# Synthesis for Testability: Polynomial Test Pattern Generation for KFDD Circuits

Martha Schnieber
University of Bremen
Bremen, Germany
schnieber@uni-bremen.de

Rolf Drechsler
University of Bremen/DFKI
Bremen, Germany
drechsler@uni-bremen.de

***Abstract*—Today, circuits are used in various safety-critical systems, therefore yielding a high demand for reliable systems. Consequently, a lot of research is conducted to improve the testability of designs by increasing the test coverage or reducing the required test time. In this context, provable computational bounds are crucial to ensure fast test pattern generation. Previous works proved polynomial test set generation for *Binary Decision Diagram* (BDD) circuits. Later, the testability of *Kronecker Functional Decision Diagrams* (KFDDs) was assessed, as KFDDs can exponentially reduce the required logic. However, the test set generation for KFDD circuits generally requires exponential resources. In this paper, we present a technique to derive circuits from KFDDs, for which a complete test set under the *Cellular Fault Model* (CFM), as well as the *Stuck-At Fault Model* (SAFM), can be generated within polynomial resources. The derived circuit is linear in size regarding the KFDD size, and in contrast to previously derived KFDD circuits, no redundant faults can occur, yielding fully testable circuits under CFM and SAFM. In our evaluation, the complete test set generation was up to 50 times faster than the previous exponential method, clearly showcasing the advantages of our polynomial approach.**

***Index Terms*—Synthesis for Testability, Kronecker Functional Decision Diagrams, Logic Synthesis**

## I. Introduction

Nowadays, *Integrated Circuits* (ICs) are omnipresent in our daily life, e.g. in smartphones, cars, etc. To address the growing functional requirements, the complexity of ICs constantly increases, where cost-effective realizations require a small circuit design. As a result, the risk for manufacturing faults steadily increases, leading to faulty chips. The resulting malfunctions can lead to tremendous damage, especially in safety-critical systems. Thus, a lot of research is conducted in the area of synthesis for testability to design circuits with a high test coverage and a reduced test time [1].

A circuit design for which the testability has been thoroughly researched is circuits derived from BDDs. While BDDs are typically used for the formal verification of circuits, they can also be converted into a circuit by replacing every node with a multiplexer. The resulting circuit can be verified in polynomial time regarding the circuit size [2] and provides excellent testability properties. First, it has been shown that all redundancies (non-testable faults) and a complete test set can be computed within polynomial resources and that full testability under CFM, SAFM and the *Path Delay Fault Model* (PDFM) can be achieved if certain redundancies don't occur within the circuit [3]. Later, the BDD circuit was adapted by adding one input and one inverter such that all redundancies are eliminated, therefore providing fully testable BDD circuits under CFM, SAFM and PDFM [4].

However, BDD circuits can generally have an exponential size, as the underlying BDDs can be exponential. For some functions, other data structures, such as KFDDs, can be exponentially smaller than BDDs [5]. As circuits can also be derived from KFDDs, a KFDD circuit can require a significantly smaller area than the respective BDD circuit, therefore overcoming drawbacks from the BDD circuits while maintaining the property of being verifiable in polynomial time [6]. The testability of KFDD circuits has also been studied before, where it has been shown that all redundancies can be computed and a full testability under CFM and SAFM can be guaranteed if certain redundancies don't occur within the circuit [7]. However, in contrast to BDD circuits, the computation of the redundancies and the test set cannot be done in polynomial time, but is generally exponential with respect to the underlying KFDD size.

In this paper, we therefore propose a synthesis for testability approach for the original KFDD circuits, such that a complete test set can be generated in polynomial time regarding the circuit size, while also guaranteeing full testability under CFM and SAFM. Therefore, the proposed method simultaneously solves the problem of exponential test pattern generation and removes all redundancies, providing fully testable circuits even if BDD circuits become impractical due to their exponential growth and hence costs. Using the KFDD circuits and test pattern generation method presented in this paper, a polynomial time complexity for the test set generation and a 100% test coverage are guaranteed. The proposed circuit design requires a linear amount of gates regarding the size of the underlying KFDD. Furthermore, the additional inputs needed for testing the design can be realized with a scan chain, therefore requiring only a constant amount of additional inputs. In our evaluation, we could show a significant speedup, where the proposed polynomial method was on average 4.19 times and up to 50.11 times faster than the exponential test set generation of the original KFDD circuit.

Parts of this work have been supported by DFG within the Reinhart Koselleck Project PolyVer: Polynomial Verification of Electronic Circuits (DR 287/36-1).

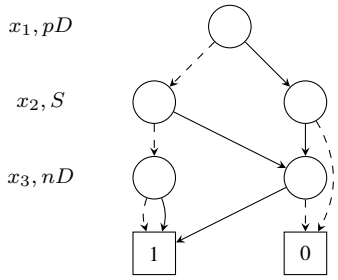



Fig. 1: Example KFDD for the function $f = x_1x_2\overline{x}_3 \oplus x_2 \oplus x_3$

## II. Preliminaries

### A. KFDDs

A KFDD [8], [9] $F$ is a directed acyclic graph representing a function with $n$ inputs $f(x_1, ..., x_n)$. The graph contains a root node and two terminal nodes representing the constant values 0 and 1. Each non-terminal node has two outgoing edges and is associated with a variable $x_i$. In an *Ordered KFDD* (OKFDD), the variables appear in the same order on every path from the root node to a terminal node. Furthermore, a *Reduced OKFDD* (ROKFDD) is minimal, i.e. no node can be removed without altering the function represented by the KFDD. For the remainder of this paper, we assume ordered and reduced KFDDs when referencing KFDDs.

Unlike BDDs [10], where a single decomposition type – i.e. the Shannon decomposition – is used for all variables, the variables in a KFDD can be decomposed according to either the Shannon decomposition (S), the positive Davio decomposition (pD) or the negative Davio decomposition (nD). Let $f_i^0$ be the cofactor of a function $f$, where the variable $x_i$ is replaced with 0, while $f_i^1$ is the cofactor where $x_i$ is replaced with 1. Furthermore, let $f_i^2$ be defined as $f_i^2 = f_i^0 \oplus f_i^1$. The function for a node using the Shannon decomposition is:

$$f = \overline{x}_i f_i^0 + x_i f_i^1 \tag{1}$$

Furthermore, the positive Davio decomposition is computed using the following equation:

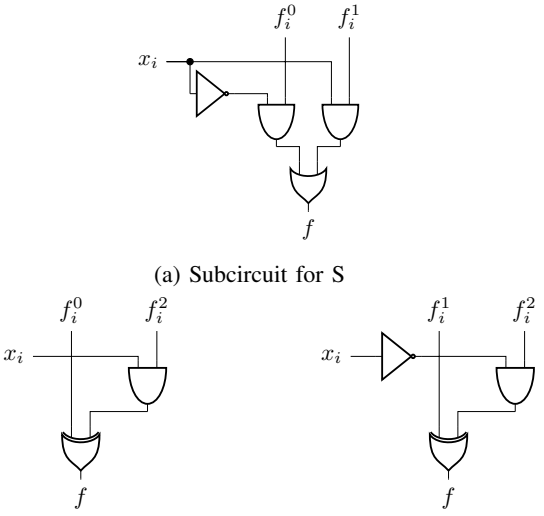
$$f = f_i^0 + x_i f_i^2 \tag{2}$$

Finally, the negative Davio decomposition is defined as:

$$f = f_i^1 + \overline{x}_i f_i^2 \tag{3}$$

For each variable $x_i$, one of the three decomposition types is chosen and defined within the *Decomposition Type List* (DTL). For a given variable ordering, e.g. $(x_1, x_2, x_3)$, the DTL lists the decomposition types used for each variable, e.g. $(pD, S, nD)$. Similar to BDDs, KFDDs are canonical for a given variable ordering and DTL. Due to the addition of the decomposition types pD and nD, KFDDs can have an exponentially reduced size compared to BDDs [5].

In Figure 1, an example of a KFDD for the function $f = x_1x_2\overline{x}_3 \oplus x_2 \oplus x_3$ with the variable ordering $(x_1, x_2, x_3)$ and the DTL $(pD, S, nD)$ is shown.




(a) Subcircuit for S

(b) Subcircuit for pD

(c) Subcircuit for nD

Fig. 2: Subcircuits for S, pD and nD nodes

While operations on BDDs, such as AND, OR, XOR and NOT, can be carried out in polynomial time [10], polynomial upper bounds are not guaranteed for all operations on KFDDs. Let $F$ and $G$ be KFDDs for the functions $f$ and $g$, where $|F|$ and $|G|$ denote the sizes of the KFDDs, i.e. the number of nodes. Similar to BDDs, the KFDD for the XOR operation $f \oplus g$ can be carried out in $\mathcal{O}(|F| \cdot |G|)$. Using complemented edges, the NOT operation can even be carried out within constant time. However, the AND and OR operations on KFDDs have an exponential worst case behavior [5], [9].

### B. KFDD Circuits

To derive a circuit computing a function $f$, a KFDD $F$ can be constructed and then converted into a circuit. To construct a circuit from a KFDD, the input variables $x_1, ..., x_n$ are turned into PIs, the root node is connected to the PO, and the terminal nodes are constant values 0 and 1. Subsequently, each node is replaced by a subcircuit corresponding to the respective decomposition type, as shown in Figure 2. As can be seen from Equation (1), a node with the S decomposition can easily be realized using a multiplexer, i.e an inverter, two AND gates and one OR gate (see Figure 2(a)). Furthermore, the pD decomposition as defined in Equation (2) can be realized with one AND gate and one XOR gate (see Figure 2(b)), while the nD decomposition from Equation (3) additionally requires an inverter (see Figure 2(c)) [11].

The circuit resulting from the application of the described method to the KFDD from Figure 1 is shown in Figure 3. For clarity, the subcircuits for all decomposition types are simplified.

While the optimization of circuits can generally negatively influence their testability by introducing redundancies, some optimizations can safely be made for KFDD circuits. If at least one input to a subcircuit is the constant 0 or 1, the respective subcircuit can be simplified, which we call *degen-*

TABLE I: Degenerated subcircuits for the S, pD and nD decomposition types

| Shannon | | positive Davio | | negative Davio | |
|---|---|---|---|---|---|
| inputs | degenerated subcircuit | inputs | degenerated subcircuit | inputs | degenerated subcircuit |
| $f_i^0 = 0, f_i^1 = 1$ | $f = x_i$ | $f_i^0 = 0, f_i^2 = 1$ | $f = x_i$ | $f_i^1 = 0, f_i^2 = 1$ | $f = \overline{x}_i$ |
| $f_i^0 = 1, f_i^1 = 0$ | $f = \overline{x}_i$ | $f_i^0 = 1, f_i^2 = 1$ | $f = \overline{x}_i$ | $f_i^1 = 1, f_i^2 = 1$ | $f = x_i$ |
| $f_i^0 = 1$ | $f = f_i^1 + \overline{x}_i$ | $f_i^0 = 1$ | $f = \overline{f_i^2 \cdot x_i}$ | $f_i^1 = 1$ | $f = \overline{f_i^2 \cdot \overline{x}_i}$ |
| $f_i^0 = 0$ | $f = f_i^1 \cdot x_i$ | $f_i^0 = 0$ | $f = f_i^2 \cdot x_i$ | $f_i^1 = 0$ | $f = f_i^2 \cdot \overline{x}_i$ |
| $f_i^1 = 1$ | $f = f_i^0 + x_i$ | $f_i^2 = 1$ | $f = f_i^0 \oplus x_i$ | $f_i^2 = 1$ | $f = f_i^1 \oplus \overline{x}_i$ |
| $f_i^1 = 0$ | $f = f_i^0 \cdot \overline{x}_i$ | $f_i^0 = f_i^2$ | $f = f_i^0 \cdot \overline{x}_i$ | $f_i^0 = f_i^2$ | $f = f_i^1 \cdot x_i$ |
| $f_i^0 = \overline{f_i^1}$ | $f = f_i^0 \oplus x_i$ | $f_i^0 = \overline{f_i^2}$ | $f = f_i^0 + x_i$ | $f_i^1 = \overline{f_i^2}$ | $f = f_i^1 + \overline{x}_i$ |

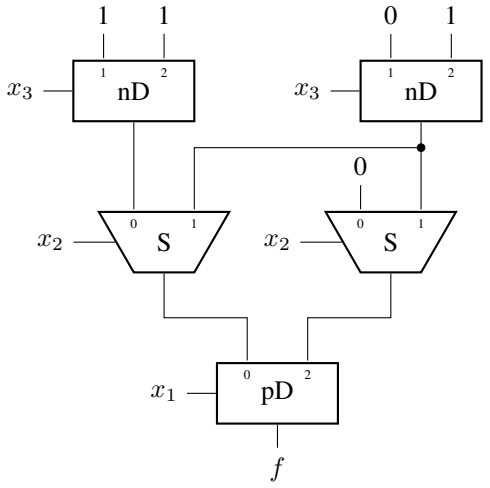



Fig. 3: KFDD circuit for the KFDD shown in Figure 1

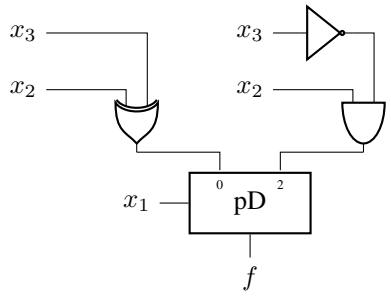



Fig. 4: KFDD circuit with degenerated subcircuits for the KFDD circuit shown in Figure 3

*erated*. Additionally, if both inputs are equal or complements of each other, the subcircuits can be degenerated as well. More precisely, the degenerations in Table I are applied for the respective subcircuits [7]. The circuit with degenerated subcircuits resulting from Figure 3 is shown in Figure 4. The two nD subcircuits are degenerated, as both inputs are constant. Furthermore, the left S subcircuit is degenerated, as the inputs of the subcircuit are complements of each other, while one input of the right S subcircuit is a constant, also resulting in a degenerated subcircuit.

The area of the resulting circuit is linear in the number of nodes in the KFDD $F$, as each subcircuit requires a constant number of at most 4 gates, resulting at most $4 \cdot |F|$ gates for the complete KFDD circuit. Additionally, the delay is linear in the number of variables $n$, as the longest path in a KFDD has at most $n$ nodes, where each node is replaced by a subcircuit with constant delay of at most 3.

In this paper, we consider KFDD circuits over two different libraries. We denote $C_{STD}$ as a KFDD circuit over $STD = \{AND, OR, NOT\}$, i.e. all XOR gates have to be realized using gates from $STD$. Furthermore, we define $C_{KFDD}$ as a KFDD circuit with gates from $KFDDLIB$ consisting of S (see Figure 2(a)), pD (see Figure 2(b)) and nD (see Figure 2(c)) subcircuits, as well as degenerated subcircuits as defined in Table I. Furthermore, $C_{KFDD}$ contains XOR gates needed for our proposed circuit transformation.

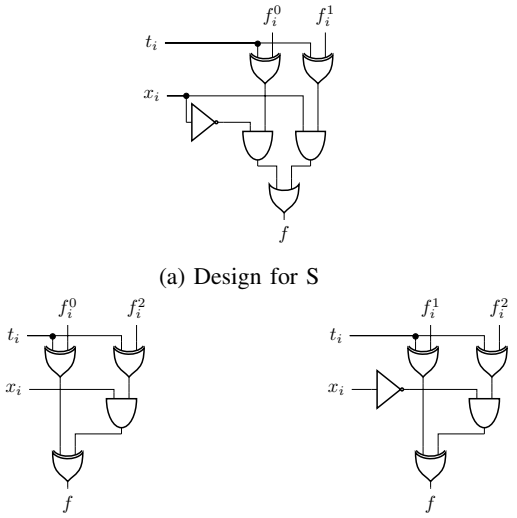
### C. Fault Models

During manufacturing, errors can occur, leading to faulty chips even if their design is correct. Using fault models, classes of faults can be defined and therefore tested. In this paper, we consider two static fault models, i.e. CFM and SAFM.

*1) Cellular Fault Model:* CFM [12], [13] assumes that a fault modifies the behavior of exactly one node in a circuit, while the faulty behavior is still combinational. Thus, a cellular fault can be defined as $(w, I, X/Y)$, where $w$ is the faulty node, $I$ is the input for which $w$ behaves faulty, and $X$ is the correct output of the circuit, whereas $Y$ is the faulty output [3].

*2) Stuck-At Fault Model:* In contrast to CFM, SAFM [14] assumes the fault within the input or output signals of nodes, instead of the nodes themselves. Here, SAFM defines faults where exactly one input or output of a node in the circuit has a constant value of 0 for a stuck-at-0 fault and 1 for a stuck-at-1 fault. A stuck-at fault at the $i$-th input of a node $w$ can be formally described as $(w[i], \epsilon)$, where $\epsilon \in \{0, 1\}$ is the respective constant value of the faulty input. A stuck-at fault at the $i$-th output of a node $w$ can analogously be described as $([i]w, \epsilon)$ [3].

A test pattern for a fault at the node $w$ in the circuit is defined as an input which results in the correct output if the fault does not occur, while it results in the wrong output if the fault occurs. If there exists no test pattern for a fault, then the respective fault is called redundant, i.e. non-testable. A complete test set contains test patterns for all testable faults, while a circuit is called fully testable if every possible fault is testable [3]. Thus, to achieve a fully testable circuit under




(a) Design for S

(b) Design for pD

(c) Design for nD

Fig. 5: Proposed addition of XOR gates to the subcircuits for S, pD and nD nodes

CFM, the test set has to test every possible input combination for every node within the circuit, while guaranteeing that the fault is propagated to the PO and not canceled out by the remainder of the circuit. Similarly, to achieve a fully testable circuit under SAFM, a test set has to contain a test for every possible stuck-at fault within a circuit, while ensuring that the fault is propagated to the output.

## III. Related Work

The testability properties of circuits derived from decision diagrams have been studied before in various works. The testability of BDD circuits was first studied in [3], where it was shown that all redundancies, as well as a complete test set under CFM, can be computed in polynomial time. Furthermore, it was shown that all subcircuits can be categorized into different *Controllability Classes* (CCs), which define redundancies at the subcircuits. Here, the circuits are fully testable under SAFM and PDFM if certain CCs are empty.

The problem of occurring redundancies in BDD circuits was later fixed by [4], where a toggle input was proposed, eliminating all redundancies and therefore enabling full testability under CFM, SAFM and PDFM without additional constraints on CCs.

The work of [3] was also extended to other types of decision diagrams in [15] and [7], where *Functional Decision Diagrams* (FDDs) and KFDDs were considered, respectively. Here, it was shown that all redundancies and a complete test set under CFM can be computed in exponential time. Again, the full testability under SAFM was shown with constraints on the CCs.

In [11], a different approach for testable KFDD circuits was presented, which was based on a BDD circuit synthesis method from [16]. Here, Boolean matrix multiplication was used to achieve a circuit with logarithmic depth. Even though the resulting circuits provided good testing capabilities in experimental results, the circuits are not generally fully testable, and no polynomially bounded algorithm for the test pattern generation has been shown.

Thus, despite the excellent testability properties of circuits derived from BDDs, no method has yet been established for the efficient generation of test sets for fully testable circuits derived from KFDDs.

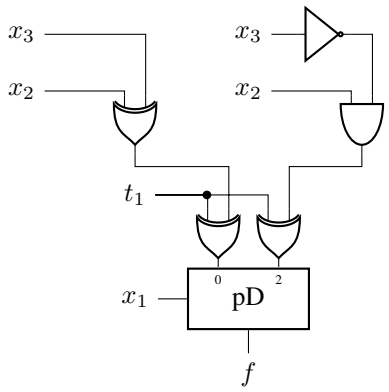



Fig. 6: Example for the proposed circuit design for the KFDD from Figure 1

## IV. Circuit Transformation

In this section, we propose a circuit design based on KFDDs that is fully testable in CFM and SAFM while guaranteeing a polynomial runtime for the test pattern generation.

Let $v$ be a KFDD node with the variable $x_i$ representing the function $f$, and let $w$ be the respective subcircuit for $v$ as shown in Figure 2. If $w$ can be degenerated according to Table I, $v$ is realized using the respective degenerated subcircuit. However, if $w$ cannot be degenerated, we add two XOR gates in our proposed circuit transformation. If $v$ uses the S decomposition, the XOR gates negate the inputs $f_i^0$ and $f_i^1$ if a toggle signal $t_i$ is set to 1, resulting in the following equation:

$$f = \overline{x}_i(t_i \oplus f_i^0) + x_i(t_i \oplus f_i^1) \tag{4}$$

The S subcircuit with the added XOR gates resulting from Equation (4) is shown in Figure 5(a). Similarly, two XOR gates and the toggle input are added for pD and nD nodes, resulting in the following equations, respectively:

$$f = (t_i \oplus f_i^0) + x_i(t_i \oplus f_i^2) \tag{5}$$

$$f = (t_i \oplus f_i^1) + \overline{x}_i(t_i \oplus f_i^2) \tag{6}$$

The pD and nD subcircuits with the added XOR gates and the toggle input are shown in Figure 5(b) and Figure 5(c), respectively. An example of the circuit derived from the KFDD in Figure 1 using the proposed method can be seen in Figure 6. As the only non-degenerated subcircuit is the pD subcircuit, the XOR gates are added to only one subcircuit, and only one toggle input $t_1$ has to be introduced.

TABLE II: Controllability classes

| class | applicable values | | | |
|---|---|---|---|---|
| 1 | 00 | 01 | 10 | 11 |
| 2 | | 01 | 10 | 11 |
| 3 | 00 | 01 | 10 | |
| 4 | 00 | | 10 | 11 |
| 5 | 00 | 01 | | 11 |

Let $F$ be a KFDD with $n$ variables from which the proposed circuit is derived. In general, at most $2 \cdot |F|$ XOR gates have to be added compared to the original KFDD circuit, as each node requires at most two additional XOR gates, resulting in at most $6 \cdot |F|$ gates in total. Furthermore, at most $n$ toggle inputs have to be added to the design.

However, as the toggle inputs are only needed for testing the circuit, while they are otherwise all set to 0, they can be realized with a scan chain [17]. For this, $n$ state elements are needed, storing the values of the toggle inputs. If a single scan chain is used, one scan input is needed for the realization. To reduce the test time, multiple scan chains can also be used, where $m$ scan inputs are required if $m$ scan chains are used. Furthermore, one scan enable input is needed to switch the chain into scan mode, as well as one test clock TCK. To test the circuit for a given pattern where no toggle input is activated, the scan chains are deactivated by setting the scan enable input to 0. If a test case requires $t_i = 1$ for some $1 \leq i \leq n$, the scan chains are activated and the respective $t_i$ can be set to 1 within a maximum of $\frac{n}{m}$ clock cycles. In the following cycle, the test pattern can be applied, testing the circuit for the respective input combination. Afterwards, the scan enable signal is set to 0 again and the toggle inputs are flushed with 0's within a maximum of $\frac{n}{m}$ clock cycles until all toggle inputs are set to 0. As the values of the state elements after the application of the test pattern are irrelevant, the scan chains do not require a scan out.

## V. Polynomial Test Pattern Generation

As described in [3] for BDDs and in [7] for KFDDs, to generate a complete test set, test patterns for each subcircuit have to be generated. The exponential blow-up from [7] results from the test pattern generation for non-degenerated subcircuits. Let $w$ be a non-degenerated subcircuit for a KFDD node $v$ with the variable $x_i$ representing the function $f$. Then, $f_{low}$ and $f_{high}$ are defined as the low- and high-children of $v$, i.e. $f_{low} = f_i^0$ for S, pD and $f_{low} = f_i^1$ for nD, whereas $f_{high} = f_i^1$ for S and $f_{high} = f_i^2$ for pD and nD. To compute test patterns for $w$, first an input assignment for all $x_j$ with $j > i$ has to be found that results in a possible fault, which then has to be propagated to the PO by an assignment of all $x_k$ with $k < i$.

Four combinations of the inputs $f_{low}$ and $f_{high}$ have to be applied to the subcircuit $w$ to test all functionality, i.e. the combinations 00, 01, 10 and 11. Assignments for all $x_j$, such that the four input combinations are applied to $w$, can be computed by building four KFDDs using four AND operations [7]:

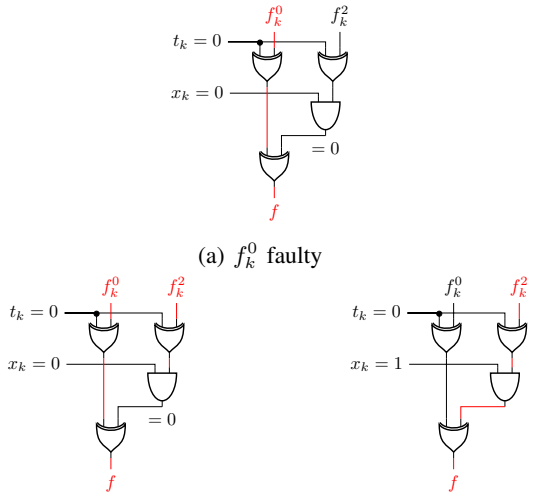
- $\overline{f_{low}} \cdot \overline{f_{high}}$ (00)
- $\overline{f_{low}} \cdot f_{high}$ (01)
- $f_{low} \cdot \overline{f_{high}}$ (10)
- $f_{low} \cdot f_{high}$ (11)

Then, a satisfying assignment for each resulting KFDD has to be determined. Determining a satisfying assignment of a KFDD with $n$ inputs can be done in $\mathcal{O}(n)$ by traversing the KFDD starting from the root and following the low-children with at most $n$ backtracking steps until the terminal 1 is reached. However, for KFDDs, the AND operation generally has an exponential complexity, therefore leading to an exponential computation time for the test pattern generation.

For the proposed circuit transformation, however, it is possible to compute satisfying assignments for all four input combinations within polynomial time using one polynomial XOR operation and one constant NOT operation. As established in [7], all non-degenerated subcircuits can be categorized into five CCs shown in Table II. The CC 1 includes subcircuits where all input combinations are applicable at $f_{low}$ and $f_{high}$, i.e. 00, 01, 10 and 11 are possible input combinations. Contrarily, for all other CCs, exactly one input combination isn't applicable at the inputs of the subcircuits categorized in the respective CC. E.g., the non-degenerated pD subcircuit in Figure 4 is in CC 4, as no assignment for $x_2$ and $x_3$ can satisfy the function $\overline{f_{low}} \cdot f_{high} = \overline{x_2 \oplus x_3} \cdot x_2\overline{x}_3$ and therefore, the input combination 01 is inapplicable at this node. As can be seen from Table II, at least three of the four input combinations are applicable at each subcircuit in an original KFDD circuit. From these CCs, it can therefore be observed that both $f_{low} \oplus f_{high}$ and $\overline{f_{low} \oplus f_{high}}$ have to be satisfiable.

To compute assignments for all $x_j$ and all four input combinations, the KFDD for $f_{low} \oplus f_{high}$ is computed within $\mathcal{O}(|F_{low}| \cdot |F_{high}|) = \mathcal{O}(|F|^2)$. A satisfying assignment for $f_{low} \oplus f_{high}$ can then be found in linear time $\mathcal{O}(n)$ and all toggle inputs are set to 0. It holds that the computed assignment results in either the input combination 01 or 10. To achieve a satisfying assignment for the respective other input combination, the toggle input $t_i$ is set to 1 while all other toggle inputs remain at 0. With $t_i$ set to 1, $f_{low}$ and $f_{high}$ are complemented and therefore, the respective other combination can be satisfied as well. Using a constant NOT operation, the process is repeated with $\overline{f_{low} \oplus f_{high}}$, yielding satisfying assignments for the input combinations 00 and 11. Overall, the satisfying assignments for all four combinations can therefore be computed within quadratic time with respect to the KFDD size.

To achieve a complete test pattern, the propagation of a fault to the PO has to be ensured, which can be done by a linear algorithm regarding the KFDD size. Let $v$ be the KFDD node with the variable $x_i$ for which the respective subcircuit $w$ is tested. As all toggle inputs except for the toggle input at the fault location are set to 0, the additional XOR gates introduced by our proposed circuit transformation do not interfere with the propagation properties of KFDD circuits. First, the KFDD is traversed to mark all nodes that can be reached from the node $v$, i.e. all nodes through which the fault at $w$ can be propagated,




(a) $f_k^0$ faulty

(b) $f_k^0$ and $f_k^2$ faulty

(c) $f_k^2$ faulty

Fig. 7: Fault propagation for pD subcircuits

which can be done in linear time $\mathcal{O}(|F|)$. Then, the values of $x_k$ with $k < i$ are set by constructing a path from the PO to the fault location as follows: We start at the subcircuit which is directly connected to the PO and run towards the fault location within linear time $\mathcal{O}(n)$. If the encountered subcircuit is degenerated, the respective variable $x_k$ can be set accordingly to propagate the fault, and the process is repeated for the input of the subcircuit. E.g., if the degenerated subcircuit computes the function $f = f_k^0 + x_k$, then $x_k$ is set to the non-controlling value 0, and the process is repeated for $f_k^0$. If the encountered subcircuit is non-degenerated, the fault can be propagated by setting $x_k$ according to the respective decomposition type as described in the following.

If the encountered non-degenerated subcircuit is an S subcircuit, it is checked from which child the fault location can be reached. If the fault can be reached from $f_k^0$, then $x_k$ is set to 0 to propagate the fault, and the process is repeated for $f_k^0$. Otherwise, the fault can be reached from $f_k^1$, therefore, $x_k$ is set to 1, and the process is repeated for $f_k^1$.

The propagation is exemplarily shown in Figure 7 for pD subcircuits, where the faulty inputs, i.e. the inputs that can be reached from the fault location, are marked in red, as well as the fault propagation through the subcircuit. As can be seen from Figure 7(a) and Figure 7(b), the fault from $f_k^0$ is propagated if $x_k = 0$, as the output of the AND gate is then 0. A fault that is only reachable from $f_k^2$ can be propagated by setting $x_k = 1$. In this case, the fault location cannot be reached from $f_k^0$, therefore, an occurrence of the fault does not change the value of the signal $f_k^0$. Thus, the XOR gate computing $f$ will output a faulty value if the fault occurs at $f_k^2$ and if $x_k$ is set to 1, as can be seen from Figure 7(c). Analogously, for negative Davio subcircuits, $x_k$ is set to 1 if the fault location can be reached from $f_k^1$. Otherwise, the fault can only be reached from $f_k^2$, therefore $x_k$ is set to 0.

The proposed circuit transformation and the test pattern generation yield the following results on the testability under CFM and SAFM.

### A. Cellular Fault Model

**Theorem 1.** *Let $F$ be a KFDD and $C_{KFDD}$ the resulting proposed circuit. Then, $C_{KFDD}$ is fully testable in CFM and has no redundancies, while a complete test set can be generated in time $\mathcal{O}(|F|^3)$.*

*Proof.* To achieve a fully testable circuit under CFM, each gate has to be tested with all possible input combinations, while faults have to be propagated to the PO. The gates in the circuit $C_{KFDD}$ are degenerated and non-degenerated S, pD and nD subcircuits, as well as the XOR gates needed for the toggle inputs. For each node in the KFDD $F$ with $n$ variables, all possible input combinations for the respective subcircuit, as well as for the added XOR gates, have to be tested.

Let $v$ be a node for the variable $x_i$ in the KFDD with a degenerated subcircuit $w$. Then, $w$ has either one input $x_i$ or two inputs $x_i$ and $f_i^c$ with $c \in \{0, 1, 2\}$. If the subcircuit has only one input $x_i$, it can simply be set to 0 and 1, respectively, to cover all input combinations. Furthermore, the algorithm for the fault propagation as described above has to be carried out, yielding an overall complexity of $\mathcal{O}(|F|)$ for this case. If the degenerated subcircuit has two inputs $x_i$ and $f_i^c$, it holds that both inputs don't share any functionality due to the read-once property of KFDDs and can therefore be set independently from each other. Thus, $x_i$ can be set to 0 and 1, and both cases have to be combined with a satisfying assignment for $f_i^c$ and a satisfying assignment for $\overline{f_i^c}$, respectively. $\overline{f_i^c}$ can be computed using a constant NOT operation, while the satisfying assignments can be computed in time $\mathcal{O}(n)$. Combined with the linear fault propagation algorithm, the test pattern computation for this case therefore has a linear complexity $\mathcal{O}(|F|)$.

Now let $v$ be a node with a non-degenerated subcircuit $w$. Then, both the subcircuit, as well as the XOR gates for the toggle input, have to be tested. To test the XOR gates, the same argument as for the degenerated subcircuit holds, as the XOR gates have two independent inputs, therefore requiring a linear complexity with respect to the KFDD size. For non-degenerated subcircuits, test patterns for all input combinations can be computed with one polynomial XOR and one constant NOT operation as described above. Including the linear algorithm for the fault propagation, the test patterns for $w$ can hence be computed in quadratic time $\mathcal{O}(|F|^2)$.

Thus, for each node in the KFDD $F$, the test pattern generation for all input combinations has at most a quadratic complexity, therefore yielding an overall complexity of $|F| \cdot \mathcal{O}(|F|^2) = \mathcal{O}(|F|^3)$. As the size of the KFDD circuit directly depends on the KFDD size, the test pattern generation is polynomial in both the KFDD size as well as the circuit size. □

### B. Stuck-At Fault Model

**Theorem 2.** *Let $F$ be a KFDD and $C_{KFDD}$ and $C_{STD}$ the resulting proposed circuit over different libraries. Then,*

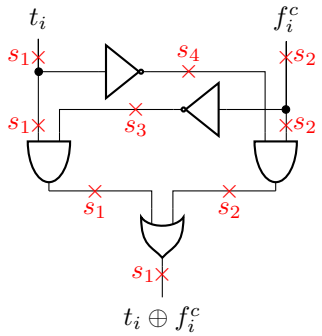



Fig. 8: Stuck-at faults for a realization of XOR in $STD$

*$C_{KFDD}$ and $C_{STD}$ are fully testable in SAFM, while a complete test set can be generated in time $\mathcal{O}(|F|^3)$.*

*Proof.* As shown in [7], for $C_{KFDD}$, all possible stuck-at faults at degenerated and non-degenerated subcircuits are testable in SAFM. As the subcircuits in our proposed circuit transformation provide the same functionality, the subcircuits are still fully testable in SAFM, where the respective test patterns can be computed with one XOR and one NOT operation in $\mathcal{O}(|F|^2)$ as described above. The only extra gates in our proposed method are the XOR gates for the toggle inputs, which can also be fully tested for stuck-at faults. Let $t_i$ and $f_i^c$ with $c \in \{0, 1, 2\}$ be the inputs to an added XOR gate. A possible input combination for testing stuck-at-1 faults at an XOR gate is 00, i.e. $t_i = 0$ and $f_i^c = 0$, while stuck-at-0 faults can e.g. be tested with the input combinations 11 and 01. Again, satisfying assignments for $f_i^c$ and $\overline{f_i^c}$ can be found in linear time $\mathcal{O}(n)$.

Stuck-at faults at the stem of an input $x_i$ or $t_i$ can be tested as well. Here, an assignment for the fault propagation is computed with the method described previously in Section V with the exception that all nodes with the faulty input $x_i$ or $t_i$ are initially marked as reachable from the fault location. After marking all other reachable nodes, a path from the root node to the fault location is constructed until a node with the respective $x_i$ or $t_i$ is encountered. A stuck-at fault at the stem of the input can then be tested at the respective encountered node. To achieve a test pattern for the stuck-at fault, a satisfying assignment for the needed input combination at the encountered node can be computed using one XOR and one NOT operation in $\mathcal{O}(|F|^2)$.

Similar to CFM, the complete test set can be computed in $\mathcal{O}(|F|^3)$, as the bottleneck for the computation using our proposed test pattern generation is the quadratic computation of the XOR operation, which has to be executed for each node in the KFDD, therefore yielding an overall cubic complexity.

In contrast to CFM, the proposed circuit design is not only fully testable for $C_{KFDD}$, but also for $C_{STD}$. Here, it was shown in [7] that the circuit is fully testable in SAFM, if the CC 3 contains no pD or nD subcircuits, CC 4 is empty and CC 5 contains no S subcircuits. Due to the toggle inputs introduced in our approach, all nodes are in CC 1 and therefore, all

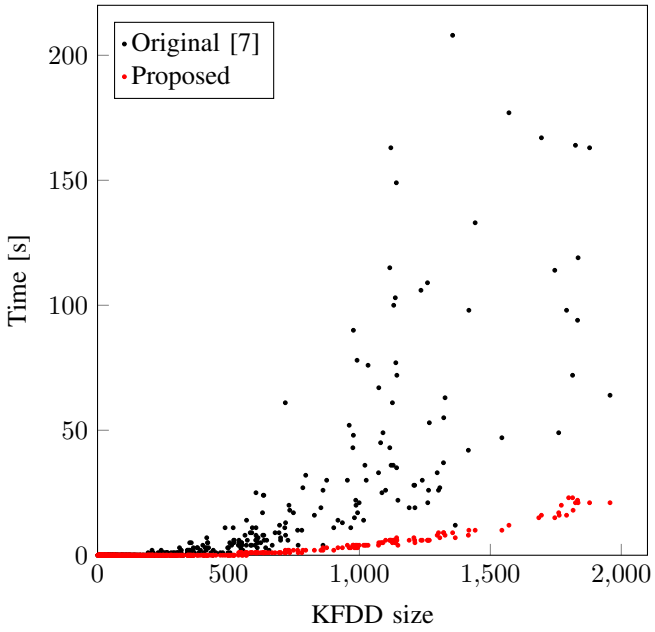



Fig. 9: Test set generation time for the original [7] and the proposed KFDD circuit

degenerated and non-degenerated subcircuits are fully testable in SAFM, where the test patterns for a subcircuit $w$ can be computed in $\mathcal{O}(|F|^2)$. To complete the proof, the testability of XOR which is realized with $STD$ gates has to be shown. Let $t_i$ and $f_i^c$ with $c \in \{0, 1, 2\}$ be the inputs to an added XOR gate. Figure 8 shows possible locations for a stuck-at fault in the $STD$ realization. In the following, we give exemplary input combinations for testing stuck-at faults at the different locations. All stuck-at-0 faults at the signals marked with $s_1$ can be tested with the input combination 10, i.e. $t_i = 1$ and $f_i^c = 0$. A stuck-at-1 fault at the same signals can be tested with 00. For signals marked with $s_2$, the stuck-at-0 and stuck-at-1 faults can be tested with 01 and 00, respectively. The combinations 10 and 11 are needed to test the signal marked with $s_3$, whereas testing $s_4$ requires the combinations 01 and 11. As $t_i$ and $f_i^c$ are independent of each other, all combinations of 0 and 1 are applicable at the inputs of the XOR gate, therefore yielding full testability of the XOR gate for the toggle inputs in $C_{STD}$. The cubic complexity of the test set generation is analogous to $C_{KFDD}$. □

## VI. Experiments

To evaluate our proposed polynomial test pattern generation and KFDD circuit transformation, we compare our approach with the approach for the original KFDD circuits in [7]. Complete test sets were computed for over 1000 random KFDDs with up to $n = 16$ inputs and a KFDD size of up to over 8000 nodes. For the evaluation, the KFDD implementation within the open-source framework FrEDDY [18], [19] was used. The timeout per circuit and method was set to one hour.

In Figure 9, the computation time is shown for random KFDDs with a size of up to 2000 nodes. The method of [7],

TABLE III: Timeouts for larger KFDD circuits

| KFDD size | Time [7] | Time proposed [s] |
|---|---|---|
| 3146 | T.O. | 81.97 |
| 4488 | T.O. | 222.97 |
| 6365 | T.O. | 796.11 |
| 8054 | T.O. | 1141.18 |

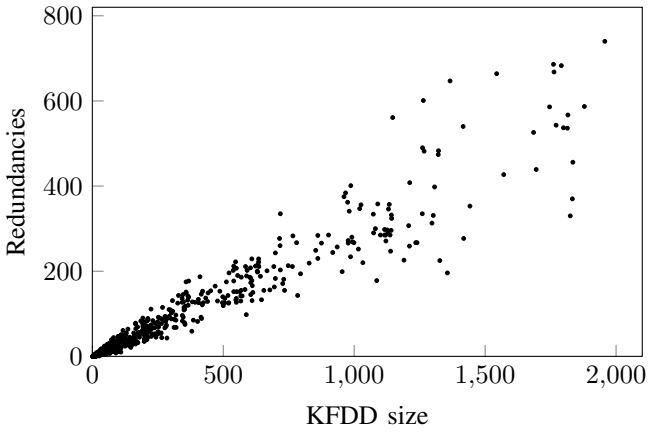



Fig. 10: Redundancies for the original KFDD circuit from [7]

where four exponential AND operations have to be carried out, is shown in black, whereas our proposed method, which computes the test patterns with one XOR and one NOT operation, is shown in red. It is apparent that our proposed method requires much less resources for the pattern generation than the method from [7]. Overall, the maximum speedup achieved on the exemplary circuits was 50.11, where the method from [7] took 323.72 seconds, whereas our proposed method took 6.46 seconds for a full test set generation. On average, our proposed polynomial approach was 4.19 times faster than the exponential method.

For some random KFDDs with more than 3000 nodes, the test pattern generation from [7] reached a timeout, i.e. the test pattern generation took longer than one hour. The results for a few test cases with large KFDD sizes where a timeout was reached are shown in Table III. As can be seen, for some large KFDDs, the test set generation of [7] reached the timeout of one hour, while our proposed method was still able to generate a complete test set within a few minutes.

In addition to the speedup, the proposed method eliminates all redundancies within the circuit. In Figure 10, the number of redundancies of [7] occurring in our evaluation is shown, i.e. the number of cases where a subcircuit is not in CC 1, leading to untestable faults. As each subcircuit can be in one of the CCs 2-5, the number of redundancies rises linearly with the KFDD size. As all subcircuits in our proposed approach are in CC 1, no redundancies occur for the proposed design. Overall, our evaluation shows that the test set generation for our proposed KFDD circuit is significantly faster than for the original KFDD circuit [7], while simultaneously eliminating all redundancies.

## VII. Conclusion

In this paper, we have shown a synthesis for testability approach for circuits derived from KFDDs that increases the testability to 100% under CFM and SAFM. For the resulting KFDD circuits, we have proposed an algorithm computing the test set in polynomial time regarding the circuit size, therefore allowing efficient automatic test pattern generation. As KFDDs can be exponentially smaller than BDDs, the resulting KFDD circuits enable full testability for some functions where BDD circuits are limited by their exponential growth. In our evaluation, the advantages of our method were clarified, overcoming the drawbacks of the existing exponential algorithm with a maximum speedup factor of 50.

## References


[1] N. K. Jha and S. Gupta, *Synthesis for testability*. Cambridge University Press, 2003, p. 799–844.

[2] R. Drechsler, "Polynomial circuit verification using BDDs," in *International Conference on Electrical, Electronics, Communication, Computer Technologies and Optimization Techniques (ICEECCOT)*, 2021, pp. 49–52.

[3] B. Becker, "Synthesis for testability: Binary decision diagrams," in *Symposium on Theoretical Aspects of Computer Science (STACS)*, ser. Lecture Notes in Computer Science, vol. 577. Springer, 1992, pp. 501–512.

[4] R. Drechsler, J. Shi, and G. Fey, "Synthesis of fully testable circuits from BDDs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 3, pp. 440–443, 2004.

[5] B. Becker, R. Drechsler, and M. Theobald, "On the expressive power of OKFDDs," *Formal Methods Syst. Des.*, vol. 11, no. 1, pp. 5–21, 1997.

[6] M. Schnieber and R. Drechsler, "Polynomial formal verification of KFDD circuits," in *International Conference on Formal Methods and Models for System Design (MEMOCODE)*, 2023, pp. 82–89.

[7] B. Becker and R. Drechsler, "Synthesis for testability: circuits derived from ordered Kronecker functional decision diagrams," in *Proceedings the European Design and Test Conference. ED&TC 1995*, 1995, p. 592.

[8] R. Drechsler, A. Sarabi, M. Theobald, B. Becker, and M. A. Perkowski, "Efficient representation and manipulation of switching functions based on ordered Kronecker functional decision diagrams," in *Design Automation Conference*, 1994, pp. 415–419.

[9] R. Drechsler and B. Becker, "Ordered Kronecker functional decision diagrams-a data structure for representation and manipulation of Boolean functions," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, no. 10, pp. 965–973, 1998.

[10] R. E. Bryant, "Graph-based algorithms for Boolean function manipulation," *IEEE Transactions on Computers*, vol. 35, no. 8, pp. 677–691, 1986.

[11] H. Hengster, R. Drechsler, B. Becker, S. Eckrich, and T. Pfeiffer, "AND/EXOR-based synthesis of testable KFDD-circuits with small depth," in *Asian Test Symposium (ATS)*, 1996, pp. 148–154.

[12] W. H. Kautz, "Testing for faults in combinational cellular logic arrays," in *Symposium on Switching and Automata Theory (SWAT)*, 1967, pp. 161–174.

[13] A. Friedman, "Easily testable iterative systems," *IEEE Transactions on Computers*, vol. C-22, no. 12, pp. 1061–1064, 1973.

[14] M. A. Breuer and A. D. Friedman, *Diagnosis & Reliable Design of Digital Systems*. Springer Berlin, Heidelberg, 1976.

[15] B. Becker and R. Drechsler, "Testability of circuits derived from functional decision diagrams," in *Proceedings of European Design and Test Conference EDAC-ETC-EUROASIC*, 1994, p. 667.

[16] N. Ishiura, "Synthesis of multilevel logic circuits from Binary decision diagrams," *IEICE TRANSACTIONS on Information*, vol. E76-D, no. 9, pp. 1085–1092, September 1993.

[17] M. Abramovici, M. A. Breuer, and A. D. Friedman, *Digital Systems Testing and Testable Design*. IEEE Press, 1990.

[18] R. Krauss, J. Zielasko, and R. Drechsler, "FrEDDY: Modular and efficient framework to engineer decision diagrams yourself," in *Design, Automation and Test in Europe Conference (DATE)*, 2025.

[19] R. Krauss, "Freddy," https://github.com/runekrauss/freddy, 2025.