Totally Self-Checking FPGA-based FSM

I. Levin , A. Yu. Matrosova , V. Sinelnikov*, S. A. Ostanin

Tel Aviv University, Israel
ilia1@post.tau.ac.il
*Center for Technological Education, Holon
sinel@barlay.cteh.ac.il

Tomsk State University, Russia
mau@fpmk.tsu.ru
ostanin@fpmk.tsu.ru

Abstract

The paper introduces a new technique for on-line checking of FPGA based Finite State Machines (FSMs). This technique is based on the architecture comprising two portions: a self-checking FSM and a separate totally self-checking (TSC) Sum-Of-Minterms (SOM) based checker. Each of these portions is implemented as a combination of an Evolution block and an Execution block. For achieving the TSC property, the proposed architecture allows avoiding any additional coding of output words and consequently allows decreasing the required overhead.

The self-checking FSM is implemented in a form of one-rail network of interconnected pre-designed LUT-based configurable logical blocks (CLB). Its TSC property is achieved by: a) constant weight state assignment, b) transforming of the FSM description to a so-called unate PLA description, which is a standard PLA description where each 0 is change

Self-checking properties of the proposed architecture are proved. Benchmark results are presented.

1. Introduction

Most of the faults that occur in VLSI circuits and systems are transient/intermittent in nature. The self-checking property allows both the transient/intermittent and permanent faults to be detected, thus preventing data contamination.

We deal with the problem of synthesis of self-checking FPGA-based Finite State Machines (FSMs). A self-checking FSM consists of circuit to be checked and a checker, which checks outputs of the circuit to see if an error has occurred. The checker has an ability to expose its own faults as well.

In paper [1] it is shown that faults, which are available for FPGA, manifest themselves as unidirectional errors on the combinational part output lines of Synchronous Sequential Scheme (SSC) that implements the FSM. The authors have proposed the known m-out-of-n encoding state and output variables of FSM separately, and observing all these variables by a checker (with insignificant increase of the number of additional input variables).

In work [2] some specific features of control FSMs were utilized for achieving a self-checking ability with very small overhead. Possibility of observing only output lines of a self-checking synchronous sequential circuit (without its state lines) was shown, keeping on mind assumption of single stuck-at faults within PLA-based implementation.

In work [3] a novel architecture of a Totally Self-checking (TSC) FSM has been proposed. This architecture allows avoiding any additional coding of output words, which is a standard solution for TSC circuits.

We relate to three above-mentioned papers as main sources for our research. Though, even when combining together these approaches, we still have two limitations preventing the TSC property of the FSM:

1. Input errors are undetectable.
2. Errors within a next state portion of the FSM can be accumulated and consequently the FSM does not satisfy the self-testing requirement.

We propose solutions of the both above problems in the present paper. This paper has combined the single stuck-at faults oriented approach [1] with the FPGA implementation of FSM (while observing only its output lines) [2]. We apply the self-checking architecture [3], as a basic scheme, to our design.

We obtain FSM with the following properties:

- Any fault appearing on the FSM output lines is either unidirectional or undetectable.
- The accumulating undetectable faults in SSC is not dangerous for the next faults from the above-mentioned class.

First, (In section 2) we discuss a synthesis of a self-checking FSM. In section 2.1 the fault model is discussed. Section 2.2 describes a method of introducing additional input variables for achieving input errors delectability. Then, (In section 3) we investigate the SSC faults properties on the assumption that only SSC outputs are observable.

2. Synthesis of the Self-checking FSM

A schematic diagram of the proposed architecture of a totally self-checking FSM is shown in Fig. 1. The TSC FSM comprises two portions: a self-checking FSM and a Sum-of-Minterms (SOM)-checker. In turn, each of these portions consists of two blocks: an Evolution block and an Execution block.

Output vectors of the Evolution blocks of the mentioned portions are compared by comparing corresponding components thereof. The resulting vector forms inputs of the Execution block of the SOM-checker.

Inputs of the Evolution block of the FSM (EV FSM) constitute working inputs of the TSC FSM and memory signals. At each clock, one and only one codeword has to be implemented, which means that the 1-out-M code is implemented on EV FSM outputs.

We use the Xilinx-4000 series architecture [4] to illustrate the proposed method of designing a self-checking FPGA-based FSM. We implement the EV FSM in a form of a tree having nodes, which are pre-designed CLBs and fan-outs. Each CLB is designed for implementation of either AND-function of eight variables or a sum of two product terms of four variables.

The Execution block of the FSM (EX FSM) effects OR-assembling of EV FSM outputs. Outputs of EX FSM are output signals Y of the FSM and memory signals D. These memory signals are coded by a "constant weight code".

The Evolution block of the SOM-checker (EVC) implements all minterms, each are corresponding to a codword of the FSM. The Execution block of the checker (EXC) implements the sum of the minterms.

Inputs of the EVC are outputs of the initial FSM. The EVC is implemented in the dual-rail style. Output codes of the EVC belong to the 1-out-of-M two-rail code.

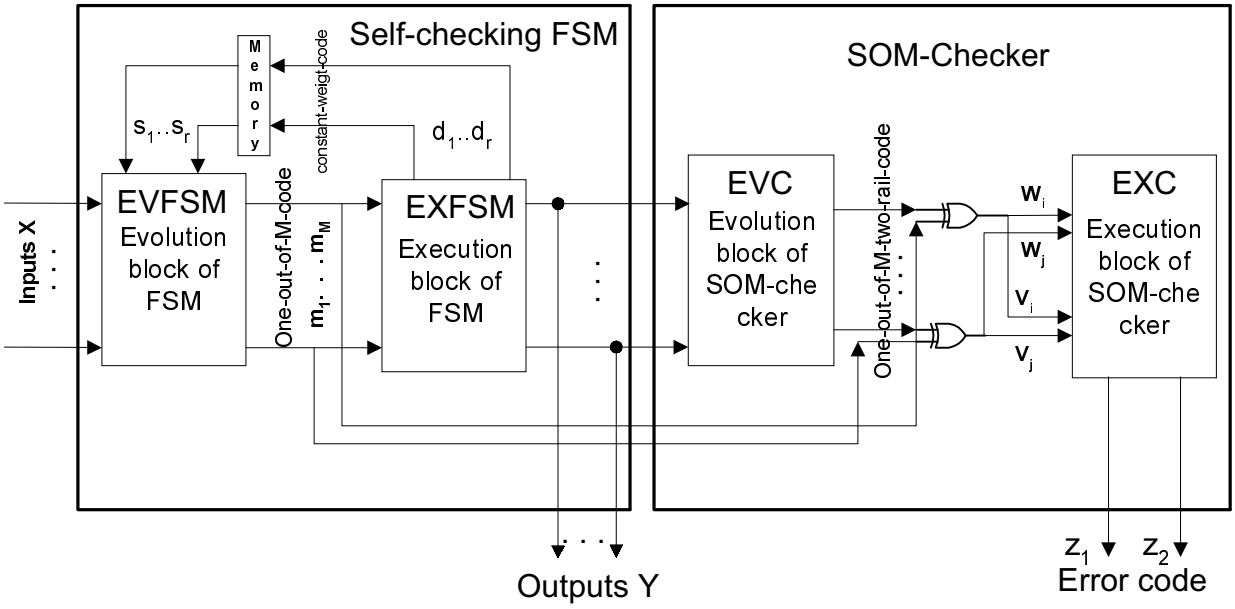

Figure 1. Schematic diagram of architecture of the TSC FSM

The proposed architecture allows to detect all unidirectional errors of the FSM by: a) comparing output lines of EV FSM with those of EVC, b) using of two-real 1-out-of-M design for the EXC implementation.

2.1. Faults of a synchronous FPGA-based sequential circuit

We will first consider the following set of faults of a synchronous sequential circuit: single stuck-at faults on the input and output poles of CLBs, and single functional faults within functions that CLB realizes. The latter means that CLB truth function can be changed for any Boolean function of the same number of variables.

Notice, that the CLB input pole single stuck-at fault can not be reduced to the single functional fault, if two functions that CLB realizes depend on the same input of CLB. Hereafter we forbid such situation and then single functional faults of CLB cover single stuck-at faults on the poles of this CLB.

We will consider only single functional faults, that is only one CLB function of SSC can be wrong.

Proposal 1. The test pattern of any cell functional fault is among the test patterns of both stuck-at faults of the corresponding cell output.

Corollary. Any single functional fault is unidirectional if both 1 stuck-at fault and 0 stuck-at fault of the corresponding cell output are unidirectional.

We can hereafter consider only stuck-at faults on the CLB outputs. A set of these faults will be called T.

Proposal 2. Any single stuck-at fault from T is either unidirectional or undetectable on the output lines of the Execution part.

Proof. We first apply the factorized synthesis to the PLA description and then cover the products and the OR functions by CLBs in the prescribed way. In this case a single stuck-at fault from T gives rise to a) an elimination of the certain products from the certain functions, b) elimination of the certain literals in the certain products, c) a conversion of the certain functions into a constant 1(0). Any of these faults manifests itself as a unidirectional error on the output lines of the Execution part. We mean output lines corresponding to the output and state variables of SSC. A fault of this kind can be also undetectable. ***Q. E. D.***

Taking into account the proposals and the corollary we can conclude that one of the approaches to self-checking design of multioutput combinational circuit (combinational part of SSC) based on FPGA is as follows.

We decompose a combinational circuit into one output combinational subcircuits so that stuck-at fault of any subcircuit output manifests itself as a unidirectional error on the circuit outputs. Then we cover the subcircuits with CLBs by any manner and as a result, obtain the self-checking combinational circuit.

There are many ways of such decomposition and the factorized synthesis [6] is one of them.

Let s consider stuck-at faults on the SSC inputs. To provide their unidirectional manifestation on the combinational part output lines we need to increase the number of the input variables of SSC [1]. Let s nclude these faults into the set T, too.

2.2. Introducing of additional input variables

Products u_i, u_j are bidirectional if there exists a component l taking the 1 value for u_i and the 0 value for u_j, and a component "r" taking the 0 value for u_i and the 1 value for u_j. For example, -10-0, -00-1 are bidirectional.

Let U be the set of products of the Evolution part. Divide U into subsets $U_1,\dots,U_{|Q|}$ in accordance with the different states of FSM. Here $|Q|$ is the number of states.

Proposal 3. Products u_i, u_j are bidirectional for $u_i \in U_i$, $u_j \in U_j$, $i \neq j$.

Proof. It follows from the encoding states when we obtain the PLA description. ***Q. E. D.***

Any product of the Evolution part has in the Execution part a corresponding full codeword representing the values of the state and output variables. Execute the following steps.

1. Divide the set U_i into subsets $U_{i1},\dots,U_{i\tau_i}$ in accordance with their different full codewords, $i \in \{1,\dots,|Q|\}$.
2. The different input codewords of the same weight correlate to different $U_{i\gamma}, \gamma \in \{1,\dots,\tau_i\}$.
3. Represent any input codeword with the proper Boolean vector α^* of the length k in additional input variables.
4. Add the same α^* to each product $u_i, u_i \in U_{i\gamma}$.
5. Execute steps 1-4 for every U_i, $i \in \{1,\dots,|Q|\}$, minimizing as much as possible the number k.

Proposal 4. Products u_k, u_s, $u_k \in U_{ik}, u_s \in U_{is}$ are bidirectional.

Change each 0-value component from the Evolution part for symbol "-". We obtain the unate products of the Evolution part and consequently the unate PLA description [6]. Call it as PLA^u.

Proposal 5. PLA^u preserves the FSM behavior in its working area.

Proof. The PLA description derived from STG represents the FSM behavior in its working area. Any minterm of the input and state variables from the working area activates the products of PLA with the same full codeword. This minterm with the corresponding additional input variables also activates the products of PLA^u with the same full codeword. Consequently the minterm gives rise to the same full codeword both for PLA and PLA^u. ***Q. E. D.***

The unate PLA description allows implementation of the factorized logic synthesis [6]. We preliminary obtain a minimized description of the PLA^u products with the same full codeword. Namely, for any such product there exists the minterm from PLA that is covered by this product only. Hereafter, we will use the minimized PLA^u description.

Table 1 illustrates the STG description of FSM, Table 2 - the corresponding PLA description and Table 3 - the minimized PLA^u description.

Table 1. STG description of the FSM

x_1	x_2	x_3	q	q	y_1	y_2	y_3	y_4	y_5
0	-	-	1	1	0	0	0	1	0
-	0	-	1	1	0	0	0	1	0
1	1	-	1	2	1	0	0	1	0
-	-	0	2	2	0	0	1	1	0
-	-	1	2	3	1	0	1	1	0
1	0	-	3	3	0	1	0	0	0
0	-	-	3	4	1	1	0	0	0
-	1	-	3	4	1	1	0	0	0
-	-	0	4	4	0	1	0	0	1
-	-	1	4	1	1	1	0	0	1

Table 2. PLA description of the FSM from Table 1

$x_1x_2x_3$	$z_1z_2z_3z_4$	$z_1z_2z_3z_4$	$y_1y_2y_3y_4y_5$
0 - -	1 0 0 0	1 0 0 0	0 0 0 1 0
- 0 -	1 0 0 0	1 0 0 0	0 0 0 1 0
1 1 -	1 0 0 0	0 1 0 0	1 0 0 1 0
- - 0	0 1 0 0	0 1 0 0	0 0 1 1 0
- - 1	0 1 0 0	0 0 1 0	1 0 1 1 0
1 0 -	0 0 1 0	0 0 1 0	0 1 0 0 0
0 - -	0 0 1 0	0 0 0 1	1 1 0 0 0
- 1 -	0 0 1 0	0 0 0 1	1 1 0 0 0
- - 0	0 0 0 1	0 0 0 1	0 1 0 0 1
- - 1	0 0 0 1	1 0 0 0	1 1 0 0 1

Table 3. Minimized PLA^u description.

$x_1x_2x_3x_4x_5$	$z_1z_2z_3z_4$	$z_1z_2z_3z_4$	$y_1y_2y_3y_4y_5$
- - - - 1	1 - - -	1 0 0 0	0 0 0 1 0
1 1 - 1 -	1 - - -	0 1 0 0	1 0 0 1 0
- - - - 1	- 1 - -	0 1 0 0	0 0 1 1 0
- - 1 1 -	- 1 - -	0 0 1 0	1 0 1 1 0
1 - - -1	- - 1 -	0 0 1 0	0 1 0 0 0
- - - 1 -	- - 1 -	0 0 0 1	1 1 0 0 0
- - - - 1	- - - 1	0 0 0 1	0 1 0 0 1
- - 1 1 -	- - - 1	1 0 0 0	1 1 0 0 1

It is possible to minimize the number k of additional input variables, applying a sophisticated algorithm which we have developed.

3. Properties of the SSC faults

We believe that SSC is derived from PLA^u by the factorized logical synthesis [6]. According to [1], we cannot observe the values of the state variables. As for the combinational part of a synchronous sequential circuit, any fault from T either appears [1] as unidirectional error on the combinational part outputs (the next state outputs and the outputs of SSC), or remains undetectable on these outputs.

A fault is detectable (for SSC), if there exists the input sequence (in the working area of FSM) for which the fault manifests itself as error on the SSC outputs. We restrict the sequence to the first manifestation. Otherwise, the fault is undetectable.
If a fault manifests itself as a unidirectional error on the SSC outputs for the certain input sequence from the FSM working area it is unidirectional.

A certain fault can appear on the next state outputs of SSC as an unidirectional error and remain undetectable on the SSC outputs in the working area of FSM. It is also possible that a certain fault appears as unidirectional error on the next state outputs of SSC several steps before than it appears on its outputs.

What will happen when the next fault (from T) appears in SSC while the previous fault (from T) was undetectable? In this case we simultaneously deal with two faults from T. It is possible that there exist several undetectable faults from T.

We have to investigate a problem of accumulating undetectable faults. First we have to study the manifestations of faults from T taking into consideration that the next state outputs of SSC are not observable. A fault from T gives rise to the following:

i) disappearance of the certain products from the Evolution block of PLA^u, or

ii) disappearance of the certain literals from the certain products of the Evolution block of PLA^u, or

iii) conversion of the certain functions that PLA^u realizes into constant 1(0).

Proposal 6. A fault that gives rise to disappearance of the certain products from the Evolution block, is unidirectional.

Proof. This fault gives rise to a none-code state vector that consists of only the 0 values. It is caused by the fact, that each product of the PLA^u description and only this product is solely activated at least once in the working area of FSM. The none-code state vector immediately results in the output vector of SSC that consists of only 0-values. ***Q. E. D.***

If we don't minimize the portion of PLA^u with the same full codeword, the fault can be undetectable.

Proposal 7. A fault that gives rise to disappearance of the certain literals from the certain products of the Evolution block is either unidirectional or undetectable.

Proof. This fault is undetectable when any minterm α of the state and input variables from the FSM working area is found among the cubes of PLA^u with the same full codeword. Assume that the certain minterm α activates simultaneously the cubes with the different full codewords. The fault is unidirectional when the parts of these codewords corresponding to the output variables of SSC are different. If these parts are the same, we obtain the wrong next state vector β that contains more 1-values than the truth next state vector. It is possible that β does not result in any errors on the SSC output lines in the FSM working area. Then, the fault is undetectable. The fault can be unidirectional when the minterm α of input variables from the FSM working area, together with the minterm representing the state resulted from β activates simultaneously several cubes with different output codewords. The latter means that the fault can manifest itself on the SSC output lines several steps later than on the next state lines. ***Q.E.D.***

Proposal 8. If a fault converts the certain transition functions into the constant 1(0), the fault is either unidirectional or undetectable.

Proof. When the fault converts the certain transition functions into the constant 0, it gives rise to the none-code state vectors. Let the none-code state vector β be derived from the corresponding truth state vector α by changing the certain 1-values for 0-values. The vector α activates the only product k_i of the state variables. It is orthogonal to all other products of these variables from PLA^u. The product k_i is derived from α by changing all 0-values for symbol - when we obtain PLA^u from PLA. Consequently, β is orthogonal to k_i and all the more to other products of the state variables. That is why the fault is immediately unidirectional on the SSC output lines.

In the case of a conversion of the certain transition functions into the constant 1, we also obtain the none-code state vectors. Let the none-code state vector β be derived from the corresponding truth state vector α by changing the certain 0 values for 1 values: β and α activate k_i. But β can also activate other products of the state variables from PLA^u. Simultaneous activating the products of the state variables from PLA^u with the different full codewords results in either an unidirectional or undetectable fault. The latter is possible as follows. Any resulted none-code state vector contains the logical sum of the truth codes of the states. If for each corresponding truth state vector and the same input minterm from the FSM working area the corresponding output vectors of SSC are the same, then this none-code state vector does not manifest itself directly on the SSC output lines. If it takes place for any resulted none-code state vector, the fault is undetectable. ***Q.E.D.***

If the fault converts the certain output functions into the constant 1(0), it is definitely unidirectional.

Taking into consideration the proposals 6-8 we conclude that undetectable faults are possible when any none-code state vector resulted from the fault does not manifest itself directly on the SSC output lines in the FSM working area.

Any none-code state vector is obtained from the corresponding truth state vector by changing the certain 0 values for 1 values. An Individual none-code state vector with a corresponding input minterm from the FSM working area will be called an undetectable portion of a fault and noticed as t. Consider t as the first fault of any pair of faults and investigate the second fault impact on the next step of the behavior in the working area.

Let t^a be a disappearance of the certain products from PLA^u.

Proposal 9. The pair (t, t^a) is either unidirectional or undetectable.

Proof. A disappearance of the certain products from PLA^u can result in appearance of the none-code state vector containing only 0-components. It means that the pair is unidirectional on the SSC output lines. It is possible that the pair remains undetectable. In fact, if t simultaneously activates several products from PLA^u and some of them disappear, then the rest products will have the same full codeword. A first undetectable fault, considering as a whole, can only increase the number of 1-values for the state followed by t. In this case, the pair (t, t^a) also results in the undetectable fault. ***Q. E. D.***

Let t^b be a disappearance of the certain literals from the certain products of PLA^u.

Proposal 10. The pair (t, t^b) is either unidirectional or undetectable.

Proof. The fault t^b can effect the additional 1 values among the corresponding state vectors resulted from t in the FSM working area. A first undetectable fault considering as a whole can only increase the number of 1-values for the state followed by t. Consequently, (t, t^b) is either unidirectional or undetectable. ***Q. E. D.***

Let t^{c_i} be a conversion of the certain state variables into constant 1.

Proposal 11. The pair (t, t^{c_1}) is either unidirectional or undetectable.

Proof. The fault t^{c_1} can effect the additional 1 values in the corresponding state vector resulted from t in the FSM working area. A first undetectable fault considering as a whole can only increase the number of 1-values for the state followed by t. Consequently, (t, t^{c_1}) is either unidirectional or undetectable. ***Q. E. D.***

Let t^{c_0} be a conversion of the certain state variables into constant 0.

Proposal 12. The pair (t, t^{c_0}) is either unidirectional or undetectable.

Proof. The fault t^{c_0} can cut the number of 1 values in the corresponding state vector resulted from t in the FSM working area. It is possible the appearance of the none-code state vector that is orthogonal to any state product of PLA^u and then (t, t^{c_0}) is unidirectional. The fault t^{c_0} can remain the pair (t, t^{c_0}) undetectable. A first undetectable fault considering as a whole, can only increase the number of the 1-values for the state followed by t. In this case the pair (t, t^{c_0}) is also either unidirectional or undetectable. ***Q. E. D.***

We have shown that if a first fault from T is undetectable, then an appearance of a next fault from T is either unidirectional or undetectable. We believe that any next fault from T appears in the SSC after exhausting the FSM working area in the presence of the foregoing fault from T. The foregoing fault being detectable has to manifest itself as unidirectional error on the SSC output lines. It means that accumulating undetectable faults from T in SSC is not dangerous.
In conclusion to this part, we illustrate the overhead required for changing STG for PLA and PLA^u description (Table 4).

Table 4

	I				II			III			
	n	m	s	q	p	Σ_1	Σ_2	n_{ad}	p	Σ_1	Σ_2
BBARA	4	2	10	60	4	410	89	4	5	270	128
BBSSE	7	7	16	56	4	350	174	4	6	317	219
BBTAS	2	2	6	24	3	120	35	2	4	90	52
BEECOUNT	3	4	7	28	3	152	79	3	5	106	112
CSE	7	7	16	91	4	631	132	5	6	576	309
DK14	3	5	7	56	3	336	154	3	5	245	213
DK15	3	5	4	32	2	160	80	3	4	108	88
DK16	2	3	27	108	5	756	305	2	7	513	412
DK27	1	2	7	14	3	56	24	1	5	42	34
DK512	1	3	15	30	4	150	65	1	6	90	73
DONFILE	2	1	24	96	5	672	304	2	7	456	384
KEYB	7	2	19	170	5	1344	130	7	6	1136	524
LION	2	1	4	11	2	40	17	2	4	28	18
LION9	2	1	9	25	4	150	55	2	5	89	67
MODULO12	1	1	12	24	4	120	40	1	6	72	48
S8	4	1	5	20	3	140	38	0	4	60	60
SAND	11	9	32	184	5	1623	675	7	7	1031	891
SHIFTREG	1	1	8	16	3	64	32	1	5	48	40
SSE	7	7	16	56	4	350	129	4	6	317	219
STYR	9	10	30	166	5	1390	489	6	7	1090	696
TAV.KIS	4	4	4	49	2	258	85	6	4	243	85
TBK.KIS	6	3	32	1569	5	16107	2235	8	7	13707	5006
TRAIN11.KIS	2	1	11	25	4	150	52	2	6	83	68
TRAIN4.KIS	2	1	4	14	2	56	26	2	4	34	24

The Table 4 is divided into 3 portions. The first portion describes STG. Here, n is the number of input variables, m - output variables, q - the number of products and s is the number of the states of FSM. The second portion describes PLA on assumption, that we use encoding states with the minimal number of state variables. Here, p is the number of the state variables, Σ_1 is the number of the literals in PLA, Σ_2 - the number of 1-values among the full codewords. The third portion describes PLA^u. Here, n_{ad} is the additional number of the input variables. We see that $\Sigma_1 + \Sigma_2$ for PLA^u is, as a rule, less than for PLA but namely, this sum represents the complexity of SSC.

Conclusion

A method of synthesis of a synchronous totally self-checking FPGA-based FSM is suggested. The method is based on specific TSC FSM architecture and on using a so called unate PLA description of the FSM, which is a standard PLA description where all of 0 values is changed for don t care .

Single stuck-at faults at the CLB poles, single functional CLB faults and single stuck-at faults on the input lines of a synchronous sequential circuit implementing the FSM are available. Additional input variables allow deriving PLA^u from the STG description.

It is shown that the use of the proposed method allows the second fault masking within the SSC to be avoided. It stems from the fact that the FPGA based implementation of the unate PLA is incapable of accumulating undetectable faults. It has to be emphasized that this important property has been achieved for the architecture, where only output lines of SSC are observable.

The obtained benchmark results show that the proposed method allows receiving of promising results from the point of required overhead.

References

[1] A. Yu. Matrosova, S. A. Ostanin, "Self-Checking Synchronous FSM Network Design", *4th IEEE Intl. On-Line Testing Workshop*, Capri, Italy, pp. 162-166, July 1998.

[2] I. Levin and M. Karpovsky, "On-Line Self-Checking of Microprogram Control Unit", *4th IEEE Intl. On-Line Testing Workshop*, Capri, Italy, pp. 152-156, July 1998.

[3] I. Levin, V. Sinelnikov. "Self-checking of FPGA based Control Units", Proceedings of 9th Great Lakes Symposium on VLSI, Ann Arbor, Michigan, March 4-6, 1999, IEEE press. (In press).

[4] Book, 1996.

[5] Fadi Y.Busaba and Parag K. Lala. Self -Checking Combinational Circuit Design for Single and Uniderectional Multibit Error. JETTA, 5, pp 19-28,1994.

[6] S. Baranov. Logic Synthesis for Control Automata. Kluwer Academic Publisher. Dordrecht/Boston/London. 1994.

.