ISSN 1392 – 124X INFORMACINĖS TECHNOLOGIJOS IR VALDYMAS, 2004, Nr. 4(33)

THE IMPROVEMENT OF TEST INDEPENDENCE FROM CIRCUIT REALIZATION

Eduardas Bareiša, Vacius Jusas, Kęstutis Motiejūnas, Šarūnas Packevičius, Rimantas Šeinauskas

Software Engineering Department, Kaunas University of Technology
Studentų 50-406., LT-51368 Kaunas, Lithuania

Abstract. We consider the possibilities of supplementing or expanding a particular realization test having a purpose to enhance test quality for detecting various defects. We suggest complementing the existing 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 experiments prove 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, because the fault simulation is not so critical task as a test generation.

1. Indroduction

System-on-a-chip (SoC) design is built on automatic topology synthesis according to behavioural descriptions of components. Such descriptions together with technological libraries are initial data for automatic topology synthesis. Due to considerable complexity the SoC test is performed by testing separate components. Components tests and defects under testing depend on the SoC production technology under usage. If technological libraries were changed or if the project was resynthesized the components test may not test all their defects. The problem how to modify the component's one realization test so that this test would test more defects after having it resynthesized and after technological libraries have changed is topical.

The possibilities of supplementing or expanding a particular realization test having a purpose to enhance test quality for detecting various defects are analysed in [1-6]. 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 [1] and [2] *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 [1], [2] and [3] show that it is possible to enhance the defect coverage using this approach. In various types of experiments performed in [4] and [5] *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. Another possibility called sensitivity of adjacent input patterns is proposed in [6]. Under a sensitive pattern, a change in a single input value causes a change in at least one output value. Such patterns are likely to be sensitive to the presence of faults, and are likely to result in fault detection [6].

In this paper we consider the sensitive adjacent input vectors and their capabilities to enhance test quality for detecting stuck-at faults. The paper is organized as follows. In Section 2 we provide definitions of sensitive patterns. In Section 3 we analyze the properties of sensitive adjacent input vectors. In Section 4 we summarize the experimental results of the improvement of test independence. Section 5 concludes the paper.

2. Definitions

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 place of the faulty pin to the output of the circuit. If such a path is created, it is likely that the presence of a fault on a path from a sensitive input to a sensitive output will be 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 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 the length n has n adjacent input vectors, from which some input vectors might be sensitive adjacent input vectors.

For example, consider an input vector 01010 that produces an output vector 101. We compare the output responses of adjacent input vectors 11010, 00010, 01110, 01000, 01011. Let these responses be 101,101,111,101,010. After comparing the output response 101 with the output responses 111 and 010, we find that these output responses are different, and we mark the input vectors 01110 and 01000 as sensitive adjacent input vectors of the considered input vector 01010. We refer to the input where vectors V and V^* differ as a sensitive input, and to the outputs that assume different values under V and V^* as sensitive outputs.

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.

In [7] a design constraint that ensures full coverage of stuck-at faults in the two-level circuit realization is derived. A sum-of-products (SOP) cover E of the function z is a set of implicants of $z = p_1+p_2+\ldots+p_k$. An implicant p_i of E is relatively essential if the result of deleting p_i from E covers fewer minterms of z than E. An input vector v is relatively essential in p if $p(v)=1$, but $p^*(v)=0$ for some other implicant $p^*/=p$ in E. A relatively essential implicant contains at least one relatively essential vector. A non redundant SOP realization is composed of relatively essential prime implicants. The following lemma proved in [7] states the sufficient conditions for a test set to detect all stuck-at faults.

Lemma 1. A test set T detects all stuck-at faults in a non-redundant SOP circuit realization if T includes:

- At least one relatively essential vector v of every prime implicant p in E
- At least one false vector adjacent to a minterm of p for every literal in p

The test set that satisfies the conditions of Lemma 1 is likely to result in high fault coverage for multi-level circuit realizations.

3. The properties of sensitive adjacent input vectors

We investigate how much the sensitive adjacent input patterns of test sets generated for the PP faults [8] and by a deterministic test generator increase the fault coverage of different realizations of ISCAS'85 benchmark circuits [9]. 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

The test sets generated for the PP faults of the black box model and the test sets extended with sensitive adjacent input vectors have been minimized by means of the fault simulation.

All results of the experiment concerning sensitive adjacent vectors are presented in Tables 1, 2 and 3**.** Test patterns were generated on the basis of the PP fault model according to the circuit black box model. This means that no information on the structure of the circuits was used during the test pattern generation. Only the functions of the circuits were taken into account. The random test pattern generation was used. When the random test pattern generation is used anyone can argue that test results can significantly differ from one run to another. In order to show the trustiness of the test generation results the experiment for every circuit was carried two times. Results in Tables 1 and 2 differ in the application of the procedure for the sensitive adjacent patterns generation. Table 1 shows results when sensitive adjacent patterns were inserted into the initial sequence of test patterns. This means that the order of the generation was as follows: the initial pattern, then its sensitive adjacent patterns, and such a mode was applied for every initial test pattern. Table 2 shows the results when adjacent patterns were added to the end of the initial test sequence. The mode of adding sensitive adjacent patterns has a big influence to the number of minimized test patterns after the fault simulation. The second mode selects much smaller number of minimized patterns. Therefore, it was decided that the second mode is the right one, and this mode was used for other experiments, which results are presented in Table 3.

Initial test patterns (Table 3) were generated by automatic test pattern generation tool for the circuit implementation R2. Test patterns presented in the first line of Table 3 for every circuit were generated in the deterministic mode. Test patterns presented in the second line of Table 3 for every circuit were generated in the random mode plus the deterministic mode in order to get 100% fault coverage.

Table 1. Application of adjacent test vectors inside the initial test sequence

Circuit	Nr.	R1		R2		R3		Nr. adjacent	R1		R2		R3	
		%	Nr.	%	Nr.	%	Nr.		%	Nr.	%	Nr.	%	Nr.
C432	109	96.1	63	98.8	58	98.9	60	986	100	81	100	68	100	72
	120	95.5	70	97.4	60	97.8	65	1091	99.6	92	100	73	100	81
C499	1035	100	67	100	87	100	96	36627	100	96	100	120	100	129
	1045	100	67	100	93	100	100	37000	100	97	100	120	100	133
C880	364	99.9	108	99.9	89	99.9	88	13962	100	203	100	178	100	180
	358	99.5	102	99.5	91	99.6	89	13627	100	188	100	161	100	164
C1355	1029	100	109	100	116	100	122	36411	100	139	100	153	100	160
	1028	100	106	100	118	100	122	36389	100	154	100	181	100	184
C1908	613	97.7	139	98.9	82	99.3	101	16389	99.8	193	99.8	121	100	151
	622	97.0	137	99.0	85	99.3	107	16657	99.7	196	99.8	124	100	155
C2670	422	99.4	155	99.3	142	99.5	146	25753	99.9	325	99.9	281	100	292
	459	99.9	160	99.7	159	99.9	160	29081	100	334	100	291	100	300
C3540	497	98.9	202	99.0	174	98.8	173	14571	100	366	100	316	100	321
	525	98.3	212	98.8	187	98.6	190	15309	100	363	100	305	100	315
C5315	1151	100	198	100	155	100	154	110221	100	520	100	468	100	480
	1161	100	203	100	153	100	151	110431	100	596	100	477	100	473
C6288	238	100	49	100	64	100	63	7842	100	144	100	207	100	187
	263	100	52	100	63	100	75	8475	100	159	100	198	100	192
C7552	1642	98.9	312	99.6	234	99.6	213	138087	99.7	702	100	520	100	554
	1727	98.9	318	99.5	245	99.6	222	144800	99.7	708	99.9	528	99.9	561
Average	720	99	142	99.5	123	99.5	125	40686	99.9	283	99.9	245	99.9	254
Aver.(%) deviation	0.54	0.03	0.49	0.03	0.54	0.03	0.76	0.56	0.01	0.76	0.003	0.64	0.001	0.36
Max.(%) deviation	1.25	0.09	1.32	0.18	1.41	1.14	2.17	1.51	0.07	1.70	0.01	2.09	0.01	1.74

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

% – The fault coverage

Table 2. Application of adjacent test vectors at the end of the initial test sequence

Circuit	Nr.	R1		R2		R3		Nr. adjacent	R1		R2		R3	
		%	Nr.	%	Nr.	%	Nr.		%	Nr.	%	Nr.	%	Nr.
C432	116	97	68	98.4	59	98.5	62	1053	100	81	100	66	100	68
	128	95.9	65	98.1	61	98	62	1148	100	77	100	69	100	71
C499	1035	100	67	100	87	100	96	36627	100	67	100	87	100	96
	1045	100	67	100	93	100	100	37000	100	67	100	93	100	100
C880	351	99.7	98	99.8	78	99.8	79	13496	100	101	100	80	100	81
	344	99.6	104	99.8	85	99.7	85	13175	100	108	100	87	100	88
C1355	1029	100	109	100	116	100	122	36411	100	109	100	116	100	122
	1028	100	106	100	118	100	122	36389	100	106	100	118	100	122
C1908	648	96.8	132	98.6	78	98.7	93	17281	99.7	158	99.8	84	100	103
	608	97.3	135	98.6	80	98.9	103	16290	99.6	154	99.8	87	100	112
C2670	460	99.9	160	99.7	159	99.9	160	29255	100	162	100	164	100	162
	428	99.7	155	99.7	147	99.8	150	27389	100	160	100	151	100	153
C3540	496	98.8	229	99.4	189	99.1	190	14488	100	255	100	201	100	208
	520	98.9	211	99.2	183	98.8	184	15199	100	242	100	202	100	211
C5315	1151	100	198	100	148	100	146	110541	100	198	100	148	100	146
	1161	100	203	100	153	100	151	110852	100	203	100	153	100	151

Circuit	Nr.	R1		R2		R3		Nr. adjacent	R1		R2		R3	
		%	Nr.	%	Nr.	%	Nr.		%	Nr.	%	Nr.	%	Nr.
C6288	268	100	50	100	62	100	68	8875	100	50	100	62	100	68
	263	100	52	100	63	100	75	8475	100	52	100	63	100	75
C7552	1768	98.9	312	99.6	225	99.7	204	148323	99.7	350	99.9	240	100	216
	1727	98.9	318	99.5	245	99.6	222	144800	99.7	359	99.9	260	99.9	233
Average	729	99.1	142	99.5	122	99.5	124	41353	99.9	153	99.9	127	99.9	129
Aver.(%) deviation	0.45	0.03	0.04	0.008	0.59	0.02	0.66	0.46	0.001	0.41	0	0.58	0.001	0.66
Max.(%) deviation	1.23	1.14	1.02	0.04	1.07	0.06	1.28	1.08	0.01	0.84	0	1.05	0.01	1.22

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

% – The fault coverage

Table 3. Application of adjacent test vectors when the test sequence is generated by ATPG

Circuit	Nr	R1		R2		R3		Nr. 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

% – The fault coverage

The left part of each of the tables presents results of the test pattern generation before the application of the procedure for the sensitive adjacent patterns generation. The right part of tables shows results with sensitive adjacent patterns. The test patterns selection was done for three implementations of every circuit.

Two columns are used to present the results of every implementation of the circuit: 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 in every table were calculated in order to prove the trustiness of the results of the test pattern generation. The line "Average" gives the average for every column. The calculation of values in the other two lines requires a longer explanation. As we remember, 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 per cents. Finally, the average deviation that is shown in the line "Average deviation" was calculated. The line "Maximum deviation" shows the maximum deviation in per cents from the average. As we see, the numbers listed in the last two lines are very small. So it 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 part of the tables, we will see bigger numbers than in the left part of the tables with exception for 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 1, 12 test sequences for 6 circuits of the R3 implementation didn't have full fault coverage (100%). After the application of the procedure for the adjacent vector generation only a single test sequence (circuit c7552) 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 parts of the table. So the left part has 688 undetected faults in total, whereas the right part has only 72 undetected faults in total. The very similar result may be confirmed in Table 2 (the left part - 670 undetected faults, the right part – 70 undetected faults).

Some attention has to be drawn to the results of Table 3. As we have said, 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 in such a way. 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 addition of adjacent patterns gives 100% fault coverage for every implementation of every circuit, except the circuit C1908. Such a result only sharpens the worth of adjacent test patterns.

Finally, Figure 1 shows how the number of test patterns subject to the test generation mode.

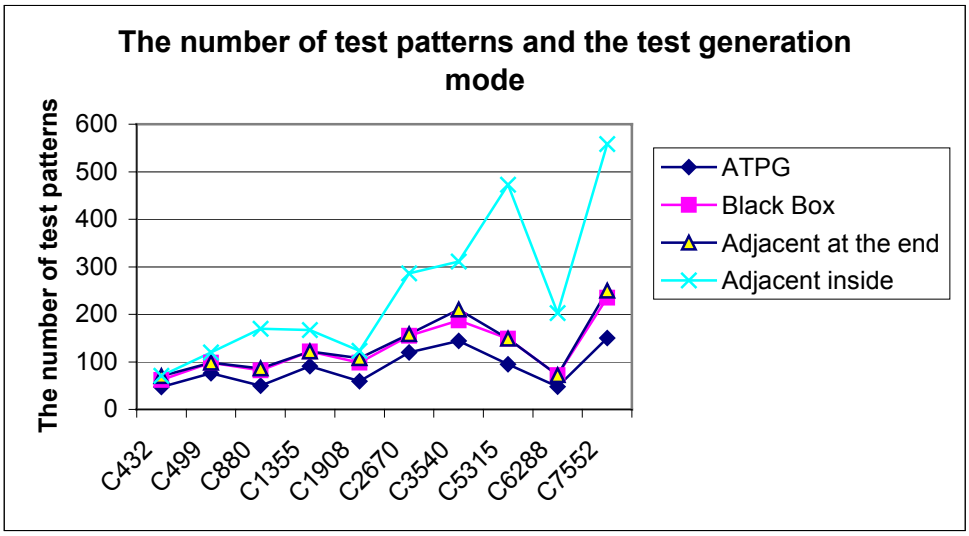

Figure 1. The number of test patterns and the test generation mode

Based on the results of the experiment we can draw the following conclusions: the addition of the adjacent test patterns showed surprisingly good results – when the circuit didn't have a 100% fault coverage, the fault coverage was improved in every case; the addition of the adjacent test patterns at the end of the test sequence is better than their insertion inside the test sequence.

4. The generalization of the improvement of test independence

Adding the sensitive adjacent patterns increases the test size by an order of magnitude or more (for considered circuits on average 53 times, the increase of test size ranges from 10 times for circuit c432 to 95 times for circuit c5315). Therefore, naturally raises a question – how much of the improvement in coverage that is being observed is coming from the fact that sensitive adjacent patterns are being used and how much is coming simply from the fact that so many more test patterns are being used? To get an answer of that, we made next experiment. Namely, for each circuit we have generated the same number of random patterns. The data for every circuit are displayed in two lines of Table 4. Test sets, whose size is presented in the first line, were generated in the random mode. Test sets, whose size is given in the second line, are deterministic tests plus the same random tests.

Table 4. Effect of application of random test vectors

Circuit	Test size	R1		R2		R3	
		%	Nr.	%	Nr.	%	Nr.
C432	500	99.8	53	99.5	48	100	48
	46+500	100	54	100	50	100	48
C499	3000	100	58	100	85	100	94
	74+3000	100	58	100	85	100	94
C880	2000	98.9	80	99.1	65	99.1	63
	49+2000	100	85	100	69	100	67
C1355	3500	100	90	100	102	100	104
	83+3500	100	90	100	102	100	104
C1908	1600	97.7	125	99.3	70	98.5	82
	57+1600	98.3	133	99.8	73	99.3	89
C2670	7800	84.7	78	80.0	61	80.8	64
	120+7800	99.9	128	100	118	99.6	120
C3540	4200	99.6	176	99.7	147	99.7	155
	143+4200	99.9	181	100	152	100	160
C5315	9200	100	156	100	121	100	117
	99+9200	100	156	100	121	100	117
C6288	1600	100	33	100	43	100	46
	47+1600	100	33	100	43	100	46
C7552	13700	96.8	203	97.9	153	98.0	135
	146+13700	98.7	245	100	194	100	174
Average	4710	97.75	105	97.55	90	97.61	91
	5246	99.68	116	99.98	101	99.89	102
	Random without C2670	99.20	108	99.50	93	99.48	94

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

The last experiment shows that the effectiveness of the sensitive adjacent test patterns is not as high as it seemed from experimental results presented in Tables 1-3. The fault coverage after appliance of pure random test patterns is 2.29% on average lower than in the case of appliance of sensitive adjacent test patterns, however if we exclude the circuit c2670 that is likely hard random pattern testable, this number will be 0.53%. Furthermore, if we compare the average fault coverage numbers between random test sets plus deterministic test sets and the test sets with sensitive adjacent test patterns (reminder – they include deterministic tests too), we get the difference only 0.08% in behalf of sensitive adjacent test patterns.

In order to reduce the size of the set of sensitive adjacent patterns, we selected only some patterns from sensitive adjacent patterns. The number of sensitive adjacent patterns for each input output pair was limited to 10 and 1. Further, on purpose to get an integrated presentation for the improvement of the independency of the test from realizations, we calculated for all considered improvement cases the total size of tests and the total number of undetected faults for three realizations. The random generated test set,

witch length is equal the deterministic test set size multiplied by the number of the circuit inputs and the deterministic test set plus the random generated test set, witch length is equal the deterministic test set size multiplied by the number of the circuit inputs were generated as well. All considered cases are presented in Table 5 and Figure 2.

Table 5. Total number of undetected faults and the total number of test patterns for three realizations of benchmark circuits

Case	Test generation mode	Test size	Undetected faults
C1	Deterministic test sets	864	721
C2	Double test sets	1724	366
C3	Deterministic test sets plus the sets of sensitive adjacent test patterns	46451	75
C4	Deterministic test sets plus the sets of sensitive adjacent test patterns limited for each input-output pair to 10	29955	84
C5	Deterministic test sets plus the sets of sensitive adjacent test patterns limited for each input-output pair to 1	7613	190
C6	Deterministic test sets plus the set of random generated test patterns of the length of sensitive adjacent patterns	47964	147
C7	Sets of random generated test patterns of the length of sensitive adjacent patterns	47100	1464
C8	Deterministic test sets plus random generated test sets, witch length is equal the deterministic test set size multiplied by the number of the circuit inputs	89070	124
C9	Random generated test sets, witch length is equal the deterministic test set size multiplied by the number of the circuit inputs	88206	1335

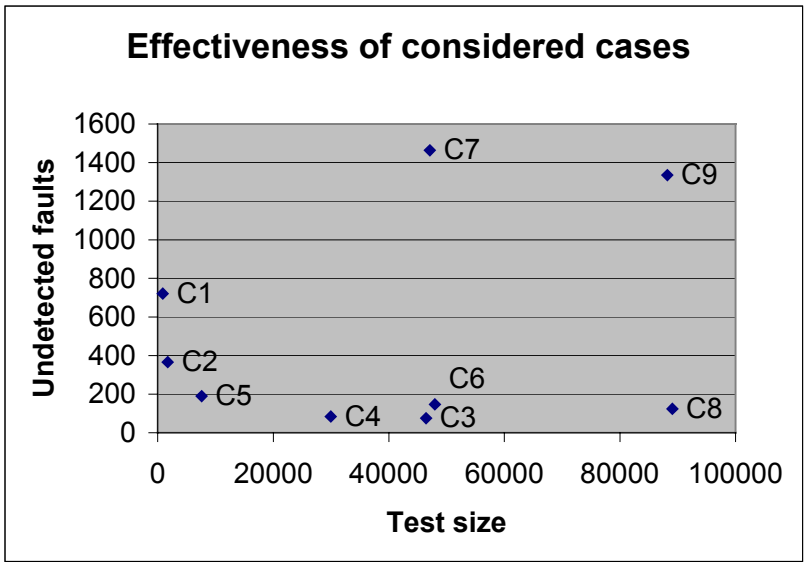

Figure 2. Effectiveness of the improvement of the test independency from realizations

5. Conclusions

On the base of presented results we can make a recommendation concerning the combinational 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 behaviour 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 all sensitive adjacent patterns or with a subset of such 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, because the fault simulation is not so critical task as test generation.

References

[1] **S.C. Ma, P. Franco, E.J. McCluskey.** An experimental chip to evaluate test techniques: experimental results. *Proc*. 1995 *Intl. Test Conf., Oct* 1995, 663 - 672.

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

[3] **E. Bareiša, V. Jusas, K. Motiejūnas, R. Šeinauskas.** The Influence of Circuit Re-synthesizing on the Fault Coverage. *Information technology and control, Kaunas, Technologija*, 2004, *No*.2(31), 7 - 15.

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

[5] **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.

[6] **I. Pomeranz, S.M. Reddy.** Pattern Sensitivity: A Property to Guide Test Generation for Combinational circuits. *Proc*. 8-*th Asian Test Symposium*, 1999, 75 - 80.

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

[8] **V. Jusas, R. Seinauskas.** Automatic Test Patterns Generation forSimulation-based Validation. *Proc. of the* 8-*th Biennal Baltic ElectronicsConference*. *ISBN* 9985-59-292-1. *Tallinn Technical University, October* 6-9, 2002, *Tallinn, Estonia*, 295 - 299

[9] **F. Brglez, H. Fujiwara.** A Neutral Netlist of 10 CombinationalBenchmark Circuits and a Target Translator in Fortran. *IEEE Int*'l *Symp. on Circuits and Systems, Vol*.3, *June* 1985, 663 - 698.