logic (BFL) depends on the test set while the other hardware only depends on the number of scan chains in the design.
- A dramatic reduction of the hardware is possible if the BFL is removed.
- As very few bits are needed to be flipped, the majority of the BFL outputs are either don't care (X) or 0 and have high potential for compression.

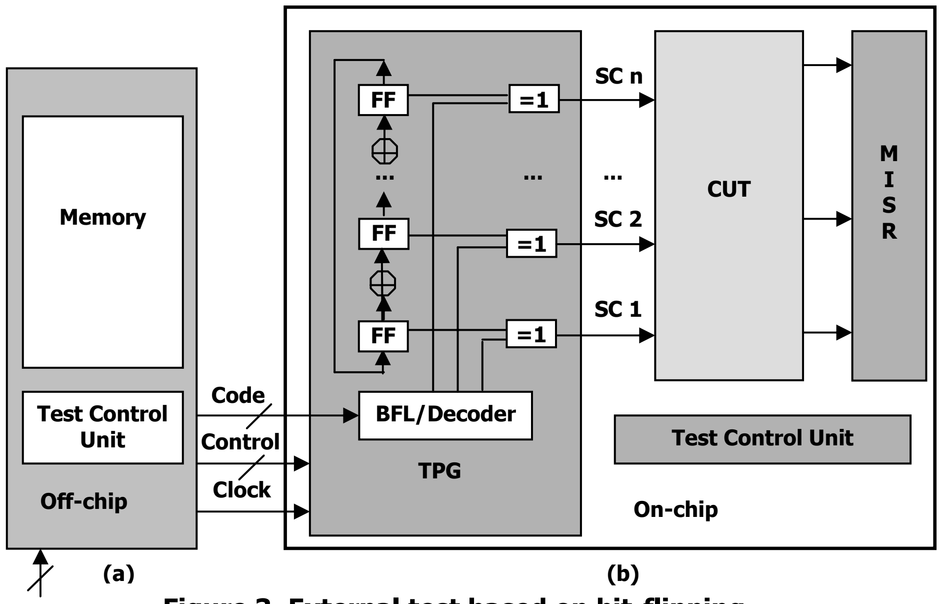

Figure 3. External test based on bit-flipping

- The LFSR, the XOR gates, the MISR and the Control Unit do not add much hardware overhead on-chip while their implementation externally would require extensive communication and very long test times.

3. Compression and decompression of bit-flip information

There is a plethora of test data compression/ decompression schemes available which could be applied to the external BIST scheme. Some of them are investigated in [1], and it has been shown that reseeding is optimal with respect to entropy [10, 13] under general assumptions. But as already pointed out, the bit-flipping output is strongly biased towards 0 and contains mainly don't cares so that other schemes look more attractive [5, 8, 11, 12, 15]. These schemes would optimize throughput and data volume while the main objective of deterministic BIST is test time under throughput and memory constraints.

To meet the speed target, new decompression architecture has to be developed which is able to generate the run-length blocks in parallel and not serially. Such a decoder along with a compression method will be presented in this section.

3.1. Compression algorithm

The compression algorithm described below exploits the fact that the majority of the BFL outputs are unspecified while among the others the number of "0"s is always much larger than the number of "1"s. We define a few basic terms:

X is used for representing a "don't care". **SC** represents the number of scan chains in the CUT.

A BFL vector *v* consists of the BFL output that converts a random test vector t_R into a deterministic test vector t_D where t_R and t_D belong to a random test set T_R and deterministic test set T_D for some design, rsp. A set of BFL vectors that changes a complete T_R into T_D will be referred as **BFL set** denoted by *V*. The lengths of a BFL vector *v*, a random vector t_R and a deterministic vector t_D are equal to SC, and the number of vectors in *V* is equal to the vectors in T_R and T_D. Each t_{Ri} has a corresponding v_i that converts it to t_{Di}. Figure 4 shows an example of the BFL set (*V*) along with the associated random test set (T_R) and deterministic test set (T_D) for a design with SC=4. All sets contain 6 vectors each of length 4. To form a deterministic vector t_{Di}, an XOR operation is performed between the bits $b_{vi,j}$ and $b_{Ri,j}$ of v_i and t_{Ri} respectively.

0 1 X X	0 1 0 0	0 0 X X	$t_{D1} = t_{R1} \oplus v_1$
X 0 X 0	1 0 1 1	X 0 X 1	where
X 0 X X	0 1 0 1	X 1 X X	$b_{D1,1} = b_{R1,1} \oplus b_{v1,1}$
			$b_{D1,2} = b_{R1,2} \oplus b_{v1,2}$
0 0 X X	0 0 1 0	0 0 X X	$b_{D1,3} = b_{R1,3} \oplus b_{v1,3}$
X X X X	1 0 0 0	X X X X	$b_{D1,4} = b_{R1,4} \oplus b_{v1,4}$
1 0 0 X	1 0 1 0	0 0 1 X	
T_D	T_R	V	

Figure 4. BFL vector and BFL set

To illustrate the compression method, assume we are implementing a test using p patterns each of length m × SC. Then the corresponding random test set T_R will contain m × p test vectors each of length SC, and the corresponding BFL set *V* will contain the same number and size of vectors. For example in figure 4, SC=4, p=2 and m=3, so each set contains 6 vectors each of length 4. We compress *V* and store it off-chip while T_R will be generated by the on-chip LFSR. As the BFL set is biased towards 0s, we will encode the 0 run-lengths.

First all the X's in the BFL set are replaced by 0s. In the next step we divide each BFL vector of the BFL set into runs of 0s having lengths equal or less than SC and call them blocks. If a BFL vector *v* ends with a 0, an imaginary 1 is assumed at its end. The only purpose for this imaginary 1 is to represent each vector as the sequence of 0 runs, each of which stops with a 1 at the end. Stopping each 0 run with 1 reduces the maximum number of different 0 runs from 2SC to SC+1. This not only helps in reducing the states of the decoder to make it small and simple but also provides a way to generate a complete 0 run within a single clock cycle. This will be shown in section 3.2. If all the SC bits are 0 in a BFL vector then we have just one block of length SC+1 and if all are 1 then we have SC blocks each of length 1.

In the next step a dictionary of these 0 run-length blocks with their occurrences is built. Using the dictionary we assign a binary code to each block such that the block with the highest frequency gets the shortest codeword. Although the Huffman codes might offer better compression they are not used as they require a hardwired decoder on-chip which depends on the BFL set *V*.

3.2. Decompression architecture

State-of-the-art decompression approaches e.g. [5, 8, 11, 12, 15] regenerate a block serially and use of them in external deterministic test will result in very long test application time. Hence, a parallel decompression technique had to be developed that produces the complete run-length block in a single clock cycle as soon as the decoder detects its codeword. As the vast majority of the BFL output vectors are X's they contain only one run-length block and most of the time the complete BFL vector is generated in a single clock cycle. This parallelism of the decoder does not only reduce the test time but also eliminates the need of the synchronisation signals used in previous techniques to stop the transfer of codes off-chip to on-chip when the decoder is busy.

Figure 5 shows the proposed decompression architecture. It mainly consists of a binary decoder, a small RAM and a generate unit. The binary decoder receives a codeword serially from the external chip and produces the address of the RAM word that stores the length of the encoded block. The length is passed to the generate unit that regenerates the run-length block. When the BFL vector is complete it is passed to the XOR inputs. Detection of a binary code, reading of a RAM word and generation of a run-length block can be performed in parallel.

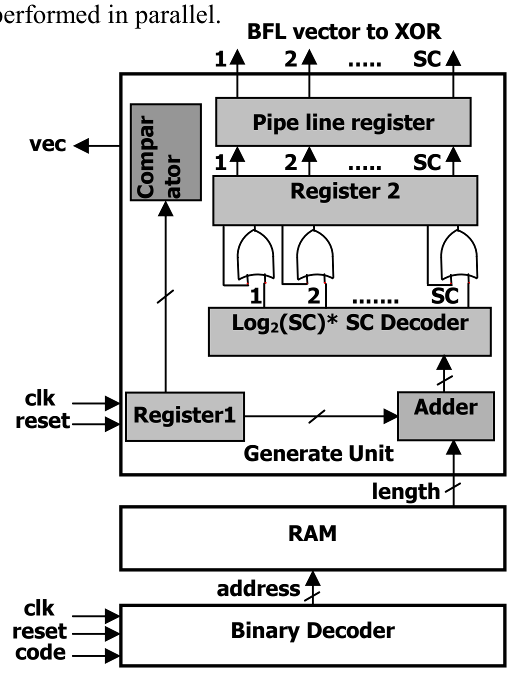

Figure 5. Decompression architecture

The binary decoder is a simple finite state machine consisting of at most SC states and implemented by $\log_2|SC|$ flip-flops. It receives a code bit by bit through the code line, and when the code is complete it generates an address of a RAM word. After detection of the shortest code (i.e. 0) the decoder gives the address of the first memory word, after the second shortest (i.e. 10) the address of the second memory word and so on. If the code is not complete, the address of the last RAM word appears at the decoder output. It depends on SC, i.e. number of scan chains in the design, and it is independent of the BFL and the test set.

The RAM stores the length of the blocks in descending order of their frequencies. In this way, the length of the block with the highest frequency and the shortest code (i.e. 0) is always stored in the first word of the RAM, the length of the block with the second shortest code (i.e. 10) is in the second word and so on. The last memory word, the address of which is produced during the reception of a code, always stores 0. The length of each RAM word is $\lceil \log_2 (SC+1) \rceil$, the maximum number of required words is SC+2, and the size of the maximally required memory becomes $(SC+2)\times\lceil \log_2 (SC+1) \rceil$ bits. If the BFL set and the frequencies of the different 0 run length blocks are changed, the RAM words can be reloaded according to the new frequencies. In this way, the decoder is independent of the BFL.

The generate unit mainly consists of an adder, a comparator and some registers. The Register1 is used to store the number of generated bits of the BFL vector while the Register2 stores the regenerated BFL vector. The Register1 is $\lceil \log_2 (SC+1) \rceil$ bits wide and the Register2 is SC bits wide. Initially all the registers are set to 0. The length of an encoded block is read from the RAM and passed to the adder. It adds this length to the contents of the Register1 and stores back the result in the Register1 while flipping the corresponding bit from 0 to 1 in Register2. This process continues until the comparator detects that the value of the Register1 is greater or equal to SC i.e. complete BFL vector has been generated and registers are reset to 0. Here simple OR gates are used to flip the bits of Register2 from 0 to 1 and also to retain the decoded blocks during the generation of the next ones.

4. Test configuration

The IEEE 1149.1 JTAG boundary scan standard is used to transfer data from the test chip to the CUT. This standard allows adding optional instructions, and for the external deterministic self-test only one extra instruction named BFTEST is added. Whenever this instruction is present in the instruction register and the TAP controller reaches the Run-Test state, it sends a start signal to the on-chip controller and the TDI is connected with the binary decoder.

The testing procedure starts by setting the start test input of the test chip. The test chip controller transfers the BFTEST instruction into the instruction register and changes the TAP controller mode to Run-Test. When the TAP controller reaches this state, code word transfer starts. The codes are read from the test chip memory and are sent to the binary decoder of the CUT through TDI. The binary decoder detects a codeword and provides the address of the memory word containing the length of the encoded block. This length is passed to the generate unit that regenerates this block using the procedure described in section 3.2. When the BFL vector is complete, the comparator sends a signal to the on-chip controller that sets the scan line to shift XOR outputs into scan chains. The controller also decrements the bit counter and changes the state of the LFSR. When the bit counter reaches 0, the pattern counter is decremented by 1 and one system clock is applied. Shift in a new vector and shift out responses go in parallel. This process continues until the pattern counter reaches 0.

The test chip is implemented on a prototyping board. For this purpose, we used a Xilinx board, of the type xc2v2000-ff896-4. This board contains a Field Programmable Gate Array (FPGA) from the Virtex-II family, which offers place for 2 millions gate-equivalents logic and a 1008 KBit Block-RAM [19].

5. Experimental results

To prove the efficiency of this external self-test technique, experiments were performed for the ISCAS benchmark circuits and some industrial circuits [7]. The industrial circuits are represented by a leading p. The memory requirements and testing time were calculated by implementing the scheme in C++. To compute the on-chip hardware overhead, on-chip test hardware was modeled using VHDL.

Table 1 shows the reduction in the test hardware size by using the proposed scheme compared to the deterministic logic BIST (DLBIST) [7]. The hardware size is computed by using a free library independent of the technology. In the external self-test (EXTEST) the bit-flipping logic (BFL) on-chip is replaced with a fixed size decoder while other components (e.g. LFSR, MISR, Bit counter, Pattern counter etc) are similar in both cases. The sub column named FIX represents the area required by these components, the test hardware is reduced in the range from 38 to 99.5%. Due to the fact that the size of the decoder is constant with respect to the number of scan chains and does not depend on circuit complexity, the hardware savings are largest for the most complex circuits. For example, p1 is more complex than s13207 but both contain the same number of scan chains (SC=10) and so the same size of the on-chip decoder (299 cells). The same is true for p2 and s38417.

Table 2 reports the gain of the external BIST architecture in terms of memory requirement and test application time. Here column header Off-chip Mem shows the size of the external chip memory needed to store BFL outputs for 10,000 patterns. It can be seen that even large industrial circuits require so less memory that a test chip can easily be implemented using an ordinary prototype board.

The column named Required Clock Cycles for Test Application tells the number of clocks required by different techniques to complete a test containing 10,000 patterns. Here it is assumed that on-chip and external chip frequencies are identical. The sub column named DLBIST represents the number of clocks required in the DLBIST approach. The sub column EXTEST shows the number of clocks when self test is implemented using the proposed scheme while the last sub column Serial shows the number of clocks, if the BFL set is compressed and decompressed by using state of the art run length compression/decompression methods [4, 5, 8, 12]. Only the ideal case for these schemes is considered when there is no data transfer overhead. The lower bounds for all are equal because their decoders create the vector serially using counters. The external self test is significantly faster compared to

Table 1. Test hardware reduction using external self test in comparison with DLBIST

Design	SC	Area DLBIST (cells)			Area EXTEST (cells)			Test Hardware Reduction [%]
		BFL	FIX	Total	Decoder	FIX	Total	
c7552	4	626	128	754	188	128	316	58.0
s13207	10	596	168	764	299	168	467	38.9
s15850	9	977	162	1139	287	162	449	60.6
s9234	4	505	128	633	188	128	316	50.0
s38417	19	2244	232	2476	470	232	702	71.6
p1	10	137847	193	138040	299	193	492	99.6
P2	19	5178	257	5435	470	257	727	86.6
P3	29	12161	323	12484	608	323	931	92.5
P4	32	199770	348	200118	680	348	1028	99.5

available techniques and is close to the time of internal BIST schemes.

Table 2. Memory and clocks required for completing a test with 10,000 patterns

Design	SC	Off-Chip Mem [MB]	Required Clock Cycles for Test Application ($\times 10^6$)		
			DLBIST	EXTEST	Serial
c7552	4	0.09	0.771	0.773	3.085
s13207	10	0.10	0.857	0.860	8.565
s15850	9	0.10	0.874	0.880	7.867
s9234	4	0.09	0.737	0.738	2.947
s38417	19	0.11	0.936	0.957	17.790
p1	10	0.86	5.210	7.195	52.095
P2	19	0.62	5.154	5.199	97.919
P3	29	0.10	0.499	0.806	14.483
P4	32	1.25	3.007	10.456	96.226

6. Conclusions

A low cost external self-test scheme has been implemented for deterministic testing. It combines the benefits of high fault coverages of deterministic test, low test application times of internal deterministic BIST and hardware savings of external self-test.

The reduction of the test time is possible due to a novel decompression scheme which generates a run-length block in parallel. Moreover, the decompression architecture on-chip is programmable and independent of the test set.

Acknowledgements

This research work was supported by the German Federal Ministry of Education and Research (BMBF) in the Project AZTEKE under the contract number 01M3063C.

We would like to thank to Yuyi Tang, Günter Bartsch and Tobias Bergmann from the University of Stuttgart for valuable discussions and help.

References

[1] K.J. Balakrishan, N.A. Touba "Relating Entropy Theory to Test Data Compression", *IEEE Proceedings of the European Test Symposium (ETS)*, 2004, pp. 187- 192.

[2] P.H. Bardell, W.H. McAnney "Self-testing of multichip logic modules", *Proceedings IEEE International Test Conference (ITC)*, 1982, pp. 200-204.

[3] C. Barnhart et all. "OPMISR: The foundation for compressed ATPG vectors", Proceedings of International Test Conference (ITC), IEEE, 2001, pp.748-757

[4] A. Chandra, K. Chakrabarty "Test Data Compression for System-on-a-Chip Using Golomb Codes", *Proceedings of VLSI Test Symposium (VTS)*, 2000, pp. 113-120.

[5] A. Chandra, K. Chakrabarty "Frequency-Directed Run Length (FDR) Codes with Application to System-on-a-Chip Test Data Compression", *Proceedings VLSI Test Symposium (VTS)*, 2001, pp. 42-47.

[6] E.B. Eichelberger, E. Lindbloom "Random-Pattern Coverage Enhancement and Diagnosis for LSSD Logic Self-Test", *In IBM J. Res. Develop., Vol.27, No.3*, May 1983.

[7] V. Gherman, H.-J. Wunderlich, H. Vranken, F. Hapke, M. Wittke, M. Garbers, "Efficient Pattern Mapping for Deterministic Logic BIST", *International Test Conference (ITC)*, *IEEE*, 2004, (to appear).

[8] P. Gonciari, B. Al-Hashimi, N. Nicolici „Improving Compression Ratio, Area Overhead, and Test Application Time for System-on-a-Chip Test Data Compression/Decompression", *Proceedings Design, Automation and Test in Europe Conference and Exhibition (DATE)*, 2002, pp 604-611.

[9] S. Hellebrand, H.-J.Wunderlich, O.F. Haberl "Generating Pseudo-Exhaustive Vectors for External Testing", *Proceedings IEEE International Test Conference (ITC)*, 1990, pp. 670-679.

[10] S. Hellebrand, S. Tarnick, J. Rajski, B. Courtois "Generation of Vector Patterns Through Reseeding of Multiple-Polynomial Linear Feedback Shift Registers", *Proceedings of International Test Conference (ITC)*, 1992, pp. 120-129.

[11] A. Jas, J. Gosh-Dastidar, N.A. Touba "Scan Vector Compression/Decompression Using Statistical Coding", *Proceedings VLSI Test Symposium (VTS)*, 1999, pp. 114-120.

[12] D. Kay, S. Mourad "Compression Technique for Interactive BIST Application", *Proceedings of 19th VLSI Test Symposium (VTS)*, Marina Del Rey, CA, USA, 2001, pp. 9-14.

[13] B. Koenemann "LFSR-Coded Test Patterns for Scan Designs", *Proceedings of European Test Conference (ETS)*, 1991, pp. 237-242.

[14] J. Rajski, J. Tyszer, M. Kassab, N. Mukherjee "Embedded deterministic test", *IEEE Trans. on CAD*, 23, May 2004, pp. 776–792.

[15] S. Reda, A. Orailoglu "Reducing Test Application Time Through Test Data Mutation Encoding", *Proceedings of the 2002 Design and Test in Europe Conference and Exhibition (DATE'02)*, pp. 387-393.

[16] A. Ströle, H.-J. Wunderlich "TESTCHIP: A chip for weighted random pattern generation, evaluation, and test control", *IEEE Journal of Solid State Circuits,* Vol. 26, No. 7, July 1991, pp. 1056-1063.

[17] E.H. Volkerink, A. Khoche, S. Mitra "Packet-based Input Test Data Compression Techniques", *Proceedings IEEE International Test Conference (ITC)*, 2002, pp. 154-163.

[18] H.-J. Wunderlich, G. Kiefer "Bit-Flipping BIST", *Proceedings International Conference on Computer-Aided Design (ICCAD), IEEE*, 1996, pp. 337-343.

[19] http://www.xilinx.com