ISSN 1392 – 124X INFORMACINĖS TECHNOLOGIJOS IR VALDYMAS, 2004, Nr. 2(31)

THE INFLUENCE OF CIRCUIT RE-SYNTHESIZING ON THE FAULT COVERAGE

Eduardas Bareiša, Vacius Jusas, Kęstutis Motiejūnas, Rimantas Šeinauskas

Software Engineering Department, Kaunas University of Technology
Studentų St., LT-3031 Kaunas, Lithuania

Abstract. The design complexity of systems on a chip drives the need to reuse legacy or intellectual property cores, whose gate-level implementation details are unavailable. The core test depends on manufacturing technologies and changes permanently during a design lifecycle. In this paper we consider the impact of circuit realization on the fault coverage of the test set. We have performed various comprehensive experiments with combinational benchmark circuits. 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 stuck-at faults of the re-synthesized circuit but in some cases this figure is more than nine percent. The double test sets declined almost twice both the maximum and the average percent of undetected faults. The experiments exhibit that the supplement of the test set with sensitive adjacent test patterns significantly increases the fault coverage of the re-synthesized core.

1. Indroduction

Many recent system-on-a-chip (SOC) ICs incorporate pre-designed and reusable circuits, 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 an 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. The tests can be generated for a high level description in order to reuse them for all possible implementations [1]. However, such tests usually can not guarantee detection of all specified faults in all possible implementations. Consequently, if we consider realization-independent testing, we can only speak about such realizations that fulfill specific requirements or have a particular structure [2, 3].

In this paper we will analyze the situation when the tests are generated for a particular implementation. In this case there naturally rises a question – can a test generated for one implementation be used for another implementation? The same core can have distinct descriptions; e. g. a parallel or sequential carry can be realized in an adder. Naturally, that a test generated according to one structure may not detect all specified faults of another structure. This case is studied in [4], where it is shown that the deviation of the stuck-at fault coverage can be up to 18% high. The employment of different synthesis tools can have an influence on the test quality as well.

In this work we will analyze only such implementations that are generated by the same synthesis tool according to the same description, changing only 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 experiments. As well we will analyze how the tests for specified faults can be modified or expanded in order to enhance the fault coverage of other realizations and we will evaluate such possibilities by an experiment.

The structure of the paper is as follows. We review the related work in Section 2. We analyze the influence of circuit re-synthesizing on the fault coverage in Section 3. We present the enhancement of the independency of the test from realizations in Section 4. We finish with conclusions in Section 5.

2. Related work

The possibilities of using a test obtained for one realization for testing faults of another realization are studied in [4] and [5]. In [4] H.Kim and J.P.Hayes synthesized two different gate-level implementations of the example circuits, one for low area and another for high speed. The stuck-at fault tests for the gate-level designs were generated using the conventional ATPG program Atalanta [6]. It is stated that Atalanta tests provide 100% stuck-at fault coverage only for the gate-level designs at which they were targeted, and fairly poor coverage for the others. The most impressive number is provided for the ISCAS'85 benchmark circuit c880, namely 100 % stuck-at fault test for high speed realization detects only 82.2% stuck-at faults of the low area realization.

The same experiments are described in [5] too. It is interesting to note that the authors of both papers for the experiments used identical tools – the Synopsys Design Compiler and ATPG program Atalanta. But in [5] it is reported that for the ISCAS'85 benchmark circuit c880 99.8 % stuck-at fault test for high speed realization detects already 99.7% stuck-at faults of the low area realization. However, more interesting is the fact that one of the authors of both articles is J.P.Hayes. Such inconsistence was the main inspiration for us to perform our own experiments.

The possibilities of supplementing or expanding a particular realization test having a purpose to enhance test quality for detecting of various defects are analyzed in [7-10]. The defect coverage that can be achieved with test sets for stuck-at faults may not be sufficient. In order to increase the defect coverage of a test set for stuck-at faults, in [7] and [8] *n*-detection test sets were considered. An *n*-detection stuck-at test set is one where each stuck-at fault *f* is detected by *n* different input patterns, or by the maximum number of input patterns if *f* has fewer than *n* different input patterns that detect it. Experiments with *n*-detection stuck at test sets reported in [7] and [8] show that it is possible to enhance the defect coverage using this approach. In various types of experiments performed in [9] and [10] *n*-detection test sets were shown to be useful in achieving a high defect coverage for all types of circuits and for different fault models.

3. The influence of circuit re-synthesising on the 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 faults of new implementations, which are done by the same synthesizer. The ISCAS'85 benchmarks have been selected for experiments. The original ISCAS'85 circuits have been re-synthesized with the Synopsys Design Compiler program by the default mode and by using the AND-NOT cell library of two inputs. The following three realizations have been analyzed:

R1 – the non-redundant ISCAS'85 benchmark circuit

R2 – Synopsys Design Optimization, the target library – class.db (the default mode)

R3 – Synopsys Design Optimization, the target library – and_or.db

We can see the number of stuck-at faults for each realization in Table 1 and Figure 1. The original benchmark realizations have more stuck-at faults in total. This means that the re-synthesized circuits were more optimized. The percent of the difference between maximum and minimum numbers of stuck-at faults to the maximum number of stuck-at faults varies from 8 to 53. It demonstrates the diversity of realizations and the impact of the target library on the design synthesis.

Table 1. The number of stuck-at faults

Circuit	R1	R2	R3	D	%
C432	507	426	460	81	16
C499	750	978	1246	496	40
C880	942	857	928	85	9
C1355	1566	1316	1406	250	16
C1908	1862	876	1224	986	53
C2670	1990	1500	1658	490	25
C3540	3126	2474	2520	652	21
C5315	5248	3879	4130	1369	26
C6288	7638	6680	7498	958	13
C7552	7039	4578	4798	2461	35
Total	30668	23564	25868		

R1 –The non-redundant ISCAS'85 benchmark circuits

R2 – Synopsys Design Optimization, the target library – class.db

R3 – Synopsys Design Optimization, the target library – and_or.db

D – The difference between maximum and minimum numbers

% – The percent of the difference to the maximum number.

The test sets have been generated for each original ISCAS'85 circuit and for each re-synthesized circuit by a deterministic algorithm and by a random & deterministic algorithm. The deterministic algorithm has been used if the random search did not reach a 100% fault coverage. The test size of test sets with a 100% stuck-at fault coverage is displayed in Table 2. The random test generation increased the test size for all realizations. In each of the test generation cases we see the test size dispersal (Figure 2) in a number of circuits. The test sizes for the realisation of the circuits c432, c880, c2670, c6288 are very similar. These circuits have the smallest dispersal of stuck-at faults after re-synthesizing.

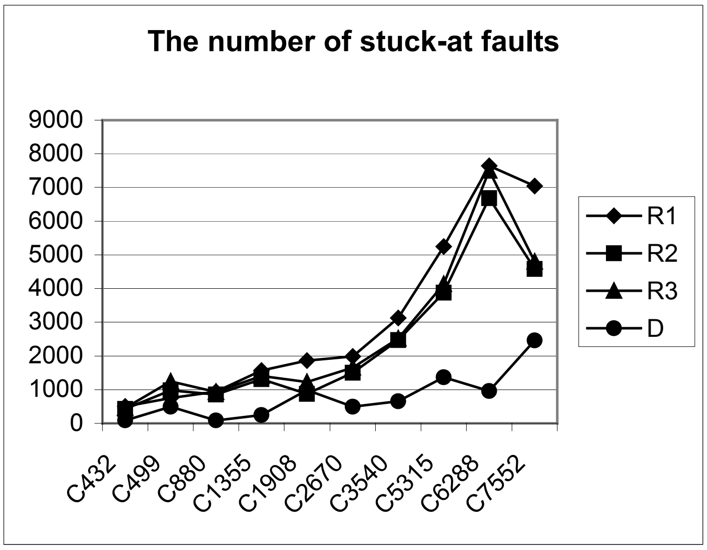

Figure 1. The number of stuck-at faults for each realization

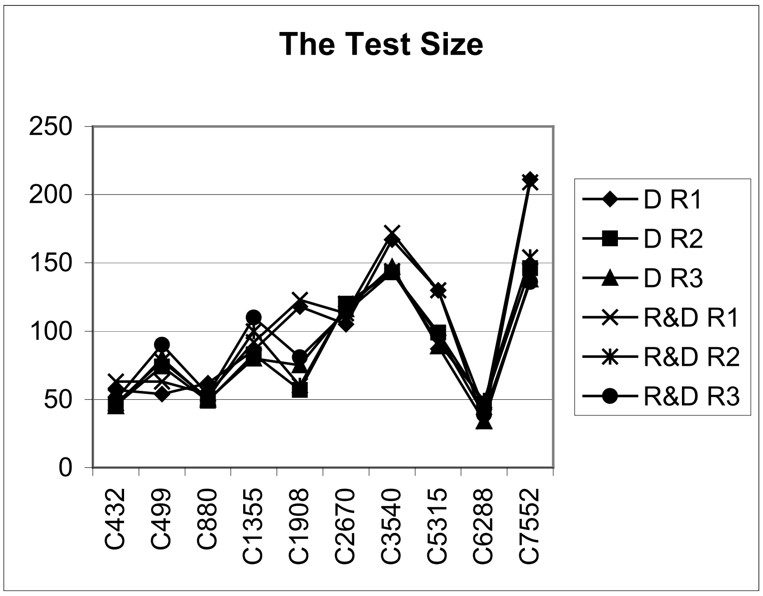

Figure 2. The test size of the deterministic, random&deterministic generated test patterns for all three realizations

Table 2. The size of test sets

Circuit	D			R&D		
	R1	R2	R3	R1	R2	R3
C432	57	46	45	63	47	50
C499	54	74	80	63	78	90
C880	62	49	50	54	51	54
C1355	86	83	80	92	100	110
C1908	118	57	75	123	60	81
C2670	105	120	116	113	120	114
C3540	167	143	147	172	144	143
C5315	130	99	89	130	92	97
C6288	43	47	34	34	49	39
C7552	211	146	138	209	154	136
Total	1033	864	854	1053	895	914

D – Synopsys deterministic test patterns

R&D – Synopsys Random and deterministic test patterns

R1 – The non-redundant ISCAS'85 benchmark circuit

R2 – Synopsys Design Optimization , the target library – class.db

R3 – Synopsys Design Optimization, the target library – and_or.db.

The number of undetected faults of each test set for each circuit realization was computed. Table 3 gives the results of the experiments. Deterministic test sets have been generated for each realization R1, R2, R3 of the benchmark circuits. The fault simulation gives the number of undetected faults for each realization (columns R1, R2, R3). Of course, the number of undetected faults is zero for realizations and tests, which were generated for that particular realization. The test set, generated for the realization R1 of the circuit c432 detects all faults of the realization R1. However, this test set doesn't detect 11 faults of the realization R2 and 1 fault of the realization R3. In general, the tests reused for other realizations in most cases fail to detect all stuck-at faults.

Table 3 Undetected faults for different realizations

Circuit		D			R&D			
		R1	R2	R3	R1	R2	R3	Total
C432	R1	0	21	16	0	13	11	61
	R2	11	0	9	4	0	7	31
	R3	1	7	0	0	8	0	16
C499	R1	0	6	16	0	8	16	46
	R2	44	0	8	34	0	8	94
	R3	116	22	0	48	12	0	198
C880	R1	0	29	18	0	13	17	77
	R2	0	0	2	1	0	1	4
	R3	0	7	0	0	4	0	11
C1355	R1	0	8	12	0	8	16	44
	R2	25	0	12	0	0	8	45
	R3	20	10	0	0	0	0	30
C1908	R1	0	158	129	0	164	120	571
	R2	3	0	12	2	0	11	28
	R3	1	41	0	0	48	0	90
C2670	R1	0	24	21	0	21	29	95
	R2	36	0	4	39	0	6	85
	R3	29	8	0	40	9	0	86
C3540	R1	0	57	53	0	61	56	227
	R2	6	0	6	4	0	4	20
	R3	6	8	0	6	8	0	28
C5315	R1	0	72	77	0	66	96	311
	R2	9	0	17	10	0	25	61
	R3	11	10	0	10	13	0	44
C6288	R1	0	0	0	0	0	0	0
	R2	39	0	21	59	0	28	147
	R3	18	27	0	41	9	0	95
C7552	R1	0	190	241	0	223	236	890
	R2	24	0	44	12	0	37	117
	R3	17	16	0	13	16	0	62
Total		416	721	718	323	704	732	

D – Synopsys deterministic test patterns

R&D – Synopsys Random and deterministic test patterns

R1 – The non-redundant ISCAS'85 benchmark circuit

R2 – Synopsys Design Optimization , the target library – class.db.

R3 – Synopsys Design Optimization, the target library – and_or.db

The random and deterministic generated test sets mainly detect more faults as compared to the deterministic generated test sets (the last row of Table 3). The test sets generated for re-synthesized benchmark circuits detect fewer faults on original benchmark circuits (the last column of Table 3 and Figure 3). Only in two cases the number of undetected faults for an original benchmark circuit is smaller than the number of undetected faults after re-synthesizing (Figure 3). The maximum percent (116/1246) = 9.3 % of undetected faults for deterministic generated test sets has been got for the original realization of the C499 circuit. The maximum percent (164/1862) = 8.8 % of undetected faults for random and deterministic generated test sets has been got for the synthesized C1908 circuit with the target library – class.db.

Each test set generated for one realization was reused for two other realizations. The average percent of undetected faults for the deterministic generated test set and for the random and deterministic generated test set is given in Table 4.

The average percent of undetected faults doesn't exceed 1.5 %. The maximum percent of undetected faults for two realizations reaches (160/2224)= 7.2% in case of test sets for the realization R1. It reaches (212/3086) = 6.9% in case of test sets for the realization R2 and it reaches (141/2738) = 5.1% in case of test sets for the realization R3.

4. Enhancement of the independency of the test from realizations

As it is reported in [7-10], *n*-detection test sets are useful in achieving a higher defect coverage for all types of circuits and for different fault models. We applied merged test sets for testing as a double-detection approach. One set is a deterministic generated test set and the other set is a random and deterministic generated test set. Both test sets have different test patterns and each of them detects all faults of the target realization. The numbers of undetected faults of double test sets are given in Table 5.

The average percent of undetected faults in case of double-detection test sets declined more than twice (Figure 4). Also the maximum percent of undetected faults declined till 2.2 %, 4.2 % and 2.9 % for the realizations R1, R2 and R3, respectively (Figure 5).

The tests reused for other realizations in most cases detect on the average more than 98% of all stuck-at faults. The maximum percent of undetected faults is significantly higher than the average percent of undetected faults. The double test sets decreased almost twice both the maximum and the average percent of undetected faults.

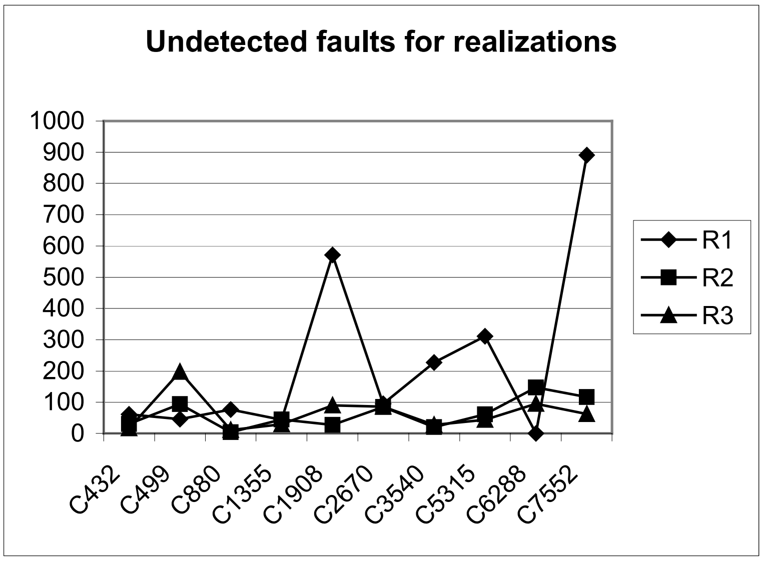

Figure 3. Total number of undetected faults for each realization of the circuit

Table 4. The number of stuck-at faults and undetected faults

Circuit	F_R2+R3	U_R2+R3		F_R1+R3	U_R1+R3		F_R1+R2	U_R1+R2	
		D_T_R1	R_T_R1		D_T_R2	R_T_R2		D_T_R3	R_T_R3
C432	886	11	5	967	29	20	933	25	18
C499	2224	160	82	1996	28	20	1728	24	24
C880	1785	0	1	1870	36	17	1799	20	18
C1355	2712	45	0	2972	18	8	2882	24	24
C1908	2100	4	2	3086	199	212	2738	141	131
C2670	3158	65	79	3648	32	30	3490	25	35
C3540	4994	12	10	5646	65	69	5600	59	60
C5315	8009	20	20	9378	82	82	9127	94	121
C6288	14178	57	100	15136	27	9	14318	21	28
C7552	9376	41	25	11837	206	239	11617	285	273
Total	49422	415	324	56536	722	706	54232	718	732
Percent		0.84%	0.66%		1.28%	1.25%		1.32%	1.35%

F-Ri+Rj – The number of stuck-at faults of two realizations Ri and Rj
U_Ri+Rj – The number of undetected faults of two realizations Ri and Rj
D_T_Ri – The deterministic test set generated for the realization Ri
R_D_Ri – The random and deterministic test set generated for the realization Ri.

Table 5. The number of undetected faults of double test sets

Circuits	F_R2+R3	TR1	U_R2+R3	F_R1+R3	TR2	U_R1+R3	F_R1+R2	TR3	U_R1+R2
C432	886	120	2	967	93	10	933	95	9
C499	2224	90	49	1996	117	10	1728	152	6
C880	1785	116	0	1870	100	5	1799	104	9
C1355	2712	178	0	2972	183	4	2882	190	8
C1908	2100	241	2	3086	117	130	2738	156	80
C2670	3158	218	27	3648	240	17	3490	230	7
C3540	4994	339	3	5646	287	24	5600	290	20
C5315	8009	260	1	9378	191	24	9127	186	30
C6288	14178	77	4	15136	96	0	14318	73	2
C7552	9376	420	4	11837	300	142	11617	274	147
Total	49422	2059	92	56536	1724	366	54232	1750	318
Percent			0.19%			0.65%			0.59%

F-Ri+Rj – The number of stuck-at faults of two realizations Ri and Rj
U_Ri+Rj – The number of undetected faults of two realizations Ri and Rj
TRi – The size of double test sets for the realization Ri.

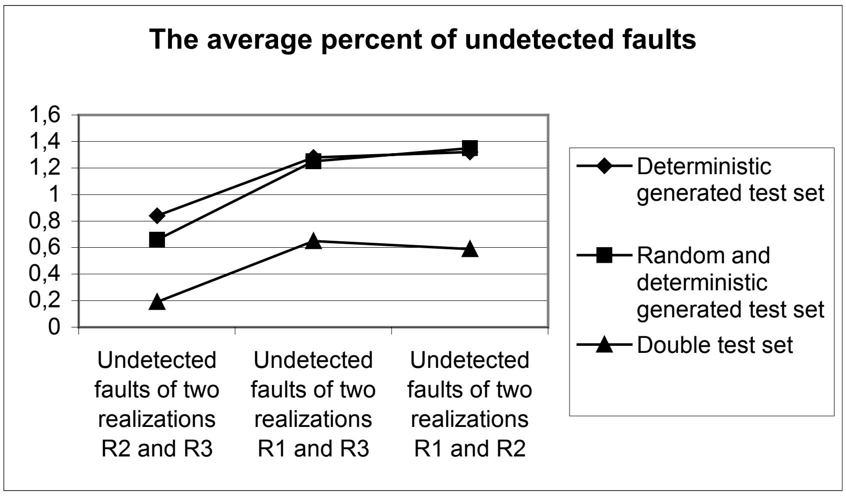

Figure 4. The average percent of undetected faults

Another possibility to enhance the test quality lies in using the sensitive adjacent input vectors. Every pin in the circuit can have two stuck-at faults: stuck-at 1, stuck-at 0. In order to detect stuck-at faults of some pin it is needed to create a sensitive path from the location of the faulty pin to the output of the circuit. If such a path was created, all the stuck-at faults along this path are detected. The creation of a sensitive path requires two test vectors. If a sensitive path starts from the input of the circuit, these test vectors differ only in the value of the input from which the sensitive path starts.

Definition 1. Two input vectors are adjacent if they differ in the value of a single input. The Hamming distance between adjacent input vectors is one.

Definition 2. The adjacent input vectors V and V^* are considered as sensitive adjacent input vectors if output vectors obtained in response to V and V^* are different.

Each input vector of length n has n adjacent input vectors, from which some input vectors might be sensitive adjacent input vectors. Sensitive adjacent input vectors can be generated for each test pattern of the test set. Since a change in the value of a single input of sensitive adjacent input vectors changes the output vector, it is likely that the presence of a fault on a path from a sensitive input to a sensitive output will be detected. The generated sensitive adjacent input vectors are likely to be sensitive to the presence of a defect, and are likely to result in higher fault coverage.

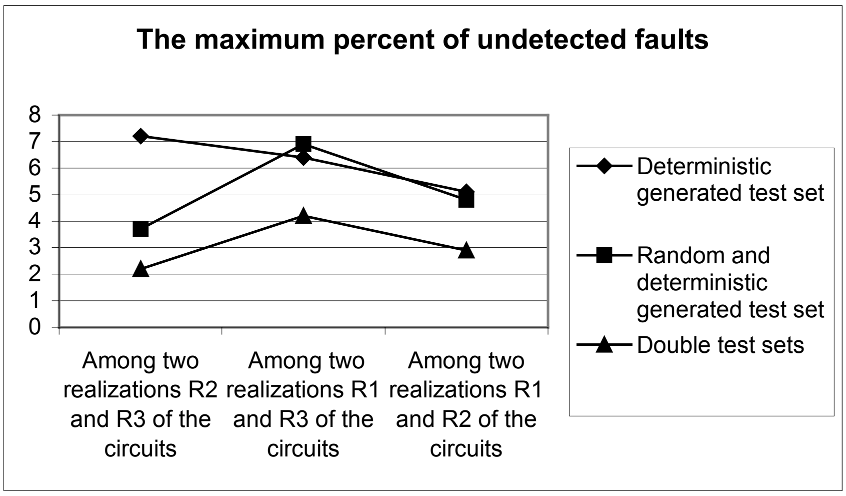

Figure 5. The maximum percent of undetected faults

Table 6. Effect of application of adjacent test vectors

Circuit	Test size	R1		R2		R3		Test size (adjacent)	R1		R2		R3	
		%	Nr.	%	Nr.	%	Nr.		%	Nr.	%	Nr.	%	Nr.
C432	46	97,4	44	100	43	98,5	40	548	100	57	100	43	100	47
	47	95,9	45	100	47	98,3	45	469	99,8	61	100	47	100	52
C499	74	98,9	52	100	74	99,0	73	2837	100	56	100	74	100	81
	78	99,2	57	100	78	98,2	74	3105	100	60	100	78	99,8	86
C880	49	98,6	46	100	46	99,6	46	2031	100	58	100	46	100	49
	51	96,9	50	100	49	99,2	49	2092	100	73	100	49	100	53
C1355	83	99,5	77	100	83	100	79	3149	100	83	100	83	100	79
	100	99,5	92	100	98	99,3	97	3868	100	95	100	98	100	102
C1908	57	91,2	56	99,7	57	96,1	54	1581	99,2	128	99,8	58	99,9	82
	60	91,5	59	99,8	59	96,7	58	1679	99,5	135	99,8	59	100	86
C2670	120	98,9	106	100	116	99,5	116	7911	100	124	100	116	100	122
	120	98,8	109	99,9	117	99,5	118	7873	100	131	100	118	100	124
C3540	143	98,0	141	100	138	99,7	137	4301	100	188	100	138	100	144
	144	98,2	143	100	144	99,7	143	4285	100	188	100	144	100	149
C5315	99	98,7	96	99,7	97	99,7	96	9314	100	150	100	106	100	106
	92	98,6	91	99,8	90	99,8	88	8713	100	146	100	98	100	98
C6288	47	100	32	100	43	99,9	43	1582	100	32	100	43	100	52
	49	100	49	100	49	99,6	49	1648	100	49	100	49	100	76
C7552	146	96,8	143	99,9	142	99,7	129	13197	98,8	229	100	146	100	142
	154	97,3	147	99,9	149	99,7	140	13836	98,8	211	100	152	100	100
Average	88	97,7	82	99,9	86	99,1	84	4701	99,8	113	99,9	87	99,9	92
Aver.(%) deviation	0,66	0,06	1,21	0,004	0,89	0,04	1,04	0,88	0,006	1,42	0	0,87	0,004	1,74
Max.(%) deviation	2,32	0,22	5,25	0,01	2,07	0,1	2,56	2,56	0,04	5,25	0	2,07	0,03	4,67

R1 – The non-redundant ISCAS’85 benchmark circuit
R2 – Synopsys Design Optimization , the target library – class.db
R3 – Synopsys Design Optimization, the target library – and_or.db
Nr. – The number of test patterns selected according to fault simulation
% – The fault coverage.

We investigated how much sensitive adjacent input patterns of test sets increase the fault coverage of different realizations of ISCAS'85 benchmark circuits. The test sets extended with sensitive adjacent input vectors have been minimized by means of the fault simulation.

Initial test patterns were generated using automatic test pattern generation tool for the circuit implementation R2. This implementation denotes the optimised library class.db. The data for every circuit are displayed in two lines of Table 6. Test sets, which size is presented in the first line, were generated in the deterministic mode. Test sets, which size is presented in the second line, were generated in the random mode plus the deterministic mode in order to get 100% fault coverage.

The left half of Table 6 presents the results of the test pattern generation before the application of the procedure for the sensitive adjacent patterns generation. The right half of Table 6 shows the results of the application of the sensitive adjacent patterns. The selection of the test patterns was done for three implementations of every circuit. The particular implementation of the circuit is presented in two columns: the fault coverage and the number of minimized test patterns. The minimization of test patterns was based on the results of the fault simulation. The simple rule was applied: the test pattern is valuable if it detects new faults. If we rearrange initial test patterns, we would get a different number of minimized test patterns.

The last three lines of Table 6 were calculated in order to prove the trustiness of the results of the test pattern generation. The average for every column is presented in the line "Average". Calculation of values in the other two lines requires a longer explanation. An experiment for every circuit was carried out two times. The average of the results was calculated for every circuit separately. Then the deviation from the corresponding average was calculated for every circuit and expressed in percents. Finally, the average deviation that is shown in the second line "Average deviation" was calculated. The last line "Maximum deviation" shows the maximum deviation in percents from the average. As we see, the last two lines have very small numbers. So this means that the distinction of the results between separate generations is very small. Therefore, these small numbers prove the trustiness of the results of the test patterns generation.

If we look at the right half of Table 6, we will see bigger numbers than in the left half, except numbers of the circuits, which have a 100% fault coverage initially. Such results mean that sensitive adjacent test patterns always add their value to the fault coverage. This conclusion is valid for any implementation of the circuit. Sensitive adjacent patterns are especially good for the and_or implementation (R3). As we can see from the left part of Table 6, one test sequence of the R3 implementation had full fault coverage (100%). After the application of the procedure for the adjacent vector generation only two test sequences (circuits c449 and c1908) didn't have full fault coverage for circuits of the R3 implementation. Another indicator that could emphasize the value of sensitive adjacent vectors is the number of undetected faults that is on the left and right halfs of the table. So the left half has 688 undetected faults jointly, whereas the right half has only 72 undetected faults jointly.

Some attention has to be drawn to other results of Table 6. As it was mentioned above, initial test patterns were generated by automatic test pattern generation tool for the implementation R2. So, we would expect 100% fault coverage for every circuit of this implementation. But the circuits C1908, C5315 and C7552 don't have 100% fault coverage initially. This could be explained as follows. The library class.db includes some hierarchical elements. The test generation was carried out at the hierarchical level, but the fault simulation was carried out at the gate level. Therefore, some circuits don't have initially 100% fault coverage for the implementation R2. But despite this drawback application of adjacent patterns gives 100% fault coverage for every implementation of every circuit, except the circuit C1908. Such a result only sharpens the value of adjacent test patterns. Hence the application of the adjacent test patterns showed surprisingly good results – when the circuit didn't have a 100% fault coverage, the fault coverage was enhanced in every case;

5. Conclusions

The tests reused for re-synthesized circuits detect on average more than 98% of all stuck-at faults. The maximum percent of undetected faults is significantly higher than the average percent of undetected faults. The double test sets declined the maximum and the average percent of undetected faults almost twice.

The extension of a test set with the sensitive adjacent patterns is a very cheap way to adopt test patterns for the re-synthesized gate level description of the IP core.

On the base of presented results we can make a recommendation concerning the IP core test suites. When the IP core is supplied to the user, it is presented at the behavioural level. Its gate-level implementation details are unavailable. Therefore the user has to synthesize gate level description herself. Test suites are supplied together with IP core. These test suites reflect the behavior of the IP core and are devoted only to a particular gate level implementation. The supplied test suites of the IP core are not able to detect all faults of any synthesized gate level implementation. Therefore there is a problem how to get a test for a re-synthesized gate level implementation of the IP core. We suggest complementing the existent test suites of the IP core with sensitive adjacent patterns. Then the suitable test patterns for the synthesized gate level implementation have to be selected on the base of the fault simulation. Our experiment proves that such a complement would enhance the test quality for any synthesized IP core gate level description. We believe that the practice of sensitive adjacent patterns is a very cheap way to adopt test patterns for the re-synthesized gate level description of IP core.

References

[1] **Y. Zorian, S. Dey, and M. Rodgers.** Test of Future System-on-Chips. *Proceedings of the 2000 International Conference on Computer-Aided Design*, *November*, 2000, 392-398.

[2] **S. B. Akers.** Universal Test Sets for Logic Networks, IEEE trans. *Computers, Vol.*C-22, 1973, 835-839.

[3] **R. Betancourt.** Derivation of Minimum test sets for Unate Logic Circuits. *IEEE Trans. Computers, Vol.*C-20, *Nov*. 1971, 1264-1269.

[4] **H. Kim and J.P. Hayes.** High-Coverage ATPG for datapath circuits with unimplemented blocks. *Proc. Int. Test Conf., Oct*. 1998, 577-586.

[5] **J. Yi and J.P. Hayes.** A Fault Model for Function and Delay Testing. *Proc. of the IEEE European Test Workshop, ETW'*01, 2001, 27-34.

[6] **H. K. Lee and D. S. Ha.** On the generation of test patterns for combinational circuits. *Technical Report* 12-93, *Department of Electrical Eng., Virginia Polytechnic Institute and State University*, 1993.

[7] **S.C. Ma, P. Franco and E.J. McCluskey.** An experimental chip to evalueta tets teshniques. Experimental results. *Proc*. 1995 *Intl. Test Conf., Oct* 1995, 663-672.

[8] **S. M. Reddy, I. Pomeranz, and S. Kajihara.** On the Effects of Test Compaction on Defect Coverage. *Proc. VLSI Test Symp., Apr*. 1996, 430--435.

[9] **I. Pomeranz and S. M. Reddy.** On n-Detection Test Sets and Variable n- Detection Test Sets for Transition Faults. *Proc*. 17th *VLSI test Symp., April* 1999, 173-179.

[10] **H. Takahashi, K.K. Sulaja, Y. Takamatsu.** An Alternative Method of Generating Tests for Path Delay Faults Using N -Detection Test Sets. *Proc. of the* 2002 *Pacific Rim International Symposium on Dependable Computing* (*PRDC'*02), 2002.

DOI: 10.5755/j01.itc.31.2.11844