Quantum Deductive Fault Simulation

V. Hahanov 1, W. Gharibi 2, M. Liubarskii 1, E. Litvinova 1, M. Gharibi 1,
S. Chumachenko 1, I. Hahanov 1, and A. Hahanova 1

1Design Automation Department, Kharkov National University of Radioelectronics, Kharkov, Ukraine
2Computer Science Department, Jazan University, Jazan, KSA
3School of Computing and Engineering, University of Missouri-Kansas City, Kansas City, MO, USA

Abstract – A structural model of the interaction of X functions and derivatives, focused on the synthesis and analysis of digital systems, is proposed in order to reduce the design and testing time of computing devices. The notion of simple X-functions of a finite number of variables, which are characterized by the absence of minimization and the presence of testability properties for solving the problems of test synthesis, simulation and diagnosis, is introduced. Metric properties of X-functions of a finite number of variables are formulated to evaluate the quality of verification tests by deductive modeling the stuck-at faults on qubit data structures. An analytical expression is proposed for the synthesis of qubit coverage of X-functions of a finite number of variables for the purpose of subsequent synthesis and analysis of tests for testing and diagnosing faults. Deductive formulas for transporting input fault lists to external outputs of X-functions of a finite number of variables have been synthesized to build a fault simulation sequencer that is invariant to input test patterns.

Keywords: Synthesis and Analysis; Digital System; X-Function; Deductive Simulation; Qubit Coverage.

1 Deductive fault analysis of logic X-functions

Deductive analysis of digital systems and components is the most effective apparatus for test synthesis and evaluation of their quality, and also for diagnosing faults in real time. The essence of deductive modeling lies in the use of synthesized digital sub-circuits, which are a complement to the original functionality intended for transporting negative reactions occurring in the faulty circuit to external outputs. As a rule, the complexity of deductive sub-circuits implemented in software or hardware is several times higher than useful functionality. Therefore, the effectiveness of deductive analysis should be considered from the point of reducing the design time of a digital system and time-to-market.

The aim of the research is to reduce the time for parallel designing a digital system-on-a-chip by synthesizing the deductive functions of logical components and circuits, including the X-functions represented by qubit forms of describing the behavior of digital devices.

Objectives are the following: 1) Develop a structural model for the interaction of X-functions and derived components, focused on the synthesis and analysis of digital systems. 2) Define simple X-functions of a finite number of variables for test synthesis, simulation and diagnosis of digital devices. 3) Define the metric properties of X-functions of a finite number of variables to evaluate the quality of the verification tests on the qubit data structures. 4) Define an analytical expression for the synthesis of qubit coverage of X-functions of a finite number of variables in order to analyze tests and fault diagnosis. 5) Synthesize deductive formulas for transporting input fault lists for X-functions of a finite number of variables. 6) Develop a quantum method of unconditional fault diagnosis, based on the use of parallel logical operations.

Deductive analysis is used to determine the test quality relatively introduced class of faults, usually stuck-at faults. There is a developed theory of deductive analysis [1], focused on the parallel processing of fault lists. An apparatus for transporting fault lists through primitive functional logic elements describes the basic concepts of the theory of fault simulation [2-8]. The deductive functions for the parallel fault simulation on an exhaustive test for the functional elements and, or, not are defined further. Getting a deductive converter for the function and is shown below:

\[ L[T = (001,001,1,1)] = (X_1 \land X_2) = \]
\[ L[(x_1 \land x_2) = (X_1 \land X_2) = \]
\[ L[(x_1 \land x_2) = (X_1 \land X_2) = \]
\[ L[(x_1 \land x_2) = (X_1 \land X_2) = \]

Similar calculations are performed for the function or:

\[ L[T = (001,001,1,1)] = (X_1 \lor X_2) = \]
\[ L[(x_1 \lor x_2) = (X_1 \lor X_2) = \]
\[ L[(x_1 \lor x_2) = (X_1 \lor X_2) = \]
\[ L[(x_1 \lor x_2) = (X_1 \lor X_2) = \]

Here \( T = (T_{11}, T_{12}, T_{13}) \) is a test-vector, which has 3 coordinates, where the last one determines the state of the output of the two-input element and (or); \( L \) is the output fault list; \( X \) is the fault list at the input of the primitive; \( x = \{0,1\} \) is
the logical value at the input of the primitive. In the next transformation, 
\(T_1 = (T_{11}, T_{12})\), \(t = T_2\) is a test vector having 2
coordinates, where the second one is the state of the inverter output:

\[
L[T = (0,1), F = \bar{x}_1] = L[(\bar{x}_1 \lor x_1)[(x_1 \oplus T_{11}) \oplus T_{12}]] = \\
\bar{x}_1[(x_1 \oplus 0) \oplus 1] \lor x_1[(x_1 \oplus 1) \oplus 0] = \bar{x}_1 \bar{x}_1 \lor x_1 x_1 = x_1 x_1 \lor x_1 x_1.
\]

The last expression illustrates the invariance of negation to the input set for fault transportation. It is transformed into a repeater. Therefore, this function does not appear at the outputs of deductive elements. The joint hardware implementation of deductive functions of two-input elements and, or on the exhaustive test is represented by the universal circuit (Fig. 1) for deductive-parallel analysis of faults [9].

A result of the synthesis is that all the deductive formulas on the four input test patterns have the same form, which means the invariance of the xor function to the input test patterns – any input word has a single deductive formula:

\[
L(\text{xor}) = [(x_1 x_2 v x_1 x_2 v x_1 x_2 v x_1 x_2 v) \lor [(x_1 x_2 v x_1 x_2 v) = \\
= (xx) \lor [(x_1 x_2 v x_1 x_2)].
\]

This means that lists of input faults will always be combined at the output of the xor-element, except when the lists are identical. In this case, no faults will be checked at the output of the xor element.

There are other functions that have a single universal deductive formula for all input test patterns. Below we propose the synthesis of the deductive formula for the operation of equivalence:

\[
L[T = (00,01,10,11), F = X_2 = X_1 X_2 v X_1 X_2]\]

\[
= L[(T_{11} x_2 v x_1 x_2 v x_1 x_2) \lor [((x_1 \oplus T_{11}) \oplus x_1 x_2) v (x_1 \lor T_{12}) v (x_1 x_2 \lor T_{00})] v (x_1 \lor T_{01}) v (x_1 x_2 \lor T_{02}) v (x_1 x_2 \lor T_{03})] = \\
L[T_{00} = 00, (x_1 x_2) \lor (x_1 \oplus 0) (x_1 \oplus 1) v (x_1 \oplus 1) (x_1 \oplus 0)] = \\
= (x_1 x_2) \lor (x_1 \lor x_2) \lor (x_1 \lor x_2) \lor (x_1 \lor x_2) \lor (x_1 \lor x_2) \lor (x_1 \lor x_2) \lor (x_1 \lor x_2) \lor (x_1 \lor x_2).
\]

Thus, the function of equivalence (not-xor) also has unique capabilities to transport input fault lists to the output of a logic function, which are independent of the input signal:

\[
L(\text{not} - \text{xor}) = [(x_1 x_2 v x_1 x_2 v x_1 x_2 v x_1 x_2 v) \lor [(x_1 x_2 v x_1 x_2 v) = \\
= (xx) \lor [(x_1 x_2 v x_1 x_2)].
\]

As a result, it turned out that deductive formulas for two different logical functions xor and not-xor are identical to each other. However, it is more significant that any test influence of a digital device consisting of the above mentioned logic elements detects 50 percent of stuck-at faults negated to the fault-free state of the input lines. To detect all faults of the input lines, only two mutually negated test patterns must be input to the circuit.

For the presentation of the subsequent material, it is necessary to compare three forms of the description of logical functions: the Truth Table (TT), the Disjunctive Normal Form (DF), the Qubit Coverage (QC) [9]:

\[
\begin{array}{cccc}
X_1 & X_2 & X_3 & Y \\
0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 \\
1 & 1 & 0 & 0 \\
1 & 1 & 1 & 1
\end{array}
\]

\[
Y(\text{DF}) = x_1 x_2 x_3 v x_1 x_2 x_3 v x_1 x_3 v x_1 x_2 x_3;
\]

\[
Y(\text{QC}) = 01101001.
\]
The qubit coverage is a vector of output states of a logical function, ordered by increasing binary addresses or input patterns of a truth table. Q-coverage is an explicit and most compact form of describing the behavior of a logical function, which has the following advantages over tabular and analytic forms: 1) The Q-vector requires less memory for data storage in n times compared to the truth table. 2) The Q-vector does not require (n**2)-complex computational procedures necessary to determine the state of the output of a logical function by a disjunctive normal form. 3) The characteristic equation for the analysis of qubit coverage contains only address write-read operations: Mi=Qi[Xi], which is characterized by high parallelism and linear computational complexity.

Taking into account the above information on the advantages of a qubit coverage, we further propose the synthesis of a deductive formula for a more complex logical circuit, which is a three-variable xor-function having a qubit coverage: (01101001). For the analytical expression of the computational complexity, which is characterized by high parallelism and linear computational complexity.

The essential fact is that the X-function of n variables, which is associated with the xor-primitive, is a single function for which the logical and deductive functions are equal to each other on any input binary pattern. In particular, for three variables the equation has the form:

\[ f(01101001) = L(01101001) = X_1\oplus X_2\oplus X_3. \]

Thus, the X function defined by the qubit coverage (01101001) on all test patterns has a deductive formula for transporting input fault lists that is equal to the original X-function:

\[ L(T_0 = 000) = (X_1\oplus X_2\oplus X_3)\oplus (X_1\oplus X_2\oplus X_3)\oplus (X_1\oplus X_2\oplus X_3) = (X_1\oplus X_2\oplus X_3) \]

On all other test patterns (010, 011, 100, 101, 110, 111) the logical X-function

\[ f = X_1\oplus X_2\oplus X_3 \]

has an analogous deductive formula equal to the original X-function.

Thus, the X function defined by the qubit coverage (01101001) on all test patterns has a deductive formula for transporting input fault lists that is equal to the original function:

\[ L(T = xxx) = f(X_1, X_2, X_3) = (X_1\oplus X_2\oplus X_3)\oplus (X_1\oplus X_2\oplus X_3)\oplus (X_1\oplus X_2\oplus X_3). \]

In order to prove the completeness of the investigation, we further develop the deductive formula for transporting the lists of input faults for the second X-function of three variables:

\[ f(Q = 10010110) = X_1\oplus X_2\oplus X_3 \]

\[ L(T_0 = 000) = (X_1\oplus X_2\oplus X_3)\oplus (X_1\oplus X_2\oplus X_3)\oplus (X_1\oplus X_2\oplus X_3) = (X_1\oplus X_2\oplus X_3) \]

The structure for fault-free simulation and deductive fault simulation of X-functions based on the use of register operations is shown in Fig. 2.

![Fig. 2. Deductive simulator of X-functions](image-url)

The advantages of the deductive simulator of logical X-functions are as follows: 1) There is no traditional hardware redundancy for deductive fault simulation of X-functions, while any other logical functions require increasing equipment in 8-10-times. 2) Two automaton clock cycles are used to process the input impact, focused on fault-free simulation and deductive fault analysis, respectively. 3) The deductive simulator uses parallel logical operations to effectively transport the input fault lists.

The formula for taking the derivatives of input Boolean variables leverages the xor operation between the cells of the adjacent qubit parts:

\[ Q'(X_k) = (Q^L_1, Q^L_1) = Q^L_1 \oplus \bigoplus_{i=1}^{k-1} Q^R_1. \]

The structure of the sequencer for taking the qubit derivative is shown in Fig. 3, where the results of the xor operation are entered in both parts of the qubit. An example of the use of a sequencer for taking the qubit derivatives of an X function of three variables is represented in the following form (the upper indices are the numbers of the cells of the adjacent qubit parts...
qubit coverage vector, where the results of the xor operation are written (w):

\[ Q = (01010011) = f = X_1 \oplus X_2 \oplus X_3; \]

\[ Q(X_1) = \theta^0 \oplus \theta^1 = 1; \]
\[ Q(X_2) = \theta^0 \oplus \theta^3 = 1; \]
\[ Q(X_3) = \theta^0 \oplus \theta^5 = 1; \]

\[ Q(X_4) = \theta^0 \oplus \theta^5 = 1; \]

In the general case, we can formulate a criterion for the existence of deductive functions, invariant to input influences: "If the qubit derivatives with respect to all variables are equal to the unit vector, then the deductive formula is invariant to the input test patterns".

For four logic functions of one variable, there are only two primitives, which have the same unit vectors of derivatives, obtained by performing xor-operation by towards shifting the qubit parts:

\[ Y = X; Q = 01, Q' = 11; \]
\[ Y = \bar{X}; Q = 10, Q' = 11. \]

For sixteen logical functions of two variables, there are also two primitives, which have the same unit vectors of derivatives obtained by performing xor-operation by towards shifting the qubit parts:

\[ Y = X_1 \oplus X_2; Q = 0110; Q' = 1111; \]
\[ Y = X_1 \oplus X_2; Q = 1001; Q' = 1111. \]

For 256 logical functions of three variables, there are also two primitives, which have the same unit vectors of derivatives obtained by performing xor-operation by towards shifting the qubit parts:

\[ Y = X_1 \oplus X_2; Q = 01101001; Q' = 11111111; \]
\[ Y = X_1 \oplus X_2; Q = 10010110; Q' = 11111111. \]

Here, two qubits of X-functions of any finite number of variables are the negation of the states of the corresponding bits. Each of them can be divided into two equal parts by the number of digits, where the left one is the negation of the right side:

\[ Q(n = 1) = 01; \]
\[ Q(n = 2) = 0110; \]
\[ Q(n = 3) = 01101001; \]
\[ Q(n = 4) = 011010011001101; \]
\[ Q(n = 5) = 011010011010011010011010011001101010011010100110101001101010. \]

Assertion 1. The set \( |(X)| = 2^n \) of logical functions of a finite number of variables \( n \) has only two functions for which the qubit derivatives are taken as unit vectors.

Assertion 2. The number of logical functions of \( n \) variables, where the condition is true \( \forall i = 1 \frac{df}{dX_i} = 1 \), always is equal to two. This means that no other input conditions are needed to activate the output when the input variable \( X_1 \) is changed.

Definition. The X-function is a simple logical function of a finite number of variables \( n = 1,2,3, \ldots \), which cannot be minimized.

Assertion 3. The chess representation of zero and unit states in the Karnaugh map defines an X-function. The following two examples of logical functions, depicted in the form of the Karnaugh map, illustrate two mutually negated Boolean X-functions of four variables:

Using the Karnaugh map, the logical X-functions of three variables can be represented, which have 8 output states:

These functions can be put in correspondence with their analytical forms, which have a useful property – each term of PDNF (the Perfect Disjunctive Normal Form) differs from the others in two variables:

\[ Y(01010011000110100); \]
\[ Y(00101001101001100); \]

Fault-free simulation of all input patterns (Table T) for the first X-circuit, fault analysis (Table D), minimization of test sequences (Table M) and the qubit form of the minimum test – Table T (Q), are presented below [9]:

Fig. 3. A sequencer of qubit derivative

\[ \text{Fig. 3. A sequencer of qubit derivative} \]
Here, each input pattern detects stuck-at faults of external inputs and outputs, which is defined in Table D. The minimum test, covering all stuck-at faults, is presented in Table M. The last row shows the result of fault coverage in the form of a vector C whose coordinates are given by the symbols $x = 0 \cup 1$. The unit values of the column $T(Q)$ create a minimum qubit form of the test (binary addresses of unit coordinates), which have to be set on external inputs to detect all stuck-at faults of internal and external lines of a digital logic circuit.

Fig. 4. Structural implementation of Boolean X-functions

Fault-free simulation of all input patterns (Table T) for the second X-circuit, fault analysis (Table D), minimization of test patterns (Table M) and the qubit form of the minimum test – Table $T(Q)$ are represented below:

<table>
<thead>
<tr>
<th></th>
<th>0 0 0 0 0 0 0 1 1</th>
<th>1 0 1 0 0 0 0 0 0</th>
<th>0 1 1 0 0 0 0 0 0</th>
<th>1 0 0 1 0 0 0 1 0</th>
<th>1 0 0 1 0 1 0 0 0</th>
<th>0 1 0 0 1 0 1 0 0</th>
<th>1 1 0 0 1 0 0 0 0</th>
<th>0 0 1 0 0 0 1 0 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>T(Q)</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Here, each input pattern detects stuck-at faults of external inputs and outputs, which is defined in Table D. The minimum test, covering all stuck-at faults, is presented in Table M. The last row shows the result of fault coverage in the form of a vector C whose coordinates are given by the symbols $x = 0 \cup 1$. The unit values of the column $T(Q)$ create a minimum qubit form of the test (binary addresses of unit coordinates), which have to be set on external inputs to detect all stuck-at faults of internal and external lines of a digital logic circuit.

2) The test in the form of PDNF of X-function is always complemented by a single input sequence – any term of the inverse PDNF $T_i \in T^0, f(T_i) = 0$ that detects all stuck-at faults of internal lines and output:

$$T^0(1) = T_i \leftarrow \forall i : f(T_i) = 0.$$  

In particular, the complement to the 1-test of the X-function ($Q=01101001$) is determined by the zero test pattern (000) for all input coordinates; the complement to the 1-test for the X-function ($Q=10010110$) is given by the first test pattern (001), containing 1 in the last digit and all other zeros. In fact, the complement to the 1-test is any input sequence that is absent in the 1-test. Therefore, the full test of the X-function is always a test of the dimension $Q = 1 + \frac{1}{2} \cdot 2^n$.

Consequently, the test of a logical X-function of $n$ variables is its PDNF, complemented by any inverse PDNF term of this function:

$$T = T^1 \lor T_i^0.$$  

$$T^1 = \forall i : f(T_i) = 1.$$  

$$T_i^0 \in T^0 = \forall i : f(T_i) = 0;$$  

$$T(01101001) = (001 \lor 010 \lor 100 \lor 111) \lor 000;$$  

$$T(10010110) = (000 \lor 011 \lor 101 \lor 110) \lor 001.$$  

Another unique property of the X-function is defined as follows: the coordinate-wise xor-sum of all codes corresponding to the PDNF (inverse PDNF) is equal to the zero (in all binary digits) vector:

$$n_l \oplus C(T_l^1) = 0;$$  

$$i=1$$

$$n_0 \oplus C(T_l^0) = 0;$$  

$$i=1$$

$$n_0 + n_l = 2^n.$$  

The fact of the convolution of the PDNF or inverse PDNF space to the zero-vector is presented for two X-functions of three variables:

$$Y(01101001) = \bar{X}_1 \bar{X}_2 X_3 \lor \bar{X}_1 X_2 \bar{X}_3 \lor X_1 \bar{X}_2 \bar{X}_3 \lor X_1 X_2 X_3 \rightarrow$$  

$$\rightarrow 001 \lor 010 \lor 100 \lor 111 = 000;$$  

$$Y(10010110) = \bar{X}_1 X_2 \bar{X}_3 \lor \bar{X}_1 X_2 X_3 \lor X_1 \bar{X}_2 \bar{X}_3 \lor X_1 X_2 X_3 \rightarrow$$  

$$\rightarrow 000 \lor 011 \lor 101 \lor 110 = 000.$$  

The property of convolution makes it possible to calculate unknown terms of the test or PDNF based on the known components by applying, for example, the following equality for the X-function of three variables:

$$T_1 \lor T_2 \lor T_3 \lor T_4 = 0;$$  

$$001 \lor 010 \lor 100 \lor 111 = 000.$$  

$$T_2 \lor T_3 \lor T_4 = T_i;$$  

$$010 \lor 100 \lor 111 = 001.$$  

For logical X-functions of two variables (Fig. 5), which are known as xor, not-xor primitives, fault-free simulation of
all input patterns (Table T), fault analysis (Table D) and the qubit form of four variants of minimum tests (Table T(Q)) are presented below:

![Fig. 5. X-functions of two variables](image)

<table>
<thead>
<tr>
<th>T(xor)</th>
<th>T(Q)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>T(Q)</td>
</tr>
<tr>
<td>0 0 0 0 0 0 1 1 1 1</td>
<td>0 1 1 1 1 1 1</td>
</tr>
<tr>
<td>1 0 1 0 1 1 1 0 0 0</td>
<td>1 1 0 1 1 1 1</td>
</tr>
<tr>
<td>2 1 0 1 0 1 0 1 0 1</td>
<td>2 1 1 0 1 1</td>
</tr>
<tr>
<td>3 1 1 0 0 0 0 0 0 0</td>
<td>3 1 1 1 1 1</td>
</tr>
</tbody>
</table>

→

<table>
<thead>
<tr>
<th>T(nx)</th>
<th>T(Q)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>T(Q)</td>
</tr>
<tr>
<td>0 0 0 0 0 0 1 1 1 1</td>
<td>0 1 1 1 1 1 1</td>
</tr>
<tr>
<td>1 0 1 0 1 1 1 0 0 0</td>
<td>1 1 0 1 1 1 1</td>
</tr>
<tr>
<td>2 1 0 1 0 1 0 1 0 1</td>
<td>2 1 1 0 1 1</td>
</tr>
<tr>
<td>3 1 1 0 0 0 0 0 0 0</td>
<td>3 1 1 1 1 1</td>
</tr>
</tbody>
</table>

For both functions, minimum tests consisting of three input patterns are obtained. This is due to the fact that the opposite input vectors have the same states of the output variable.

A test based on T-axioms for defining input sequences, which verify all stuck-at faults of digital circuits (xor, nx), has the following form:

\[ T(xor, nx) = T_1 \lor T_i^0, \]

\[ T(Q = 1100)(xor) = (01 \lor 100) \lor 00; \]

\[ T(Q = 0101)(nx) = (00 \lor 111) \lor 01. \]

For two logical X-functions of one variable (Fig. 6), fault-free simulation of all input patterns (Table T), fault analysis (Table D) and the qubit form of minimum tests (table T(Q)) are represented below, leveraging the functional elements repeater-rep, inverter-not:

![Fig. 6. X-functions of one variable](image)

A test that leverages T-axioms to define input sequences detecting all stuck-at faults of digital primitives (rep, not), has the following form:

\[ T(rep, not) = T_1 \lor T_i^0, \]

\[ T(Q = 01)(rep) = 1 \lor 00; \]

\[ T(Q = 10)(not) = 0 \lor 1. \]

Thus, leveraging two T-axioms mentioned above makes it possible to define a complete test for stuck-at faults of the input, internal and output lines of any complex logical X-function.

The following 12 properties of X-functions, integrated into the model of relationships shown in Fig. 7, can be useful for the synthesis and analysis of digital circuits.

![Fig. 7. Structural model of interaction of X-functions](image)

1) The qubit coverage of the X-function has equal number of zero and unit coordinates.

2) The number of X-functions of n Boolean variables is always equal to two:

\[ Q^2_X(n) = Q^X(n) \lor Q^X(n). \]

The states of the coordinates of the qubit coverage of both X-functions of n variables are mutually inverse.

3) X-functions of one logical Boolean variable are represented by a repeater and an inverter: \( Y = X; Y = \overline{X} \).

4) The X-functions of two Boolean variables are represented by known logic primitives xor, not-xor:

\[ Y = X_1 X_2 \lor X_1 X_2; \]

\[ Y = X_1 X_2 \lor X_1 \overline{X_2}. \]

5) The qubit derivative with respect to any variable of the X-function is equal to the unit vector. The Boolean derivative with respect to any variable of the X-function is equal to 1.

6) To activate the input variable of the X function for changing the output, no conditions are required for the states of the other variables.

7) A pair of input patterns that have negated signals in all coordinates always changes the output state of the X-function of an odd number of variables. Changing the input state of the X function always changes the output state.

8) A synthesis of two X-functions of n variables is performed by concatenating (*) the qubit vectors of the X-functions of the n-1 variable:

\[ Q^{2X}(n) = Q^X(n-1) \ast Q^X(n-1) \lor Q^X(n-1) \ast Q^X(n-1). \]
The structure of the sequencer for the synthesis of the qubit coverage of the X function of n variables is shown in Fig. 8.

Fig.8. Sequencer for the synthesis of X-function Q-coverage

Another interpretation of the X-function is related to their identification with respect to the first and last bit of the qubit coverage: 01, 10, 00, 11. Then the synthesis of the qubit vectors of the X-functions of k variables can be performed by applying the concatenation operation to two Q-coverages of the X-functions of k-1 variables:

\[
Q_k^{01} = Q_{k-1}^{01} * Q_{k-1}^{10};
Q_k^{10} = Q_{k-1}^{10} * Q_{k-1}^{00};
Q_k^{00} = Q_{k-1}^{01} * Q_{k-1}^{00};
Q_k^{11} = Q_{k-1}^{11} * Q_{k-1}^{01}.
\]

Naturally, there is a one-to-one inversion relationship between two qubit coverages of X-functions of k variables:

\[
Q_k^0 = Q_{k-1}^1;
Q_k^1 = Q_{k-1}^0;
Q_k^0 = \overline{Q_{k-1}^1};
Q_k^1 = \overline{Q_{k-1}^0}.
\]

This means that if one X-function of k variables is known, then it is easy to obtain the second function, as a binary complement to the first one.

9) Any input pattern of the X-function detects 50% of faults on external inputs, which are inverted to the states of fault-free lines. Two mutually inverse test patterns detect all stuck-at faults of the input variables and output of the X-function of an odd number of variables.

10) The deductive formula of the X function transports the symmetric difference of the input fault lists to the output. This means combining the input lists of detected faults, except when the fault lists at all inputs are identical.

11) The test of stuck-at faults of all lines of a logical X-function of n variables is its PDNF complemented by any term of negated PDNF of this function: \( T = T^1 \lor T^0 \). The dimension of the complete test of the X-function is always equal to \( Q = 1 + \frac{1}{2} \times 2^n \).

12) The coordinate-wise xor-sum of all codes of the X-function corresponding to the PDNF (inverse PDNF) is equal to the zero all coordinates vector.

Thus, logical X-functions with unique test properties can be used for the synthesis of testable and self-repairable logic digital devices, and also for transporting faults from external inputs to the outputs of a Boolean structure.

3 Conclusion

1) A structural model for the interaction of X-functions and derivative components aimed at the synthesis and analysis of digital systems has been developed in order to obtain testable solutions related to the reduction of the design and testing time of computing devices.

2) The concept of simple X-functions of a finite number of variables, which are characterized by the absence of minimization and the presence of testability properties is introduced for the first time, which makes it possible to synthesize digital devices, applicable for solving testing, simulation and diagnosing problems.

3) Metric properties of X-functions of a finite number of variables are formulated, which make it possible to leverage them in developing testable digital devices, generating tests and evaluating their quality by deductive simulation of stuck-at faults on qubit data structures.

4) An analytical expression for the synthesis of qubit coverages of X-functions of a finite number of variables is proposed, which makes it possible to create testable logic circuits that do not require exponential costs for the synthesis and analysis of tests for diagnosing faults.

5) Deductive formulas for transporting input fault lists to external outputs for X-functions of a finite number of variables are obtained, which are characterized by unit vectors of derivatives with respect to all variables that makes it possible to build a sequencer for fault simulation, invariant to the input test patterns.

6) Further research will be focused on the transformation of logic circuits to the form of X-functions, which are technologically convenient for solving the problems of testing and verification of digital systems.

4 References