DOI 10.31114/2078-7707-2019-2-18-24

Register Duplication for Scan Designs

M.S. Ladnushkin

Federal State Institution «Scientific Research Institute for System Analysis of the Russian Academy of Sciences»

***Abstrac* — This paper proposes a method for reducing the fault testing times for digital VLSI circuits by duplicating individual functional flip-flops. The reduction in testing time is due to better signal testability and fewer mutual conflicts between faults occurring in the logic paths of VLSI circuits. Proposed here is an algorithm for selecting flip-flops to be duplicated based on a search for logic paths with the largest number of signal sources, which was used in the design of built-in test tools for a number of custom-designed units and system-on-a-chip designs. The results showed a decrease in test time by an average of 14.4 % with hardware costs not exceeding 1.2 % of the total VLSI chip area.**

***Keywords* — testing, microcircuit rejection, flip-flop duplication, test signal compression, simulation.**

I. INTRODUCTION

Inserting additional test structures for control and observation of internal nodes in VLSI circuits makes it possible to increase the test coverage and reduce the test time. Creating a test operation mode in which the VLSI's flip-flop subsystem is used as a shift register (scan path) [1] helps achieve full controllability and observability of all VLSI flip-flops and use them to test the VLSI's combinatorial portion.The amounts of test data required for circuit testing is growing, and built-in compression tools [2], [3] are not always capable of reducing the test times to acceptable levels for many reasons, including the growing lengths of logic paths, the growing numbers of mutual conflicts between faults, and the need to shift from logic cell-level abstraction down to transistor-level abstraction when generating test patterns for VLSI circuits [4], [5].

The growing lengths of combinatorial logic paths and the growing numbers of reconvergent fanouts result in poorer observability and controllability of the nodes of these paths in test mode [6], [7], [8]. To solve this problem, additional test paths – control and observation test points – are embedded in certain VLSI nodes. These points are additional control-logic flip-flops that make it possible to increase the observability and controllability of individual nodes of the VLSI's combinatorial subsystem [9].Test point insertion helps reduce the test time by up to 35 % [10], [11]. A modern test point insertion method, which reduces the number of mutual conflicts between stuck-at faults, yields a 2.2 times shorter average test time for compression-enabled scan paths by increasing the number of faults tested by each test vector [12]. However, test point insertion reduces the system's tracing capabilities, increases signal propagation delays, and increases the test logic area, which is only used in test mode and does not function in normal operation mode. Modern test point creation methods employ the VLSI's existing functional flip-flops instead of adding new ones, which helps reduce the test logic-specific hardware costs but, at the same time, unavoidably increases the delays in critical paths after the test point insertion [13], [14]. The increase in controllability and observability of speed-critical paths poses a serious problem when designing test tools.

One known method used by VLSI topology designers for reducing the conductor lengths in the circuit to provide for shorter signal propagation delays relies on creating copies of (*i. e.* duplicating) individual cell elements [15]. A flip-flop cell and its copy, in the functional mode, are in identical logic states at any time. It is believed that, in scan test mode, these flip-flops must remain in identical states, since loading different values into duplicated scan flip-flops can cause the chip to fail if its circuits contain logic cell elements that can draw leakage current [16]. If, however, the chip contains but few such cells, or none at all, then copies of duplicated flip-flop cells can be used as independent variables when generating test patterns to increase the node testability for the VLSI's combinatorial portion.

II. INCREASING THE TESTABILITY OF COMBINATORIAL CIRCUIT NODES

In scan test mode, the testability of (or the probability of detecting a fault in) a logic node of a combinatorial circuit can be calculated as a minimum number of test patterns required for fault detection divided by a maximum number of unique input stimuli [6]. Consider a combinatorial circuit design with n inputs $\vec{x} = (x_1, x_2, ..., x_n)$ and m outputs $\vec{F} = (F_1, F_2, ..., F_m)$. Let $g(\vec{x})$ be the internal signal across the circuit; then the testability of stuck-at-0 and stuck-at-1 faults in the g[th] signal can be expressed, respectively, as [6], [17]:

$$t(g/0) = S\left(g(\vec{x}) \sum_{j=1}^{m} \frac{\partial F_j}{\partial g} \right), \quad (1)$$

$$t(g/1) = S\left(\overline{g}(\vec{x}) \sum_{j=1}^{m} \frac{\partial F_j}{\partial g} \right), \quad (2)$$

where t(g/0) is the testability of the stuck-at-0 fault and t(g/1) is the testability of the stuck-at-1 fault. S(F) is the syndrome of the function F:

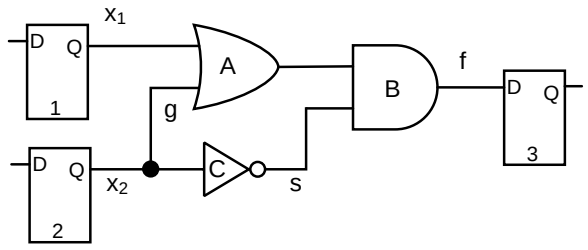
$$S(F) = \frac{K}{2^n} \qquad (3)$$

where K is the number of minterms of the function F and n is the number of its inputs. The product of functions in parentheses is the intersection of the sets of input stimuli that specify the required value for signal $g(\vec{x})$ and ensure that this value is observable at outputs $\vec{F}$. If the circuit contains reconvergent fanouts, then this intersection of sets can turn out to be rather small, which results in a decrease in the values of t(g/i), i ∈ {0;1}.

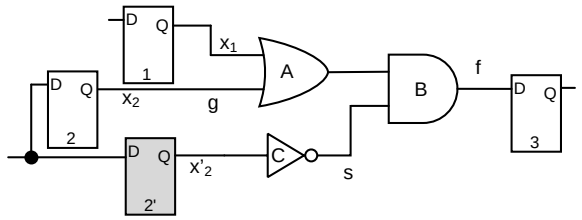
Input signals that have a fanout at the inputs can be divided into separate independent signals by duplicating the source flip-flops, which helps reduce the number of reconvergent fanouts in the circuit and, hence, increase the testability of this circuit.

Consider how the testability of the nodes of a combinatorial circuit changes with flip-flop duplication compared to the use of standard control and observation test points.

Fig. 1a shows an example of a scan chain fragment (flip-flops 1-2) and its associated combinational circuit. The combinatorial circuit has 2 independent inputs, which are designated by variables x_1 and x_2, and one output, f. This combinational circuit contains a reconvergent fanout across node x_2. Imagine that there is a stuck-at-0 fault (g/0) on signal g. The testability of signal g = x_2 according to formula (1) will be equal to $t(g/0) = S(x_2\overline{x_1 x_2}) = 0.$.



(a)



(b)

Fig. 1. An example of how testability can be increased in a combinational circuit: a) the baseline circuit design; b) the same design modified *via* flip-flop duplication

The testability of fault g/0 is 0, which means that this node cannot possibly be tested for stuck-at-0 faults using the test tools available. Higher testability figures $t(g/0)$ for this node can only be achieved *via* hardware-level modification to the circuit design.

Consider a modified circuit design in which flip-flop 2 is duplicated (see Fig. 1b). In this design, the fanout at the output of flip-flop 2 was split into 2 independent logic signals controlled independently by flip-flops 2 and 2'. Independence of control is achieved by loading different values into duplicated flip-flop cells during the testing process. To ensure that the combinatorial circuit design is functionally equivalent to the baseline one, the inputs, D, to flip-flops 2 and 2' are combined. Evaluation of fault g/0 testability for the modified circuit design with flip-flop duplication showed an increase in the value of $t(g/0)$, which amounted to:

$$t(g/0) = S(x_2\overline{x_1 x_2'}) = 1/8.$$

III. Reducing Mutual Conflicts Between Faults

Let us now consider flip-flop duplication as a means of reducing conflicts between faults. To reduce the number of test vectors required to test for VLSI faults, each test vector must be able to detect as many defects as possible, which ability is hampered by mutual conflicts between faults. Let s_0 be the source from which signals are fanned out to branches $s_1,...,s_n$, then the degrees of conflicts for any branch s_k being set, respectively, to 0 and 1, k ∈ [1; n], can be computed as follows [18]:

$$c_{s_k} = \min\{b_{s_k}; F_{s_k}\}, \qquad (4)$$

$$C_{s_k} = \min\{B_{s_k}; f_{s_k}\}, \qquad (5)$$

where c_{s_k} and C_{s_k} are the numbers of conflicts where node s_k is set, respectively, to logic values 0 and 1, b_{s_k} and B_{s_k} are the numbers of logic states that branch s_k must have when being set, respectively, to 0 and 1 to ensure that faults occurring on all the remaining fanout branches with source s_0 are observable, and f_{s_k} and F_{s_k} are the numbers of logic states that branch s_k must have when being set, respectively, to 0 and 1 to ensure that all the faults that source signal s_0 are observable, and

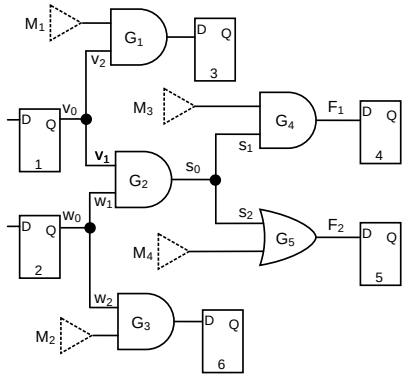
$$F_{s_k} = F_{s_0} + \sum_{i=1}^{n} B_{s_i}, i \neq k, \qquad (6)$$

$$f_{s_k} = f_{s_0} + \sum_{i=1}^{n} b_{s_i}, i \neq k, \qquad (7)$$

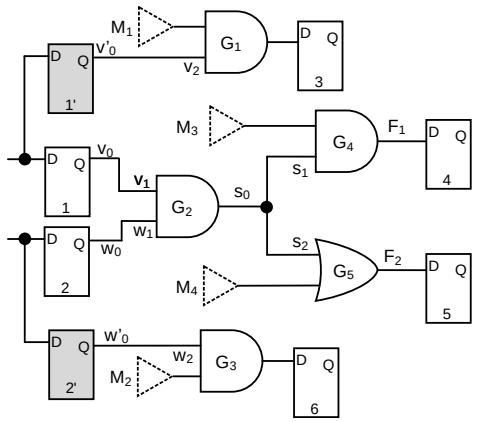
For the fanout of signal s at the output of the flip-flop, the value of $F_{s_0} = f_{s_0} = 0$, *i. e.* $F_{s_k} = \sum_{i=1}^{n} B_{s_i}, i \neq k,$, $f_{s_k} = \sum_{i=1}^{n} b_{s_i}, i \neq k,$. In this case, if the number of fanout

branches is reduced to n = 1, then $F_{s_k} = f_{s_k} = 0$. Hence, in this fanout, $C_{s_k} = c_{s_k} = 0$, *i. e.* there will be no conflicts between faults in it. Moreover, for any signal x in the paths from s_0 to the target receivers of the signal (flip-flops or output ports), the values of f_x and F_x for all fanouts will be reduced, respectively, by $\sum_{i=1}^{n} b_{s_i}$ and $\sum_{i=1}^{n} B_{s_i}$. Hence, using formulas (4) and (5), we find that the number of conflicts c_x and C_x for any signal x can be reduced, respectively, by $\sum_{i=1}^{n} B_{s_i}$ and $\sum_{i=1}^{n} b_{s_i}$.

Duplication of a flip-flop at the fanout source reduces the number of fanout branches to 1, meaning that the number of mutual conflicts on all logic paths sourced from these flip-flops will decrease.



(a)



(b)

Fig. 2. An example of how conflicts between faults can be reduced in a combinational circuit: a) the baseline circuit design; b) the same design modified *via* flip-flop duplication

Fig. 2a shows an example of a combinatorial path design showing a mutual conflict between faults in signal s. Individual components of the combinatorial path are designated with triangles, in which the number of faults is M_i. Using formulas (4)–(7), let us compute the conflict metrics for nodes s_1 and s_2; we will get:

$$c_{s_1} = \min\{b_{s_1}; F_{s_1}\} = \min\{0; \min\{M_2; M_1\}\} = 0,$$
$$C_{s_1} = \min\{B_{s_1}; f_{s_1}\} = \min\{M_3; M_4\},$$
$$c_{s_2} = \min\{b_{s_2}; F_{s_2}\} = \min\{M_4; \min\{M_1; M_2\} + M_3\},$$
$$C_{s_2} = \min\{B_{s_2}; f_{s_2}\} = \min\{0; 0\} = 0.$$

According to the assessment, a conflict occurs when node s_2 is set to 0 and when node s_1 is set to 1. To resolve this conflict, the authors of [18] propose that an OR-type control test point be deployed on signal branch s_2, as a result of which the values of c_{s_2} and C_{s_1} in this design go down to 0. Consider an equivalent modified circuit design in which flip-flop cells 1 and 2 are duplicated (see Fig. 2b). We get the following set of expressions characterizing signals s_1 and s_2 in the circuit:

$$c_{s_1} = \min\{0; 0\} = 0,$$
$$C_{s_1} = \min\{M_3; M_4\},$$
$$c_{s_2} = \min\{M_4; M_3\},$$
$$C_{s_2} = \min\{0; 0\} = 0.$$

Comparing the results of fault conflict assessments for the modified and baseline circuit designs, we can conclude that the degree of conflict where node s_2 is set to 0 is no longer dependent on the variables M_1 and M_2, which causes c_{s_2} to decrease to the value of $\min\{M_2; M_1\}$. Since the proposed flip-flop duplication affects any logic path leading from this flip-flop to the target receivers of the signal by reducing the degrees of mutual conflicts occurring on fanout s at the output of this flip-flop, we can infer that this method will yield the greatest effect when duplicating flip-flops with the greatest number of combinatorial cell elements in their output paths.

IV. An Algorithm for Searching for Source Flip-Flops to be Duplicated

There are known search methods that identify all reconvergent fanouts based on a circuit logic description [7], [8]. The option of duplicating all the flip-flops that have a fanout at the output results in an excessive increase in area. It is, thus, advisable to select for duplication only those flip-flops that produce the maximum effect in terms of increasing testability and reducing the number of conflicts between faults. This paper analyzes an algorithm based on a search for logic trees with the greatest number of sources.

Long reconvergent paths usually represent logic functions of a large number of variables. An analysis of logical paths of a number of VLSI circuits based on 65-nm through 250-nm design rules was carried out by counting the number of target signal sources for each flip-flop in the circuit (Table 1). The analysis showed that a significant part (more than 70 %) of the flip-flops had one target signal source.

Table 1

Parameters of VLSI circuits under analysis

Characteristic	VLSI-1	VLSI-2	VLSI-3	VLSI-4
Technology	3D 65 nm CMOS		250 nm SOI	
Number of flip-flops, '000	76.1	103.0	344.8	513.8
Number of combinatorial cell elements, '000	191.2	528.3	878.0	2084.7
Number of I/O ports	140	353	311	639
Number of RAM units and custom-designed units	24	110	371	1055

Fig. 3 shows the distribution of flip-flops by the number of signal sources with an increment of 100 units. VM4 is one VLSI system in which we observed individual flip-flop cells with big numbers of different signal sources – up to 7400. Logic functions with so many sources are difficult to test, so it makes sense to duplicate them first.

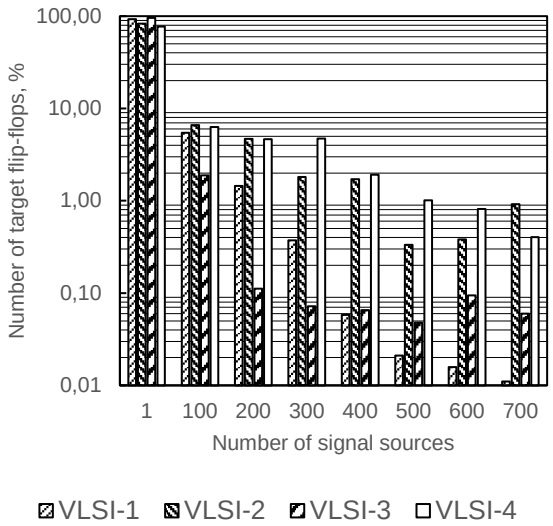


Fig. 3. Distribution of flip-flops by the number of signal sources

A search algorithm that identifies flip-flop cells at the source of critical paths, the number of such cells being bounded by threshold d_{max}, comprises the following sequence of steps:

1) setting the number of candidate flip-flops for duplication, d_{max};

2) analyzing the circuit and obtaining a set of low-testability signals, L;

3) obtaining a set of target sources, S_i, for each signal L_i;

4) excluding from each set S_i the I/O ports and flip-flops that have no fanout at the output;

5) sorting the sets S_i by the number of elements in descending order;

6) successively adding elements from each set S_i to set D until their number exceeds d_{max}.

This algorithm outputs a set of flip-flops, D, which are to be duplicated.

V. VLSI DESIGN FLOW WITH FLIP-FLOP DUPLICATION

Flip-flop duplication is carried out after a logic cell-level model (netlist) of the VLSI has been obtained. The first step is to search for flip-flop cells at the source of critical paths, their number being bounded by threshold d_{max}. A flip-flop whose output logic path includes a cell element that can draw leakage current is to be excluded from the list. The resulting list of flip-flops, D, is duplicated, after which the circuit design is logically optimized, since flip-flop duplication may lead to changes in signal propagation delays. Such optimized design then serves as a basis for creating a compression-enabled scan path.

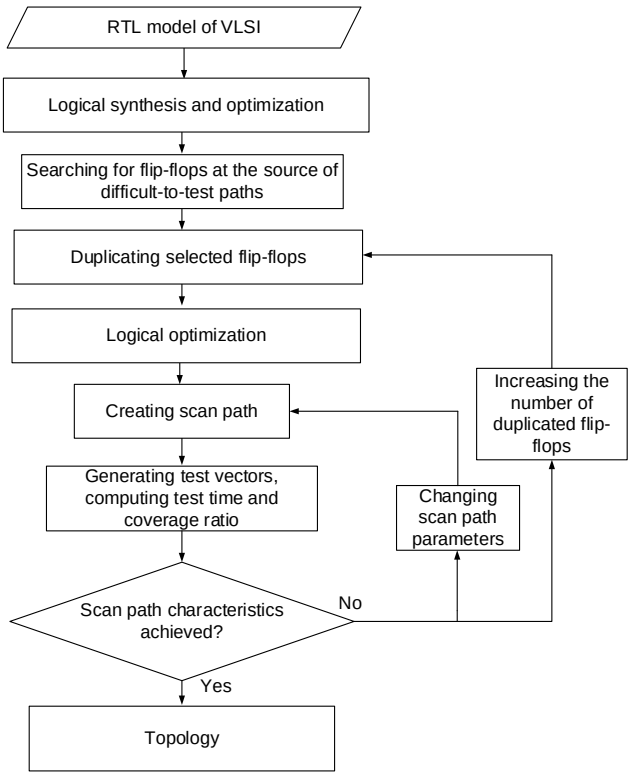


Fig. 4. Design flow

Generation and simulation of test vectors for the resulting circuit design allows the designer to estimate the test time and the test coverage ratio. If the required parameters have been obtained, or if the chip occupancy limit has been reached, the next step is to design the device topology. If the required coverage ratio has not been achieved, or if there is at the same time a need to reduce

the test time and available space for additional flip-flops, the designer can either change the scan path parameters [19] or duplicate more flip-flop cells in critical paths. Fig. 4 illustrates a design flow based on the methodology described above.

VI. Results of Simulation of VLSI Scan Paths

The proposed methodology was used in designing the test tools for seven IP units: a 64-bit microprocessor core (cpu), a 2D graphics controller (2d), a 10/100/1000 Mbps Ethernet controller (eth), a PCI-E 2.0 8x controller (pcie_8x), a 3.125 Gbps RapidIO 4X serial controller (rio), a 3 Gbps SATA controller (sata), and a USB 2.0 controller (usb). The same methodology helped design test tools for three systems-on-a-chip (SoC): a 64-bit microprocessor implementing the KOMDIV architecture with built-in serial RapidIO links (proc), a six-lane switch for high-speed 10 Gbps serial RapidIO links (smpo), and a six-lane PCI Express 2.0 switch (basis). The VLSI products listed above differ in the number of flip-flops, the amount of RAM, the number of built-in custom-designed units, the number of clock domains, the clock rates, and the combinatorial and flip-flop logic areas. All the VLSI circuits were synthesized based on standard cell elements from TSMC's library using 65-nm design rules as well as with built-in interface transceivers, custom-designed units, and built-in RAM units. Table 2 shows the design parameters for all these products.

All the VLSI products listed above were examined for difficult-to-test nodes, and as a result, sets of low-testability nodes, L, were obtained for each product.

Table 2

Parameters of VLSI circuits under modification

Project	Number of flip-flops, '000	Area, mm^2				Number of I/O ports
		of flip-flop cells	of combinatorial logic cells	of RAM and custom-designed units	Total	
cpu	96.2	1.04	2.98	6.58	10.59	840
eth	31.2	0.33	0.23	2.03	2.59	602
pcie_8x	98.3	1.05	0.90	1.23	3.18	3241
rio	65.0	0.71	0.91	0.35	1.96	1638
sata	14.0	0.15	0.21	0.11	0.47	2465
usb	25.8	0.28	0.32	3.51	4.11	617
2d	6.0	0.06	0.07	0.65	0.79	1857
proc	513.8	6.21	8.55	43.18	57.95	639
smpo	344.8	3.79	4.06	51.78	59.62	311
basis	187.6	2.20	2.77	22.31	27.28	226

The threshold, d_{max}, on the number of additionally inserted elements to be used by the test tools was set at 2 % of the total number of VLSI flip-flops, which is an acceptable value for test logic area growth [20]. After that, according to the algorithm described above, and subject to the specified threshold d_{max}, sets of candidate flip-flop cells, D, were obtained for each circuit design. All the elements in each such set were subsequently duplicated.

Then, compression-enabled scan testing tools (scan paths) were created in each VLSI circuit so obtained [21]. The VLSI test system design flow described above was implemented in Synopsys' DFT Compiler. The parameters of the resulting circuits, such as the number of internal scan chains and their length, the width of the external test data bus, as well as the number of flip-flops inserted in the course of flip-flop duplication are shown in Table 3. The increase in VLSI logic area due to introduction of additional logic cells was calculated after the respective circuits had been optimized in terms of area and speed. On average, the said increase in area amounted to 0.38 %.

The scan paths obtained as shown above were then simulated. The test patterns were generated using Synopsis' Tetramax ATPG (Automatic Test Pattern Generation) tool.

As a result, the relevant scan path parameters, *i. e.* test coverage and number of test vectors, were determined. Based on the test clock frequency of 10 MHz, the length of the scan chains, and the number of vectors, test durations were computed for each VLSI circuit (see Table 4). Scan path designs with duplicated functional flip-flops required 4.8–39.1 % fewer test patterns to achieve a given test coverage compared to the baseline designs without duplication, which, across all the projects studied, made up an average of minus 16.1 %. As additional flip-flops were inserted, the lengths of scan chains in the circuits increased, which resulted in longer propagation times for each test vector. However, since there were now fewer vectors, the resulting test times reduced by anywhere between 3.3 % (rio IP unit) and 37.1 % (proc VLSI), or, on average, by 14.4 %.

Table 3

Parameters of VLSI scan paths

Project	Number of scan chains	Width of scan path data bus, input/output	Scan chain lengths for scan paths, flip-flops		Number of flip-flops inserted	Growth in total logic area, %
			without dupl.	with dupl.		
cpu	164	6/6	586	598	1828	0.16
eth	51	5/5	612	622	505	0.21
pcie_8x	164	6/6	601	609	1411	1.18
rio	111	6/6	587	598	1276	0.97
sata	23	4/4	610	622	254	0.09
usb	43	5/5	602	618	684	0.14
2d	10	3/3	595	615	177	0.47
proc	2000	10/10	346	351	7207	0.36
smpo	1000	10/10	252	260	5018	0.09
basis	1000	9/9	189	193	3725	0.14

Table 4

Results of simulation of VLSI scan paths

Project	Test coverage, %	Number of test vectors		Scan path test time, s		Test time reduction, %
		without dupl.	with dupl.	without dupl.	with dupl.	
cpu	93.6	4840	4418	0.284	0.264	7.0
eth	94.1	1672	1565	0.102	0.097	4.9
pcie_8x	95.3	5365	5077	0.322	0.309	4.0
rio	96.6	2270	2138	0.133	0.128	3.8
sata	86.2	816	593	0.050	0.037	26.0
usb	91.4	504	480	0.030	0.029	3.3
2d	79.6	421	261	0.025	0.016	36.0
proc	91.6	55236	33656	1.391	0.875	37.1
smpo	90.0	19374	14993	0.670	0.526	21.5
basis	91.5	10401	10147	0.196	0.195	0.5

VII. CONCLUSION

A method was proposed for increasing the testability and reducing mutual conflicts between faults in digital VLSI circuits, which consists in duplicating flip-flops in difficult-to-test paths. It was shown that duplication of flip-flops in paths with reconvergent fanouts makes it possible to increase the testability of nodes of these paths, and duplication of flip-flops at the source of logic paths makes it possible to reduce the number of mutual conflicts between faults at all the nodes of these paths. The proposed methodology involving the duplication of individual flip-flops was implemented in 3 VLSI projects and 7 IP unit projects when designing compression-enabled scan testing tools for them. The results showed a decrease in test time by an average of 14.4 % with hardware costs not exceeding 1.2 % of the total VLSI chip area.

REFERENCES

[1] Abramovici M., Breuer M. A., and Friedman A. D. Digital Systems Testing and Testable Design, Computer Science Press, 1990. 364–366 P.

[2] Touba N. A. Survey of test vector compression techniques // IEEE Design & Test of Computers. – July–August, 2006. – Vol. 23. No. 4. – P. 294–303.

[3] Kapur R., Historical perspective on scan compression // IEEE Design & Test of Computers. – March–April, 2008. – Vol. 25. No. 2. – P. 114–120.

[4] Hapke F., Redemund W., Glowatz A., Rajski J., Reese M., Hustava M., Keim M., Schloeffel J., Fast A. Cell-Aware Test // IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. – SeP. 2014. – Vol. 33. – No. 9. – P. 1396–1409.

[5] Acero C., Feltham D., Hapke F., Moghaddam E., Mukherjee N., Neerkundar V., Patyra M., Rajski J., Tyszer J., Zawada J. Embedded deterministic test points for compact cell-aware tests // Test Conf ITC 2015 IEEE Int. – Oct. 2015. – P. 1–8.

[6] Savir J. Good Controllability and Observability Do Not Guarantee Good Testability // IEEE Transactions on Computers. – 1983. – V. 32. No. 12. – P. 1198–1200.

[7] Robert M. W., Lala P. K. Algorithm to Detect Reconvergent Fanout in Logic Circuits // IEEE Proceedings. – 1987. – Vol. 134. No. 2. – P. 105–111.

[8] Xu S., Edirisuriya E. A new way of detecting reconvergent fanout branch pairs in logic circuits // Asian Test Symposium (ATS'04). – 2004. – P. 354–357.

[9] Pomeranz I., Reddy S.M. Test-point insertion to enhance test compaction for scan designs // Proc. ICDSN. – 2000. – P. 375–381.
[10] Geuzebroek M. J., Linden J. T., Goor A. J. Test point insertion that facilitates ATPG in reducing test time and data volume // Proc. ITC. – 2002. – P. 138–147.
[11] Kumar A., Rajski J., Reddy S. M., Rinderknecht T. On the Generation of Compact Deterministic Test Set for BIST Ready Designs // Asian Test Symposium. – 2013. – P. 201–206.
[12] Acero C., Feltham D., Liu Y., Moghaddam E., Mukherjee N., Patyra M., Rajski J., Reddy S. M., Tyszer J., Zawada J. Embedded deterministic test points // IEEE Transactions on Very Large Scale Integration (VLSI) Systems. – 2017. – P. 1–13.
[13] Ren H., Kusko M., Kravets V., Yaari R. Low cost test point insertion without using extra registers for high performance design // Proc. ITC. – 2009. – P. 1–8.
[14] Yang J., Touba N. A., Nadeau-Dostie B. Test Point Insertion with Control Points Driven by Existing Functional Flip-Flops // IEEE Transactions on Computers. 2012. V. 61. P. 1473–1483.
[15] Srivastava A., Kastner R., Sarrafzadeh M. Timing driven gate duplication: Complexity issues and algorithms // Proc. ICCAD. – 2000. – P. 447–450.
[16] Goessel M., Singh A., Sogomonyan E. Scan-path with directly duplicated and inverted duplicated registers // Proceedings 20th IEEE VLSI Test Symposium. 2002. P. 47–52.
[17] Savir J. Syndrome-testable design of combinational circuits // IEEE Transactions on Computers, 1980. V.29. – P. 442–451.
[18] Liu Y., Moghaddam E., Mukherjee N., Reddy S. M., Rajski J., Tyszer J. Minimal area test points for deterministic patterns // Proc. ITC. – NoV. 2016. – P. 1–7
[19] Ladnushkin M. S. Snizheniye apparaturnykh zatrat i uvelicheniye koeffitsienta kompressii sredstv testirovaniya konstantnykh neispravnostey KMOP tsifrovykh SBIS (Reducing area and increasing compression ratio of scan compression system for digital VLSI using stuck-at fault model) // VII Vserossiyskaya nauchno-tekhnicheskaya konferentsiya «Problemy razrabotki perspektivnykh mikro- i nanoelektronnykh sistem. – 2016». Sbornik trudov / pod obshch. red. akademika RAN A. L. Stempkovskogo. M.: IPPM RAN, 2016. Ch. 2, S. 68–75.
[20] Acero C., Feltham D., Patyra M., *et al*. On new test points for compact cell-aware tests // *IEEE Des. Test*. – Dec. 2016. – Vol. 33 No. 6. – P. 7–14.
[21] Wohl P. , Waicukauski J. A., Ramnath S. Fully X-Tolerant Combinational Scan Compression // International Test Conference. – 2007. – P. 1–10.