Structural Testing of Modern Reconfigurable Chips

Michel Renovell

STRUCTURAL TESTING OF MODERN RECONFIGURABLE CHIPS

RENOVELL M.

Laboratoire d'Informatique, Robotique et Microélectronique de Montpellier, LIRMM-UM2 161 Rue Ada 34392 Montpellier France; renovell@lirmm.fr Tel (33)467418523 Fax (33)467418500

Abstract. This paper presents recent developments for testing SRAM-based FPGAs using a structural approach. The specific architecture of these new chips is first presented identifying the specific FPGA test problems as well as the FPGA test properties. The FPGA architecture is then conceptually divided into different architectural elements. For each architectural element test configurations and test vectors are derived targeting the assumed fault models.

1. Testing Reconfigurable Circuits

Field Programmable Gate Arrays (FPGAs) are digital devices that can implement logic circuits by programming the required function [1,2]. Because of their short turnaround time and programmability in the field, they have been widely used for rapid prototyping or reconfiguration of complex digital systems. One important class is the SRAM-based FPGAs which can be easily reprogrammed any number of times. This paper focus on SRAM-based FPGAs. In such a programmable circuit, an array of logic cells and interconnection cells can be configured in the field to implement a desired designed function.

Testing for FPGAs as well as conventional digital ICs, is a difficult and important problem [3-25]. A very simple approach consists in considering the FPGA as a classical digital ASIC making the problem of testing the FPGA equivalent to the problem of testing a classical ASIC. Such a classical test approach fails when applied on FPGA. The main reason is because FPGAs have a heterogeneous architecture mixing interconnect elements, logic modules and RAM cells. Indeed, usual digital VLSI circuits include a few number of large blocks, each block being of a specific nature: a random logic block, a RAM block, buses… In a classical digital test approach, a specific test technique is consequently used for each block according to its nature. On the contrary, typical FPGAs can be viewed as composed of a large number of small blocks, all the blocks are similar but include logic modules, RAM cells and interconnect elements. These heterogeneous elements are strongly interconnected making the usual test techniques difficult to apply. As the matter of fact, a typical heterogeneous FPGA block can be viewed as a good recollection of all the problems encountered in testing:

a) Sequential elements (flip-flops),

b) Mixed architecture (logic/interconnect/RAM),

c) Multiple representation level (module/gate/transistor)

d) Fault model definition (stuck/short/open/memory).

To face the difficulty of handling an heterogeneous block, a practical solution consists in conceptually dividing the heterogeneous block into different homogeneous sub-blocks as represented in figure 1. An interesting solution can be to divide the heterogeneous block into the following homogeneous blocks:

a) The logic cell,

b) The interconnect cell,

c) The RAM cells.

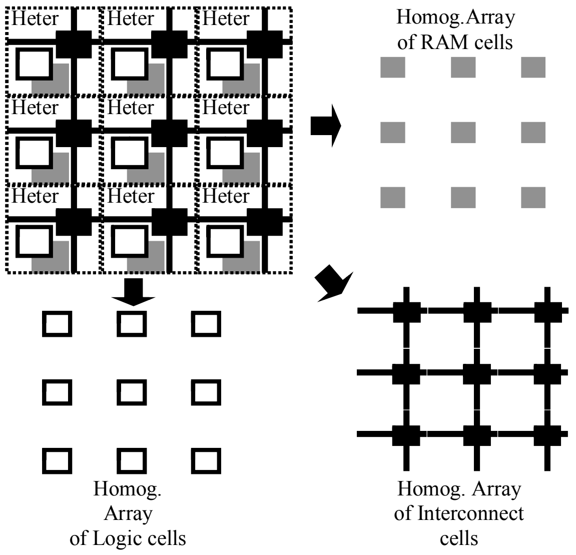
Heter
Homog. Array of RAM cells
Homog. Array of Logic cells
Homog. Array of Interconnect cells

Fig. 1. Homogeneous Arrays

Then, a specific and adequate test approach can be applied to each homogeneous cell according to its nature. But, it is obvious that each specific test approach must take into account the connection of the considered cell with the other ones. For example, the test approach proposed for the RAM cells must account for the connections between the RAM cells and the logic cell and interconnect cell.

These remarks try to give a general idea of the complexity of FPGA testing. On the other side, typical SRAM-based FPGAs have an important test property that must absolutely be considered and used in a test approach. As previously mentioned, almost all the blocks into the FPGAs are similar meaning that the small heterogeneous block is repeated forming a regular two-dimensional array of mxm heterogeneous blocks. From the test point of view, the regular structures present some properties that simplified the test approach. So, applying the above conceptual division to the whole FPGA means that we have to propose a specific test approach for each two-dimensional array:

a) a two-dimensional array of logic cells,

b) a two-dimensional array of interconnect cells,

c) a two-dimensional array of RAM cells.

According to this practical test strategy, the published works usually target specific array [3-25]. As an example, Inoue and al. address the problem of testing the array of look-up table in [13], Huang and al. address the problem of testing the array of logic cells in [20]. Other authors focus on different test strategies as Abramovici and al. dealing with BIST for FPGA in [17,18,19], Lombardi and al. dealing with diagnosis in [21].

Following this practical divide and conquer test strategy, the author has proposed first a test procedure targeting the array of interconnect cells [4,5,8], second another test procedure targeting the array of logic cells in [6,7,10], third a test procedure for the array of LUT/RAM modules in [9,11], and finally a test procedure for the array of interconnect/logic interface cells in [12]. In this paper the test of the two-dimensional array of logic cells is presented in section 2, the test of the two-dimensional array of interconnect cells in section 3 and the test of the two-dimensional array of RAM cells in section 4.

2. The logic cells

This section is devoted to the test of the logic cells. This problem has been discussed in detail by the authors in [4,5,8]. This section summarizes the main results. The logic cells usually consists of three types of logic modules: D flip-flops, multiplexers and look up table units (LUT). The multiplexers and the look-up tables are typical configurable devices while the D flip-flop are not really configurables. In fact, the control signals of the flip-flop (reset, clock...) are configurables by means of multiplexers. Because we concentrate here on the definition of test configuration for configurable devices, only multiplexers and look-up tables are considered in this section.

In figure 2, we have an example of classical 4-to-1 multiplexer with of 2 bit address A0,A1. In a typical FPGA representation, the data inputs E0,E1,E2,E3 are represented because they are Operation Inputs while the 2 bit address are not represented because they are Configuration Inputs not available during normal operation. In practice the internal logic structure can vary or is not really known, and so the fault model associated to this device is the stuck-at of the 6 different inputs E0,E1,E2,E3,A0,A1 and the single output S.

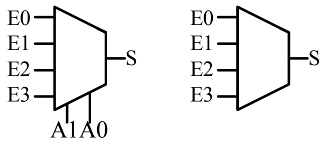

Fig. 2. Classical Mux and FPGA Mux

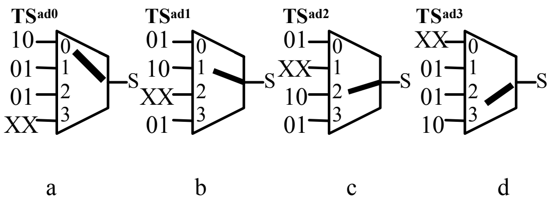

Fig. 3. Test of the FPGA Mux: a – **TC**^{ad0}**;** b – **TC**^{ad1}; c – **TC**^{ad2}; d – **TC**^{ad3}

It can be demonstrated that all the stuck-at-0/1 of the inputs and output of the FPGA multiplexer can be detected by using 4 test configurations. As the matter of fact, a test configuration is associated to each multiplexer address. For each test configuration, a sequence of 2 test vectors is applied. These 4 test configurations are illustrated in figure 3 where the configurations are symbolically represented by a connection between an input and output. In a more general way, we can say that a multiplexer with 2^n addresses (n address bits) require 2^n test configurations.

Assuming now that the LUT is a particular type of multiplexer as illustrated in figure 4, the test conditions are identical for the multiplexer and the LUT. Hence, we can use the vectors previously defined for the multiplexer. In such conditions, we found that the exclusive-OR and complemented exclusive-OR vectors must be applied on the LUT configuration inputs and an exhaustive sequence of 2^n vectors must be applied on the LUT operation Inputs. This is equivalent in defining 2 test configurations called **TC**^{XOR} and **TC**^{XNOR} and defining 2 corresponding test sequences called **TS**^{XOR} and **TS**^{XNOR}. The 2 test configurations are symbolically represented by a XOR (E) or XNOR symbol inside the LUT.

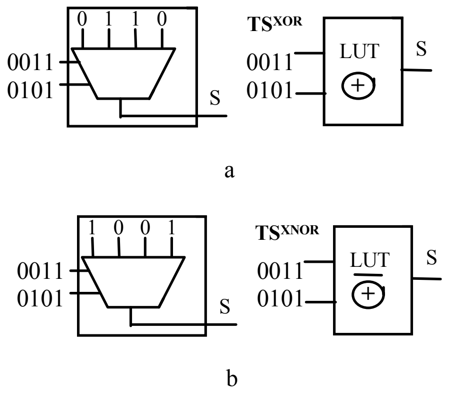

Fig. 4. Test of the Look-up Table: a – **TC**^{XOR}; b – **TC**^{XNOR}

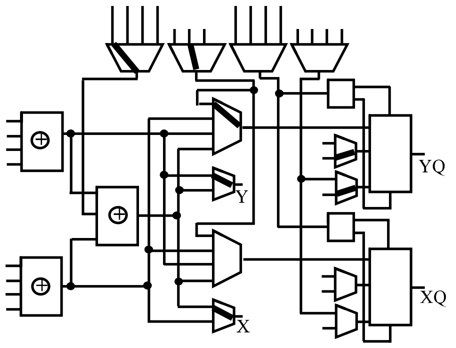

Fig. 5. Example of logic cell test configuration

The test configurations and test vectors defined for the isolated multiplexer and LUT are now used to define test configurations and test vectors for the logic cell. The logic cell is an interconnection of modules such as multiplexers, LUTs and flip-flops and so, a test configuration of a logic cell is an aggregate of the test configurations of the modules as illustrated in figure 5 with the Xilinx 4000 logic cell [25]. In order to 'cover' all the test configurations of all the modules in the cell, several test configurations must obviously be defined for the logic cell. We demonstrated that using our technique, only 5 test configurations are required for the Xilinx 4000. Figure 5 gives an example of test configuration for the logic cell of the Xilinx 4000. Concerning, the test sequences associated to each test configuration, they are obtained from the test sequences of each module. In fact, the test sequences of the modules are simply justified through the other modules.

The minimization of the number of Test Configuration using the module Test Configurations leads to only 5 Test Configurations for completely testing the complex XILINX 4000 CLB. The problem now is to define Test Configurations and Test Sequences for the mxm array of logic cells. In case of an array of logic cells, the problem consists in controlling and observing the whole logic cells. Individual access to each logic cell is not possible in practice. Indeed, a FPGA does not have enough I/O pads to control and observe each logic cell in parallel from outside.

For these reasons, the logic cells are interconnected in a special way forming one-dimensional arrays of cascaded logic cells. The length of the one-dimensional array is not important. The number and length of the arrays only depends on the number of available I/O pads. In practice, the most convenient solution is illustrated in Figure 6 where a mxm array of logic cells are distributed in m one-dimensional arrays of m logic cells. Using this scheme, the m one-dimensional arrays are tested in parallel.

In fact, a one-dimensional array of m cascaded logic cells can be viewed as an iterative circuit. Each logic cell received a local input from the previous logic cell and produces a local output connected to the next logic cell. The left most local inputs are controllable primary inputs and the right most local outputs are observable primary outputs. In addition, each logic cell receives a number of controllable primary inputs (black arrows) that are common to every logic cell in the FPGA.

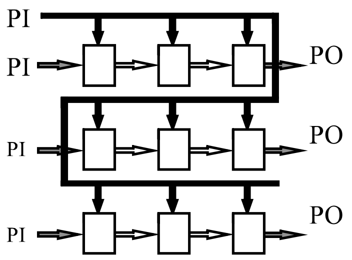

Fig. 6. One-dimensional arrays of 3 CLBs

Using the concept of one-dimensional array, it is clear in Figure 6 that an 'embedded' logic cell is controlled through logic cells located on its left side and observed through logic cells located on its right side. Consequently, the proposed test configurations and sequences for the whole array of logic cells must guarantee propagation through a given logic cell of signal controlling other logic cells on its right side, and guarantee propagation through a logic cell of observing signals from other logic cells on its left side. In the example of the Xilinx 4000, we simply define 5 test configurations for the whole array corresponding to the 5 test configurations of the single logic cell. In our approach, all the m^2 logic cells in the FPGA are configured exactly in the same way. Knowing that the 5 basic Test Configurations guarantee the complete test of any isolated CLB, we have demonstrated that the proposed interconnecting principle guarantees the controllability and observability of any embedded CLB. The demonstration is based on the fact that the test configurations used for the CLB implement in fact XOR or XNOR functions. The one-dimensional array corresponds consequently to a cascade of XOR (XNOR) functions that have well-known controllability and observabillity properties.

At this point, it must be noted that the complete Test Procedure has been simulated using an iterative array of 4 CLBs giving 100% coverage of the assumed fault models. These simulations validate the proposed test configurations and test sequences.

3. The interconnect

This section is devoted to the test of the interconnect cells. This problem has been discussed in detail by the authors in [4,5,8]. The interconnect structure of this type of RAM based FPGA is composed of a mxm array of « Switch Matrix » interconnected by k metal lines. This regular array of interconnect cells is illustrated in the simplified example of figure 7 where m=5 and k=4.

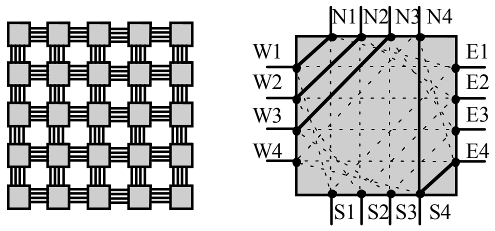

Fig. 7. FPGA Interconnect Structure

A switch matrix is a programmable connecting element receiving k lines on each side. The lines are connected to the switch matrix pins called North pins (N_1...N_k), East pins (E_1...E_k), South pins (S_1...S_k) and West pins (W_1...W_k). Inside the Switch Matrix, some pairs of pins can not be connected and are called non-connectable pins. Some pairs of pins can be connected and are called connectable pins. Figure 7 gives an example of switch matrix with 4 pins on each side. In figure 7 the connectable pins are linked by a dotted line. Figure 7 gives also an example of configuration where some connectable pins are connected (full lines). In the remainder of the paper we consider as an example that any set of pins with the same (different) number i are connectable (non-connectable). The considered set of connectable pins illustrated in figure 7 corresponds to the Xilinx 4000 family and so, the results presented here can be directly applied.

It is now necessary to define adequate fault models for this particular interconnect cells and lines. Due to the nature of the elements, we consider fault models classically used for interconnections i.e. Opens and Shorts:

1) For any line: Open called **Line-Open**.

2) For any pair of line : Short called **LinePair-Short**.

3) For any pair of non-connectable pins: Short called **Permanent-Connection**.

4) For any pair of connectable pins: Short called **Permanent-Connection**.

5) For any pair of connectable pins: Open called **Permanent-Disconnection**.

Obviously, the faults concerning the non-connectable pins are independent of the switch matrix configuration while faults concerning the connectable pins depend on the switch matrix configuration. A test configuration that connects for example pins N4 and S4 makes the Permanent-Connection fault between connectable Pins N4 and S4 redundant and so, un-testable. While a test configuration that does not connect the pins make the fault non-redundant. It is easy to demonstrate that a minimum of 3 test configurations are required to make all the faults under consideration non-redundant. Several set of 3 test configurations can be defined and figure 8 gives an example of 3 test configurations called the Orthogonal, the Diagonal-1 and the Diagonal-2.

The problem now is to use the 3 previous test configurations not to test a single isolated interconnect cell but to test the complete mxm array of interconnect cells. The approach used here is similar to the one used in the previous section for the logic cells i.e. all the interconnect cells in the array have the same test configurations. This method gives obviously 3 test configurations for the complete array that are illustrated in figure 8.

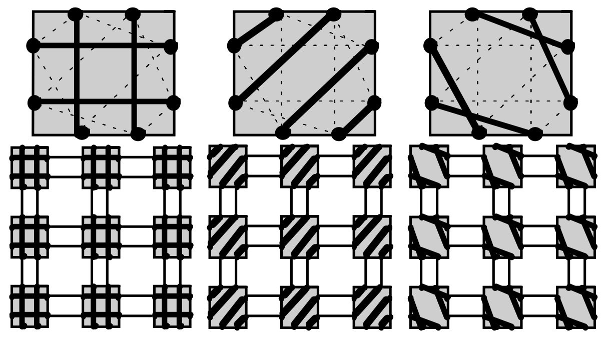

Fig. 8. The 3 Test Configurations: a – orthogonal; b – Diagonal-1; c – Diagonal-2

Using these 3 test configurations, the complete array can be conceptually considered as a global bus. The concept of bus with shorts and opens allow to use the previously published works about the bus testing problem [25-28]. It has been demonstrated that a n bits bus can be tested for any short and open with $\log_2(n)$ vectors. For the considered array, the resulting number of test vectors is: $\log_2(2km)$.

4. The RAM

This section is devoted to the test of the array of RAM cells that are embedded in the LUT/RAM modules. The LUT/RAM module is assumed to be part of a logic cell and it is assumed to be configured in RAM mode. This problem has been discussed in detail by the authors in [9,11]

Considering first a single isolated module in RAM mode, the module operates as a classical RAM and any type of existing RAM test can be used. This is very interesting because the problem of RAM testing has been investigated for a long time and very mature algorithms exist [29-30]. A well-known class of test algorithms for RAM circuits are the MARCH algorithms. The RAM fault models usually considered in the march tests are:

– SAF: The stuck-at fault can be defined as follows: The logic value of a stuck-at cell or line is always 0 or 1 and cannot be changed to the opposite value.

– AF: The address decoder faults concern faults in the address decoder. Different types of faults are usually considered. The first fault assumes that no cell will be accessed with a certain address. The second fault assumes that a certain cell is never accessed. The third fault assumes that multiple cells are accessed simultaneously with a certain address. And finally, the last fault assumes that a certain cell can be accessed with multiple addresses.

– TF: The transition fault is a special case of the SAF. It is defined as follows. A cell or line which fails to undergo a $0 \rightarrow 1$ transition when it is written is said to contain an up transition fault; similarly, a down transition fault is the impossibility of making a $1 \rightarrow 0$ transition.

– CF: The coupling fault involves 2 cells and can be defined as follows. A write operation which generates a up or down transition in one cell changes the contents of a second cell. Different types of coupling faults are usually considered. The inversion coupling fault (CFin) assumes that an up or down transition in one cell inverts the contents of a second cell. And the idempotent coupling fault (CFid) assumes that an up or down transition in one cell forces the contents of a second cell to a certain value, 0 or 1.

Note that DRAM circuits are more sensitive to CF than SRAM circuits. Dealing in this paper with SRAM based FPGA, we will use the SAF, AF and TF fault models. It can be observed in Table 1 that MATS, MATS+, Marching 1/0 and MATS++ are able to detect some of the fault models under consideration. It seems interesting to use the MATS++ test because it covers all the considered fault models and the number of test vectors is very low.

Note that only one test configuration is required to test a single module in RAM mode. Considering now the complete array of LUT/RAM module, the approach used here is similar to the one used in the previous sections for the logic cells and the interconnect cells i.e. all the LUT/RAM modules in the array have the same test configurations. This method gives obviously 1 test configuration for the complete array. In order to guarantee full controllability and observability of each module, we propose to connect the output of the LUT/RAM module in RAM mode to the input of the DFF included in the logic cell. The output of the Dff is connected to the data in of the following LUT/RAM module. This particular test configuration called pseudo shift register is illustrated in figure 9 with m=3.

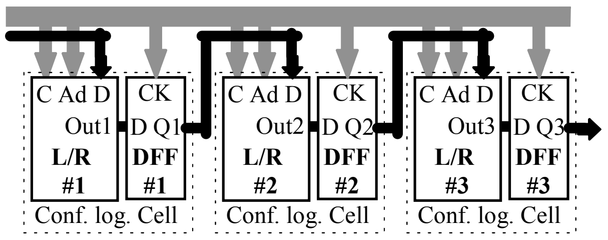

Fig. 9. The pseudo shift register

In this unique test configuration, the common primary inputs connected to every module include the control signals (Read/Write, Enable..) and address inputs of the LUT/RAM module, and the clock (CK) of the Dffs. The Read/Write control signal and the Dff clock can be adequately used to shift a value from the primary data input to the primary output, through the Dffs and through the modules. In this configuration, the MATS++ algorithm can be adapted taking into account the shift trough the different cells: This adaptation is called the shifted MARCH++ algorithm.

March tests

Algorithm	Covered Fault
MATS	some AFs, SAFs
MATS+	AFs, SAFs
Marching 1/0	AFs, SAFs, TFs
MATS++	AFs, SAFs, TFs

5. Conclusion

This paper gives a general overview of a structural test approach proposed for testing RAM-based FPGA taking into account their configurability. The SRAM-based FPGA architecture is first discussed identifying the specific FPGA test problems as well as the FPGA test properties. The FPGA architecture is then conceptually divided into different architectural elements such as the logic cells, the interconnect and the RAM cells. For each architectural element appropriated fault models are proposed, and test configurations and test vectors are derived targeting the fault models under consideration.

References: **1.** *S.D. Brown, R.J. Francis, J. Rose, S.G. Vranesic,* «Field-Programmable Gate Arrays», Kluwer Academic Publishers, 1992. **2.** *Trimberger S.M. (ed).* Field-Programmable Gate Array Technology // Kluwer Academic Publishers, 1994. **3.** *Jordan C. and MarnaneW.P.* Incoming Inspection of FPGAs // Proc. of IEEE European Test Conference, 1993. P. 371-377. **4.** *Renovell M., Figueras J. and Zorian Y.* Testing the Interconnect Structure of Unconfigurated FPGA», IEEE European Test Workshop, pp. 125-129, Sиte (Montpellier), FRANCE, June 1996. **5.** *Renovell M., Figueras J. and Zorian Y.* Test of RAM-Based FPGA: Methodology and Application to the Interconnect», 15th IEEE VLSI Test Symposium, pp. 230-237, Monterey, CA, USA, May 1997. **6.** *Renovell M., Portal J.M., Figueras J. and Zorian Y.* Test Pattern and Test Generation Methodology for the Logic of RAM-Based FPGA // IEEE Asian Test Symp., pp. 254-259, Akita, Japan, November, 1997. **7.** *Renovell M., Portal J.M., Figueras J. and Zorian Y.* Testing the Configurable Logic of RAM-based FPGA // IEEE Int. Conf. on Design, Automation and Test in Europe, pp. 82-88, Paris, France, Feb 1998. **8.** *Renovell M., Portal J.M., Figueras J. and Zorian Y.* Testing the Interconnect of RAM-Based FPGAs», IEEE Design & Test of Computer, January-March 1998. Vol. 15, №1. P.45-50. **9.** *Renovell M., Portal J.M., Figueras J. and Zorian Y.* SRAM-based FPGAs: Testing the RAM mode of the LUT/RAM modules // IEEE European Test Workshop, Barcelone, Spain, May 1998. **10.** *Renovell M., Portal J.M., Figueras J. and Zorian Y.* SRAM-based FPGAs: A Fault Model for the Configurable Logic Modules // Field programmable Logic Conference, Tallin, Estonia, Sept. 1998. **11.** *Renovell M., Portal J.M., Figueras J.and Zorian Y.* SRAM-Based FPGA: Testing the LUT/RAM modules // IEEE International Test Conferenc, Washington, DC, USA, Nov 1998. **12.** *Renovell M., Portal J.M., Figueras J. and Zorian Y.* SRAM-Based FPGA: Testing the Interconnect/Logic Interface // IEEE Asian Test Symposium, Singapore, Nov 1998. **13.** *Inoue T., Fujiwara H., Michinishi H., Yokohira T. and Okamoto T.* Universal Test Complexity of Field-Programmable Gate Arrays // 4th Asian Test Symposium. Bangalora, November 1995, India. P. 259-265. **14.** *Michinishi H., Yokohira T., Okamoto T., Inoue T., Fujiwara H.* A Test Methodology for Interconnect Structures of LUT-based FPGAs», IEEE 5th Asian Test Symposium, pp. 68-74, November 1996. **15.** *Michinishi H., Yokohira T., Okamoto T., Inoue T., Fujiwara H.* Testing for the Programming Circuits of LUT-based FPGAs // IEEE 6th Asian Test Symposium, pp. 242-247, November 1997. **16.** *Inoue T., Miyazaki S.and Fujiwara H.* Universal Fault Diagnosis for Lookup Table FPGAs», IEEE Design & Test of Computer, special Issue on FPGAs, pp.39-44, January-March 1998. **17.** *Abramovici M. and Stroud C.* No-Overhead BIST for FPGAs», 1st IEEE International On-line Testing Workshop. Nice, FRANCE, 1995. P. 90-92. **18.** *Stroud C., Chen P., Konala S., Abramovici M.* Evaluation of FPGA Ressources for Built-In Self Test of Programmable Logic Blocks», Proc. of 4th ACM/SIGDA Int. Symposium on FPGAs, 1996. P. 107-113. **19.** *Abramovici M., Stroud C.* ILA BIST for FPGAs: A Free Lunch with Gourmet Food // 2nd IEEE International On-line Testing Workshop, pp. 91-95, Biarritz, FRANCE,1996. **20.** *Huang W.K. and Lombardi F.* An Approach for Testing Programmable / Configurable Field Programmable Gate Arrays, 14th IEEE VLSI Test Symposium, Princeton, NJ, USA, May 1996. P. 450-455. **21.** *Lombardi F., Ashen D., Chen X.T., Huang W.K.* Diagnosing Programmable Interconnect Systems for FPGAs, FPGA'96, pp. 100-106, Monterey CA, USA, 1996. **22.** *Ashen D.G., Meyer F.J., Park N. and Lombardi F.* Testing of Programmable Logic Devices (PLD) with Faulty Resources», IEEE International Workshop on Defect & Tolerance in VLSI Systems, Paris, October 1997. P.76-84. **23.** *Huang W.K., Meyer F.J., Park N. and Lombardi F.* Testing Memory Modules in SRAM-based Configurable FPGAs», IEEE International Workshop on Memory Technology, Design and Test, August, 1997. **24.** *Hermann M. and Hoffmann W.* Fault modeling and test generation for FPGAs», in R.W. Hartenstein and M.Z. Servit (eds), Lecture Notes in Computer Science, Field Programmable Logic, Springer-Verlag, 1994. P. 1-10. **25.** *Xilinx.* The Programmable Logic Data Book . San Jose, USA, 1994. **26.** *Kautz W.H.* Testing for Faults in Wiring Networks » IEEE Transactions on Computers, Vol. C-23, № 4. 1974. P. 358-363. **27.** *Goel P. and McMahon M.T.* Electronic Chip-in Place Test » Proc. of International Test Conference, 1982. P. 83-90. **28.** *Jarwala N.and Yau C.W.* A New Framework for Analyzing Test Generation and Diagnosis Algorithms for Wiring Networks // Proc. of International Test Conference, 1989. P. 63-70. **29.** *Abadir M.S. and Reghbati J.K.* Functional Testing of Semiconductor Random Access Memories // ACM Computing Surveys, 15 (3), 1983. P. 175-198. **30.** *Van de Goor A.J.* Testing Semiconductor Memories: Theory and Practice , John Willey & Sons, 1991.