ELECTRONCS AND ELECTRICAL ENGINEERING
ISSN 1392 – 1215 *2006. Nr. 3(67)*
ELEKTRONIKA IR ELEKTROTECHNIKA

ELECTRONICS
T 170
ELEKTRONIKA

Transition Test Supplement

E. Bareiša, V. Jusas, K. Motiejūnas, R. Šeinauskas

Software Engineering Department, Kaunas University of Technology
Studentų St. 50-406, Kaunas, Lithuania
e-mail: eduardas.bareisa@ktu.lt, vacius.jusas@ktu.lt, kestutis.motiejunas@ktu.lt, rimantas.seinauskas@ktu.lt

Introduction

Many recent system-on-a-chip (SoC) integrated circuits incorporate pre-designed and reusable components, variously referred to as intellectual property (IP) circuits or cores. Such circuits are frequently supplied by third-party vendors and are extremely hard to test when embedded in a SoC because their functions are specified only in high-level terms. This is done either to protect the circuits' IP content or else to allow system designers to synthesize their own low-level (gate-level) implementations. Tests can be generated for a high level description in order to reuse them for all possible implementations [1]. However, such tests usually cannot guarantee the detection of all specified faults in all possible implementations. Consequently, if we consider realization-independent testing, we can only speak about such realizations that fulfil specific requirements or have a particular structure [2, 3].

Conventional fault models like the standard single stuck-at model were developed for gate-level logic circuits. Regardless of the stuck-at fault model's efficiency for several decades, alternative models need to account for deep sub-micron manufacturing process variations [4]. Increasing performance requirements for circuits makes it difficult to design them with large timing margins. Thus imprecise delay modelling, statistical variations of the parameters during the manufacturing process as well as physical defects in integrated circuits can sometimes degrade circuit performance without altering its logic functionality. These faults are called delay faults.

In this paper we will analyse the situation when tests are generated for a particular implementation and are used for the other possible implementations of the same circuit. The same core can have distinct implementations. Naturally, that a test generated according to one structure may not detect all specified faults of another structure. The employment of different synthesis tools can have an influence on the test quality as well. The problems of generation of realization-independent tests for stuck-at faults were addressed in [5-7]. We will investigate the delay faults coverage in various implementations of the same circuit.

In general case, the re-synthesized IP core for a new technology requires a test generation from scratch. But there always is a problem - are the existing tests for the old technology useful for the new technology? This question is answered for the stuck-at faults [7]. The transitions faults still have no answer. Therefore the purpose of this paper is to estimate how much a test of one implementation is suitable for the other implementation of the same circuit. On the base of this information the designer can reasonably decide how to test the transition faults of the re-synthesized core. There are three possibilities:

1. To start the test generation from scratch;
2. To use the test of some other implementation.
3. To augment the test of the other implementation in order to compensate the loss of the fault coverage.

In this work we will analyse such implementations that are generated by the synthesis tool according to the same description, changing the synthesis tool and the target library used during the synthesis. We will explore the test quality of one realization for detecting faults of other realizations. The ISCAS'85 benchmark circuits will be used for the experiments. As well we will analyse how the tests for delay faults can be modified or expanded in order to enhance the fault coverage of other realizations and we will evaluate such possibilities by experiment.

The conventional synthesis goal is to find a trade-off between the minimal area and the maximal performance. The different implementations could be based on these extremities: low area and high speed [5]. We have tried to synthesize the circuits targeted on the low area and the high speed. But the obtained results were very similar. Then we changed a target library. The obtained results of different target libraries were quite different. Therefore, the choice was made for the implementations based on different target libraries.

The structure of the paper is as follows. We review the related work in Section 2. We analyse the influence of circuit re-synthesizing on the transition fault coverage in Section 3. We explore the application of functional delay tests to detect transition faults in Section 4. We present the enhancement of the independency of the test from implementations in Section 5. We finish with conclusions in Section 6.

Related work

Two general types of delay fault models - the gate delay fault model [8, 9], and the path delay fault model [10] - have been used for modelling delay defects. Although the path delay fault model is generally considered to be more realistic and effective in modelling physical delay faults, it is often difficult to use in practice due to a huge number of paths in the circuit. Therefore, the gate delay fault model is more feasible for large circuits. The most commonly used gate delay fault model is the transition fault model 8. According to this model, every line in the circuit is associated with two transition faults: a slow-to-rise fault (rising fault) and a slow-to-fall fault (falling fault). To simplify the analysis of transition faults, it is often assumed that the extra delay caused by a transition fault on a line is sufficiently large such that the delay of every path passing through this line exceeds the maximum allowed value, which is usually the system clock period for synchronous sequential circuits.

The possibilities of using a test obtained for one realization for testing delay faults of another realization are studied in [5]. The suggested fault model is called a coupling fault, which is devoted to testing stuck-at faults and is applicable to test path delay faults. The corresponding coupling delay tests detect all robust path delay faults in any realization of the function. The size of a coupling delay test set is very large compared to that of a typical path delay test set, however [5].

The realization-independent tests are generated when functional level test generation is performed. Several high-level delay fault models have been proposed that aim to cover timing faults, especially path delay faults [11-13]. Tests based on the fault model [11] result in the sets of practical sizes, but their coverage of path delay faults in an arbitrary gate level implementation of the circuit is low. The gross [12] and function-robust [13] delay fault models assume that a delay fault in a module is large enough to cause excessive delay in the whole circuit. A test set for gross delay faults is composed of all adjacent vector pairs, hence $n2^n$ tests are used to detect all gross delay faults in an n-input module. Function-robust delay faults require tests that meet a special propagation condition [13]. It is hard to generate test vectors for function-robust delay faults because all vector pairs must be examined to see if they meet the function-robust propagation condition.

The universal model for functional test generation based on input-output path testing and called pin pair fault model has been suggested in [14] and generalized in [15]. The experiments show that the test sets generated according to pin pair fault model detect in average more than 99 percent of the stuck-at faults of the three different circuits implementations at gate level [15]. The pin pair fault model can be easily extended to generate functional delay test. In Section 4, we will introduce briefly pin pair fault model and will present its extension that to generate functional delay tests. The generated tests will be examined at the gate level of the circuit for detection transition faults.

The possibilities of supplementing or expanding a particular realization test having a purpose to enhance test quality for detecting of delay faults are analysed in [16-19]. Test sets for path delay faults in circuits with large numbers of paths are typically generated for path delay faults associated with the longest circuit path. This may lead to undetected failures since a shorter path may fail without any of the longest paths failing. The paper [16] proposes a test enrichment procedure that significantly increases the number of faults associated with the next-to longest paths that are detected by a compact test set. The alternative approach to this problem is an optimisation of the critical path selection [18] or a selection of the longest testable path [17, 19]. The papers [17, 19] combine the merits of both the transition fault model and the critical path delay model. Both papers agree that more automatic test pattern generation efforts are required to produce tests for all faults in this model than that given by the single transition fault model. Therefore the paper [17] suggests that to obtain a high quality transition fault test set using reasonable run times, initially a conventional transition fault test set can be generated and then augmented by a test based on the longest testable path passing through the fault site.

The impact of implementation to the transition fault coverage

The core can be synthesized by different electronic design automation systems and mapped into different cell libraries and manufacturing technologies. An important issue is how the test set of the core covers the transition faults of new implementations, which are done by the same synthesizer or by a different one. The ISCAS'85 benchmark circuits have been selected for experiments. The similar experiment was performed in [20]. But there were used the original ISCAS'85 circuits, some of which are redundant and cannot be by accepted by the synthesis tool. That complicates an experiment. The synthesis tool after re-synthesizing leaves some untestable faults for some versions of circuits. That fact was neglected in [20]. As we can see later in this paper, the untestable faults have an influence to the fault coverage.

The non-redundant ISCAS'85 circuits have been re-synthesized by the Synopsys Design Compiler program in three modes and by the Cadence BuildGates synthesis program in one mode. The following five implementations have been analyzed:

V1 – the non-redundant ISCAS'85 benchmark circuit,
V2 – Synopsys Design Optimization,
target library – class.db,
V3 – Synopsys Design Optimization,
target library – and_or.db,
V4 – Synopsys Design Optimization,
target library – virtex.db,
V5 – Cadence BuildGates synthesis,
target library – lca300k.alf.

We can see the number of transition faults for each realization in Table 1. The second row for each circuit shows the number of untestable faults, which were found during test pattern generation by Synopsys test generation tool TetraMAX. It needs to draw attention to the fact that originally the circuits C432 and C499 have XOR gates. When a test is generated for XOR gate, it does not detect 2

faults of the equivalent circuit constructed of NOT, AND and OR gates. All the other versions of the circuits are constructed from primitive gates: AND, OR, NAND, NOR and NOT. In order to have equal conditions for all versions of circuits, the original circuits C432 and C499 were expanded to the NOT, AND and OR gates. The version V4 of the circuits is constructed from FPGA cells. The transition faults of these circuits were simulated at the gate level, too.

Table 1. Transition and Untestable Transition Faults

Circuit	V1	V2	V3	V4	V5
C432	141236	10020	11720	122827	10500
C499	34300	26460	29820	31380	26460
C880	23960	21460	22800	3040144	21700
C1355	33500	32740	36180	33060	32740
C1908	48480	21762	27960	29960	24402
C2670	56460	41340	44860	49220	41620
C3540	89600	61540	64480	69420	802459
C5315	138160	103120	103640	133823101	106521
C6288	144220	135280	147900	181800	25678605
C7552	191600	119620	120480	141360	128981

The original non-redundant benchmark realizations have more transition faults in total. This means that the re-synthesized circuits were optimised. The percent of the difference between the maximum and minimum numbers of transitions faults to the maximum number of transition faults varies from 10 to 55. It demonstrates the diversity of realizations and the impact of the target library on the design synthesis.

The number of transition faults has to be equal to the number of stuck-at faults because every pin of the gate has two transition faults (a slow-to-rise fault and a slow-to-fall fault) and two stuck-at faults (stuck-at 0 and stuck-at 1). If we look to the paper [7] which presented the numbers of stuck-at faults for benchmarks circuits we would see two or three times smaller numbers. The reasons are the following: 1) a simulation program of the stuck-at faults for two inputs AND gate includes 4 stuck-at faults into a fault list meanwhile a simulation program of the transition faults for the same gate includes all 6 transitions faults; 2) the equivalence and dominance relations are applied for stuck-at faults. Therefore, for example, the circuit C880 has 942 stuck-at faults [7] and 2396 transition faults.

Synopsys test pattern generator TetraMAX was used to generate the test sets for transition faults. The test sets have been generated for version V1 of the original ISCAS'85 circuits and then the obtained test set was applied to all the other implementations of the same circuit. The size of test sets that obtain 100% transition fault coverage is displayed in Table 2.

Table 2. The Size of Transition Test Sets

C432	C499	C880	C1355	C1908
268	434	282	620	628
C2670	C3540	C5315	C6288	C7552
510	836	598	236	912

The fault coverage and the number of undetected faults for each realization of the circuit were computed. Table 3 presents the results of the experiments. Of course, the transition fault coverage is 100% for the version V1 of all circuits. The faults of all the other versions of the circuits were not covered completely, except the version V3 of the circuit C432. In general, the version V3 was checked the best for all the circuits. Recall that this version is implemented on the base of two inputs gates. The obtained results of version V5 were worse than for the other versions of the circuits. We would like to remind that version V5 was synthesized by Cadence BuildGates synthesis tool, whereas all the other versions were synthesized by Synopsys Design Optimization tool. That points to the conclusion that the transition fault coverage is depended on the synthesis tool used. But on the other side, it needs to bear in mind that our designer always uses Synopsys synthesis tool, therefore he is not used to work with Cadence synthesis tool and he does not know all the specific features of it.

Table 3. Transition Fault Coverage

Circuit	V1	V2	V3	V4	V5
C432	100	97.41	100	98.45	97.52
C499	100	99.51	99.97	99.75	99.51
C880	100	99.95	99.96	97.79	99.95
C1355	100	96.09	96.93	98.91	96.09
C1908	100	98.90	99.39	99.37	99.02
C2670	100	97.24	97.10	99.43	97.26
C3540	100	99.16	99.02	99.28	96.80
C5315	100	99.61	99.38	99.19	99.51
C6288	100	99.06	99.48	99.20	92.89
C7552	100	99.46	99.49	99.82	99.40
%	100	98.64	99.07	99.19	97.82

The average percent of all four implementations is 98.69.

If we compare the results of the stuck-at faults for the other realizations presented in the paper [7], we could find that the biggest average percent of undetected faults among different versions of circuits is 1.35. The biggest average percent of undetected transition faults from Table 3 is 2.18. But this result includes the worst case of the circuit C6288 which could be excluded. If the result of the circuit C6288 is excluded, then the biggest average percent of undetected transition faults is 1.56. But when the comparison is accomplished the attention has to be paid to the fact that stuck-at faults were exercised only for two implementations: V2 and V3. The biggest average percent of undetected transition faults of these implementations is 1.36, which is very similar as for stuck-at faults.

Transition fault coverage by functional delay test

The universal possibility to reuse test is to use test generated at the functional level of circuit. The test at the functional level is not related to the particular implementation of the circuit. Therefore the functional test

has to suit to any implementation of the circuit. An interesting issue is how the functional test is able to detect transition faults at the gate level of the circuit. The pin pair fault model [15] was chosen as the functional level fault model.

We provide a brief presentation of the main concepts of this model. Let the circuit have a set of inputs $X = \{x_1, x_2, ... ,x_i, ... ,x_n\}$ and a set of outputs $Z = \{z_1, z_2, ... ,z_j, ... ,z_m\}$. The pin fault model considers the stuck-at-0/1 faults occurring at the module boundary. We write x_i^1 and x_i^0 for the input stuck-at-1/0 faults, and z_j^1 and z_j^0 for the output stuck-at-1/0 faults. There are 2n +2m possible pin faults. Input-output pin stuck-at fault pairs (x_i^t, z_j^k), t=0,1, k=0,1 are called pin pair faults (PP). The number of possible pin pair faults of the circuit is at most 4*n*m.

The test vector detects the pin pair fault (x_i^t, z_j^k) of the module if the test vector detects both the pin faults x_i^t, and z_j^k of the pair on the output z_j of the module. It may appear that there exist no electric connections between the input and the output, and the pin pair fault defined by these inputs and outputs can't be detected. These faults are not testable. The PP fault (x_i^t, z_j^k) of a module is testable if a conventional deterministic test generator for a realization of the module finds a test vector, which detects a pin fault x_i^t on an output z_j while the input x_i and the output z_j are set up to the opposite values (not t on the input and not k on the output). The number of testable PP faults equals to 4*n*m minus the number of not testable PP faults. Note that in general it is not possible relate the PP fault with the defects of the module unambiguously, because the PP fault doesn't fix exactly the signal propagation path in the circuit.

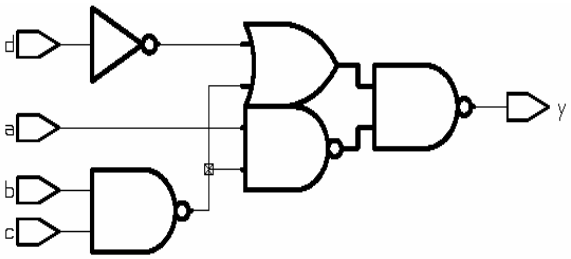

Fig. 1. The example circuit

If we compare the functional delay [21, 22] and pin pair models we see that both models have almost the same meaning with one distinction: the functional delay model is intended for detection of malfunctions in the dynamic behaviour of module and the pin pair model – for detection of malfunctions in the steady state of module. For example, consider the PP fault (b^1,y^0) and functional delay fault (b, y, rb, fy) (r – rising transition, f – falling transition) (see 0). The PP fault (b^1,y^0) is detected by an input pattern 1010 and functional delay fault (b, y, rb, fy) is detected by a pair of input patterns <1010, 1110>. Based of this observation, we can define the rule how to extend the PP fault test to the functional delay fault test.

Rule. If the input pattern *q* detects the PP fault (x_i^t, z_j^k), t=0,1, k=0,1, than the pair of input patterns $<q, p>$, where the signal value of input x_i in the pattern *q* is 1 (0) and the signal value of input x_i in the pattern *p* is 0 (1) detects the functional delay fault (x_i, z_j, tr x_i, tr z_j), tr=r,f (r – rising transition, f – falling transition).

Suppose, we have an input pattern *w* that detects *l* PP faults. Than the transforming this pattern for detection of *l* corresponding functional delay faults needs to build maximum *l* pairs of input patterns according to the Rule (signal transition on one input can cause signal transitions on *s* outputs, therefore, only one pair of input patterns is needed for detection of *s* functional delay faults). For example, the transformation of input pattern 1010 that detects the PP faults (b^1,y^0), (a^0,y^0) results into the sequence of input patterns (<1010, 1110>, <1010, 0010>) that detects functional delay faults (b, y, rb, fy) and (a, y, fa, fy) (see 0).

Note, that all test pairs composed from PP tests according to the Rule are single-input transition tests, and, therefore, all test pairs satisfy the function robust propagation property.

The non-redundant ISCAS'85 benchmark circuits have been selected for experiments (version V1). The functional delay tests have been composed from PP fault tests according to defined Rule. The test sets for PP faults were generated for the black-box model of the circuits using the random search procedure.

Table 4. Transition fault coverage

Circuit	Test size 1	Test size 2	Number of faults	Detected	Coverage (%)
C432	117	610	1412	1288	91.22
C499	1077	10302	3430	3418	99.65
C880	381	1920	2396	2302	96.08
C1355	1011	10292	3350	3317	99.01
C1908	620	4612	4848	4594	94.76
C2670	448	3584	5646	5447	96.48
C3540	515	2954	8960	7533	84.30
C5315	1169	9604	13816	13565	98.18
C6288	268	2064	14422	14386	99.75
C7552	2115	11602	19160	18494	96.52
Average	772	5754			95.59

Test size 1 - size of test for PP faults
Test size 2 - size of test for functional delay faults composed from the test for PP faults according to Rule

The comparison of test set sizes and detected transition faults of the non-redundant ISCAS'85 benchmark circuits are given in Table 4 Note that the functional test sets detect 100% of targeted faults.

If we examine the results of experiments presented in Table 4, we can see that the test size enlargement due to transformation of PP into functional delay tests is on the average 7.45 for considered circuits. Thus, on the average almost four test pairs are generated for each separate test pattern.

The numbers of detected faults and test coverage are given in Table 4. The average percent of detected transition faults by the tests generated for the function delay faults exceeds 95.59 %, but the minimum percent of detected transition faults is 84.3 % (circuit c3540).

The reuse of transition fault test augmented by functional delay test

In Section 3, we presented the results of the application of transition fault test of one implementation

for the other re-synthesized implementations of the same circuit. The obtained fault coverage is very high, but it never reaches 100%. Of course, such a result is quite good, but in most cases is not satisfactory, because the fault coverage is not 100%. In Section 4, we presented the results of application of functional delay test to test transition faults. The obtained fault coverage is high, too, except the circuit C3540.

Table 5. The Undetected Transition Faults and the Fault Coverage

Circuit	V1	V2	V3	V4	V5	No
C432	0 100	5 99.50	0 100	2 99.83	5 99.52	196 +18
C499	0 100	0 100	0 100	0 100	0 100	376 +14
C880	0 100	0 100	0 100	0 100	0 100	252 +4
C1355	0 100	0 100	0 100	0 100	0 100	500 +154
C1908	0 100	0 100	0 100	0 100	0 100	360 +28
C2670	0 100	1 99.98	1 99.98	1 99.98	1 99.98	422 +72
C3540	0 100	0 100	0 100	0 100	17 99.79	656 +132
C5315	0 100	0 100	0 100	0 100	0 100	484 +48
C6288	0 100	0 100	0 100	0 100	69 99.72	236 +560
C7552	0 100	1 99.99	1 99.99	1 99.99	1 99.99	716 +100
%	100	99.99	99.99	99.99	99.94	

The average percent of all four implementations is 99.98.

The functional delay test is universal in the sense, that it is not related to the particular implementation. It is applicable for any implementation of the circuit. Therefore, the coupling of transition fault test with the functional delay test could improve the fault coverage for any implementation of the circuit. On the base of this idea, we performed the following experiment. The test used in the experiments of Section 3 was complemented by the functional delay test used in the experiments of Section 4. Then we checked the transition fault coverage for all re-synthesized implementations of the circuits as in Section 3. The obtained results are very promising (Table 5). The full transition fault coverage was not obtained only for a few implementations of the circuits. The only single undetected fault was left in all implementations of the circuit C2670 and C7552.

The value of high transition fault coverage obtained after coupling of transition fault test with the functional delay test is overshadowed by one disadvantage – a quite enough big number of test patterns. But this problem can be resolved on the base of information obtained from the test coverage evaluation tool (in our case, TetraMAX), which indicates the test patterns that do not contribute to the value of test coverage. These patterns can be dropped from the whole test set. We have performed an appropriate experiment. The results are presented in the last column of Table 5 for version V5, where the first number represents the amount of patterns selected from transition test set, and the second number is the amount of patterns selected from functional delay test set.

Conclusions

The comparison of the detection of the transition faults for different implementations of the circuit was carried out. The results show that the tests reused for re-synthesized circuits detect on average more than 98% of all transition faults. The maximum percent of undetected faults 7.11% is significantly higher than the average percent of undetected faults 1.31%.

The test sets, which are generated according to the functional delay fault model, obtain high fault coverage of transition faults at the gate level of the circuit. It is very likely the test vectors based on the functional fault model can cover other kinds of the faults. Another advantage of test set generated at the functional level is that it is independent of and effective for any implementation and, therefore, can be generated at early stages of the design process.

The transition fault test augmented by functional delay test achieves full fault coverage almost for all the re-synthesized implementations of the circuits. Such a result is very promising.

References

1. **Y. Zorian, S. Dey, M. Rodgers**. Test of Future System-on-Chips. Proceedings of the 2000 International Conference on Computer-Aided Design, pp. 392-398, November, 2000.
2. **S. B. Akers**, Universal Test Sets for Logic Networks, IEEE trans. Computers, vol. C-22, pp. 835-839, 1973.
3. **R. Betancourt**, Derivation of Minimum test sets for Unate Logic Circuits, IEEE Trans. Computers, vol. C-20, pp1264-1269, Nov. 1971.
4. **S.Ohtake, K.Ohtani, H.Fujiwara**, A Method of Test Generation for Path Delay Faults Using Stuck-at Fault Test Generation Algorithms, Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'03), 2003, pp.310-315.
5. **J. Yi, J.P. Hayes**, A Fault Model for Function and Delay Testing, Proc. of the IEEE European Test Workshop, ETW'01, 2001, pp. 27-34.
6. **H. Kim, J. P. Hayes,** Realization independent ATPG for designs with unimplemented blocks, IEEE Transactions on CAD, vol. 20, no. 2, pp. 290-306, February 2001.
7. **E. Bareiša, V.Jusas, K.Motiejūnas, R.Šeinauskas**, The Influence of Circuit Re-Synthesizing on The Fault Coverage, Information Technology and Control, ISSN 1392-124X. Kaunas, Technologija, 2004, Nr. 2 (31), pp.7-15
8. **J.A. Waicukauski, E. Lindbloom, B.K. Rosen**, V.S. Iyengar, Transition fault simulation, IEEE Design and Test, April 1987, pp.32-38.
9. **V.S. Iyengar, B.K. Rosen, J.A.Waicukauski**, On Computing the Sizes of Detected Delay Faults, IEEE TCAD, March 1990, pp.299-312.
10. **C.J.Lin, S.M. Reddy**, On Delay Fault Testing in Logic Circuits, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 6, September 1987, pp.694-703.
11. **B. Underwood, W.O. Law, S. Kang, H. Konuk**. Fastpath: A path-delay test generator for standard scan designs. Proceedings of International Test Conference, 1994, pp.154–163.

12. **I. Pomeranz, S.M. Reddy**. On testing delay faults in macro-based combinational circuits. Proceedings of International Conference on Computer-Aided-Design, 1994, pp.332–339.
13. **I. Pomeranz, S.M. Reddy**. Functional test generation for delay faults in combinational circuits. Proceedings of International Conference on Computer-Aided-Design, 1995, pp. 687–694.
14. **E. Bareisa, R. Seinauskas**. Test Selection Based on the Evaluation of Input Stuck-at Faults Transmissions to Output. Information Technology and Control, Kaunas, Technologija, 1996, No.2(3), pp.15-18.
15. **E. Bareiša, V. Jusas, K. Motiejūnas, R. Šeinauskas**. The Realization-Independent Testing Based on the Black Box Fault Models. Informatica, 2005, Volume 16, Nr. 1, pp. 19-36.
16. **I. Pomeranz, S. M. Reddy**, Test Enrichment for Path Delay Faults Using Multiple Sets of Target Faults, Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'02), 2002, pp.722-729.
17. **I. Pomeranz, S. M. Reddy**, On Generating High Quality Tests for Transition Faults, Proceedings of the 11th Asian Test Symposium (ATS'02) November 18-20, 2002, pp. 1-8.
18. **J.-J. Liou, L.-C. Wang, K.-T. Cheng**, On theoretical and practical considerations of path selection for delay fault testing, Proceedings of the 2002 IEEE/ACM International Conference on Computer-aided Design, 2002, San Jose, California, USA, November 10-14, 2002, pp.94-100.
19. **K. Yang, K.-T. Cheng, L.-C. Wang**, TranGen: A SAT-Based ATPG for Path-Oriented Transition Faults, Proceedings of the 41th Design Automation Conference, DAC 2004, San Diego, CA, USA, June 7-11, 2004, pp.92-97.
20. **E.Bareiša, V.Jusas, K.Motiejūnas, R.Šeinauskas**. Transition Fault Coverage for Different Implementations of the Circuit. Electronics and Electrical Engineering, ISSN 1392-1215, Kaunas, Technologija, 2005, No. 3(59), pp. 79 – 83.
21. **B. Underwood, W.O. Law, S. Kang, H. Konuk**. Fastpath: A path-delay test generator for standard scan designs. In Proceedings of 1994 International Test Conference (1994), pp.154–163.
22. **I. Pomeranz, S.M. Reddy**. On testing delay faults in macro-based combinational circuits. In Proceedings of 1994 International Conference on Computer-Aided-Design (1994), pp.332–339.

Presented for publication 2006 02 15

E. Bareiša, V. Jusas, K. Motiejūnas, R. Šeinauskas. Transition Test Supplement // Electronics and Electrical Engineering. - Kaunas: Technologija, 2006. – No. 3(67). – P.19 – 24.

The design complexity of systems on a chip drives the need to reuse intellectual property cores, whose gate-level implementation details are unavailable. The core test depends on manufacturing technologies and changes permanently during a design lifecycle. The purpose of this paper is to assist the designer in the decision making how to test transition faults of re-synthesized intellectual property cores. The comparison of the detection of the transition faults for different implementations of the circuit was carried out. Our experiments show that the test sets generated for a particular circuit realization fail to detect in average only less than one and a half percent of the transition faults of the re-synthesized circuit but in some cases this figure is more than 7%. The possibility of reuse the functional delay test was studied, too. Ill.1, bibl. 22 (in English, summaries in English, Russian and Lithuanian).

Э. Барейша, В. Юсас, К. Мотеюнас, Р. Шейнаускас. Пополнение тестов задержки // Электроника и электротехника. – Каунас: Технология, 2006. – № 3(67). – С. 19 – 24.

При проектировании сложных современных систем используются уже готовые блоки, детали которых на вентильном уровне являются неизвестными. Тест для такого блока зависит от технологии изготовления и может каждый раз меняться. Цель этой работы помочь проектировщику принять решение по поводу тестирования дефектов задержки таких блоков. Мы провели много экспериментов, используя комбинационные схемы. Наши эксперименты показывают, что тесты одной реализации в среднем не обнаруживают только 1,5% дефектов задержки других реализаций, однако в некоторых случаях эта цифра равна 7%. Та же тенденция наблюдается и для константных одиночных неисправностей. Ил. 1, библ. 22 (на английском языке; рефераты на английском, русском и литовском яз.).

E. Bareiša, V. Jusas, K. Motiejūnas, R. Šeinauskas. Vėlinimo testų papildymas // Elektronika ir elektrotechnika. – Kaunas: Technologija, 2006. – Nr. 3(67). – P.19 – 24.

Projektuojant lustų sistemas, naudojami iš anksto paruošti blokai, kurių ventilinio lygmens įgyvendinimo detalės yra nežinomos. Šių blokų testai priklauso nuo gamybos technologijos ir kiekvieną kartą gali keistis. Straipsnio tikslas – padėti projektuotojui priimti sprendimą dėl tokių blokų vėlinimo gedimų testavimo. Atlikus daug eksperimentų su standartinėmis kombinacinėmis schemomis, nustatyta, kad vienos realizacijos testai neaptinka vidutiniškai tik 1,5% vėlinimo gedimų persintezuotose kitose tos pačios schemos realizacijose, nors kai kuriais atvejais šis skaičius viršija 7%. Testų kokybę kitoms realizacijoms galima pagerinti naudojant funkcinius vėlinimo testus. Il. 1, lent. 5, bibl. 22 (anglų kalba; santraukos , anglų, rusų ir lietuvių k.).