Circuit Resynthesis for Improved Testability

Dr. BORISLAV ROUSSEV

Computer Science Department, University of the Witwatersrand, Private Bag 3, Wits 2050, South Africa
borislav@cs.wits.ac.za

Abstract. This work proposes a *modus operandi* for testable resynthesis of combinational circuits. Observational test-equivalence relation is introduced as a generalization of the well-known fault equivalence relation. Circuit decomposition and composition are defined using the graph model of a combinational circuit. The underlying notion of the paper is local transformations preserving test-equivalence based on observational test-equivalence relation and circuit decomposition. The application of transformations allows the initial circuit to be transformed to a test-equivalent circuit, which is easier to test. We exemplify the feasibility of the proposed method.

1. Introduction

The fabrication process for digital integrated circuits is prone to random defects, which may affect the functionality of a device. Testing of finished circuits is essential in weeding out defective parts to ensure that electronic systems built using the circuits function correctly. Circuit complexity increases every year with the advancing fabrication technology, but the number of input/output (I/O) connections on a chip decreases relative to the number of transistors. Therefore, techniques for improving the controllability and observability of internal circuit nodes become more important as time goes on. Methods for improving circuit testability have advanced from the level of design for testability (DFT) [1], in which additional logic is added to improve the testability of an existing design, to synthesis for testability [2], in which testability improvements are added before a gate-level circuit description is actually synthesized. The work that we are proposing falls between these two extremes, in the sense that it aims to improve the testability of an existing design through resynthesis, rather than simply adding additional logic. Sequences of local transformations are used to modify a circuit into a simpler representation with improved testability. Only transformations which preserve the original design functionality are used.

2. Previous Research

High test complexity results in high cost for the test generation process. This cost includes expenditures for test generation, fault simulation, and use of test equipment. The cost of testing a digital system has become the main determining factor in the overall cost of a new design, over-shadowing the costs of the other three design phases -- design specification, verification, and implementation. Therefore, techniques for synthesis of easily-testable circuits are quite necessary. Testability is an inherent property of a circuit's structure which measures quantitatively the difficulty of generating tests for the circuit. The basic attributes of testability are controllability and observability. Controllability is the ability to establish a specified signal value at each node in a circuit by setting values on external inputs. Observability is the ability to observe at an external output the signal value at any node in a circuit by setting values on external inputs. DFT techniques aim to increase the testability of a circuit to ensure that it is testable by enhancing the controllability and observability of internal circuit nodes.

Various approaches have been used in the past to improve the testability of combinational circuits without the added overhead of control and observation points. Automatic test generators (ATGs) have been used to identify redundant faults so that the corresponding redundant logic can be removed [3, 4, 5]. Structural analysis was used to remove many of the redundancies in combinational circuits [6, 7, 5], and the approach was extended to sequential circuits in [10]. Logic synthesis techniques have also been used to reduce redundancies [8, 5]. De and Banerjee [8] used probabilistic techniques to partition a circuit into several blocks such that resynthesis of the blocks results in a circuit implementation containing a near minimal number of redundancies. Pomeranz and Reddy [9] developed a resynthesis approach in which subcircuits are replaced by comparison units, making the resulting circuits fully testable for both stuck-at faults and path delay faults.

3. Definitions and Notations

In the paper we consider multi-input multi-output combinational digital circuits composed of *AND*, *OR*, *NOT*, *NAND*, and *NOR* gates. For an n-input circuit C which realizes $f_i(\underline{x})$, $i \in \{1, 2, ..., m\}$ and $\underline{x}=(x_1, x_2, ..., x_n)$, a fault α changes the function realized to $f_i^{\alpha}(\underline{x})$. $U=\{0,1\}$ is the universe of input vectors. M is the set $\{1, 2, ..., m\}$. $F(C)=\{\alpha_1, \alpha_2, \alpha_r\}$ is the prescribed fault set of interest. The type of fault model used here is the classical single stuck-at-fault (*ssaf*) model, where $l$sa1 ($l$sa0) denotes that line l is stuck at 1 (0). $T(\underline{x})=\{\underline{x} \in U \mid \exists i \in M: f_i(\underline{x}) \oplus f_i^{\alpha}(\underline{x})=1\}$ is called the test set of α. We say that two circuits are behavior equivalent iff they realize one and the same set of logic functions.

In what follows we define the decomposition of a combinational circuit using graph theory.

Let $G=(V, E)$ be a directed graph and C a combinational circuit. Primary inputs, primary outputs, and gates of C each have a distinct representative node in the graph. The lines of C are represented by edges oriented in the direction of the information flow. The function of an edge is the function on its corresponding line in C. The function of a node is the same as the function on the outgoing edge from the node.

Let $G'=(V', E')$ be an induced subgraph of G and $v \in V$. Then $inp(v)$ is the set of incoming edges incident on v, and $out(v)$ is the set of outgoing edges incident on v. The entry and exit nodes of the induced subgraph G' are defined as $entry(G')=\{v \mid inp(v)=0\}$ and $exit(G')=\{v \mid out(v)=0\}$, respectively.

Definition 1.

Let G' be an induced subgraph of G. G' is called E-component of G iff for every $v \in V' \setminus exit(G')$ the set of outgoing edges of v, $out(v)$, belongs to E' and for every $v \in V' \setminus entry(G')$ the set of incoming edges of v, $inp(v)$, belongs to E'. In other words if one of the outgoing (incoming) edges of a node belongs to E' then the rest of the outgoing (incoming) edges of the node also belong to E' except for the set of nodes $exit(G') \cup entry(G')$.

This means that one E-component is defined by its set of edges and has some nodes to share with other E-components. E-composition of several E-components is done by fusing their common nodes.

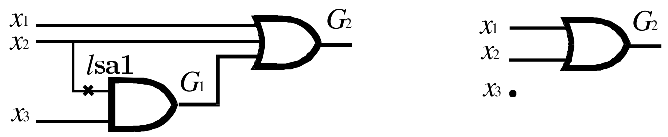
Definition 2.

The E-decomposition of the directed graph $G=(V, E)$ into two subgraphs is done by partitioning the set of edges E into two disjoint sets so that the induced subgraphs are proper E-components of G.

$G=G_1 \| G_2$ denotes the E-decomposition (E-composition) of G_1 and G_2. E-composition is obviously commutative and associative operation. E-composition is done by merging together the common nodes of the corresponding E-components.

4. Methodology

We propose to improve the testability of a circuit through sequences of local transformations which modify a circuit into a simpler, more testable representation. The approach is similar to the one developed by Pomeranz and Reddy [9]. The example on figure 1 illustrates the underlying idea. The test set for all ssaf's of circuit C_2 is also a test set for all detectable faults of circuit C_1 except for $l$sa1. Therefore, we can say that C_2 is test-equivalent to C_1 w.r.t. $F(C_1) \setminus \{l\text{sa}1\}$. It is clearly seen from the example that the test generation from C_2 is much faster than the test generation from C_1 because the number of lines and logic elements of C_2 is two times smaller than that of C_1. Besides there are three undetectable faults in C_1 which are not present in C_2. The question arises how circuit C_1 can be transformed into circuit C_2.



(a)Circuit C_1 (b)Circuit C_2

Figure 1. Introductory example.

To apply the resynthesis-for-testability approach we need tools to verify the application of transformations. The observation test equivalence relation is defined in [11] as follows.

Definition 3.

Let C_1 and C_2 be combinational circuits with fault sets $F(C_1)=\{\alpha_1, \alpha_2, ..., \alpha_3\}$ and $F(C_2)=\{\beta_1, \beta_2, ..., \beta_m\}$, respectively. Circuits C_1 and C_2 are observational test-equivalent, $C_1 t{\approx} C_2$, w.r.t. $F(C_1)$ and $F(C_2)$ iff they are behavior equivalent and $T(F(C_1))= T(F(C_2))$.

If C_1 coincides with C_1 and the power of both $F(C_1)$ and $F(C_2)$ is one we obtain the well-known fault-equivalence relation. Two faults α and β are equivalent iff $f_i^{\alpha}(\underline{x})=f_i^{\beta}(\underline{x})$, $i\in\{1, 2, ..., m\}$. We can consider $t{\approx}$ a generalization of the fault-equivalence relation.

We shall prove that the observational test-equivalence relation is equivalence relation over the set of combinational circuits. Let $C_1 t{\approx} C_2$ w.r.t. $F(C_1)$ and $F(C_2)$, and $C_2 t{\approx} C_3$ w.r.t. $F(C_2)$ and $F(C_3)$. $C_1 t{\approx} C_2$ implies that $T(F(C_1))=T(F(C_2))$. $C_2 t{\approx} C_3$ implies that $T(F(C_2))=T(F(C_3))$. Therefore, $T(F(C_1))=T(F(C_3))$ and according to definition 3 $C_1 t{\approx} C_3$ w.r.t. $F(C_1)$ and $=F(C_3)$ since they are behavior equivalent.

The criterion for determining which of two observational test-equivalent circuits is easier to test is called circuit size.

Definition 4.

Circuit size of the combinational circuit C is the ordered quadruple $size(C)=(G, L, R, U)$, where G is the number of gates, L is the number of line, R is the number of reconvergent fanouts, and U is the number of undetectable faults of the circuit C. Circuit sizes are compared as vectors, e.g. $(a_0, a_1, a_2, a_3)\leq (b_0, b_1, b_2, b_3)$ iff $a_i\leq b_i$, $1\leq i\leq 4$.

The defined circuit size is relative in the sense that not every two behavior equivalent circuits can be compared. Furthermore, the information for undetectable faults is completely missing. The defined circuit size is used to compare the test generation cost of the circuit before and after the application of a given transformation.

Example 1.

The circuit size of C_1 from figure 1 is $size(C_1)=(2, 7, 1, 3)$ while that of C2 is $size(C_2)=(1, 3, 0, 0)$.

All transformations introduced in the next section are defined over simple circuits. They preserve the observation test equivalent relation between the initial and the modified circuit; i.e., after the application of each transformation, the initial and the modified circuits will be observation test equivalent. Each transformation consists of a premise circuit and a resultant one. The resultant circuit is observation test equivalent to the premise w.r.t. its complete test set and not vice versa. This requirement guarantees that new faults won't be added during the resynthesis process. A given transformation may be applied only if the graph of the circuit can be E-decomposed into two E-components and one of them is isomorphic to the graph of the premise of the transformation. We use the graph model only to define formally the decomposition operation. In other words, the premise acts as a template that is compared to subcircuits of the resynthesized unit. When a match is found and if the application of the chosen transformation increases the testability of the current representation, the subcircuit isomorphic to the premise of the transformation is substituted by the resultant circuit. Then the subcircuits are composed to form the new representation using E-composition. Since transformations must be used in conjunction with E-composition, observational test-equivalence must be congruent w.r.t. E-composition.

Theorem 1 Substitution theorem.

Let C be the E-composition of C_1 and C_2, $C=C_1\|C_2$, and $F_1=F|C_1$ ($F_2=F|C_2$) be the fault set of interest $F(C)$ restricted to the lines of the subcircuit C_1 (C_2). If C_2' is a combinational circuit such that $C_2't{\approx}C_2$ w.r.t. F_2' and F_2 then the circuit $C'=C_1\|C_2'$ is observational test-equivalent with C w.r.t. F and F', $Ct{\approx}C'$, where $F'=F_1\cup F_2'$.

The proof of the Substitution theorem is given in [12].

5. Transformations and Algorithm

In this section we present a new method for circuit resynthesis that increases the testability of the circuit under consideration making use of a set of local transformations.

Seventeen transformations preserving observational test-equivalence are shown on figure 2. Each transformation preserves observational test-equivalence w.r.t. all ssaf's in the premise circuit except if not stated otherwise. *LB* denotes a logic block consisting of logic elements. All transformations can be generalized for an arbitrary number of inputs of the comprising logic elements. Transformation TR_3 is based on the DeMorgan theorem. Transformation TR_7 corresponds to the theorem $x_i f(\underline{x})=x_i f_i(1)$ which follows from the Expansion theorem, where the term $f_i(1)$ results from substituting 1 for x_i in $f(\underline{x})$. Transformations TR_7' to TR_9' inclusive correspond to theorems following directly from the Expansion theorem. Transformation TR_{10} differs from the rest in introducing extra logic elements. Line i in the premise of this transformation is a stem of a reconvergent fanout. Let i be assigned a logic value v_i, $v_i\in\{0, 1\}$. Suppose that v_i implies the logic value v_j at line j. Then the effect of the fault i″stuck-at-v_i′, $v'=v_i\oplus 1$, cannot be propagated through logic block LB_j since its output is driven to a fixed logic value by the signal necessary to excite the fault. On the other, hand nothing has been changed w.r.t. LB_m. The latter justifies the introduction of the new logic block. As a result of the application of a given transformation a line can be set to logic 1 or 0. In this case, three-valued logic simulation is carried out. The three values used in the simulation are 0, 1, and X; X is the *unknown* or *don't care*. All lines except for the fixed one from the transformation line are set to *don't care* values. Each line and logic element forced to logic value other than X by the simulation is cut. We shall call this procedure reduction.

Having introduced the theoretical framework we can present the method for testable resynthesis of combinational circuits. The flow diagram is shown on figure 3. C_I is the circuit which should be resynthesized. $F(C_I)$ is the fault set of interest. *Limit* and *Th* are constants and i is an index variable. C is current representation of the initial circuit C_I. In step 4, TR denotes the transformation which must be applied to the current representation. It is chosen among the transformations shown above.

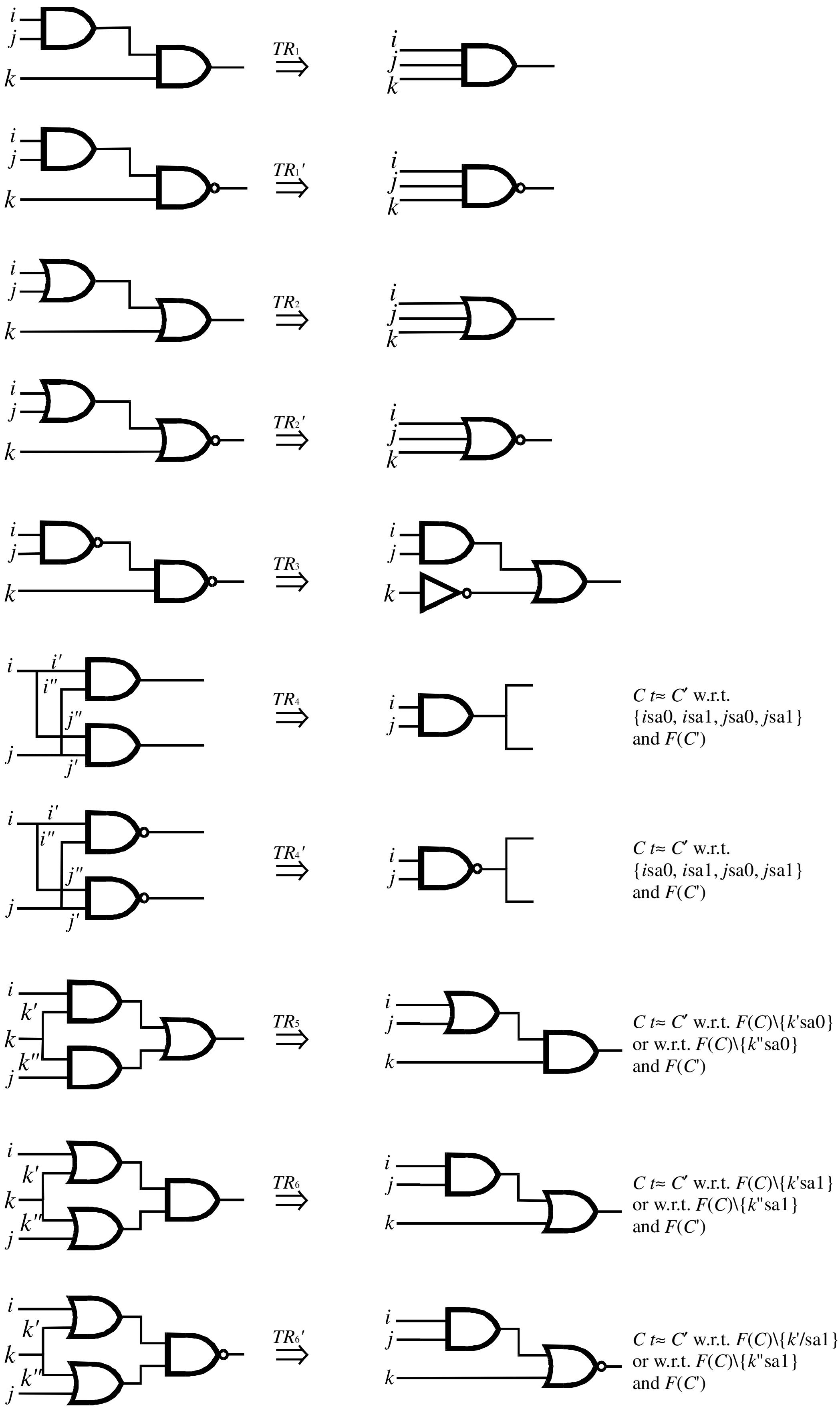


Figure 2. Transformations preserving test-equivalence.

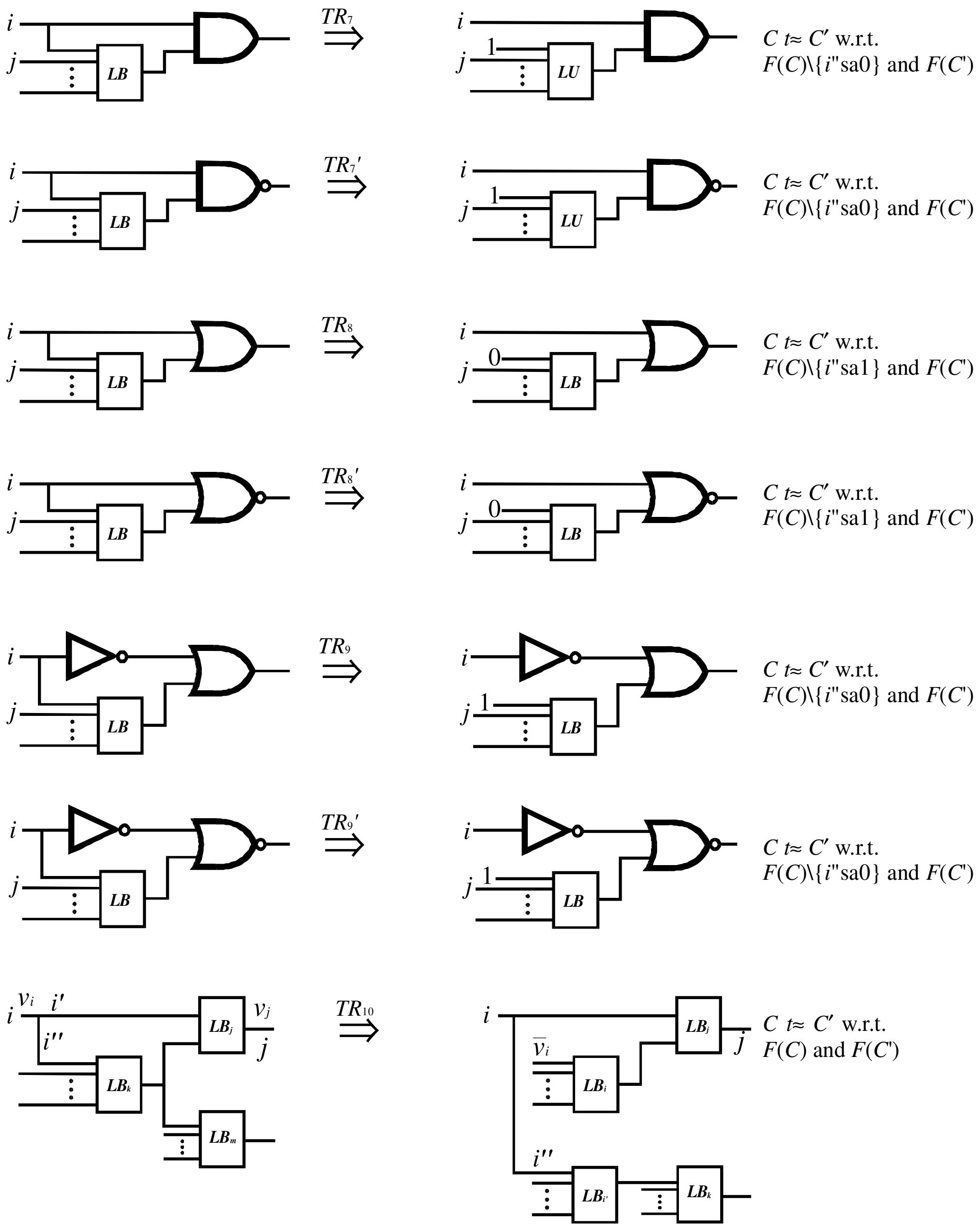


Figure 2. Transformations preserving test-equivalence.

In step 5, C is decomposed to C_1 and C_2 using the operation E-decomposition in such a way that the graph model of C_1 is isomorphic with that of the premise of the chosen transformation. The chosen transformation is applied to C_1 in step 6 which results in C_1 being transformed to C_1'. Reduction is carried out in step 7. In the next step C_1' is composed with its environment C_2 and circuit C' is obtained. C' is observational test-equivalent with the circuit representation before the application of the transformation *TR*. The described process continues iteratively until the integral testability measure goes beyond the predefined threshold *Th* or the number of iterations becomes greater than or equal to *Limit*. *Th* depends on the method for evaluation of the testability of the circuit. The constant *Limit* is a function of the number of lines in the initial circuit C_I. The function is to be identified experimentally. At present the gathered experimental data is not sufficient to solve this problem. Our estimate is that *Limit* should not be greater than the number of lines in the initial circuit.

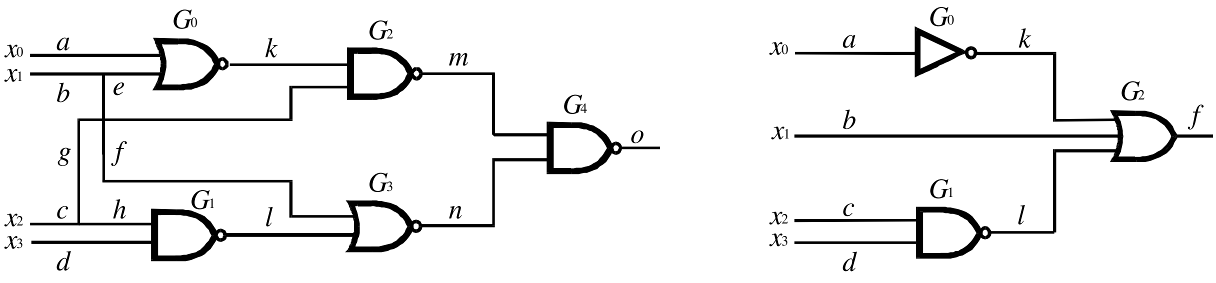
step 1. *initialize*(C_1, $F(C_1)$, *Th*, *Limit*);
step 2. $i = 0,\ C = C_1$;
repeat
    begin
step 3. $i = i + 1$;
step 4. TR = *chose_transformation*( );
step 5 $C = C_1 \parallel C_2$;
step 6 $C_1' = transform(C_1)$;
step 7 reduce( );
step 8 $C' = C_1' \parallel C_2$;
    end
until ($(TM(C) < Th)$ <u>or</u> $(i \leq Limit)$)

Figure 3.

Next we exemplify the proposed method.

Example 2.

Figure 4 shows the initial circuit C_I and the resynthesized circuits C_R. The graph model of C_I is shown on figure 5a. The circuit C_1 on figure 6a is a subcircuit of C_I. The graph model of C_1 shown on figure 5b is an *E*-component of the graph model of C_I . First, C_1 is transformed into C_2 using transformation TR_3 (figure 6b). Then, C_2 is modified to C_3 by fusing gates G_3 and G_4 (figure 6c). Figure 6d shows the result of the composition of C_3 with its environment. Next, transformation TR_8 is applied for lines b and e. As a result, line e is driven to logic 0. In the subsequent reduction line e is removed, then G_0 is replaced with an invertor and C_4 is obtained (figure 6e). Finally, after the application of TR_7 and the merging of G_0 and G_2 (G_2 has only one input) we obtain circuit C_2 from figure 4.



Circuit C_1 , size(C_1)=(5,13,2,2)
$F(C_1)$={*a*sa1,*a*sa0,*d*sa1,*d*sa0,*e*sa0,*f*sa0,*g*sa1,*h*sa1}

(a)

Circuit C_2 , size(C_2)=(3,7,0,0)
$F(C_2)$={*a*sa1,*a*sa0,*b*sa0,*c*sa1,*c*sa0,*d*sa1}

(b)

Figure 4. (a) Initial circuit C_1. (b) Resynthesized circuit C_2 obtained from C_1 by a sequence of local transformations preserving test-equivalence.

Example 3.

Circuit C_1 shown on figure 7 is a subcircuit of the ISCAS benchmark circuit c432.ckt, a circuit with 36 inputs and 432 lines. Note that v_i=1 implies G_m=0. First, we apply transformation TR_{10} and add a copy of the logic element G_i labeled G_i'. Finally, as a result of the subsequent reduction gate G_k and line i_1 are removed and circuit C_2 is obtained.

Example 3 clearly illustrates the advantages of the proposed approach. First, we have found out that there are no sensitized paths through gate G_k for anyone fault of either the analysed subcircuit or its environment. Second, we have discovered that the effect of anyone fault reaching gate G_j cannot be propagated through G_m. The removal of gate G_k together with its inputs and the output line of gate G_j feeding into G_m is equivalent to pruning the search space before the beginning of the test generation.

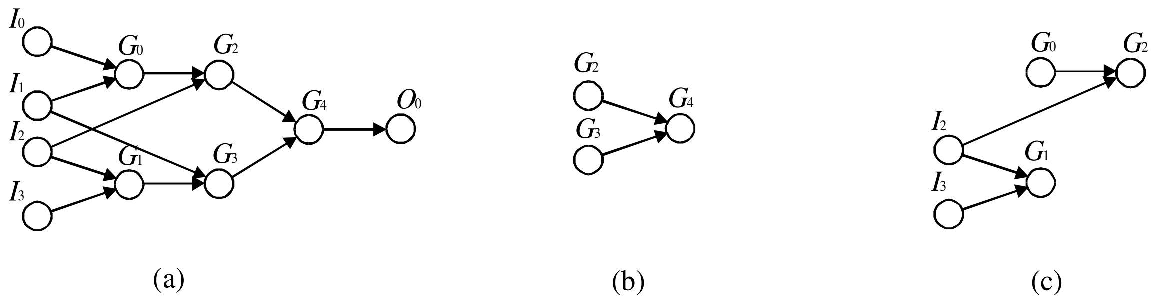


Figure 5. (a) Graph model of C_1 from figure 4. (b) and (c) Proper *E*-components of the graph model of C_1.

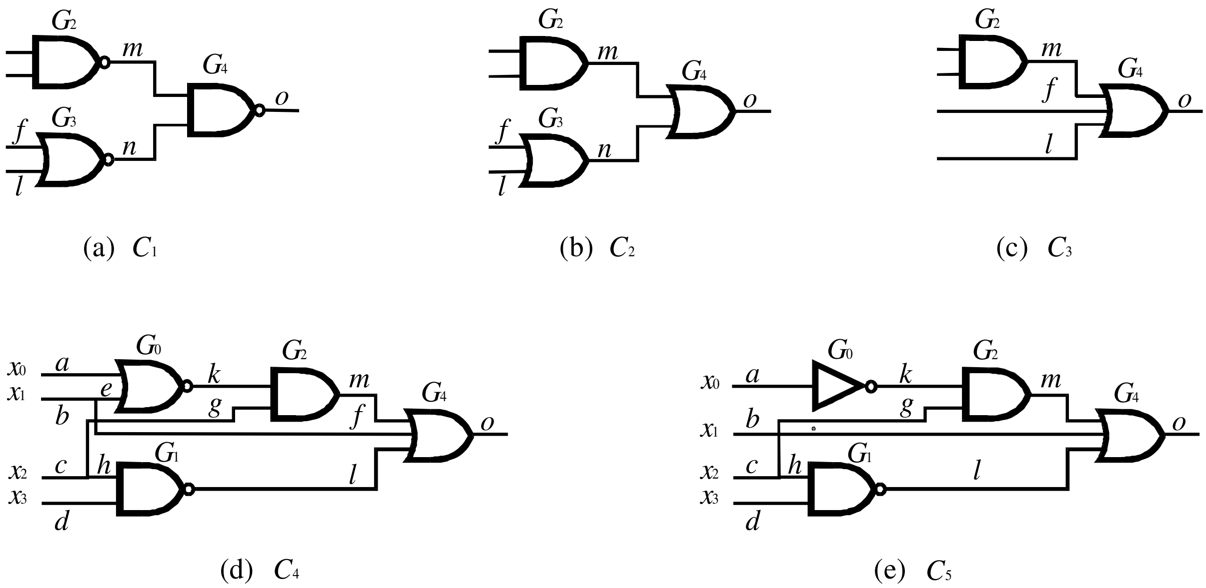


Figure 6. Example tracing the resynthesis of circuit C_1 from figure 4.

6. Experimental Results

Experimental results obtained for the circuit 74F181--a 4-bit ALU and an ISCAS benchmark circuit are shown on figure 8. We have implemented a version of PODEM[13] in C++ making use of SCOAP testability measures[14] for guiding the search. Only faults that might be detected at primary output number 17 are considered. Since the transformations are applied manually their cost is not included in the presented results. The cost of applying transformations grows proportionally to the number of lines in the circuit, i.e. the procedure of transformation application has linear complexity.

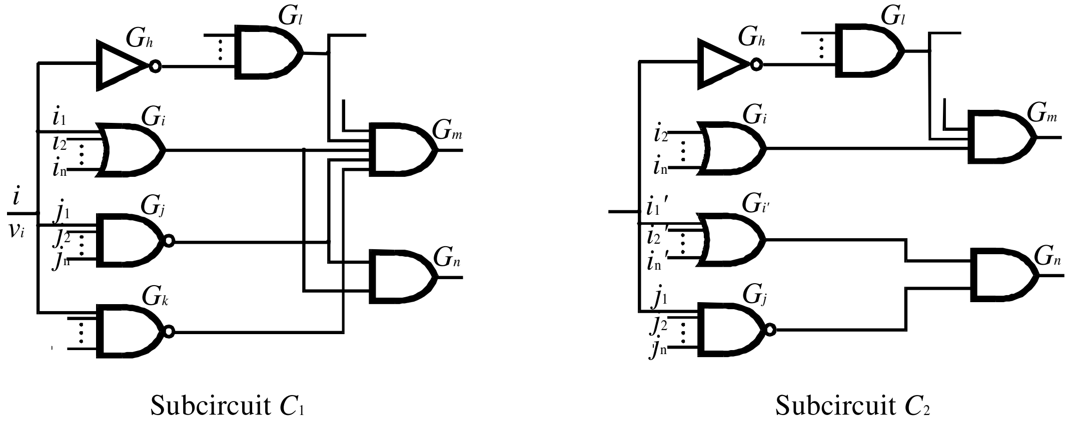


Figure 7. Example for the application of transformation TR_{10}.

The test set consists of 72 saf's including 14 undetectable ones. Test generation run time for the initial circuit is 29.8% greater than the test generation run time for the resynthesized circuit. For all faults the tests for the resynthesized circuit are generated faster than the tests for the initial one.

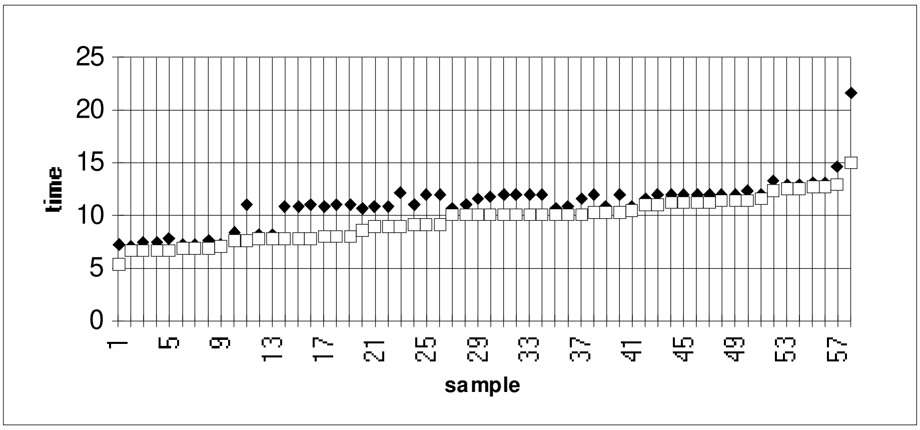


Figure 8. Experimental results for 74F181; ◆–initial circuit; □–resynthesized circuit.

7. Interpretation of the Results

The presented approach is advantageous for three reasons. First, it is based on local analysis of the circuit topology. Therefore, it is possible to construct a parallel algorithm for the identification and the application of transformations. The graph of the circuit might be used as a shared resource, accessed by a set of processes, transforming different parts by using an appropriate synchronization mechanism. In general, test generation is not amenable to parallel processing because of the global dependencies of the signals due to reconvergent fanouts. In the proposed approach, we can implicitly parallelize test generation in the resynthesis step. Second, by replacing complex subnets in the logic gate model by simple ones, the number of gates can be substantially reduced, which in turn reduces the size of the test generation problem. Third, the resynthesis procedure is only applied during a preprocessing phase, i.e., once for the circuit, while the test generation process is repeated for all faults of interest. As a result of the resynthesis, the fault coverage of the simulation-based test generators will improve, and the number of backtracks during the deterministic test generation process and the time taken for each backtrack will be reduced.

It should be noted that the proposed set of transformations is neither complete nor minimal. Owing to this, the resulting circuit could not be guaranteed to be minimal w.r.t. any criterion, since circuits obtained via different transformation sequences might differ a lot.

8. Conclusion

We presented a new approach to testable resynthesis of combinational circuits, one that improves the testability of the logic networks. It is based on local transformations of the circuits' topology. The proposed method is based on the observational test-equivalence relation. Seventeen transformations preserving observational test-equivalence have been shown. Employing these transformations we have worked out an algorithm. This algorithm is a sequence of three main steps that are repeatedly executed: identification of a portion of the circuit to be transformed, a decision-making step, and resynthesis. In the decision-making step, testability analysis is performed to decide whether to apply the chosen transformation. Finally, experimental results were obtained proving the feasibility of the proposed approach.

Future research includes: identification of new transformations and inclusion of all transformations in a library; parallel algorithm for applying the defined transformations.

Bibliography

[1] M. Abramovici, M. A. Breuer, and A. D. Friedman, *Digital Systems Testing and testable Design*, New York, NY: Computer Science Press, 1990.

[2] L. J. Avra and E. J. McCluskey, "High-level synthesis of testable designs: an overview of university systems," *Int. Test Conf.*, Test Synthesis Seminar Digest of Papers, Paper 1.1, 1994.

[3] K.-T. Cheng, "On removing redundancy in sequential circuits," *Proc. Design Automation Conf.*, pp. 164-169, 1991.

[4] M. Abramovici and M. A. Iyer, "One-pass redundancy identification and removal," *Proc. Int. Test Conf.*, pp. 807-815, 1992.

[5] S. K. Gupta, "Synthesis of testable combinational circuits: an overview of university activities," *Int. Test Conf.*, Test Synthesis Seminar Digest of Papers, Paper 1.2, 1994.

[6] P. R. Menon and H. Ahuja, "Redundancy removal and simplification of combinational circuits," *Proc. VLSI Test Symp.*, pp. 268-273, 1992.

[7] M. A. Iyer and M. Abramovici, "Low-cost redundancy identification for combinational circuits," *Proc. Int. Conf. VLSI Design*, pp. 315-318, 1994.

[8] K. De and P. Banerjee, "Logic partitioning and resynthesis for testability," *Proc. Int. Test Conf.*, pp. 906-915, 1991.

[9] I. Pomeranz and S. M. Reddy, "On synthesis-for-testability of combinational logic circuits," *Proc. Design Automation Conf.*, pp 126-132, 1995.

[10] A. Iyer and M. Abramovici, "Sequentially untestable faults identified without search," *Proc. Int. Test Conf.*, pp. 259-266, 1994.

[11] B. Roussev, ``Transformations speeding up the test generation for combinational circuits", 8th Int. Conf. Systems for Aut. of Engineering, 1994.

[12] B. Roussev, *Test Generation for Digital Circuits Modeled with Functional Graphs*, Ph.D. dissertation, Technical University-Sofia, 1996.

[13] P. Goel, "An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits," *IEEE Trans. Comp.*, Vol.C-30, pp.77-84, June 1980.

[14] L. Goldstein, "SCOAP: Sandia Controllability/Observability Analysis Program," *Proc. 17th Des.Auto.Conf.*, Minneapolis, pp.190-196, June 1980.