Local Transformations and Robust Dependent Path Delay Faults

Institut für Informatik — Report 81 *

Harry Hengster[1] Uwe Sparmann[2] Bernd Becker[1] Sudhakar M. Reddy[3]

[1] Institute of Computer Science, Albert-Ludwigs-University,
Am Flughafen 17, 79110 Freiburg, Germany
Email: {hengster,becker}@informatik.uni-freiburg.de

[2] Computer Science Department, University of Saarland,
Postfach 15 11 50, 66041 Saarbrücken, Germany
Email: sparmann@cs.uni-sb.de

[3] Department of ECE, University of Iowa,
Iowa City, IA 52242, USA
Email: reddy@eng.uiowa.edu

Abstract

Local transformations are used in several synthesis approaches. During application of such transformations attention has to be paid to many important properties, e.g. area, speed, power consumption, and testability. In this paper we study relations between local transformations and delay fault testability. In delay testing it is not necessary to test every path in a circuit to ascertain correct timing behavior. For example, a set of robust dependent path delay faults need not be considered for testing if all paths that are not robust dependent are tested. We present sufficient conditions for local transformations which ensure that a test set for all non-robust-dependent paths in the original circuit is also a test set for all non-robust-dependent paths in the transformed circuit. These conditions are applied to some local transformations which are often used in logic synthesis and it is shown that they preserve testability. The impact of local transformations on robust dependent testability is demonstrated by experimental results performed on benchmark circuits.

*This work has been supported by DFG grants Be 1176/8-2 (H. Hengster) and SFB 124 – "VLSI Entwurfsmethoden und Parallelität" (H. Hengster and U. Sparmann) and by von Humboldt Foundations Senior Research Fellowship (S.M. Reddy).

1 Introduction

A *local transformation* is the substitution of a subcircuit by a different realization. Such transformations are often used in logic synthesis to optimize a given design with respect to the current goal of optimization. Besides the optimization of area, delay, and power consumption one of the most important aspects during logic synthesis is testability.

Therefore, testability properties of local transformations have also been examined considerably in the literature. In [1] a synthesis approach for area optimization was presented and it was shown that the proposed transformations preserve stuck-at fault testability, i.e. if the original circuit is testable then the transformed circuit is testable.

For high speed circuitry not only logical faults but also delay faults have to be considered to check the correct behavior of the circuit. This led to the definition of the path delay fault model [2]. Initially, it was assumed that all possible path delay faults in a circuit have to be checked by *robust tests* [3] to ascertain correct timing behavior of the circuit. In this paper we call robust path delay fault testability of a circuit *complete path* (CP) testability, if the ratio of robustly testable paths to all paths in the circuit is considered. It has already been observed in [4] that CP testability is quite poor for typical combinational circuits. In [5, 6] it has been shown that all but one of the local transformations of [1] also preserve CP testability, i.e. if all paths in the original circuit are robustly testable then all paths in the transformed circuit are robustly testable. Some of these transformations even improve CP testability [5]. A tool using local transformations to improve testability was presented in [7]. In [8] similar local transformations were suggested to improve testability.

Although CP testability can be improved considerably by local transformations it is still very low for most circuits. One source for low testability is the assumption that all possible path delay faults have to be considered. But it has been shown in [9] and [10] that the set of *robust dependent* (RD) path delay faults need not be considered in order to ascertain correct timing behavior of a circuit. Thus, it is only necessary to robustly test the non-RD paths. We call path delay fault testability of a circuit $\overline{\text{RD}}$ testability, if the ratio of robustly testable paths to *non-RD* paths in the circuit is considered. Experimental results have shown that $\overline{\text{RD}}$ testability results in higher fault coverages than CP testability. A heuristic to determine RD paths which is applicable to large circuits was presented in [11].

In our paper we examine the effect of local transformations upon $\overline{\text{RD}}$ testability. We present sufficient conditions which ensure that a local transformation preserves $\overline{\text{RD}}$ testability, i.e. if all non-RD paths in the original circuit are robustly testable then all non-RD paths in the transformed circuit are robustly testable. These conditions are more complex than the conditions for CP testability, because the supposition that all path delay faults of the original circuit are testable does not hold for $\overline{\text{RD}}$ testability. Nevertheless, the presented conditions also refer only to the subcircuits describing a local transformation. Once the conditions are verified for a local transformation this transformation can be performed at any location in any circuit without deteriorating $\overline{\text{RD}}$ testability of the circuit. Furthermore, the transformed circuit is tested by the same test set as the original circuit. Thus, there is no need to construct new test sets by pattern generation after local transformations have been applied.

After the conditions for $\overline{\text{RD}}$ testability are presented, we show by application of these conditions that all but one of the local transformations of [1] preserve $\overline{\text{RD}}$ testability.

Considering $\overline{\text{RD}}$ testability instead of CP testability more realizations of a function are fully testable. This larger search space can be helpful if other optimization goals like area, delay, and power consumption are optimized simultaneously with testability (cf. Example 4).

Most practically used circuits are neither fully testable with respect to CP testability nor with respect

to $\overline{\text{RD}}$ testability. It has already been observed in [7] that local transformations improve CP testability. We give experimental results which demonstrate that application of local transformations to benchmark circuits does also improve $\overline{\text{RD}}$ testability.

The paper is structured as follows: In the next section some basic definitions are reviewed. In Section 3 the sufficient conditions ensuring $\overline{\text{RD}}$ testability preservation are presented and these conditions are applied to some local transformations. The effect of local transformations on benchmark circuits is examined in Section 4. The paper is concluded in Section 5.

2 Basic Definitions

For simplicity of presentation we will restrict to single output combinational circuits in the following. For multi output circuits the theory is applied for each output cone separately.

The following definitions concerning delay testing are taken from [11]. A combinational circuit C consists of leads and gates. As gate types we will consider simple gates (AND, OR, NAND, NOR, and NOT), as well as primary inputs (PIs) and primary outputs (POs). A lead is connecting two gate pins with each other.

A *physical path* $p = (g_0, l_0, g_1, \ldots, l_{m-1}, g_m)$ in C is an alternating sequence of gates and leads leading from a PI g_0 to a PO g_m of C. For simplicity we will denote a path only by the leads which uniquely determine the path. Inputs of gates on the path which are not connected to a lead l_i of the path are called *side inputs*. As usual in delay fault test generation we will associate two *logical paths* with each physical path. A logical path is given as a tuple (t, p) with $t = \overline{x} \rightarrow x,\ x \in \mathbf{B} = \{0, 1\}$, being a transition at the PI of p.

A manufactured implementation of circuit C is denoted by C_m. We assume that C_m has the gate level structure of C, but the delays of its gates and lines may take arbitrary values due to variations of the fabrication process or manufacturing defects. Let τ be the required operation time (clock period) for C_m. C_m is said to have a *delay fault* $> \tau$ on logical path (t, p) iff its delay for propagating t over p exceeds τ. A *robust test* for (t, p) [3] is a two pattern sequence which can be used to measure the delay of (t, p) in any implementation C_m of C. From the fact that C_m does (does not) operate correctly for a robust test under clock period τ it can be concluded that the delay of C_m for propagating t over p is $\leq \tau$ ($> \tau$).

In [9, 10] it has been shown that in order to guarantee that the delay of a manufactured circuit meets its specification not all logical paths need to be checked robustly.

Definition 1 ([9])
Let $\mathcal{LP}(C)$ be the set of all logical paths in a circuit C and $\mathcal{R}$ a subset of $\mathcal{LP}(C)$.

$\mathcal{R}$ is said to be a robust dependent set (RD-set) *iff for all implementations C_m of C and all clock periods τ the following holds: The absence of delay faults $> \tau$ on the logical paths from $\mathcal{LP}(C) \setminus \mathcal{R}$ implies the delay of C_m is $\leq \tau$.*

From Definition 1 and the above stated property of robust tests it follows directly: For verifying that the delay of C_m does not exceed the clock period it is sufficient to check all non-RD paths ($\mathcal{LP}(C) \setminus \mathcal{R}$) with robust tests.

Definition 2
A set of two pattern sequences is called a full $\overline{\text{RD}}$ test set *for a circuit C iff it contains a robust test for every logical path in $\mathcal{LP}(C) \setminus \mathcal{R}$ for some RD-set $\mathcal{R}$.*

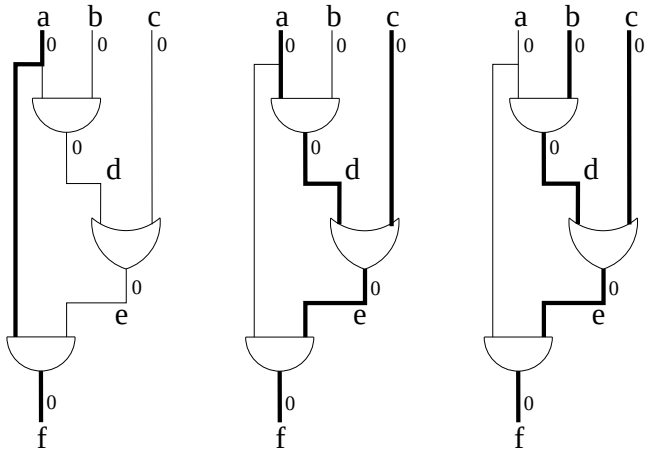


Figure 1: Stabilizing systems S_1, S_2, S_3 for input vector $\langle 000 \rangle$

In the following we briefly review basic definitions of [11] which are important for the understanding of the following theorems (cf. Examples 1 and 2).

Definition 3

i) For circuit C and input vector $v \in \mathbf{B}^n$ a stabilizing system *S is a minimal subcircuit of C such that S stabilizes the* Po *of C on its final (stable) value $C(v)$ independent of the circuitry of C not included in S.*

ii) The set of logical paths included in stabilizing system S for input vector v is denoted by $\mathcal{LP}(v, S)$.

iii) The set of all possible stabilizing systems for circuit C and for input vector v is denoted by $\mathcal{S}(v, C)$.

There can exist many different stabilizing systems for one input vector v. An algorithm to compute the stabilizing systems was presented in [11]. Note, that a stabilizing system normally only constitutes a small portion of the overall circuit.

Example 1 *The stabilizing systems for a circuit C and input vector $v = \langle 000 \rangle$ are shown by bold lines in Figure 1. Each path drawn with bold lines represents the logical path with a transition switching to the final value given at the* Pi *of the path. There exist exactly three stabilizing systems S_1, S_2, and S_3 for input vector $v = \langle 000 \rangle$. The logical paths included in the stabilizing systems are: $\mathcal{LP}(v, S_1) = \{(1 \rightarrow 0, \mathtt{af})\}$, $\mathcal{LP}(v, S_2) = \{(1 \rightarrow 0, \mathtt{adef}), (1 \rightarrow 0, \mathtt{cef})\}$, $\mathcal{LP}(v, S_3) = \{(1 \rightarrow 0, \mathtt{bdef}), (1 \rightarrow 0, \mathtt{cef})\}$. The set of all possible stabilizing systems for $v = \langle 000 \rangle$ is $\mathcal{S}(v, C) = \{S_1, S_2, S_3\}$.*

To obtain the final (stable) value 0 at Po `f` *of circuit C for input vector $v = \langle 000 \rangle$ it suffices to switch the paths included in one stabilizing system. For example in stabilizing system S_1 it suffices that a falling transition at* Pi `a` *travels along path* `af` *in order to guarantee that the* Po `f` *obtains the final (stable) value 0. To ensure that* `f` *obtains the final (stable) value 0 in S_2 it is not sufficient that a falling transition at* Pi `a` *travels along the path* `adef`*. This is, because the output* `e` *of the* Or*-gate evaluates to the value 0 only if both inputs* `d` *and* `c` *are 0. Therefore, also the falling transition at* Pi `c` *must travel along the path* `cef` *to guarantee the final value 0 at the* Po `f`*.*

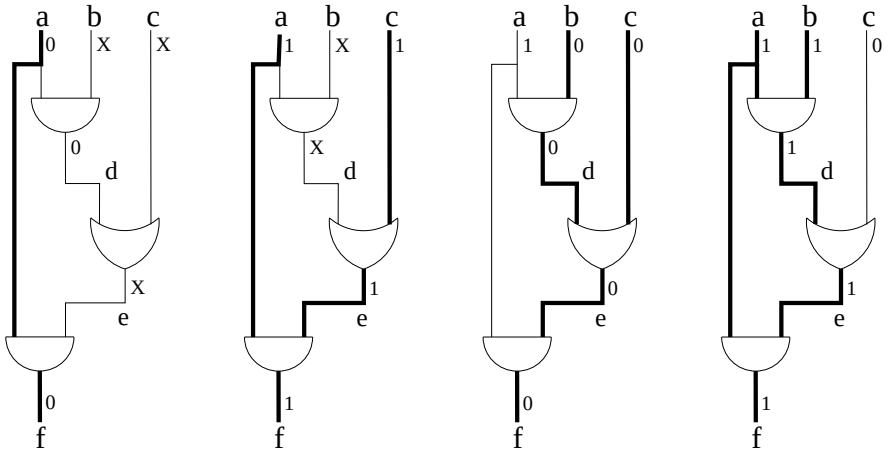


Figure 2: Complete stabilizing assignment

Definition 4

i) *A* complete stabilizing assignment σ *is a mapping which assigns to each input vector* $v \in \mathbf{B}^n$ *a stabilizing system, denoted by* $\sigma(v)$.

ii) *The set of all logical paths included in the stabilizing systems of a complete stabilizing assignment* σ *is denoted by* $\mathcal{LP}(\sigma)$.

Example 2 *To define a complete stabilizing assignment* σ *for circuit* C *of Figure 1 one stabilizing system has to be selected for every input vector* $v \in \mathbf{B}^3$. *In Figure 2 such a complete stabilizing assignment* σ *is given. An* `X` *at a signal symbolizes a don't care, i.e. the values* 1 *or* 0 *can be substituted for each* `X`. *For example, the leftmost stabilizing system was selected for the input vectors* $\langle 000 \rangle$, $\langle 001 \rangle$, $\langle 010 \rangle$, *and* $\langle 011 \rangle$. *The set* $\mathcal{LP}(\sigma)$ *of all logical paths included in the stabilizing systems of* σ *consists of all logical paths of circuit* C *except* $(1 \rightarrow 0, \texttt{adef})$.

Now, we can present the theorem from [11] which shows that it is sufficient to only test the logical paths from $\mathcal{LP}(\sigma)$ robustly.

Theorem 1 ([11])
Let $\mathcal{LP}(C)$ *denote the set of all logical paths in* C *and consider an arbitrary complete stabilizing assigment* σ *for* C. *Then* $\mathcal{LP}(C) \setminus \mathcal{LP}(\sigma)$ *is an RD-set.*

The formal proof of the theorem can be found in [11]. It is based on the fact that it suffices to check the logical paths of one stabilizing system for each input vector of the circuit. The selection of the stabilizing systems is given by the complete stabilizing assignment σ.

Example 3 *The logical path* $(1 \rightarrow 0, \texttt{adef})$ *was not included in any stabilizing system of* σ *in Example 2. This is the only path of circuit* C, *which is not robustly testable. Since all other logical paths are robustly testable, we conclude that circuit* C *is fully* $\overline{RD}$ *testable.*

The following example shows that it is crucial to consider $\overline{\text{RD}}$ testability instead of CP testability, if the objective of a syntesis tool is to obtain fully delay testable and area optimal circuits.

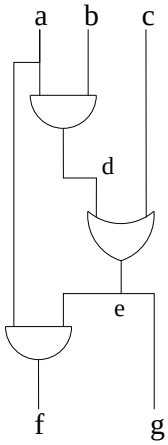


Figure 3: Fully $\overline{\text{RD}}$ testable and area optimal realization

Example 4 *In Figure 3 a circuit C' is depicted which is obtained from circuit C of Example 1 by connecting signal* `e` *to an additional* PO `g`.

Analogously to Example 2 a complete stabilizing assigment can be constructed which does not include the logical path $(1 \rightarrow 0, \texttt{adef})$. *Again, this is the only path which is not robustly testable. Thus, circuit C' is fully $\overline{RD}$ testable. It can be shown that any realization, which is fully CP testable (i.e. all path delay faults are robustly testable), needs more gates to implement* `f` *and* `g` *simultaneously.*

3 Local Transformations

A *local transformation* is the substitution of a subcircuit by a new realization which has better properties with respect to the current goal of optimization. Local transformations are often used in logic synthesis to improve the area and the depth of a given design while testability properties are retained. In [1] an algorithm for multi-level logic synthesis based on local transformations has been presented. In this paper it was also shown that these transformations preserve stuck-at fault testability. In [5, 6] it has been shown that all but one of these local transformations preserve CP testability. Some of these transformations even improve CP testability [5]. In [7] SALT (System for Application of Local Transformations) was introduced and this tool was used to improve CP testability of circuits by local transformations.

In the following we will present a theoretical framework for local transformations with respect to $\overline{\text{RD}}$ testability. Sufficient conditions are given, which ensure that a local transformation preserves $\overline{\text{RD}}$ testability, i.e. if all non-RD paths in the original circuit are robustly testable then all non-RD paths in the transformed circuit are robustly testable.

Definition 5
Let C_1 and C_2 be two single output circuits with the same input sequence which realize the same function. $\pi : \mathcal{LP}(C_1) \longrightarrow \mathcal{LP}(C_2)$ *is a* path mapping *iff for all paths $p \in \mathcal{LP}(C_1)$ the following holds:*
p and $\pi(p)$ have the same transition and start at corresponding PIs.

A path mapping for the circuits describing the local transformation is used to derive stabilizing systems for the transformed circuit from the stabilizing systems of the original circuit. These stabilizing systems are used to argue about $\overline{\text{RD}}$ testability of the transformed circuit.

Theorem 2
Let T be a full $\overline{RD}$ test set for N_1 and let C_1 be a subcircuit of N_1 with m inputs and 1 output. T is a full $\overline{RD}$ test set for the circuit N_2, which is obtained from N_1 by substituting C_2 for C_1, if conditions (i) - (iii) *hold for circuit C_2.*

(i) *C_2 has the same input sequence and realizes the same function as C_1.*

(ii) *There exists a path mapping* $\pi : \mathcal{LP}(C_1) \longrightarrow \mathcal{LP}(C_2)$:
$\forall\, v \in \mathbf{B}^m : \forall\, S_1 \in \mathcal{S}(v, C_1) : \exists\, S_2 \in \mathcal{S}(v, C_2)$:
(ii-a) $\mathcal{LP}(v, S_2) \subseteq \{\pi(p) \mid p \in \mathcal{LP}(v, S_1)\}$ *and*
(ii-b) $\forall\, p \in \mathcal{LP}(v, S_1)$: *Any robust test for p is a robust test for $\pi(p)$.*

(iii) *If a two pattern sequence t produces a stable value at the output of C_1 then t produces a stable value at the output of C_2.*

Conditions (i) and (iii) are identical to the conditions, which were presented in [5] for CP testability. Compared to the conditions for CP testability condition (ii) is more complex. Intuitively, it allows to choose for each stabilizing system S_1 to circuit C_1 a stabilizing system S_2 to C_2 such that the following holds: Each full delay test for the logical paths of S_1 is also a full delay test for the logical paths of S_2.

Proof:
We will show how to construct stabilizing systems for all input vectors of N_2 and then demonstrate that all paths included in these stabilizing systems are robustly testable by T. According to Theorem 1 the paths not included in the stabilizing systems form an RD-set. Thus, according to Definition 2 the set T is also a full $\overline{\text{RD}}$ test set for circuit N_2.

Consider an arbitrary input vector $v' \in \mathbf{B}^n$ for the circuit N_2, which clearly is also an input vector for N_1. We will show how a stabilizing system for circuit N_2 and input vector v' can be constructed. Since T is a full $\overline{\text{RD}}$ test set for N_1 a stabilizing system $S_1' \in \mathcal{S}(v', N_1)$ exists, such that T contains a robust test for every logical path in $\{p_1', p_2', \ldots, p_k'\} = \mathcal{LP}(v', S_1')$. Let $t_i' \in T$ be the robust test for logical path p_i' $(1 \leq i \leq k)$.
We distinguish two cases:

Case 1 : No logical path in $\mathcal{LP}(v', S_1')$ runs through C_1

It is guaranteed by condition (i) that the value at the output of circuit C_1 is equal to the value at the output for the alternative realization C_2. We conclude that all values in circuit N_2 outside of C_2 are also identical to the values of N_1 outside of C_1. Thus, the stabilizing system S_1' is also a stabilizing system for circuit N_2. We only have to show that all logical paths are robustly testable.

If there does not exist any path from the output of circuit C_1 leading to a side input of a logical path p_i' the proof is trivial, since the transformation does not influence the paths to be tested. Otherwise, the output of C_1 leads to at least one logical path p_i'. Let t_i^* be the values at the inputs of C_1 which are induced by the two pattern sequence t_i'.
Assume t_i^* induces a static value at the output of C_1, i.e. the output remains stable during application of both input vectors of the two pattern sequence t_i^*. Then t_i^* induces the same static value at the output of C_2 because of (i) and (iii). Thus, all values in N_2 outside C_2 are identical to the values in N_1 outside C_1 and t_i' is a robust test for p_i' in N_2.
Otherwise, a transition or a hazard may occur at the output of C_1, i.e. the output may change during application of the two input vectors. Since t_i' is a robust test for p_i' the effect of this hazard cannot invalidate the robustness of the test in N_1. Thus, a hazard at the output of C_2 cannot invalidate the test for p_i' in N_2.

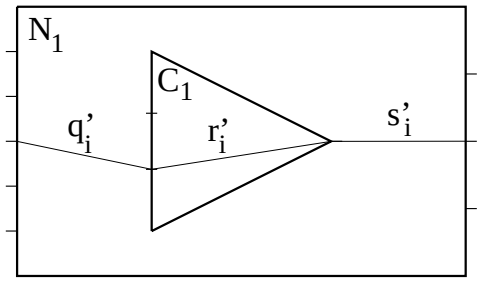


Figure 4: Logical path segments of p'_i

Case 2 : Logical paths $p'_1, p'_2, \ldots, p'_l$ **do** and $p'_{l+1}, p'_{l+2}, \ldots, p'_k$ **do not** run through C_1

Since the paths $p'_1, p'_2, \ldots, p'_l$ pass through C_1 in N_1 they no longer exist after the local transformation is applied. In the following we describe how the path mapping π can be used to transform these paths to new paths for circuit N_2. Together with $p'_{l+1}, p'_{l+2}, \ldots, p'_k$ these new paths form a stabilizing system for N_2.
Each logical path p'_i running through C_1 is subdivided into 3 segments (cf. Figure 4):

q'_i: starts from a PI of N_1 and leads to an input of C_1

r'_i: starts from an input of C_1 and leads to the output of C_1

s'_i: starts from the output of C_1 and leads to a PO of N_1

Let v^* be the input vector for circuit C_1, which is induced by the input vector v' for circuit N_1. The logical path segments $r'_1, r'_2, \ldots, r'_l$ form a stabilizing system $S_1^* \in \mathcal{S}(v^*, C_1)$, which stabilizes the output of C_1 to a stable value, which again is necessary to stabilize the output of N_1. It follows from (ii) that a stabilizing system $S_2^* \in \mathcal{S}(v^*, C_2)$ exists which fulfills conditions (ii-a) and (ii-b). Condition (ii-a) and Definition 5 guarantee that the inputs of C_2 which are included in S_2^* are a subset of the corresponding inputs of C_1 which are included in S_1^*.
Then a stabilizing system $S'_2 \in \mathcal{S}(v', N_2)$ can be constructed as follows:

- Include the logical path segments $s'_1, s'_2, \ldots, s'_l$ in S'_2
- Include S_2^* in S'_2
- Include the logical path segment q'_i in S'_2, iff it leads to an input of C_2 which is included in S_2^*
- Include the logical paths $p'_{l+1}, p'_{l+2}, \ldots, p'_k$ in S'_2

i.e. S'_2 is obtained from S'_1 by replacing S_1^* by S_2^* and leaving away those paths of S'_1 leading to inputs of C_2 which are not included in stabilizing system S_2^*.
Now, we have to show, that all paths in S'_2 are robustly testable by T.

Consider any logical path p^* in S'_2 running through C_2. p^* is subdivided into 3 logical path segments q^*, r^*, and s^* analogously to the subdivision of the paths in N_1 running through C_1. By construction of S'_2 there exists some $j \in \{1, 2, \ldots, l\}$ such that $r^* = \pi(r'_j)$, $q^* = q'_j$ and $s^* = s'_j$. Let t'_j be a robust test in T for the path p'_j and t_j^* be the values induced at the inputs of C_1. t_j^* is a robust test for $r^* = \pi(r'_j)$

$A_1B_1 + \ldots + A_1B_m + \ldots$ $+A_nB_1 + \ldots + A_nB_m$	$\stackrel{\mathcal{T}_F}{\Longrightarrow}$	$(A_1 + \ldots + A_n)(B_1 + \ldots + B_m)$
$a_1 \cdot a_2 \cdot \ldots \cdot a_n$	$\stackrel{\mathcal{T}_M}{\Longrightarrow}$	$\overline{\overline{a_1} + \overline{a_2} + \ldots + \overline{a_n}}$
$(ab)A_1 + (ab)A_2 + \ldots + (ab)A_n$	$\stackrel{\mathcal{T}_1}{\Longrightarrow}$	$CA_1 + CA_2 + \ldots + CA_n$ with $C = ab$
$A_1B_1 + A_1B_2$	$\stackrel{\mathcal{T}_2}{\Longrightarrow}$	$A_1(B_1 + B_2)$
$abA + \bar{a}B + \bar{b}B$	$\stackrel{\mathcal{T}_3}{\Longrightarrow}$	$MA + \overline{M}B$ with $M = ab$
$(a\bar{b} + \bar{a}b)A + (\bar{a}\bar{b} + ab)B$	$\stackrel{\mathcal{T}_4}{\Longrightarrow}$	$MA + \overline{M}B$ with $M = a\bar{b} + \bar{a}b$
$\bar{a}bA + acA + \bar{a}\bar{b}B + a\bar{c}B$	$\stackrel{\mathcal{T}_5}{\Longrightarrow}$	$MA + \overline{M}B$ with $M = \bar{a}b + ac$

Table 1: Local transformations

because of condition (ii-b). The transition is propagated along q^* in N_2, since N_1 and N_2 are identical outside of C_1 and C_2, respectively. It is propageted along s^* since C_2 has only one output where the same transition as at the output of C_1 occurs and thus the values at all side inputs of s^* in N_2 are identical to the values at the side inputs of s'_j in N_1.

It follows analogously to Case 1 that all paths $p'_{l+1}, p'_{l+2}, \ldots, p'_k$, which are not running through C_2 are also robustly testable. ■

Remark 1
In condition (ii) *it is not necessary to consider all stabilizing systems of* $\mathcal{S}(v, C_1)$. *A stabilizing system* S_1 *for the subcircuit* C_1 *need not to be examined if some paths in* $\mathcal{LP}(v, S_1)$ *are not robustly testable, because any stabilizing system for* N_1, *which contains* S_1 *as a subsystem, would also include untestable paths. But all logical paths of all stabilizing systems selected by the complete stabilizing assignment for* N_1 *have to be robustly testable, since* T *is a full* $\overline{RD}$ *test set for circuit* N_1.

Since the conditions of Theorem 2 refer only to the substituted region of the circuit it suffices to examine only the representation of the subcircuits. Once the conditions are verified for a given local transformation this transformation can be performed at any location in any circuit without deteriorating $\overline{\text{RD}}$ testability of the circuit. Additionally, the transformed circuit is tested by the same test set as the original circuit. Thus, there is no need to construct new test sets by pattern generation after local transformations have been applied.

Table 1 shows some local transformations which are wide-spread in logic synthesis. The circuit realizations are given in Figure 5 and Appendix A. The first and second local transformations describe factorization and De Morgan's law, respectively. The remaining five transformations correspond to different kinds of cube extraction as introduced in [1]. Note, that transformation $\mathcal{T}_2$ is a special case of factorization.

The conditions of Theorem 2 are applied to these local transformations. Their effect upon $\overline{\text{RD}}$ testability is described by the following lemma:

Lemma 1

a) The application of transformations $\mathcal{T}_F$, $\mathcal{T}_M$, and $\mathcal{T}_1$ – $\mathcal{T}_4$ to a subcircuit in a fully $\overline{RD}$ testable circuit preserves $\overline{RD}$ testability. Any full test set for the original circuit is also a full test set for the transformed circuit.

b) The application of transformation $\mathcal{T}_5$ to a subcircuit in a fully $\overline{RD}$ testable circuit does not preserve $\overline{RD}$ testability in general.

Proof:
a) The proofs are very technical and very similar for all local transformations. If the circuits describing the local transformations and the path mappings of Appendix A are used, it is possible to develop a program, which automatically verifies the conditions of Theorem 2.
Therefore, we restrict ourselves to one example. We consider transformation $\mathcal{T}_3$, because the proof for this transformation is the most interesting one.

In Figure 5 transformation $\mathcal{T}_3$ is depicted. The circuit at the left-hand side is called $\mathcal{T}_3^a$ and the other one is called $\mathcal{T}_3^b$ in the following. For both circuits some additional marks are defined to identify all paths. For example `ay` in $\mathcal{T}_3^a$ denotes the path starting at input `a`, running through the upper inverter, passing the middle AND gate and the OR gate.

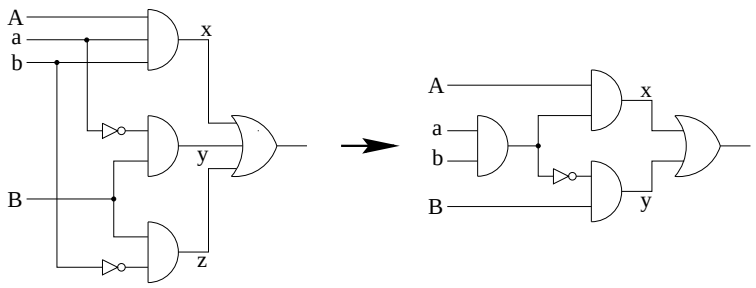


Figure 5: Circuits $\mathcal{T}_3^a$ and $\mathcal{T}_3^b$ describing local transformation $\mathcal{T}_3$

To prove the lemma for transformation $\mathcal{T}_3$ we have to show that conditions (i) - (iii) of Theorem 2 are fulfilled.

Condition (i) : follows from de Morgan's laws and the distributive law.

Condition (ii) : The path mapping π is given in Table 2. For shortness we have denoted only the physical paths. Conditions (ii-a) and (ii-b) have to be shown for all input vectors. Here, we examine the conditions only for input vector $v = \langle 1010 \rangle$ where the values of the inputs are given for the input sequence $\langle \mathtt{A}, \mathtt{a}, \mathtt{b}, \mathtt{B} \rangle$. The other input vectors can be treated analogously.

p	ax	ay	bx	bz	Ax	By	Bz
$\pi(p)$	ax	ay	bx	by	Ax	By	By

Table 2: Path mapping π for transformation $\mathcal{T}_3$

In $\mathcal{S}(v, \mathcal{T}_3^a)$ there exist exactly two stabilizing systems: $S_1^a = \{(1 \to 0, \mathtt{ax}), (1 \to 0, \mathtt{By}), (1 \to 0, \mathtt{Bz})\}$ and $S_2^a = \{(1 \to 0, \mathtt{ax}), (1 \to 0, \mathtt{By}), (0 \to 1, \mathtt{bz})\}$. $\mathcal{S}(v, \mathcal{T}_3^b)$ consists only of the stabilizing system $S_1^b = \{(1 \to 0, \mathtt{ax}), (1 \to 0, \mathtt{By})\}$. The stabilizing systems are shown by bold lines in Figure 6. Using the path mapping π in Table 2 it is easy to see that $\{\pi(p) \mid p \in \mathcal{LP}(v, S_1^a)\} = \{(1 \to 0, \mathtt{ax}), (1 \to 0, \mathtt{By})\}$ and $\{\pi(p) \mid p \in \mathcal{LP}(v, S_2^a)\} = \{(1 \to 0, \mathtt{ax}), (1 \to 0, \mathtt{By}), (0 \to 1, \mathtt{by})\}$ which both contain $\mathcal{LP}(v, S_1^b)$. Hence, condition (ii-a) is fulfilled for $v = \langle 1010 \rangle$.

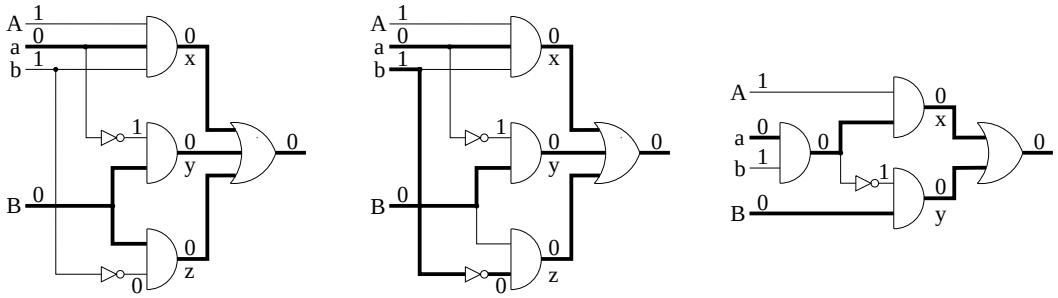


Figure 6: Stabilizing systems S_1^a, S_2^a, and S_1^b

The vector pairs which can be used to test the logical paths in $\mathcal{LP}(v, S_1^a)$ and $\mathcal{LP}(v, S_2^a)$ are given in Table 3. As introduced in Section 2 an `X` symbolizes a don't care. It can be seen that each test for a path p is also a test for path $\pi(p)$. Thus, condition (ii-b) also holds.

logical path	robust tests
$(1 \to 0, \mathtt{ax})$	$\langle\langle \mathtt{111X} \rangle, \langle \mathtt{1010} \rangle\rangle$
$(1 \to 0, \mathtt{By})$	$\langle\langle \mathtt{X0X1} \rangle, \langle \mathtt{X0X0} \rangle\rangle$
$(1 \to 0, \mathtt{Bz})$	$\langle\langle \mathtt{XX01} \rangle, \langle \mathtt{XX00} \rangle\rangle$
$(0 \to 1, \mathtt{bz})$	$\langle\langle \mathtt{XX01} \rangle, \langle \mathtt{0111} \rangle\rangle$

Table 3: Robust tests for paths in $\mathcal{T}_3^a$

Examining the stabilizing systems and the corresonding tests for all other input vectors shows that condition (ii) holds.

Condition (iii) : All two pattern sequences resulting in the stable values 1 and 0 at the output of circuit $\mathcal{T}_3^a$ are given in Table 4. It can be seen, that a stable value at the output of circuit $\mathcal{T}_3^b$ is produced by all of these two pattern sequences.

stable 1	stable 0
$\langle\langle \mathtt{111X} \rangle, \langle \mathtt{111X} \rangle\rangle$	$\langle\langle \mathtt{X0X0} \rangle, \langle \mathtt{X0X0} \rangle\rangle$
$\langle\langle \mathtt{X0X1} \rangle, \langle \mathtt{X0X1} \rangle\rangle$	$\langle\langle \mathtt{XX00} \rangle, \langle \mathtt{XX00} \rangle\rangle$
$\langle\langle \mathtt{XX01} \rangle, \langle \mathtt{XX01} \rangle\rangle$	$\langle\langle \mathtt{0XX0} \rangle, \langle \mathtt{0XX0} \rangle\rangle$
	$\langle\langle \mathtt{011X} \rangle, \langle \mathtt{011X} \rangle\rangle$

Table 4: Two pattern sequences for stable values at $\mathcal{T}_3$

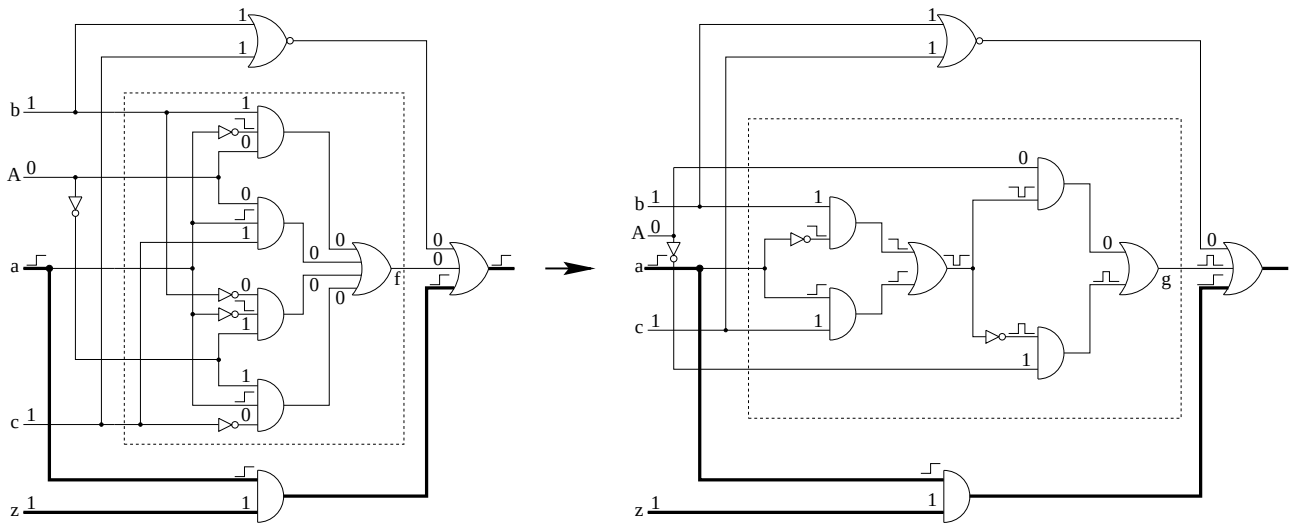


Figure 7: Transformation which destroys $\overline{\text{RD}}$ testability

b) Transformation $\mathcal{T}_5$ does not fulfill condition (iii). An example is given in Figure 7. The dashed boxes show transformation $\mathcal{T}_5$. For each of the circuits in Figure 7 there exists exactly one stabilizing system for the input pattern $\langle 10111 \rangle$ for the input sequence $\langle$`bAacz`$\rangle$, which is shown by bold lines. Thus, in both circuits the path drawn with bold lines and starting at input `a` has to be tested robustly. The only possible robust tests for the path in the left-hand side circuit is $\langle\langle 10011 \rangle, \langle 10111 \rangle\rangle$. But applying this two pattern sequence to the right-hand side circuit a hazard can occur at line `g` and no robust test exists for the path drawn with bold lines after the transformation was applied.
Additionally, it can be shown that it is not possible to find a path mapping for local transformation $\mathcal{T}_5$ such that the conditions of (ii) are fulfilled. We omit details for briefness. ■

Although $\mathcal{T}_5$ does not preserve $\overline{\text{RD}}$ testability in general, the lemma does not prove that testability of a circuit is always destroyed by application of $\mathcal{T}_5$. For many instances of the transformation testability will still be preserved. It is also interesting to note that $\mathcal{T}_5$ was the only transformation which did not preserve CP testability [5, 6].

Furthermore, it should be mentioned that in some cases only a subset of the original $\overline{\text{RD}}$ test set is needed to test the transformed circuit. It is also possible that the transformed circuit is fully $\overline{\text{RD}}$ testable even if the original circuit was not. In this sense the local transformations do even improve testability.

Example 5 *Consider a circuit C^* which contains $\mathcal{T}_3^a$ (cf. Figure 5) as a subcircuit. Assume that for one stabilizing system of C^* a stabilizing subsystem for input $\langle 1010 \rangle$ of $\mathcal{T}_3^a$ has to be selected (cf. Figure 6). We will examine two different situations:*

a) *C^* is fully testable and none of the vector pairs covered by $\langle\langle$`XX01`$\rangle, \langle$`XX00`$\rangle\rangle$ (cf. Table 3) are applicable to subcircuit $\mathcal{T}_3^a$. Then stabilizing system S_2^a must have been selected in $\mathcal{T}_3^a$ because C^* is fully testable. In this case three different robust tests have to be applied to test the paths in S_2^a. After transformation $\mathcal{T}_3$ was used to substitute subcircuit $\mathcal{T}_3^a$ by $\mathcal{T}_3^b$ in C^* only two robust tests are necessary to test the logical paths of the only stabilizing system S_1^b. Thus, the full $\overline{RD}$ test set for C^* can be decreased for the transformed circuit.*

b) *None of the vector pairs covered by* $\langle\langle \mathtt{X0X1}\rangle, \langle \mathtt{X0X0}\rangle\rangle$ *are applicable to subcircuit* $\mathcal{T}_3^a$*. Then circuit* C^* *cannot be fully testable because both stabilizing systems* S_1^a *and* S_2^a *include path* $(1 \to 0, \mathtt{By})$*. After application of transformation* $\mathcal{T}_3$ *the paths of stabilizing system* S_1^b *can be tested robustly by vector pairs covered by* $\langle\langle \mathtt{111X}\rangle, \langle \mathtt{1010}\rangle\rangle$ *and* $\langle\langle \mathtt{X101}\rangle, \langle \mathtt{X100}\rangle\rangle$*. Thus, the transformed circuit can be fully* $\overline{RD}$ *testable.*

In the following section we will show that local transformations considerably improve $\overline{\text{RD}}$ testability for typical combinational circuits.

In addition to the local transformations already presented the absorbtion rule $a \cdot (a + b) = a$ and Boolean reductions like $a \cdot a = a$, $a \cdot \overline{a} = 0$, etc. can improve $\overline{\text{RD}}$ testability and the size of the transformed circuits. Moreover, for every local transformation $\overline{\text{RD}}$ testability properties can be directly transfered to the dual transformation [12].

As motivated in [13] reduction of the number of paths while preserving testability is also very important for delay testing. Since some of the local transformations reduce the number of paths in a circuit while they preserve testability they are well suited for that purpose as well.

4 Experimental Results

In this section we examine the effect of local transformations on $\overline{\text{RD}}$ testability for some ISCAS85 circuits [14] and the combinational parts of some ISCAS89 circuits [15].

In Table 5 we give two rows for each circuit. In the first row we give the statistics for the original benchmark circuit. The second row shows the statistics for the circuits which are optimized by SALT [16]. We applied the same local transformations which were used in [16] to improve CP testability. The results can be improved if the selection among instances of local transformations which exclude one another is directed by special strategies for $\overline{\text{RD}}$ testability.

In the first column of Table 5 the name of the circuit is given. The number of logical paths in the circuit is put in the second column. The third number gives the number of logical non-RD paths determined by the heuristic of [11]. Since the non-RD sets computed by this heuristic are not guaranteed to be minimum, the actual number of non-RD paths might be slightly smaller. The non-RD paths have to be tested by robust tests to verify that the delay of the circuit does not exceed the clock period. The number of logical path, which are testable by robust tests, is given in the fourth column.

The next two columns give the fault coverages with respect to CP testability (CP-FC) and $\overline{\text{RD}}$ testability ($\overline{\text{RD}}$-FC), respectively. CP fault coverage is computed by dividing the number of testable paths by the number of all logical paths in the circuit. To calculate $\overline{\text{RD}}$ fault coverage only the non-RD paths are considered instead of the number of all logical paths. Since the number of non-RD paths has been computed by a heuristic, the values given in column $\overline{\text{RD}}$-FC are lower bounds on the actual fault coverages.

The advantages of local transformations can be seen in Table 5. For CP testability as well as for $\overline{\text{RD}}$ testability the fault coverage is increased by local transformations, i.e. more untestable, non-RD paths than robustly testable paths are removed. For example for circuit c1355 CP fault coverage is increased from 0.27% to 19.60% and $\overline{\text{RD}}$ fault coverage is improved by local transformations from 2.05% to 37.78%. Also for circuits with high initial testability (c0880) local transformations can still improve testability considerably.

In the last two columns the number of gates and the depth is given for each circuit. These numbers are determined for circuit realizations which consist of 2-input gates only. Application of local transformations does not only improve testability but also the depth remains nearly the same and the number of gates is even decreased.

circuit		log. paths	non-RD	testable	CP-FC	$\overline{\text{RD}}$-FC	gates	depth
c0880	orig.	17284	16731	16083	93.05	96.13	346	25
	transf.	15092	14834	14438	95.67	97.33	340	27
c1355	orig.	8346432	1110304	22784	0.27	2.05	518	26
	transf.	682112	353920	133696	19.60	37.78	408	22
c2670	orig.	1359920	243556	15218	1.12	6.25	883	30
	transf.	240932	110214	14417	5.98	13.08	815	28
c3540	orig.	57353342	2872037	88408	0.15	3.08	1270	42
	transf.	42306378	2485872	83986	0.20	3.38	1252	44
c5315	orig.	2682610	433596	81435	3.04	18.78	2079	37
	transf.	2081924	347042	70000	3.36	20.17	1876	37
c7552	orig.	1452988	336502	86250	5.94	25.63	2632	29
	transf.	1152772	266045	74780	6.49	28.11	2478	30
cs00713	orig.	43624	4996	1184	2.71	23.70	198	37
	transf.	35134	4722	1126	3.20	23.85	182	36
cs13207	orig.	2690738	527998	27603	1.02	5.23	3214	36
	transf.	855740	325120	35432	4.14	10.90	3023	33
cs15850	orig.	329476092	13548808	182673	0.06	1.35	3873	47
	transf.	38341164	5692014	211142	0.55	3.71	3659	41
cs38584	orig.	2161446	447497	92239	4.27	20.61	13502	37
	transf.	571982	251428	105038	18.36	41.78	13022	37

Table 5: Effect of local transformations upon testability

5 Conclusions

Sufficient conditions which ensure that a local transformation preserves $\overline{\text{RD}}$ testability have been presented. The conditions refer only to the subcircuits which describe the local transformation. Once the conditions are checked for a local transformation this transformation can be performed at any location in any circuit. A test set for the original circuit will also test the transformed circuit.

Using these conditions we proved for a set of local transformations which are frequently used in logic synthesis that they preserve $\overline{\text{RD}}$ testability. Experimental results showed that these transformations also improve $\overline{\text{RD}}$ testability for circuits which are not fully $\overline{\text{RD}}$ testable.

Although $\overline{\text{RD}}$ testability can be considerably improved by local transformations it is still very poor for several circuits. In some approaches only the paths with an expected delay higher than a given threshold are considered. Of course it is possible to use local transformations favorably if they improve testability of these paths. Additionally, it is possible to obtain 100% CP testable ($\overline{\text{RD}}$ testable) circuits by insertion of test points [17] ([18]). An interesting subject for future work would be to combine our approach with the other approaches to obtain highly testable circuits at a low cost.

References

[1] J. Rajski and J. Vasudevamurthy. The testability - preserving concurrent decomposition and factorization of boolean expressions. *IEEE Trans. on CAD*, 11(6):778–793, 1992.

[2] G.L. Smith. Model for delay faults based upon paths. In *Int'l Test Conf.*, pages 342–349, 1985.

[3] C.-J. Lin and S.M. Reddy. On delay fault testing in logic circuits. *IEEE Trans. on CAD*, 6(5):694–703, 1987.

[4] S.M. Reddy, C.J. Lin, and S. Patil. An automatic test pattern generator for the detection of path delay faults. In *Int'l Conf. on CAD*, pages 284–287, 1987.

[5] H. Hengster, R. Drechsler, and B. Becker. Testability properties of local circuit transformations with respect to the robust path-delay-fault model. In *VLSI Design Conf.*, pages 123–126, 1994.

[6] A. El-Maleh and J. Rajski. Delay-fault testability preservation of the concurrent decomposition and factorization transformations. In *VLSI Test Symp.*, pages 15–21, 1994.

[7] H. Hengster, R. Drechsler, and B. Becker. On the application of local circuit transformations with special emphasis on path delay fault testability. In *VLSI Test Symp.*, pages 387–392, 1995.

[8] I. Pomeranz and S.M. Reddy. On the number of tests to detect all path delay faults in combinational logic circuits. *IEEE Trans. on Comp.*, 45(1):50–62, 1996.

[9] W.K. Lam, A. Saldanha, R.K. Brayton, and A.L. Sangiovanni-Vincentelli. Delay fault coverage and performance tradeoffs. In *Design Automation Conf.*, pages 446–451, 1993.

[10] K.-T. Cheng and H.-C. Chen. Delay testing for non-robust untestable circuits. In *Int'l Test Conf.*, pages 954–961, 1993.

[11] U. Sparmann, D. Luxenburger, K.-T. Cheng, and S.M. Reddy. Fast identification of robust dependent path delay faults. In *Design Automation Conf.*, pages 119–125, 1995.

[12] S. Kundu and A.K. Pramanick. Testability preserving boolean transforms for logic synthesis. In *VLSI Test Symp.*, pages 131–138, 1993.

[13] I. Pomeranz and S.M. Reddy. On synthesis-for-testability of combinational logic circuits. In *Design Automation Conf.*, pages 126–132, 1995.

[14] F. Brglez and H. Fujiwara. A neutral netlist of 10 combinational circuits and a target translator in fortran. In *Int'l Symp. Circ. and Systems, Special Sess. on ATPG and Fault Simulation*, pages 663–698, 1985.

[15] F. Brglez, D. Bryan, and K. Kozminski. Combinational profiles of sequential benchmark circuits. In *Int'l Symp. Circ. and Systems*, pages 1929–1934, 1989.

[16] H. Hengster, R. Drechsler, and B. Becker. On local transformations and path delay fault testability. *Jour. of Electronic Testing: Theory and Applications*, 7(3):173–191, 1995.

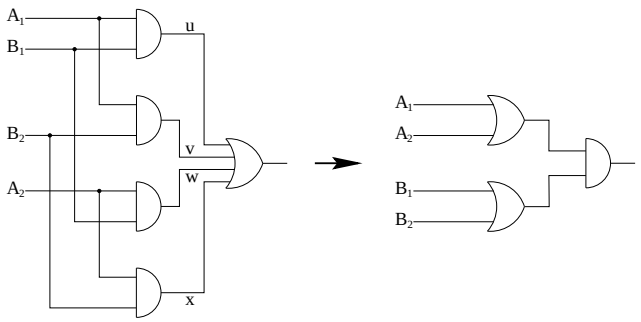
[17] I. Pomeranz and S.M. Reddy. Design-for-testability for path dealy faults in large combinational circuits using test-points. In *Design Automation Conf.*, pages 358–364, 1994.

[18] P. Uppaluri, U. Sparmann, and I. Pomeranz. On minimizing the number of test points needed to achieve complete robust path delay fault testability. In *VLSI Test Symp.*, pages 288–295, 1996.

A Local Transformations and Path Mappings

For the local transformations presented in Table 1 the circuit realizations and the path mappings are given in the following figures.

For parameterized local transformations the circuit realizations and path mappings are given for fixed values. Local transformation $\mathcal{T}_3$ was already examined in Section 3. The circuit realization was depicted in Figure 5 and the path mapping was given in Table 2. As shown in Lemma 1 local transformation $\mathcal{T}_5$ does not preserve $\overline{\text{RD}}$ testability in general. Thus, there is no path mapping given for $\mathcal{T}_5$.

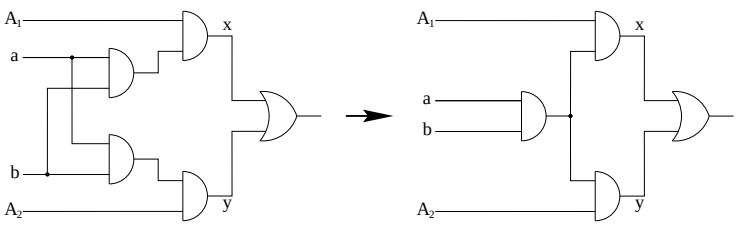


p	A_1u	A_1v	A_2w	A_2x	B_1u	B_1w	B_2v	B_2x
$\pi(p)$	A_1	A_1	A_2	A_2	B_1	B_1	B_2	B_2

Figure 8: Local Transformation $\mathcal{T}_F$ for $n = 2$ and $m = 2$

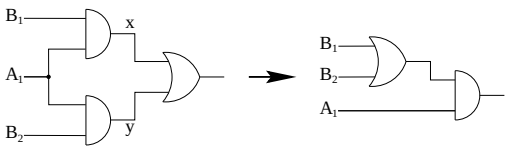
p	a_1	a_2
$\pi(p)$	a_1	a_2

Figure 9: Local Transformation $\mathcal{T}_M$ for $n = 2$



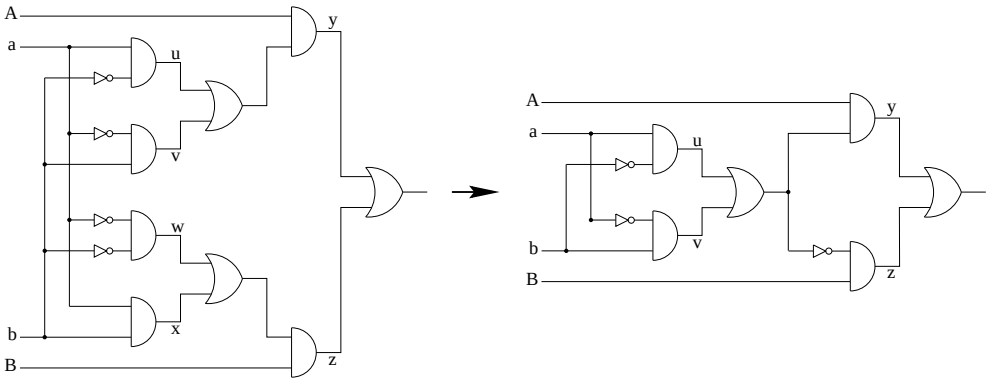
p	ax	ay	bx	by	A_1x	A_2y
$\pi(p)$	ax	ay	bx	by	A_1x	A_2y

Figure 10: Local Transformation $\mathcal{T}_1$



p	$\mathtt{A_1x}$	$\mathtt{A_1y}$	$\mathtt{B_1x}$	$\mathtt{B_2y}$
$\pi(p)$	$\mathtt{A_1}$	$\mathtt{A_1}$	$\mathtt{B_1}$	$\mathtt{B_2}$

Figure 11: Local Transformation $\mathcal{T}_2$



p	auy	avy	awz	axz	buy	bvy	bwz	bxz	Ay	Bz
$\pi(p)$	auy	avy	auz	avz	buy	bvy	bvz	buz	Ay	Bz

Figure 12: Local Transformation $\mathcal{T}_4$

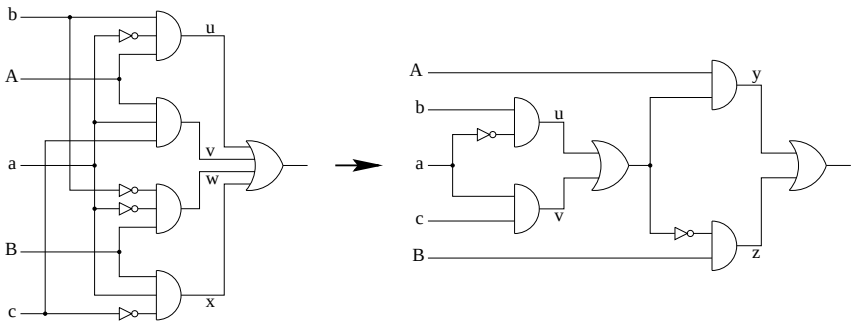


Figure 13: Local Transformation $\mathcal{T}_5$