Computing and Informatics, Vol. 25, 2006, 1001–1024, V 2006-Sep-8

TESTABILITY ANALYSIS AND IMPROVEMENTS OF REGISTER-TRANSFER LEVEL DIGITAL CIRCUITS

Josef Strnadel

Faculty of Information Technology
Brno University of Technology
Božetěchova 2
612 66 Brno, Czech Republic
e-mail: strnadel@fit.vutbr.cz

Manuscript received 20 June 2005; revised 23 May 2006
Communicated by Elena Gramatová

Abstract. The paper presents novel testability analysis method applicable to register-transfer level digital circuits. It is shown if each module stored in a design library is equipped both with information related to design and information related to testing, then more accurate testability results can be acieved. A mathematical model based on virtual port conception is utilized to describe the information and proposed testability analysis method. In order to be effective, the method is based on the idea of searching two special digraphs developed for the purpose. Experimental results gained by the method are presented and compared with results of existing methods.

Keywords: Digital circuit, testing, register-transfer level, data-path, testability analysis, design for testability, scan technique

1 INTRODUCTION

Problems related to testing digital systems belong to most important but also most expensive and difficult parts of a design cycle. Classical test approach deals with selecting test vectors for detection/localization of a physical defect in a manufactured electronic system. This paper is related to testing of a *register-transfer level* (RTL) digital system. Usually, RTL circuit is used to be an output of the high-level

synthesis and input of low-level synthesis. Those are places where concepts and methods presented in this paper are supposed to be applied.

During solving problems related to circuit testing, many computationally sophisticated algorithms for *generating tests* have been developed – especially for lower levels of design description [1]. But, their time complexity practically limited them to smaller circuit designs. This fact caused that *hierarchical test generation* methods appeared, e.g. [18, 19]. They are based on modular decomposition of a circuit structure, generation of (local) tests for circuit modules and their translation to desired (global) test. Also, this problem can be seen as the problem of construction of a test data for each module and the problem of transmitting them to/from the module through the structure of the circuit. Using the modular approach, serious problem of selecting proper module granularity appears. As an alternative to structural-based test generation methods, other methods appeared – e.g., *pseudo-random* or *functional* test generation methods. But, they generate tests of lower quality than structural methods in general.

Imperfections caused by test generation methods can be withdrawn, e.g., using proper combination of several test generation methods or modifying original circuit structure by proper combination of *design/synthesis* (DFT/SFT) for testability technique(s). Modification of a circuit structure in order to increase its testability became an integral part of a digital circuit design cycle. Of course, there are general consequences of this approach: area&pin overhead, variation of dynamic&power consumption parameters, etc. To be able both to satisfy design constraints posed on the resulting circuit and to find a highly-testable modification of the original circuit structure, the modification process is required to be informed about the quality (in view of testability parameters and level of satisfaction of design constraints) of a proposed modification. Such an information can be provided by a *testability analysis* (TA) method. Using results of the method, acceptable trade-off between testability parameters and design constraints can be found. In the paper, principle of a new TA method is presented together with experimental results gained so far using the method.

The paper is organized as follows. First, state of the art and our research motivation in TA is presented in brief. After that, principles of proposed virtual port conception and TA method are summarized. The end of the paper deals with experimental results and conclusions connected to our research activities.

2 STATE OF THE ART IN TESTABILITY ANALYSIS

At present, exact (standardized) definition of testability does not exist. Generally, *testability* is understood as a characteristic involving various costs related to digital circuit testing. It is talked about fault testability, in-circuit node testability (a fault or a node is classified as testable if it is possible to detect the fault in the node), etc. Usually, the classification is done by a TA method using special *testability measures*. However, differences in existing testability definitions lead to various

understanding of some testability-related concepts, components and consequently of measures used for testability evaluation. Most of those drawbacks should disappear after involving definitions in IEEE P1522 standard [27]. Even though existing TA methods differ in their goals, it can be said they share the following attribute: their effort is to provide sufficiently precise information about circuit testability (usually evaluated by means of controllability and observability factors). Circuit structure modification (hopefully) leading to better circuit testability is affected by the information significantly.

Existing TA methods can be classified as gate-level methods (e.g., SCOAP [8], TMEAS [9]), RTL methods (e.g., TMEAS [9], CAMELOT [20], COPS [2], ITA [25], ASCOPA [7]), high-level methods (e.g., [6], SATAN [21], FACTOR [34]). RTL methods can be divided into two following groups:

- Methods based on a *probability model* of a diagnostic data flow through modules within the circuit structure. For each module, several characteristics exist determining probability of a certain input-output data transfer.
- Methods based on *modeling of diagnostic properties* of in-circuit modules. They allow diagnostic data flow be modeled more precisely and in more detail, but they are more complex in general.

As a TA method is inspired by a certain test generation principle [33], results of the former methods are practically applicable when a pseudo-random test generator is to be used. Results of the others are more general, and thus applicable when deterministic test generator is to be used or combined with pseudo-random or other test generation method.

3 MOTIVATION OF OUR RESEARCH

Because our research is related to the methods belonging to the second group, let some of their drawbacks be presented now:

- testability interpretations and definitions differs – this leads to disunion, incompatibility and difficult comparability of existing algorithms, because they solve different tasks and try to reach different goals,
- some of them are very closely related to particular DFT technique so they are not general-purpose,
- they are often based on a simple model of diagnostic properties, which results in inaccurate testability information and, consequently, in redundant set of difficult-to-test parts identified in the circuit, in redundant involving of diagnostics resources into the circuit structure etc.

As an example of a very often used and simple model of diagnostic properties, let us present basics of so called *I path* conception [1] in the following text. The conception supposes n-bit diagnostic data transfer is possible between x and y iff

both x and y are n-bit ports and each n-bit data can be transferred unchanged in the direction from x to y (i.e., one-to-one identity mapping exists between x-data and y-data). This concept is clear, but its application can lead to inexact information about circuit testability. In Figure 1, example of modules that are able to use the I-path concept to model their diagnostic properties is presented.

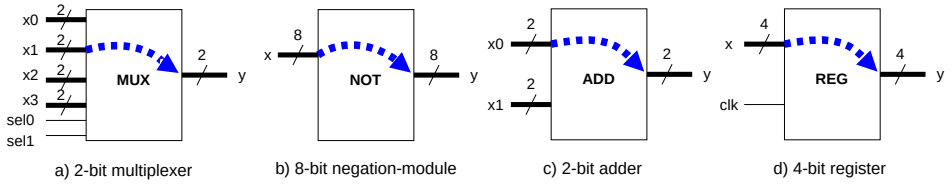

Fig. 1. Illustration to I-path concept

Example of a module M that cannot be modeled by means of I path conception is presented in Figure 2. In the example, two ports with the same bit-width do not exist, so it is not possible to describe diagnostic data transfer using I-path concept. However, as can be seen in tables d) to h), due to existence of several partial mappings it is possible to transfer "partial" diagnostic data through the structure of M. As a result, it can be seen the I-path concept is too strict in definition of a *data path* (DP) suitable for diagnostic data transfer. Its "one-to-one identity mapping" requirement is too strict and leads to unneeded restriction of set of DPs suitable for transferring diagnostic data.

In contrast to the strict requirement, it was shown [18, 19, 31] it is advantageous to analyze DP circuit separately for transferring test vectors (responses) between ports x and y belonging to an interface of the same in-circuit module:

- test vectors can be transferred through a module structure iff a surjection exists between x-data y-data,
- responses can be transferred through a module structure iff an injection exists between x-data y-data.

By means of the above-mentioned divide & conquer principle, the strict "I path requirement" can be softened with no impact to accuracy of TA results.

4 OUR RESEARCH GOALS

Research related to this paper is specialized in design of a new RTL testability analysis method and was motivated especially by the above-outlined drawbacks of existing methods. Usually, RTL digital circuit is supposed to consist of the following two parts: a description of its DP and a circuit controller used to control a data flow in the DP. Our approach deals only with problems related to testability of a DP.

Surely – precision of a TA method highly depends on how precisely diagnostic properties of in-circuit modules are modeled. Especially, it is important how

module ability to transport test vectors or responses through its structure (so-called *transparency* of the module) is modeled. For practical reasons, it is advantageous to gather both transparency-related and design-related information (HDL description, technological information etc., required by design tools) and store it in a library.

Regarding previous text, our research goals can be summarized as follows:

- to develop a more precise transparency model for RTL modules,
- to design and describe novel TA method using instruments of the model,
- to apply our TA method in selected areas of digital circuit testing,
- to summarize results and to compare them with other approaches.

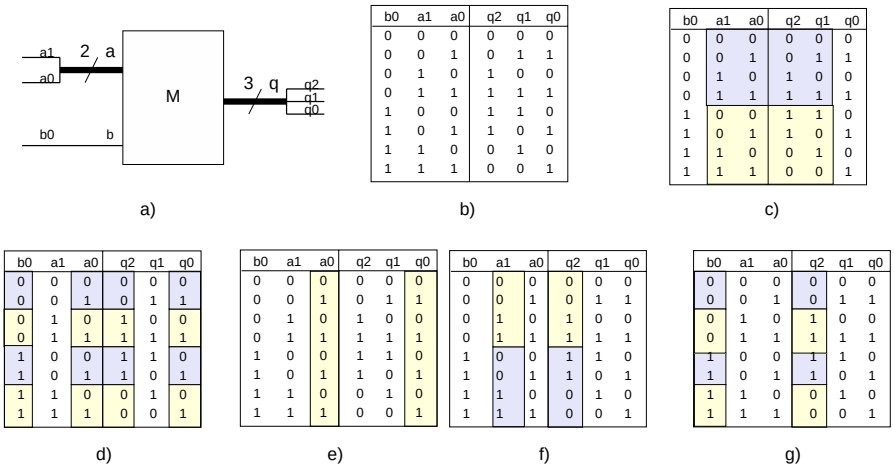

Fig. 2. Illustration to transparency modes based on virtual port concept

5 BASIC CONCEPTS OF PROPOSED SOLUTION

5.1 Virtual Port Concept Basics

In our approach, DPs suitable for transferring diagnostic data are considered between so called *virtual ports*. Virtual port (i.e. n-tuple of 1-bit inputs/outputs) abstracts from the module interface defined by a module designer and can be understood as a generalized port belonging to the interface of the module. The main reason of using virtual ports is to ease a mathematical description of the conception proposed in our work. Let us give some examples of what virtual port means. In Figure 2, module M is presented. Its designer-created interface consists of input ports $a = (a_1, a_0)$, $b = (b_0)$ and output port $q = (q_2, q_1, q_0)$. The set of all *virtual input ports* of M is a subset of $\{a1, a0, b0\}^+$ and set of all *virtual output ports* is

a subset of $\{q2, q1, q0\}^+$. Using virtual ports, it is easy to describe existence of partial (surjective/injective) mappings between data at module inputs and module outputs.

As an example, it can be seen in Figure 2 c) there is an unconditioned 3-bit one-to-one mapping between b_0, a_1, a_0 and q_2, q_1, q_0 data. In Figure 2 d), 2-bit one-to-one mapping (determined/conditioned by b_0-value) between a_1, a_0 and q_2, q_1 is highlighted. Similarly, in Figure 2 e) there is a 2-bit mapping between b_0, a_0 and q_2, q_0 conditioned by a_1-value. In Figure 2 f)–h), 1-bit mappings are highlighted.

Information about mappings can be utilized in the following way. Let us suppose module M is to be used as a module through which test data will be propagated to/from other modules within DP circuit. Suppose that all 3 input bits, i.e. a_1, a_0 and b_0 of module M are fully controllable. Then, there is no problem to propagate any 3-bit data through M's structure to outputs of M. But, what are transfer capabilities of M in case some of its input bits are not fully controllable? Then only 2 bit (1 bit) test data can be transferred through structure of M. The remaining bits (i.e., not influenced by particular mapping) make portion of circuit DP difficult to control and/or observe. Consequently, they become candidates for testability enhancement, e.g. by insertion of DFT resources at proper places in the circuit structure.

5.2 Transparency Digraphs

Having information about how interfaces of various modules are interconnected and about transformation (mapping) of diagnostic data between each pair of virtual ports, it is possible to construct two special digraphs for the circuit: *test pattern data flow digraph* (G_S) and *test response data flow digraph* (G_I). Vertices of G_S (G_I) are virtual ports. An oriented edge exists between two vertices iff surjection (injection) exists between the start-vertex and end-vertex data, i.e. iff it is possible to transfer test vectors (responses) from start-vertex to end-vertex. Pairs of vertices between which the transfer is possible, together with information about required flow-condition, are put in special relations forming a basis for constructing edges of G_S (G_I).

In Figure 4, example (associated with circuit depicted in Figure 3) of a portion of G_S (Figure4 a)) for adjusting test data from primary input *tst_in* to input b of module *MOD1* (*MOD1.b*) is presented together with a portion of G_I (Figure 4 b)) for observing test data from output y of module *MOD3* (*MOD3.y*) at primary output *out*. In Figure 4, the following graphical notation is used. In full-line circles ports of in-circuit modules are depicted, in dash-line circles primary ports are depicted and in a double-line circle, the port to which the digraph-portion belongs to is depicted. Circles connected by a full-line represent test path for the double-lined port and circles connected by dash-line represent paths to be controlled in order to ensure the data flow through full-line path.

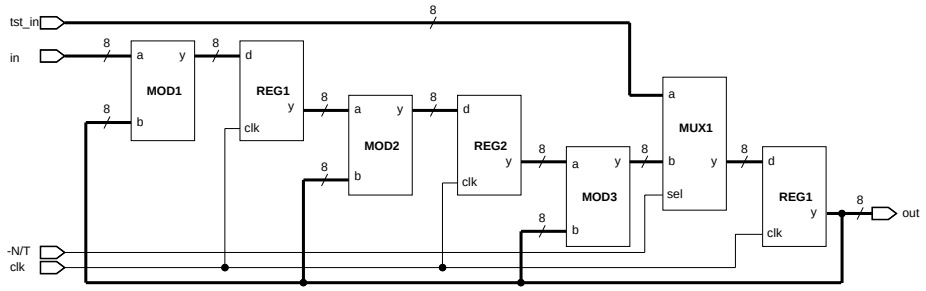

Fig. 3. Example of an RTL circuit

5.3 Proposed Testability Analysis Algorithm

Proposed TA algorithm is constructed as a graph-searching algorithm over G_S and G_I. During the search process, accessibility of virtual ports from circuit primary inputs is analyzed in G_S (this step corresponds to controllability analysis) first, and then accessibility of virtual ports at circuit primary outputs is analyzed in G_I (this step corresponds to observability analysis).

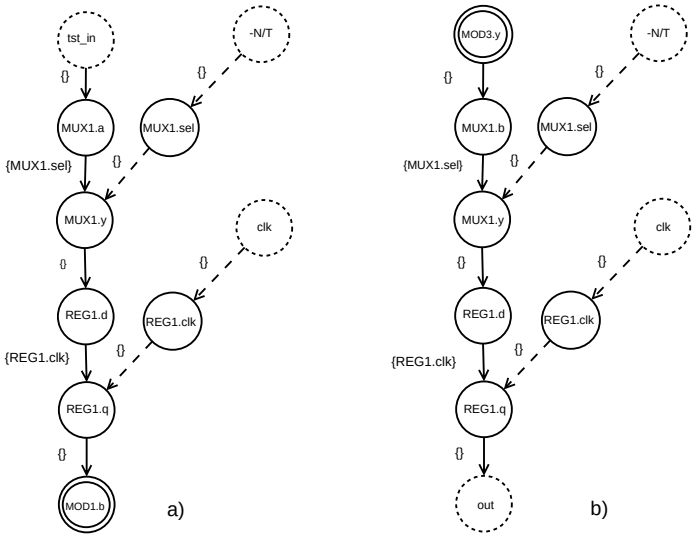

Fig. 4. Illustration to G_S and G_I

5.3.1 Principle

Principle of a basic *graph-search* (vertex marking) *algorithm* is as follows. During searching process, marks are being assigned to vertices within graph in such a way

that if a mark is assigned to a vertex x a path from initial vertex r to vertex x exists in the graph. The principle of this basic algorithm is as follows.

1. [*Initialization*] Assign an initial mark to r; other vertices are markless.
2. [*Edge selection*] Select an edge e, whose start-vertex is marked and end-vertex is markless; continue with step 3. If there is no such an edge, end the algorithm.
3. [*Marking*] Assign a mark to the end-vertex of e, go to step 2.

Proposed TA algorithm performs both operations during graph-searching process: marking of vertices and evaluating easiness of accessing marked vertices by means of primary inputs and outputs of the circuit. While a mark from an edge start-vertex (where conductive connection is represented by the edge) can be transported without any condition, it does not hold for the second edge type of an edge (modelling a data-flow through module structure). For as much as the second edge type depends on (is conditioned by) controllability of certain control inputs of the module, the transfer of the mark through this edge is conditioned by the ability to control those inputs. Thus, each such an input is required to be marked before the mark is transferred through the edge. As an example, sets of such input ports are present by edges in Figure 4.

The structure of the mark consists of several attributes, each of them representing certain diagnostic property of the place the mark is assigned to. Each measured property is evaluated by a real number from $< 0; 1 >$ interval, where 0 indicates absence of the property and 1 indicates occurrence of the property in its best form. Using the evaluation, it is possible to state which vertex is "better" than another one by simple numerical comparison of the vertices. In the following sections, '.' will be utilized to access attributes of a mark meanwhile a multiplication operator will be represented by '$\times$'.

Principle of proposed G_S searching method can be expressed by the following algorithm:

1. [*Initialization*] Assign marks to primary input vertices of G_S; leave other nodes markless.
2. [*Edge selection*] Select such edges from G_S, whose start-vertex together with "edge condition inputs" are marked and whose end-vertex is either markless nor it has worse mark than the mark to be transported; continue with step 3. If there is no such edge, finish the algorithm.
3. [*Marking*] Assign marks to end-vertices of selected edges and go to step 2.

5.3.2 Detail Description

In the following text, detailed description of controllability-analysis portion of our TA algorithm is presented. Its step-by-step functionality is shown on a simple feedback-loop circuit (NL1 – see Figure 5) and its modified version NL3 (see Figure 6). For the purposes of the following text, let each marked (to-be-marked)

vertex from G_S be denoted as n (m) where $(n, m) \in G_S$. Before detailed principle of the algorithm will be presented, it is necessary to explain the following:

Structure of a mark. Mark (n.mark) of virtual port n consists of the following components:

- $\#p$, the number of 1-bit signals needed to control n,
- $\#c$, the number of clk-pulses needed to control n,
- $\prod_{c-}$, the product of controllabilities of all ports needed to control a data flow ending in the predecessor of n in the circuit data path,
- $\prod_{c}$, the product of controllabilities of all ports needed to control data transfer from predecessor of n to n in the circuit data path,
- *con*, the controllability value of n.

Meaning of other symbols utilized in the algorithm:

- $\sum_p$, sum of all non-primary 1-bit signals that can be used for control,
- $\sum_c$, sum of worst-case sequential lengths of in-circuit elements.
- $(n, m).controls$, set of 1-bit signals needed to control data flow from n to m,
- $(n, m).\#controls$, number of elements in $(n, m).controls$ set,
- $(n, m).\#clks$, number of clock elements in $(n, m).controls$ set.

Below, let us present the detail principle of the algorithm. The algorithm (see the next page) consists of three phases: *phase 1*, *phase 2* and *phase 3*. In phase 1, initial marks are assigned to primary input (PI) virtual ports, which means:

- no additional ports or clk-pulses are needed to control PIs,
- controllability of each PI is set to the best possible.

Phases 2 and 3 are repeated until no port is marked by new (i.e., better controllable) mark. In phase 2, marks from last newly-marked virtual ports are simply transported (copied) to their data-path successors physically connected with them. If at least one mark was transported, phase 3 starts. In phase 3, marks that were newly assigned to module inputs in phase 2 are transported to outputs of those modules if to-be-placed mark ($m'.mark$) is better than existing-one mark (m.mark). For the purpose, temporary mark (m'.mark) is used. Quality of a mark is evaluated by the following τ function returning controllability value of the mark:

$$\tau(n) = (1 - \frac{n.mark.\sharp p}{1 + \sum_p}) \times (1 - \frac{n.mark.\sharp c}{1 + \sum_c}) \times (n.mark.\prod_{c-}) \times (n.mark.\prod_{c}). \quad (1)$$

After marking G_S vertices (controllability analysis) is completed, marking G_I vertices (observability analysis) can start. G_I marking is similar to G_S marking, the only difference is that marks are being transported from initial POs in the direction to PIs contrary to the orientation of edges within G_I. G_I marking process can start after G_S marking process is completed, because it needs an information about controllability marks.

Controllability Analysis Algorithm
Begin

```
[Initialization]
    marked_ports_C=∅;
    newly_marked_ports_C=∅;
    for each port n do n.mark.con=0.0;
[Phase 1: PI-marking]
    for each PI-port n do
        n.mark.#p=n.mark.#c=0;
        n.mark.∏_{c−}=n.mark.∏_c=n.mark.con=1.0;
        newly_marked_ports_C ∪= {n};
[Phases 2&3: Controllability marking process]
    while(newly_marked_ports_C != ∅)
    [Phase 2: Transport mark(s) through connections]
        transported_to = ∅;
        for each n ∈ newly_marked_ports_C do
            for all (n,m) ∈ E(G_S) do
                m.mark = n.mark;
                trasported_to ∪= m;
    [Phase 3: Transport mark(s) through modules]
        transported_to = ∅;
        for each n ∈ newly_marked_ports_C do
            for all (n,m) ∈ E(G_S) do
                m'.mark.#p = n.mark.#p + (n,m).#controls;
                m'.mark.#c = n.mark.#c + (n,m).#clks;
                m'.mark.∏_{c−} = (n.mark.∏_{c−}) × (n.mark.∏_c);
                m'.mark.∏_c = ∏_{x∈(n,m).controls} x.mark.con;
                if(τ(m') > m.mark.con)
                    m.mark=m'.mark;
                    transported_to ∪= m;
                newly_marked_ports_C -= n;
        newly_marked_ports_C = transported_to;
        marked_ports_C ∪= newly_marked_ports_C;
```

End

After G_I marking is completed, (local) controllability and observability mark is available for each in-circuit port. Then, (global) testability of a circuit x can be evaluated using the function τ_T defined by the formula presented below, where $x.con_{ratio}$ ($x.obs_{ratio}$) represents the percentage of controllable (observable) 1-bit signals in interfaces of in-circuit modules, and $x.con_{avg}$ ($x.obs_{avg}$) represents the arithmetic average of controllabilities (observabilities) of marks obtained during the marking process.

$$\tau_T(x) = x.con_{ratio} \times (1 + x.con_{avg}) \times x.obs_{ratio} \times (1 + x.obs_{avg})/4 \qquad (2)$$

5.3.3 Step-by-Step Example

Let us present step-by-step functionality of the whole TA algorithm briefly when applied to circuits NL1 (Figure 5) and NL3 (Figure 6). The following information is available for circuit NL1 (NL3) in Tables 1, (2&3):

- for each port n (2nd column from the left), information in $i - t(co)$ format (1st column from the left) describes in which step (i) a mark was assigned to n during controllability ($t = C$) or observability ($t = O$) analysis process,
- other columns inform about attributes that were assigned to $n.mark$ in the particular step. Character "-" means particular attribute remains unchanged.

For example, the information stored in two "1-C rows" of Table 1 informs about controlability marks and their attributes that were assigned to primary ports NL1.in and NL1.clk in step 1 of TA algorithm. Likewise, the information stored in "6-O" row of the same table informs about observability mark that was assigned to MOD3.y in step 6 of TA algorithm.

As a more detailed illustrating example, let the mark-propagation process for NL1 circuit be presented now (cf. Figure 5 and Table 1). For NL1 circuit ($\sum_p = 75$, $\sum_c = 3$ for NL1), TA runs in the following 6 steps:

1-C: each PI (NL1.in, NL1.clk) is marked by the best-controllable mark,

2-C: marks are transported unchanged to ports (MOD1.a, REG1.clk, REG2.clk, REG3.clk) physically connected to PIs that were newly marked in the 1st step. Because no other port can be marked (due to feedback loop dependency in NL1 data path it is impossible to transport mark from MOD1.a to next ports in the data path), controllability analysis ends in this step;

3-O: observability analysis starts with marking primary output NL1.out;

4-O: mark from NL1.out is transported unchanged to REG.y output port. The mark is unaffected by the transport because it is transported through a physical connection;

5-O: mark is transported through REG3 structure to REG3.d input port. The transport is possible because there is only 1 port (REG3.clk) controlling data flow through REG3 structure and REG3.clk is fully controllable (marked in 1-C). Because of going through REG3 structure, mark attributes are penalized using τ function applied in phase 3 of the observability analysis algorithm.

6-O: mark is transported unchanged from REG3.d to MOD3.y. In this step TA applied to NL1 ends.

Unmarked ports remain uncontrollable and unobservable. In total, there are only 6 controllable ports and 4 observable ports from 21 ports in NL1. According to marks assigned during TA process, testability of NL1 is set to 0.011, which means NL1 is a low-testable structure.

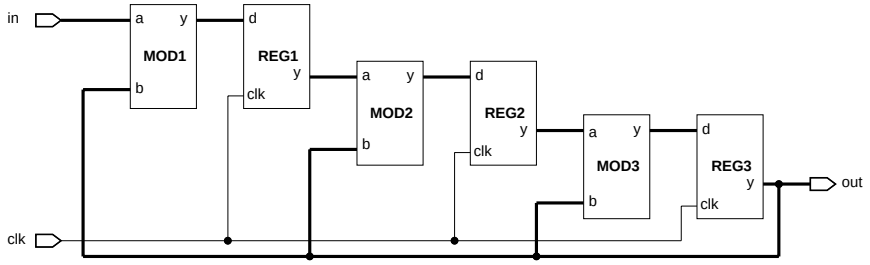

Fig. 5. NL1 structure

Because of high number of uncontrollable and unobservable ports in the above-mentioned example of NL1, REG3 was modified to scan-register to break the most-nested loop – and, thus all loops – in the NL1-structure (see Figure 6, NL3 circuit. $\sum_p = 78$, $\sum_c = 10$ for NL3). Significant enhancement of NL3 testability was expected as a consequence of the modification.

Step # (C/O)	**Newly colored ports**	**Mark of the port**				
		#p	#c	$\prod_{c-}$	Con.	Obs.
1-C	NL1.in	0	0	1.0	1.0	–
	NL1.clk	0	0	1.0	1.0	–
2-C	MOD1.a	0	0	1.0	1.0	–
	REG1.clk	0	0	1.0	1.0	–
	REG2.clk	0	0	1.0	1.0	–
	REG3.clk	0	0	1.0	1.0	–
3-O	NL1.out	0	0	1.0	–	1.0
4-O	REG3.y	0	0	1.0	–	1.0
5-O	REG3.d	1	1	1.0	–	0.74
6-O	MOD3.y	1	1	1.0	–	0.74

Table 1. Testability marking process for NL1 circuit

In Tables 2 and 3, it can be seen that TA of NL3 took 28 steps (controllability analysis process took 14 steps as well as observability analysis process). It can be seen more ports are marked (i.e. seen as controllable) in NL3-case than in NL1-case. As a result, all 25 ports are controllable and 20 ports are observable in NL3 structure. According to marks assigned during TA process, testability of NL3 is set to 0.434, which means NL3 is a well-testable structure with 95.9 % of testable nodes.

Some of the significant properties of the proposed TA algorithm are proved in [31]: especially, worst-case time complexity $O(|V(G_S)|\cdot|E(G_S)|+|V(G_I)|\cdot|E(G_I)|)$ of the algorithm and correctness of the algorithm in view of testability-evaluating formulas. Because of their sizes, proofs are omitted in this text. Instead of formal proofs, experimental results related to the method are presented in Section 6.

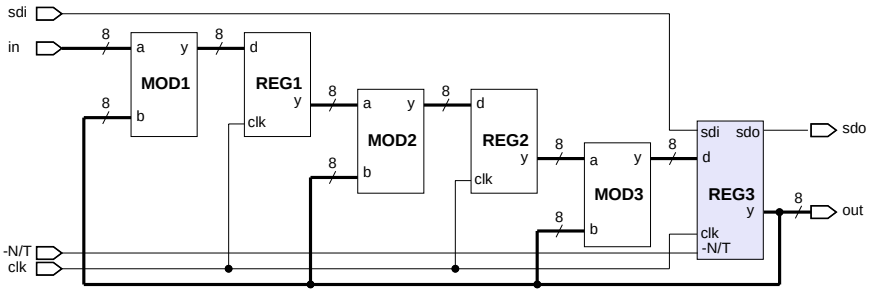

Fig. 6. NL3 structure (NL1 with REG3 modified to scan register)

Step # (C)	Newly colored ports	Mark of the port				
		#p	#c	$\prod_{c-}$	Con.	Obs.
1-C	NL3.in	0	0	1.0	1.0	–
	NL3.clk	0	0	1.0	1.0	–
	NL3.-N/T	0	0	1.0	1.0	–
	NL3.sdi	0	0	1.0	1.0	–
2-C	MOD1.a	0	0	1.0	1.0	–
	REG1.clk	0	0	1.0	1.0	–
	REG2.clk	0	0	1.0	1.0	–
	REG3.clk	0	0	1.0	1.0	–
	REG3.sdi	0	0	1.0	1.0	–
	REG3.-N/T	0	0	1.0	1.0	–
3-C	REG3.sdo	2	8	1.0	0.266	–
	REG3.y	2	8	1.0	0.266	–
4-C	NL3.out	2	8	1.0	0.266	–
	MOD3.b	2	8	1.0	0.266	–
	MOD2.b	2	8	1.0	0.266	–
	MOD1.b	2	8	1.0	0.266	–
	NL3.sdo	2	8	1.0	0.266	–
5-C	MOD1.y	8	0	0.266	0.238	–
6-C	REG1.d	8	0	0.266	0.238	–
7-C	REG1.y	9	1	0.266	0.214	–
8-C	MOD2.a	9	1	0.266	0.214	–
9-C	MOD2.y	10	8	0.214	0.051	–
10-C	REG2.d	10	8	0.214	0.051	–
11-C	REG2.y	11	9	0.214	0.033	–
12-C	MOD3.a	11	9	0.214	0.033	–
13-C	MOD3.y	10	8	0.033	0.008	–
14-C	REG3.d	10	8	0.033	0.008	–

Table 2. Controllability marking process for NL3 circuit

Step # (O)	Newly colored ports	Mark of the port				
		#p	#c	$\prod_{c-}$	Con.	Obs.
15-O	NL3.sdo	0	0	1.0	–	1.0
	NL3.out	0	0	1.0	–	1.0
16-O	REG3.y	0	0	1.0	–	1.0
	REG3.sdo	0	0	1.0	–	1.0
17-O	REG3.d	2	1	1.0	–	0.886
	REG3.sdi	2	8	1.0	–	0.266
18-O	MOD3.y	2	1	1.0	–	0.886
	NL3.sdi	2	8	1.0	–	0.266
19-O	MOD3.b	10	1	0.033	–	0.026
	MOD3.a	2	1	0.266	–	0.235
20-O	REG2.y	2	1	0.266	0.235	–
21-O	REG2.d	3	2	0.266	0.209	–
22-O	MOD2.y	3	2	0.266	0.209	–
23-O	MOD2.b	11	2	0.057	–	0.040
	MOD2.a	3	2	0.070	–	0.055
24-O	REG1.y	3	2	0.070	–	0.055
25-O	REG1.d	4	3	0.070	–	0.049
26-O	MOD1.y	4	3	0.070	–	0.049
27-O	MOD1.b	12	3	0.070	–	0.043
	MOD1.a	4	3	0.019	–	0.013
28-O	MOD1.y	4	3	0.019	–	0.013

Table 3. Observability marking process for NL3 circuit

5.4 Research Related to Application of Our Method

In order to verify theoretically proven properties of proposed TA algorithm practically, we have decided to apply the algorithm in the area dealing with automatic generation of RTL benchmark circuits [24, 22] and in DFT area [13, 14, 30, 31, 32].

The first-mentioned area deals with a method for evolving a circuit that is the worst testable within a given sub-class of RTL circuits. Our TA algorithm was utilized there to evaluate testability of a particular solution during the evolution process. As an output of the evolution process, set of RTL benchmark circuits has been generated and offered for a public use [23]. Actually, it is the only set containing RTL benchmark circuits of such complexities – up to thousands of PIs/POs and in-circuit modules, hundreds of thousands of gates, etc. The latest results from this area were published in [22].

To be able to apply the algorithm in the second-mentioned area, i.e., to be able to determine which *scan layout* (particular way of selecting and chaining registers in scan chains) represents the most feasible alternative according to given design constraints and desired diagnostics properties of resulting design, it was necessary to find answers to following questions first:

How can particular scan layout be described mathematically, and what data structure is suitable to represent such a mathematical notation? Proposed *notation* [14, 30, 31] is related to the below-mentioned analysis of scan layout search state-space and can be summarized as follows:

- scan chain is represented by a sequence of registers,
- special character (period, dot) is used to separate particular scan chains,
- if the ordering of scan registers within the scan chains is not important, registers belonging to the same scan chain are chained in the left-right direction according to increasing values of their indices (registers with higher indices are on the right),
- scan chains are ordered in left-to-right direction according to increasing index of the first register in particular scan chain (scan chains with higher index of the first register are placed on the right),
- if there is no register selected into scan, notation contains no character.

For an example of scan layouts corresponding to the notation, see Figure 7. In Figure 7 a), registers R_1, R_2 and R_5 are modified to scan registers. The registers are placed in two scan chains in such a way the first scan chain consists of registers R_2 and R_1 (in given order), and the second one consists of register R_5. Notation $R_2R_1.R_5$ represents the scan layout. In Figure 7 b), registers R_1, R_2 and R_5 are included into scan again, but the first scan chain consists of R_1 and R_2 registers in reverse order than in the previous case. Notation $R_1R_2.R_5$ represents the scan layout. In Figure 7 c), all registers are included into scan in a following way. Each of registers R_2, R_3, R_6 forms separate scan chain and registers R_1, R_5 and R_4 form (in given order) further scan chain. Notation $R_1R_5R_4.R_2.R_3.R_6$ represents the scan layout. In total, there are two scan chains in Figure 7 a) and 7 b) and four scan chains in Figure 7 c). As another example of scan layouts described by the notation, let us present $R_1R_2.R_5$, $R_1R_4R_5.R_2.R_3.R_6$ where ordering of registers in scan chain is not important.

It can be seen that a scan layout can be encoded by means of a one-dimensional structure (e.g. an array). Element within the structure either identifies particular register or contains the separator.

What is the size of scan layout search/state-space? During our previous research activities (e.g., [13, 14, 30, 31]), we have concluded that the size can be determined by means of

- a Bell number [5] when ordering of registers within scan chains is not important or
- a Lah number [5] when ordering of registers within scan chains is important.

In both cases, the size grows exponentially with the number of registers within the circuit structure.

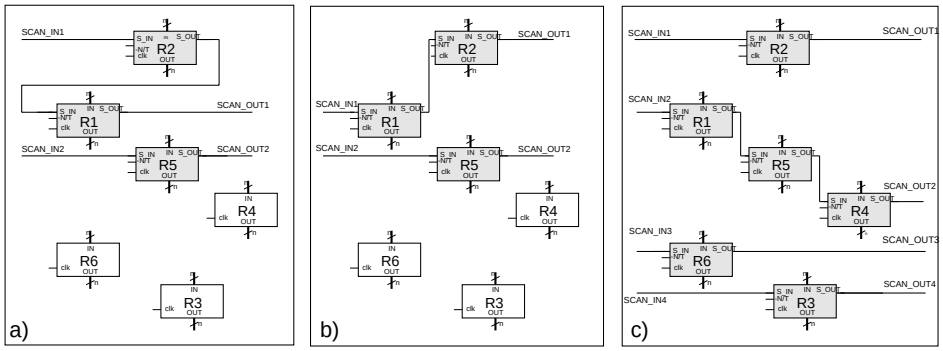

Fig. 7. Illustration to scan layout variants and their notations

6 EXPERIMENTAL RESULTS

To experimentally prove some of algorithm properties, algorithm was implemented in C++ and applied to several benchmark circuits. The goal was to verify theoretically proven properties by series of experiments and to be able to compare the results obtained by the method with those obtained with other existing approaches solving the same problem.

6.1 Testability Analysis

First of all, our TA method was applied to circuits from the [22] benchmark set. In Figure 8, the relation between number of modules within the circuit structure and average time needed to perform TA is presented. The numbers at the horizontal and vertical axis represent thousands of modules present in the circuit structure, and average time in seconds needed to perform TA of a circuit consisting of such a number of modules, respectively. The time is average because it differs a little for various circuit structures consisting of the same number of modules. In the figure, it can be seen that duration of the algorithm for "small" circuits consisting of approx. 1 000 modules is almost constant.

This is because it takes longer time to construct dynamic data-structures (start-phase) for such "small" circuits than to analyze their testability. For "bigger" circuits, total TA overhead becomes greater than overhead of the start-phase because of more complex circuit structures. In case of circuits from the [22] benchmark set, CPU time can be seen as a linear function of the number of in-circuit modules.

Below, testability results of common RTL benchmark circuits are presented in Table 4. The table informs about:

- the number of elementary data ports within the circuit structure,
- circuit controllability and ratio of controllable data ports,
- circuit observability and ratio of observable data ports,
- circuit testability and ratio of testable data ports.

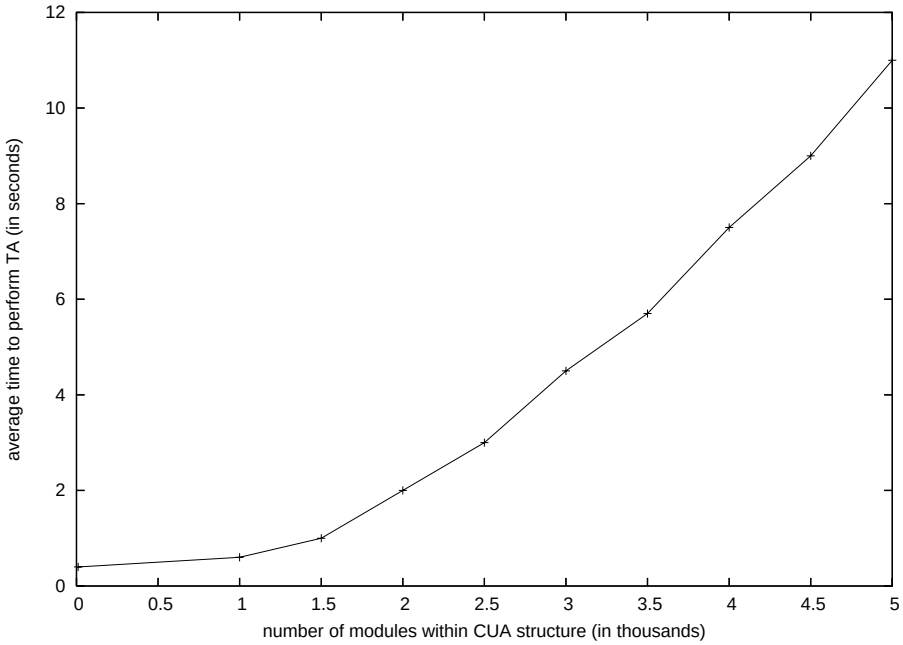

Fig. 8. Average CPU time needed for proposed TA algorithm

Circuit	**# of data ports**	**Con. (%)**	**Obs. (%)**	**Tst. (%)**
Bert	440	0.943 (100)	0.765 (100)	0.721 (100)
Diffeq	400	0.423 (56)	0.173 (28)	0.073 (0)
Paulin	512	0.912 (100)	0.833 (100)	0.760 (100)
Tseng	360	0.921 (100)	0.847 (100)	0.780 (100)

Table 4. Testability results of selected benchmark circuits

While for Bert results it is evident that all ports within the circuit are testable, in Diffeq results it can be seen Diffeq is a difficult-to-test circuit. Its result says that only 56 % of its ports within its DP are controllable, only 28 % are observable and none is both controllable and observable, i.e. none is testable. Diffeq testability is evaluated by value 0.073, which is very close to 0, and thus this is an untestable circuit. Third analyzed circuit was Paulin and the last one was Tseng.

6.2 Application in DFT

In DFT, results provided by our TA method were utilized for the following purpose. It was required to select registers into scan chains and to chain them in such a way the highest possible testability is achieved together with the lowest price proportional to size of modifications caused by scan application in the original circuit structure. The method for selection of registers into scan chains (i.e., from a scan layout) was

implemented using several optimizing techniques. First, it was implemented using a genetic algorithm, then using greedy-search algorithm and simulated annealing algorithm. Inputs of the method can be summarized in the following list:

- model of the original circuit structure,
- set of DFT techniques allowed for testability enhancement,
- desired diagnostic properties of the final circuit (e.g., fault coverage, test application time, number of testable nodes),
- design constraints (e.g., maximum area&pin overheads allowed),
- optimization algorithm.

At the output, a modification of original circuit structure fulfilling desired diagnostic properties and design constraints maximally appears. [11] deals with comparison of optimizing algorithms used to solve the problem. As the result, greedy-search algorithm was found as the best for the purpose. In Tables 5 and 6, results of various methods dealing with the same problem are summarized.

Benchmark circuit	**Scan technique**	**# of scan registers**	**Fault** coverage (%)	**Test** application time (cycles)	**Area** overhead (%)
Bert	None, **Our**	0	93.64	4 813	0
	[25]	–	–	–	–
	[3]	4	96.03	5 160	11.80
Diffeq	None	0	84.88	2 960	0
	[17]	4	93.19	14 221	14.29
	[16]	3	92.83	20 520	14.18
	[35]	2	94.88	23 021	13.96
	Full scan	13	97.47	15 935	17.06
	[35]	3	97.01	4 966	13.02
	[25]	3	93.75	21 542	14.18
	[3]	4	97.47	14 200	14.29
	Our	2	97.47	20 284	13.96
Paulin	None, **Our**	0	85.82	10 043	0
	[17]	3	90.05	14 221	14.29
	[16]	2	91.15	20 520	14.18
	[35]	2	94.61	23 021	13.96
	Full scan	10	93.67	15 935	17.06
	[35]	2	95.66	4 966	13.02
	[25], IDAT [3]	–	–	–	–
Tseng	None, [25] **Our**	0	95.10	6 823	0
	[3]	2	96.23	7 350	10.08

Table 5. Detailed comparison of experimental results Gained by various methods

It can be seen our method is able to produce scan layouts with a very good cost/quality trade-off between diagnostic properties and their costs (comparing to

other existing methods dealing with the same problem). In Table 5, the number of scan registers included in scan chains, estimated fault coverage, test application time and area overhead are selected as comparison basis. In Table 6, results are compared by means of particular scan layouts published and seen as the best in [3, 25]. The results are written according to the notation presented at the end of subsection 5.4. The symbol '*' in a table cell means that application of scan technique does not lead to desired testability improvement at given design constraints. The symbol '-' means that the circuit was not analyzed by the method.

Method	**Scan layout found for benchmark circuits**			
	Bert	*Diffeq*	*Paulin*	*Tseng*
[3]	$R_5R_4R_1R_2$	$R_4R_6R_1R_5$	-	R_5R_1
[25]	-	$R_1R_2R_4$	-	*
Proposed method	*	R_1R_6	*	*

Table 6. Comparison of results gained by several scan layout selection methods

6.3 Distribution of Testabilities in Scan Layout State-Space

During the experiments presented in the previous subsection, the scan layout was being searched that meets given design-constraints maximally and is characterized by maximal value of testability. In connection with the experiments, we were interested how testabilities are distributed within the scan layout state-space for a particular circuit. The goal was to approve or dismiss following hypothesis: testability of the circuit grows with growing number of registers and multiple scan chains.

In the experiments, the hypothesis has been approved. Experimental results gained for *Diffeq* benchmark circuit are presented in Table 7 and Figure 9. In Table 7, distribution of testabilities for particular scan layouts is presented. In each inner cell of the table, scan layout (testability value) fulfilling given pin and area overheads is presented in the upper (bottom) line of the cell. The distribution is visualized in Figure 9. It can be seen that in case of low-overhead constraints (upper-left cell), scan layout contains less registers and less multiple scan chains than in case of high-overhead constraints (bottom-right cell).

7 CONCLUSIONS

Novel TA method applicable to RTL digital circuits has been presented in the paper. The main goal of the method was to analyze an input circuit structure more precisely in order to achieve more detailed testability information about the structure. This effort was motivated by lack of existing TA methods. To achieve the goal, we have utilized generalized virtual port concept to model transparency of in-circuit modules first. On the basis of the concept, test pattern data flow (G_S) and test response data flow (G_I) digraphs were defined. Our TA method has been constructed

	Area overhead (%)				
Pin over. (%)	8	10	14	16	20
5	R_6R_1 0.631	$R_6R_1R_3$ 0.658	$R_6R_1R_3R_4$ 0.678	$R_6R_1R_4R_2R_3$ 0.689	$R_6R_1R_2R_4R_3$ 0.7
10	R_6R_1 0.631	$R_6R_1R_2$ 0.658	$R_6R_1R_4R_3$ 0.678	$R_6R_1R_4R_2R_3$ 0.689	$R_6R_1R_3R_1R_4$ 0.747
15	$R_1.R_6$ 0.638	$R_1.R_3.R_6$ 0.662	$R_1R_4.R_6R_3$ 0.758	$R_1R_2.R_6R_3R_4$ 0.772	$R_1R_4R_2.R_6R_5R_3$ 0.747
20	$R_1.R_6$ 0.638	$R_1.R_2.R_6$ 0.706	$R_1.R_3.R_6R_2$ 0.763	$R_1R_3.R_2.R_6R_4$ 0.786	$R_1R_5.R_3R_2.R_6R_4$ 0.797
25	$R_1.R_6$ 0.638	$R_1.R_3.R_6$ 0.706	$R_1.R_2.R_3.R_6$ 0.748	$R_1R_4.R_2.R_3.R_6$ 0.784	$R_1R_4.R_2R_5.R_3.R_6$ 0.798
30	$R_1.R_6$ 0.638	$R_1.R_2.R_6$ 0.706	$R_1.R_2.R_3.R_6$ 0.748	$R_1.R_2.R_3.R_4.R_6$ 0.799	$R_1.R_2R_4.R_3.R_5.R_6$ 0.812
35	$R_1.R_6$ 0.638	$R_1.R_2.R_6$ 0.706	$R_1.R_2.R_3.R_6$ 0.748	$R_1.R_2.R_3.R_4.R_6$ 0.799	$R_1.R_2.R_3.R_4.R_5.R_6$ 0.819

Table 7. Exploration of state-space of *Diffeq* scan layouts: Summary

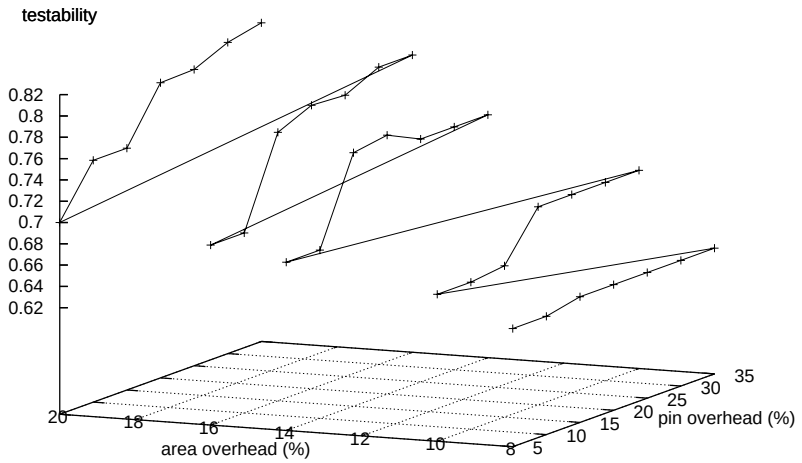

Fig. 9. Visualisation of testability distribution in *Diffeq* scan layout state-space

as a searching method over those digraphs – searching G_S (G_I) corresponds to controllability (observability) phase of the method. The main principles of the method are presented in the paper, together with illustrative step-by-step examples of its application, controlability analysis phase of the algorithm, testability evaluating formulas and its worst-case time complexity.

At the end of the paper, results gained by our TA method are presented and compared with those of existing methods. The results show that the method informs more precisely about circuit testability than existing methods do. In practice this can result e.g., in significant savings in the area of DFT application. This fact is accompanied by relevant experimental results presented in the paper. The second area our TA method was applied in was that dealing with generating RTL benchmark circuits. Our method was utilized there to evaluate testability of a particular solution during the evolution process. As the result, set of actually the most complex RTL benchmark circuits has been generated.

Nowadays, sophisticated experiments observing correlation between results of our TA method and fault-coverage results retrieved by commercial test generation tools are being performed. The first results look hopefully – they show our TA method is able to estimate fault coverage very narrowly. Completion of the experiments belongs to one of our main research activities for the near future.

The proposed TA method has been implemented in C++ language and – as a part of educational/experimental toolsets – it is available for public use in [28].

Acknowledgements

The work related to the paper has been financially supported by the Grant Agency of the Czech Republic (GACR) under contracts No. GP102/05/P193 "Optimizing Methods in Digital Systems Diagnosis" and GP102/04/0737 "Modern Methods of Digital Systems Design".

REFERENCES

[1] Abramovici, M.—Breuer, M. A.—Friedman, A. D.: Digital Systems Testing and Testable Design. IEEE Press, Piscataway, 1990, 670 p., ISBN 0-7803-1062-4.

[2] Abramovici, M.—Parikh, P. S.: Testability-Based Partial Scan Analysis. Journal of Electronic Testing: Theory and Applications, Vol. 7, 1995, No. 1. pp. 62–70.

[3] Bukovjan, P.: Allocation for Testability in High-Level Synthesis. Ph. D. thesis, Institute National Polytechnique de Grenoble, 2000, 130 p.

[4] Chen, C. H.—Saab, D. G.: A Novel Behavioral Testability Measure. Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 12, 2002, No. 12, pp. 1960–1993.

[5] Convay, J. H.—Guy, R. K.: The Book of Numbers. Springer-Verlag, New York, 1996, 320 p., ISBN 0-387-97993-X.

[6] Chen, C. H.—Saab, D. G.: A Novel Behavioral Testability Measure. Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 12, 2002, No. 12, p. 1960–1993.

[7] Fernandes, J. M.—Santos, M. B.—Olivera, A. L.—Teixeira, J. C.: Probabilistic Testability Analysis and DFT Methods at RTL. In Proceedings of 9th IEEE Workshop on Design and Diagnostics of Electronics Circuits and Systems, 2006, pp. 216–217.

[8] Goldstein, L. H.: Controlability/Observability Analysis for Digital Circuits. IEEE Transactions on Circuits and Systems, Vol. 26, 1979, No. 9, pp. 685–693.

[9] Grason, J.: TMEAS – a Testability Measurement Program. In Proceedings of IEEE/ACM Design Automation, 1979, pp. 156–161.

[10] Gu, X.: RT Level Testability Improvement by Testability Analysis and Transformations. Ph. D. thesis, Linkoping University, 1996.

[11] Herrman, T.: Optimalization in the Area of Selecting Registers into Scan Chains (in Czech). Diploma Work, FIT VUT Brno, 2004, 50 p.

[12] Kotásek, Z.—Strnadel, J.: Normalized Testability Measures at RT Level: Utilization and Reasons for Creation. In Proceedings of 36th International Conference on Modeling and Simulation of Systems, MARQ, 2002, pp. 297–304.

[13] Kotásek, Z.—Strnadel, J.: Optimising Solution of the Scan Problem at RT Level Based on a Genetic Algorithm. In Proceedings of 5th IEEE Workshop on Design and Diagnostics of Electronics Circuits and Systems, FIT VUT v Brně, 2002, pp. 44–51.

[14] Kotásek, Z.—Strnadel, J.: Testability Improvements Based on the Combination of Analytical and Evolutionary Approaches at RT Level. In Proceedings of Euromicro Symposium on Digital System Design – Architectures, Methods and Tools, IEEE Computer Society Press, 2002, pp. 166–173.

[15] Kuchcinski, K.—Peng, Z.: Testability Analysis in a VLSI High-Level Synthesis System. The Euromicro Journal, Microprocessing and Microprogramming, Vol. 28, 1990, Nos. 1–5, pp. 295–300.

[16] Lee, T. C.: Behavioral Synthesis of Highly Testable Data Path in VLSI Digital Circuits. Ph. D. Thesis, Department of Electrical Engineering, Priceton, University, 1993.

[17] Lee, D. H.—Reddy, S. M.: On Determinig Scan Flip-Flops in Partial Scan Designs. In Proc. of Intenational Conference on Computer-Aided Design, 1990, pp. 322–325.

[18] Makris, Y.—Collins, J.—Orailoglu, A.: Fast Hierarchical Test Path Construction for Circuits with DFT-Free Controller-Datapath Interface. Journal of Electronic Testing: Theory & Applications, Vol. 18, 2001, No. 1, pp. 29–42.

[19] Makris, Y.—Orailoglu, A.: Efficient Transparency Extraction and Utilization in Hierarchical Test. In Proc. of the IEEE VLSI Test Symposium, 2001, pp. 246–251.

[20] Maunder, C. M.—Bennetts, R. G.—Robinson, G. D.: CAMELOT: A Computer-Aided Measure for Logic Testability. In Proceedings of Intenational Conference on Computer Communication, 1980, pp. 1162–1165.

[21] Novák, F.—Khalil, M.—Robach, C.—Biasizzo, A.: System Level Diagnostic Strategies and Tools. In Proceedings of 4th IEEE Design and Diagnostics of Electronic Circuits and Systems, 2001, pp. 251–258.

[22] Pečenka, T.—Kotásek, Z.—Sekanina, L.: FITTest_BENCH06: A New Set of Benchmark Circuits Reflecting Testability Properties. In Proc. of 9th IEEE Workshop on Design and Diagnostics of Electronics Circuits and Systems, 2006, pp. 285–289.

[23] Pečenka, T.: FITTest_BENCHxx Benchmarks & Cirgen, 2005 [2006]. Available at `http://www.fit.vutbr.cz/~pecenka/cirgen/`.

[24] Pečenka, T.—Kotásek, Z.—Sekanina, L.—Strnadel, J.: Automatic Discovery of RTL Benchmark Circuits with Predefined Testability Properties. In Proceedings of the 2005 NASA/DoD Conference on Evolvable Hardware, IEEE Computer Society Press, 2005, pp. 51–58.

[25] Růžička, R.: Formal approach to the Testability Analysis of RT Level Digital Circuits (in Czech). Ph. D. Thesis, VUT Brno, 2002, 102 pp.

[26] Sekanina, L.—Strnadel, J.—Kotásek, Z.—Pečenka, T.: Evolutionary Design of Synthetic RTL Benchmark Circuits. In Proceedings of 9th European Test Symposium, IEEE Computer Society Press, 2004, pp. 107–108.

[27] Sheppard, J.—Kaufman, M.: IEEE P1522 Standard for Testability and Diagnosability Characteristics and Metrics (Draft), 2000 [2004]. Available at `http://grouper.ieee.org/groups/1232/pubs/`.

[28] Strnadel, J.: Educational tools for digital circuit diagnosis, 2005 [2006]. Available at `http://www.fit.vutbr.cz/~strnadel/diag/`.

[29] Strnadel, J.: Normalized Testability Measures Based on RTL Digital Circuit Graph Model Analysis. In Proceedings of 5th International Scientific Conference Electronic Computers and Informatics, TU Košice, 2002, pp. 200–205.

[30] Strnadel, J.: Scan Layout Encoding by Means of a Binary String. In Proceedings of of 37th International Conference on Modelling and Simulation of Systems, MARQ, 2003, pp. 115–122.

[31] Strnadel, J.: Testability Analysis and Improvements of Register-Transfer Level Digital Circuits (in Czech). Ph. D. Thesis, VUT v Brně, 2004, 150 p.

[32] Strnadel, J.: VIRTA: Virtual Port Based Register-Transfer Level Testability Analysis and Improvements. In Proc. of 8th IEEE Design and Diagnostic of Electronic Circuits and Systems Workshop, University of West Hungary, 2005, pp. 190–193.

[33] Ubar, R.: Functional Level Testability Analysis for Digital Circuits, 1992 [2004]. Technical report LiTH-IDA-R-92-03 available on: `http://the-compost-system.org/publications/cgi-bin/tr-fetch.pl?r-92-03`.

[34] Vedula, V. M.—Abraham, J. A.: FACTOR: A Hierarchical Methodology for Functional Test Generation and Testability Analysis. In Proceedings of the 2002 Design, Automation and Test in Europe Conference and Exhibition, IEEE Computer Society, 2002, pp. 730–735.

[35] Yang, T.—Peng, Z.: Incremental Testability Analysis for Partial Scan Selection and Design Transformations. In Proceedings of IEEE European Test Workshop, IEEE Computer Society, 1998.

Josef Strnadel received the M. Sc. degree in computer science and electrical engineering at the Faculty of Electrical Engineering and Computer Science, Brno University of Technology (BUT), in 2000, and the Ph. D. degree in information technology at the Faculty of Information Technology (FIT) of BUT, in 2004. Now he works as an assistant professor at the Department of Computer Systems at FIT BUT. His main research interests are related to testability of digital circuits.