Scan Layout Encoding by Means of a Binary String

Josef Strnadel[*]
strnadel@fit.vutbr.cz

Abstract: The paper deals with a problem of modelling a scan layout by means of a binary string. Using the proposed binary encoding, it is possible to identify concrete scan layout within a digital circuit. Encoded binary string can be seen as a unique identification of a set of registers that will be modified into scan registers and included into scan chains, parallel scan chains and (optionally) the order of scan registers within each scan chain. It can be also seen as a prescription how to modify digital circuit to include certain scan layout into original circuit structure.

Key Words: design for testability, scan layout, model, state-space, analysis, set partition, BELL number, LAH number, STIRLING number

1 Introduction

There are several steps that are necessary to be performed when a high-quality digital circuit design is expected, i.e., a design fulfilling certain requirements (functionality, pin and area overhead, delay constraints etc.) and also, a design fulfilling certain diagnostic properties. A digital circuit that is under the design process is denoted as *circuit under the design* (*CUD*) in this paper.

Basically, there are two big design areas that can be distinguished in the CUD design process. The first one can be seen as an area dealing with a required CUD function design, its verification and synthesis. The second one can be seen as an area dealing with CUD diagnostic properties.

There are two basic fields of research in the second area - the first one dealing with a *test generation* and the second one dealing with a *design for testability* (*DFT*), i.e., with modifying a CUD structure, e.g., by inserting *test-structures* in order to improve a *testability* [7] of the CUD.

It can be stated that a testability improvement (e.g., see [8]) of the CUD makes both the test generation of the CUD and test application to the CUD more effective. In general, goals of these areas are contradictory, so a cost/quality trade-off [6] is needed. In this paper, we do not deal with the problem of generating a test nor offer a solution to a problem of selecting registers into *scan chains* (for the second problem, see e.g. [5]). But, we present here a scan layout binary representation that is needed when DFT process is to be automated.

Dealing with a CUD testability improvement means either to design a CUD with a respect to testable-design rules (e.g., respecting *modeling for testability* (*MFT*) hints) or to use some of existing DFT techniques or their combination. Also, combination of DFT techniques and MFT rules is feasible. Currently, a big variety of DFT techniques is available. Applying a DFT technique (inserting a DFT structure) into a CUD brings about both CUD testability improvement that is desired and degradation of CUD parameters (growth of a die-area, pin number, delay), which is inconvenient because it makes the final CUD die more expensive.

[*] Faculty of Information Technology, Brno University of Technology, Božetěchova 2, CZ-612 66 Brno

Thus, it is necessary to find an acceptable cost/quality trade-off [5, 6] between CUD testability requirements and CUD costs. We will focus only on the scan DFT technique, its analysis and encoding by means of a binary string in this paper.

Before inserting a DFT structure into a CUD, it is necessary to consider, which concrete parts of the CUD are to be modified by means of selected DFT techniques. Generally, this is related with a location of all hard-to-be-tested CUD nodes [5, 7]. By a node, we can mean a concrete port, connection, state or an element within CUD structure. To identify these nodes, number of various methods can be used. Usually, so-called *testability analysis* (*TA*) [7] methods or methods based on *feedback-loops* (*FL*) *identification* [9] are used for these purposes. Also, on the basis of TA and FL method results, it is necessary to determine which previously identified hard-to-be-tested nodes are those ones which have the biggest (we can say critical) impact on the CUD testability. It is important to locate critical nodes within CUD structure exactly because improving their testability properties will consequently cause a significant CUD testability improvement with minimal costs [5, 6, 7]. In [2], the solution of controller testability is presented. It measures the combinational and sequential hardness to reach each state in the controller. Based on this result, hard to reach states are detected and testability enhancement techniques are developed to improve the state reachability.

We did our research in two stages. During the first one we analysed the number of all possible solutions (state-space size) of *scan layout* problem [5] and during the second one we dealt with a design and implementation of an effective methodology solving this problem in an acceptable way, i.e., in the way fulfilling the requirements of a customer or a designer. In this paper, we will primarily focus on the state-space analysis and modelling of a scan layout by means of a binary string. This binary representation is required, e.g., when automated DFT process is being designed.

The paper is organized as follows. First, a mathematical description of the scan layout together with proposed notation is introduced, illustrated and analysed. Second, the binary encoding of a scan layout model is described. Finally, the perspectives of future activities in this research area are discussed.

2 On Using Scan DFT Technique

When scan is used to improve CUD testability parameters, it is necessary to state which concrete memory elements (flip-flops (FFs) and registers (REGs) respectively) are to be modified to their scan-version, i.e., to scan flip-flops (SFFs) and scan registers (SREGs) respectively. It is known that using the scan, CUD parameters will change (growth of a die-area, number of pins, delay etc.), which makes the final CUD die more expensive. An *m*-bit SREG consists of *m* SFFs, which are connected in such a way that SREG can also operates as *m*-bit shift register in the *test mode* of operation.

In practice, this technique can be then applied only to a CUD containing memory elements. For the CUDs having no memory elements, it is necessary to consider the application of this technique because besides to-scan-version modification costs there is an additional need of inserting new FFs or REGs into a CUD which usually causes extra (and usually unacceptable) CUD costs.

Aside from the type of used SFF, common properties of scan memory cells can be summarized. Besides the *normal* (*function*) *mode*, SFFs and SREGs can operate in the *scan* (*test*) *mode* which is used to load *diagnostic data* into the scan memory and to read scan memory content respectively. The next property is that scan memory cells can be connected to a so-called *serial scan chain*. In the test mode, the scan chain is used to transport diagnostic data through the CUD structure.

Generally speaking, certain number of scan chains each containing at least one scan memory element can exist in a CUD structure. The problem is, how big this number is and which of the scan layout possibilities is characterized by the best cost/quality trade-off between diagnostic properties and acceptable costs.

With respect to used SFF types and comparing with original CUD parameters, there is a certain area overhead, pin overhead and performance degradation, which are the costs of using scan in the CUD. The benefits from implementing the scan can be seen in increasing diagnostic properties (testability, fault coverage, test application time, etc.) of CUD. Both the costs and testability depend highly on the way in which the scan memory elements are formed to scan chains, SFF type influences both parameters as well.

Scan approaches fall into two main groups: *full scan* (when all memory elements are included in the scan layout) and *partial scan*. In partial scan, flip-flops are selected in such a way that the remainder of the circuit has certain desirable testability properties.

3 Mathematical Description of a Scan Layout

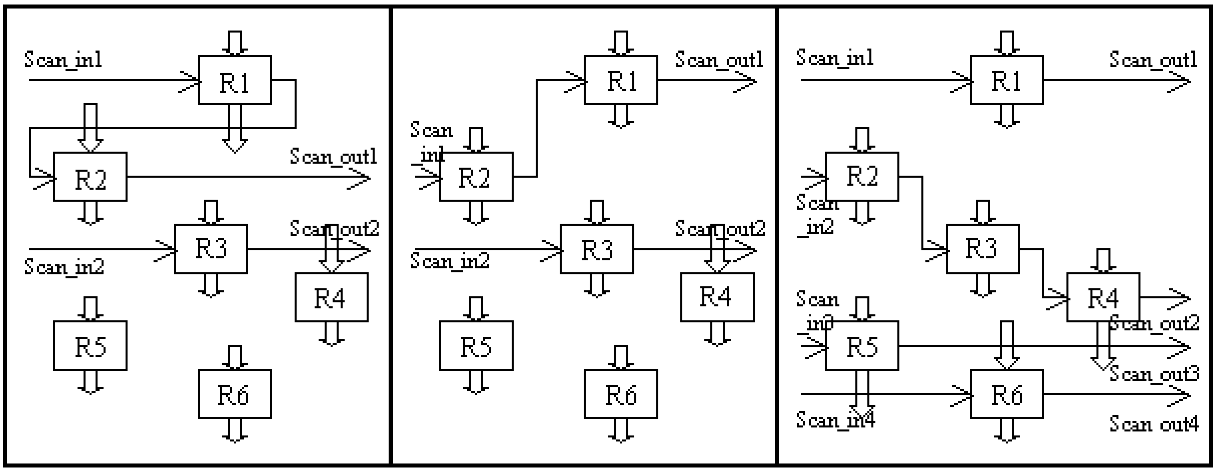
For the purposes of the modelling, we denoted the set of all registers in CUD as $\mathrm{REGS}_{\mathrm{CUD}}=\{\mathrm{R}_1, \mathrm{R}_2, \dots \mathrm{R}_n\}$. The basic idea of the first stage of our research is as follows. It is true that numerous alternatives how to implement a scan layout in a CUD exist supposing that each such layout can consist of several parallel (multiple) scan chains in which the order of registers is not important for the result (let this case be marked as A1) or is important for the result (let this case be marked as A2). The problem of repositioning (ordering) FFs in a CUD to minimize their number is solved in [3]. It is assumed that the goals of the analysis in terms of area, pin overhead, testability properties etc. are defined before the analysis starts. In this paper, we shall deal primarily with the A2 case, which is more general, but both A1 case and A2 case will be described in detail.

As the first step of our analysis, it is determined how big the partial state space is, i.e., how many scan structures can exist within concrete CUD, each of them consisting of *k* scan registers exactly, $\mathrm{SCAN}_{\mathrm{CUD}} \subseteq \mathrm{REGS}_{\mathrm{CUD}}$, where $1 \leq k \leq n$, $k = |\mathrm{SCAN}_{\mathrm{CUD}}|$. First, the partial state-space analysis (for given *k*), then the complete state-space analysis (for general *n*) for both A1 case and A2 case will be presented.

For the modelling, the notation is introduced:

- the ordered (A1 case) or non-ordered (A2 case) sequence of scan registers represents scan registers belonging to the same scan chain within the scan layout configured in CUD,
- the dot will separate scan chains that are parallel within the scan layout configured in the CUD,
- every $r\in \mathrm{SCAN}_{\mathrm{CUD}}$, will be involved exactly in one scan chain the scan layout configured in the CUD.

It is seen that by means of the above described notation, any realizable scan layout (containing any realizable combination of multiple scan chains and containing any realizable combination of scan registers within each scan chain (including their order in A2 case)) can be identified uniquely and if the order of registers is not taken into account (in A1 case), then some (e.g., R1R2.R3 and R2R1.R3 – see Fig. 1 solutions are identical, i.e., they represent identical scan layouts.



$R_1R_2.R_3$ (k=3, n=6) $R_2R_1.R_3$ (k=3, n=6) $R_1.R_2R_3R_4.R_5.R_6$ (k=6, n=6)

Fig. 1 – Example related to the proposed notation

From the mathematical point of view, each notation statement represents $\text{SCAN}_{\text{CUD}} = \{r_1, r_2, \ldots, r_k\}$ set partition into j partition classes (called also blocks), where $1 \le j \le k$ (j is the „number of dots in the notation statement increased by 1“ (i.e., the number of parallel scan chains within the scan layout configured in the CUD), and (in A2 case only) the ordering of elements in each block (such an ordered set is called a list) is required, rather than the ordering of blocks. As an example, $\{\{r_1\}, \{r_2\}\}$ and $\{\{r_1, r_2\}\}$ are all $\{r_1, r_2\}$ set partitions for A1 case and $\{\{r_1\}, \{r_2\}\}$, $\{\{r_1, r_2\}\}$ and $\{\{r_2, r_1\}\}$ are all $\{r_1, r_2\}$ set partitions for A2 case.

Then, our partial task is defined as ”how many different solutions of scan layouts based on including just k scan registers into the scan layout exist?”. It means that we must determine the number of partitions of the k-element SCAN_{CUD} set when the order of the elements in the blocks is not taken into account (A1 case) or is taken into account (A2 case). Let the number of partitions for given k be denoted as $nparts_k$, which represents the partial state-space size.

If the order of registers in the scan chain is not taken into account (A1 case) then the sequence of registers according to the notation introduced will not be seen as ordered and then $nparts_k = b_k$, where recurrent relation

$$b_k = \sum_{i=0}^{k-1} \left[\binom{k-1}{i} \times b_i \right], \tag{1}$$

with the initial value $b_0 = 0$ is a BELL number determining the number of partitions of a k-element set. BELL number can be also evaluated by means of

$$b_k = \sum_{i=1}^{k} S2(k,i), \tag{2}$$

where S2(k, i) is a STIRLING number of the second kind representing the number of set partitions of a k-element set into i blocks.

If it is possible to determine the value of $nparts_k$ for given k, i.e., if it is possible to determine how many unique scan layouts (each containing exactly k scan registers from concrete SCAN_{CUD}, $1 \le k = |\text{SCAN}_{\text{CUD}}| < n$) can exist within certain CUD, then it is also possible to determine the complete state-space size, i.e., how many different scan layouts can be configured within a CUD for general *n* if we systematically analyse the total number of all unique scan layouts each containing just 1, 2,... up to exactly *n* registers from REGS$_{\mathrm{CUD}}$ set (all possible 1, 2,... , *n*-element subsets of REGS$_{\mathrm{CUD}}$ set are created and all possible scan layouts are formed from each such subset). We have derived that the number of scan layouts is

$$nstructs_n = \sum_{k=1}^{n}\left[\binom{n}{k} \times nparts_k\right], \tag{3}$$

where the value of $\binom{n}{k} \times nparts_k$ represents the product of two values: the number of all possible *k*-element subsets (here, SCAN$_{\mathrm{CUD}}$ sets) of an *n* element set (here, REGS$_{\mathrm{CUD}}$ set) and the number of various layouts, each containing exactly all of *k* scan registers from a given SCAN$_{\mathrm{CUD}}$ set.

Now, the problem of *nstructs$_n$* evaluation for A1 case is solved. The problem of *nstructs$_n$* evaluation for A2 case is more complicated because the order of the elements in each block (the order of scan registers within each scan chain) must be taken into account, which means that the formula for *nparts$_k$* evaluation will be more complicated for the A2 case than for the A1 case. For each block, all possible permutations must be taken into account, which was not required in A1 case. For this purpose it is necessary to determine a formula evaluating *nparts$_k$* in A2 case as well. The procedure of deriving the formula is quite complicated, therefore just the resulting formula is given here as

$$nparts_k = \sum_{i=1}^{k}\left[\frac{k!}{i!} \times \binom{k-1}{i-1}\right] = a_k, \tag{4}$$

where a_k represents the number of partitions of a *k*-element set, where each block is a list. Let it be noted that the value

$$\mathrm{L{\scriptsize AH}}(k, i) = \left[\frac{k!}{i!} \times \binom{k-1}{i-1}\right] \tag{5}$$

represents the number of ways in which a *k*-element set can be partitioned exactly into *i* blocks (lists), where the order of elements in each block is taken into account. This value is being denoted as an unsigned LAH number (for more detail information see, e.g., [1, 4]).

4 Binary Encoding of a Scan Layout

To be able to utilize a GA for our purpose, it was necessary to model the problem by means of a bit string, denoted as a chromosome. In the previous section, it was shown that for CUD containing *n* registers, *nstructs$_n$* scan layouts can exist within the CUD. Also, the notation and analysis of scan layout state-space for both A1 case and A2 case were presented. Besides this theoretical background, it is required (for scan layout state-space exploration) any scan layout to be addressable in a certain way. In this section, encoding of a scan layout by means of a binary string will be presented. The proposed encoding can represent any set partition of a REG$_{\mathrm{CUD}}$ set, i.e., also any scan layout that can be represented by our notation.

Let the *chromosome* $\in \{0, 1\}^w$ (i.e., scan layout model) be defined as a bit string of the length w, where

$$w = n \times (\lceil 1 + \log_2 n \rceil) \qquad (6)$$

for A1 case and

$$w = n \times (\lceil 1 + 2\times\log_2 n \rceil) \qquad (7)$$

for A2 case.

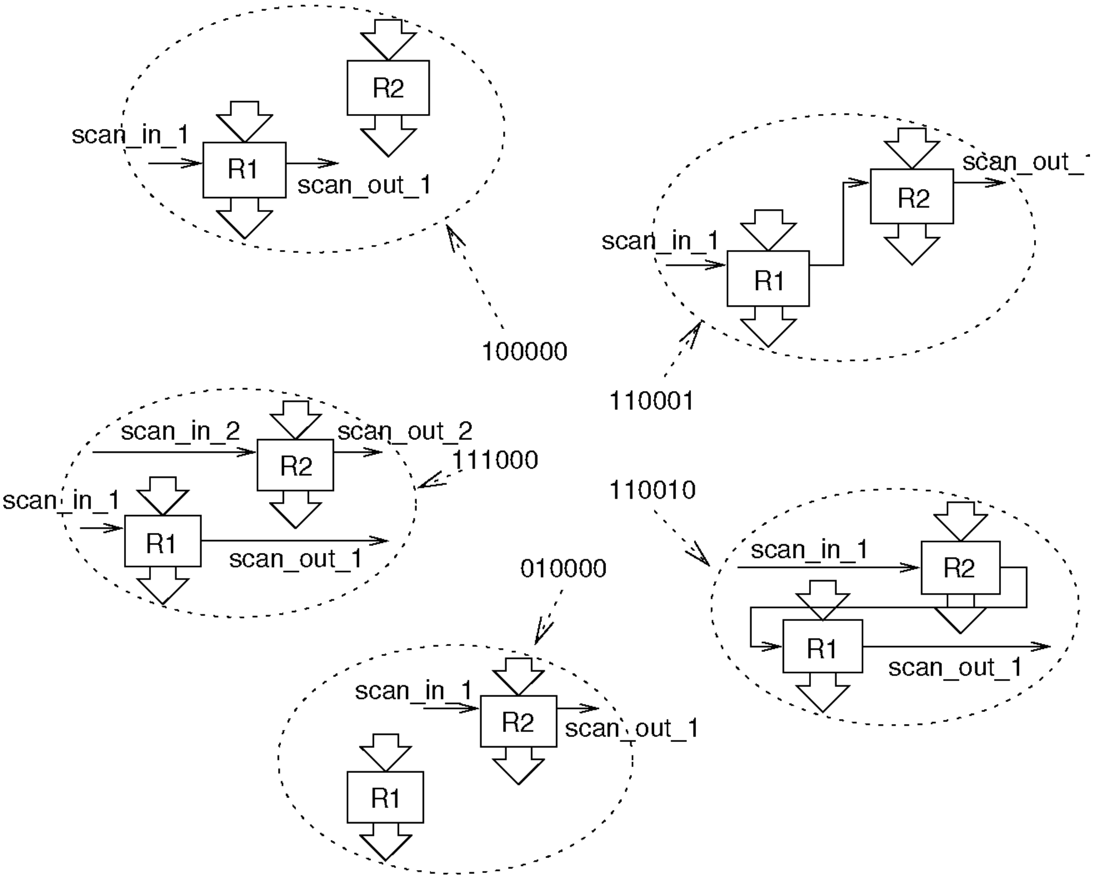


Fig. 2 – Example of a chromosome to scan layout mapping

The meaning of chromosome bits for both A1 case and A2 case will be explained in the next text. Suppose that chromosome bits are indexed in such a way that the highest (the most left) bit has index equal to (w-1) and the lowest (i.e., the most right) bit has index equal to 0. The chromosome bits are organized into three parts, each describing a special group of chromosome bits:

- **The highest *n* bits of a chromosome**: for both cases, bits with indexes (w-1), . . . , (w-n) represent information about registers ($\mathrm{REG_{CUD}}$ elements) that are selected to be also $\mathrm{SCAN_{CUD}}$ set elements, i.e., information whether $\mathrm{R}_i \in \mathrm{REG_{CUD}}$ belongs (then $(w\text{-}i)^{\mathrm{th}}$ bit is set to 1) or does not belong (then $(w\text{-}i)^{\mathrm{th}}$ bit is set to 0) also to the $\mathrm{SCAN_{CUD}}$ set.

- **The following n × ($\lceil 1 + \log_2 n \rceil$) bits of a chromosome**: for both cases, each group of $\lceil 1 + \log_2 n \rceil$ bits (i.e., chromosome bits with indexes $(w - n - 1 - (i - 1) \times \lceil \log_2 n \rceil), \dots , (w - n - i \times \lceil \log_2 n \rceil)$) belongs to a concrete $R_i \in REGS_{CUD}$, the most left group belongs to the R_1, . . ., the most right group belongs to the R_n. Each group of such bits identifies the number of the scan chain R_i belongs to. Let us note that a scan chain is identified by a unique number from $\{0, \dots , (n-1)\}$ set and that the group of bits belonging to R_i is valid iff $(w-i)^{th}$ chromosome bit is set to 1. These are also the last n × ($\lceil 1 + \log_2 n \rceil$) bits of A1-chromosome.
- **The following n × ($\lceil 1 + \log_2 n \rceil$) bits of a chromosome**: each group of $\lceil 1 + \log_2 n \rceil$ bits (i.e., chromosome bits with indexes $(w - n - 1 - (n + i - 1) \times \lceil \log_2 n \rceil), \dots , (w - n - (n + i) \times \lceil \log_2 n \rceil)$) belongs to a concrete $R_i \in REGS_{CUD}$, the most left group belongs to the $R_1, \dots$, the most right group belongs to the R_n. Each group of bits represents a scan register sequence-number within the scan chain. In the scan chain, a scan register with a lower sequence-number is included prior to a scan register with a higher sequence-number. This information is valid only for A2-chromosomes.

An example of a mapping from chromosome set to scan layout set for A2 case is presented in Fig. 2.

Let it be summarized what is the meaning of a chromosome in our case. First, the chromosome can be seen as a scan layout address. There are 2^w possible addresses, i.e., binary strings, each having a value from $<0; 2^w-1>$ interval. Concrete chromosome models concrete scan layout within CUD. Second, the chromosome can be seen as a unique identification of 1) a set of CUD registers that will be modified into scan registers and included into scan chains, 2) parallel scan chains that will be inserted into a CUD and 3) the order of scan registers within each scan chain in the A2 case). Third, the chromosome can be also seen as a prescription how to modify CUD to include certain scan layout into original CUD structure.

5 Conclusions and Future Research Perspectives

In the paper, a problem of modelling a scan layout by means of a binary string was dealt. It was shown that using the proposed binary encoding, it is possible to model any concrete realizable scan layout within a digital circuit.

To be able to design and implement automated DFT process (as the logical consequent in our research line), it is necessary to define the way in which the quality (reflecting constraints posed on a final CUD) of a chromosome will be evaluated and select proper state-space exploration method (e.g., genetic algorithm).

For the purpose of evaluating chromosome quality, a function have to be developed that assigns a numeric value to a chromosome (i.e., concrete scan layout model). The assigned value should be proportional to the cost/quality trade-off level (testability factors, chip area overhead and number of I/O pins etc. are processed in the function). Thus, the quality (specified by a customer or designer) of two unique chromosomes (each representing unique CUA modification, i.e., solution of the scan layout problem) can be easily compared numerically and the solution (scan layout) automated DFT process is searching for would be represented by a chromosome with the highest possible value that was assigned to it.

Also, there is an intention to develop state-space restriction methods and to extend proposed method to more test techniques than scan. This will offer the possibility to improve digital circuit testability through an optimum-close combination of selected DFT techniques in an acceptable time, which is a feature that is not available and published yet.

6 Acknowledgements

This work has been financially supported by the Grant Agency of Czech Republic grant No. 102/01/1531 "Formal Approaches in Digital Circuit Diagnostics – Testable Design Verification" and the Research Intent of FEI, Brno University of Technology, Czech Republic, grant No. CEZ: J22/98:262200012 "Research of the Information and Control Systems".

References

[1] Comtet, L., Advanced Combinatorics: The Art of Finite and Infinite Expansions, Reidel Publishing Company, Dordrecht, 1974.

[2] Gu, X., Larsson, E., Kuchinski, K., Peng, Z., A Controller Testability Analysis and Enhancement Technique, In: Proceedings of European Design & Test Conference 1997, Paris, France, 1997, pp. 153–157 .

[3] Higami, Y., Kajihara, S., Kinoshita, K., Partial Scan Design and Test Sequence Generation Based on Reduced Scan Shift Method, Journal of Electronic Testing: Theory and Applications, Vol. 7, 1995, pp. 115–123.

[4] Sloane, N. J. A., Plouffe, S., The Encyclopedia of Integer Sequences, Academic Press, Orlando, 1995.

[5] Strnadel J., Kotásek Z.: 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 DSD'2002, Los Alamitos, US, ICSP, 2002, pp. 166-173.

[6] Strnadel J.: Evaluating Cost/Quality Trade-off Solutions Proposed During a DFT Process, In: Proceeding of 8th Conference Student EEICT 2002, Brno, CZ, VUT v Brně, 2002, pp. 506-510.

[7] Strnadel J.: Normalized Testability Measures Based on RTL Digital Circuit Graph Model Analysis, In: Proceedings of The fifth International Scientific Conference Electronic Computers and Informatics 2002, Košice, SK, TU v Košiciach, 2002, pp. 200-205.

[8] Yang, T., Peng, Z., An improved register-transfer level functional partitioning approach for testability, Journal of Systems Architecture, Vol. 46, No. 3, 2000, 209–223.

[9] Zbořil F., Kotásek, K., Strnadel J., Mika D., The Identification of Feedback Loops in RTL Structures, Proceedings of the 5th International Scientific Conference Electronic Computers and Informatics 2002, Košice, SK, TU v Košiciach, 2002, pp. 142–147.