Elena Zaitseva - Miroslav Kvassay - Vitaly Levashenko - Jozef Kostolny *

RELIABILITY ANALYSIS OF LOGIC NETWORK WITH MULTIPLE OUTPUTS

Reliability analysis of a logic network with multiple outputs is considered in this paper. One of the principal tasks of reliability engineering is identification of those system components that have the most influence on the system activity. There exist several measures that are used for this purpose. One of them is Structural Importance Measure (SIM) which focuses on topological importance of individual system components. This measure is calculated from the system structure function, i.e. function that defines dependency between system activity and activity of its components, using logical differential calculus. In this paper, we present a method that can be used to identify the structure function of a logic network with multiple outputs and, based on logical differential calculus, we propose several definitions of the SIM for this type of systems. As a case study, the topological analysis of a one-bit full adder is considered in the last part of this paper. This study demonstrates usefulness of our approach in reliability analysis of logic networks and indicates that its further development and practical implementation could be beneficial.

Keywords: *Logic network, structure function, availability, logical differential calculus, structural importance measure.*

1. Introduction

Reliability has been considered as an important design measure in many technical systems [1 - 6]. A logic network is one of them [1, 4 - 6]. One of principal problems in reliability analysis of a logic network is investigation of influence of breakdown of each gate upon the network failure [1, 5 and 6]. The system reliability modeling and calculation of reliability indices and measures are principal steps in such analysis.

A real system contains a lot of components. From reliability point of view, the system and all its components can be in one of two possible states: functional (presented as 1) and failed (presented as 0). The dependency between system state and states of its components is defined by the structure function [7 and 8]:

$$\phi(x_1,x_2,...,x_n) = \phi(\boldsymbol{x}): \{0,1\}^n \rightarrow \{0,1\}, \quad (1)$$

where n is the number of system components, x_i is the state of component i, for $i = 1,2,...,n$, and $\boldsymbol{x} = (x_1, x_2,..., x_n)$ is a vector of components states (state vector).

The structure function of many systems is monotonic. This means that there exists no situation in which the failure of some components causes the system repair. This type of systems is known as coherent. Coherency is a typical property of many systems in reliability analysis. However, there exist some systems that are noncoherent [9] - [12]. Their structure function is non-monotonic, which means that there exist situations in which the failure of some component results in system repair. The logic networks are typical example of such systems and, therefore, classical approaches of reliability engineering cannot be used in their analysis [12].

The structure function does not take the availabilities of individual system components into account and, therefore, it allows analyzing only topological characteristics of the system. When other aspects of system availability have to be studied, then the probabilities of working/non-working state of every component should be known:

$$p_i = \Pr\{x_i = 1\}, \quad q_i = \Pr\{x_i = 0\}, \quad p_i + q_i = 1. \quad (2)$$

Probabilities p_i and q_i are known as the availability and unavailability of component i.

When the system structure function and availabilities of all system components are known, then system availability and unavailability can be computed as follows [7 and 8]:

$$A = \Pr\{\phi(\boldsymbol{x}) = 1\}, \quad U = \Pr\{\phi(\boldsymbol{x}) = 0\}, \quad A + U = 1. \quad (3)$$

The availability is one of the most important characteristics of any system because it defines the proportion of time in which the steady-state system will be working. It can also be used to compute other reliability characteristics, e.g. mean time to failure, mean time to repair, some types of importance measures, etc. [7].

* **Elena Zaitseva, Miroslav Kvassay, Vitaly Levashenko, Jozef Kostolny**
Department of Informatics, Faculty of Management Science and Informatics, University of Zilina, Slovakia
E-mail: elena.zaitseva@fri.uniza.sk

2. Logic Network with Multiple Outputs

In reliability analysis of logic networks, the studied system is a logic network of k logic inputs and n logic gates, which realizes a logic function:

$$F(y_1, y_2,..., y_k) = F(\boldsymbol{y}): [0, 1]^k \rightarrow [0, 1], \quad (4)$$

where k is a number of input signals, y_l is the l-th variable of the logic function and it corresponds to the l-th input of the logic network, for $l = 1, 2,...,k$, and $\boldsymbol{y} = (y_1, y_2,..., y_k)$ is a vector of input signals (input vector).

A lot of logic networks have more than one output and realize a set of logic functions. Therefore, the general logic network is a realization of m-dimensional vector logic function:

$$\boldsymbol{F}(y_1, y_2,..., y_k) = \boldsymbol{F}(\boldsymbol{y}): [0, 1]^k \rightarrow [0, 1]^m, \quad (5)$$

where k is a number of input signals and m is a number of output signals.

With relation to reliability analysis, a logic network has two different types of components:

- n logic gates – they can be working or failed;
- k inputs – they can be correct or incorrect.

In paper [12], there has been considered assumption that the input signals are always correct and, therefore, only logic gates are relevant system components. Using this assumption, the structure function of a logic network and the *Structural Importance Measure* (SIMs) of individual logic gates have been defined. However, they have been proposed for networks with only one output. Now, we concentrate on a logic network with m outputs.

3. Reliability Analysis of Logic Network with Multiple Outputs

A. Structure Function

The output of a real logic network is determined not only by values of inputs but also by proper work of individual logic gates. Therefore, the real output of a logic network has to be defined by logic function $\boldsymbol{F}_o(\boldsymbol{y}; \boldsymbol{x})$ which takes into account not only the values of the input signals (input vector $\boldsymbol{y}$) but also the states of logic gates of the network (state vector $\boldsymbol{x}$) [5]:

$$\boldsymbol{F}_o(\boldsymbol{y}; \boldsymbol{x}): [0, 1]^{k+n} \rightarrow [0, 1]^m, \quad (6)$$

where k is a number of input signals, m is a number of output signals, and n is a count of logic gates that are used in the logic network.

When the expected output $\boldsymbol{F}(\boldsymbol{y})$ and real output $\boldsymbol{F}_o(\boldsymbol{y}; \boldsymbol{x})$ of a logic network are known, then its availability can be defined as probability that these two output signals have the same values [5]:

$$A = \Pr[\boldsymbol{F}_o(\boldsymbol{y}; \boldsymbol{x}) = \boldsymbol{F}(\boldsymbol{y})]. \quad (7)$$

Using relation between the system structure function and its availability (3), the structure function of a logic network with m outputs can be defined in the following way:

$$\phi(\boldsymbol{x}; \boldsymbol{y}) = \boldsymbol{F}_o(\boldsymbol{y}; \boldsymbol{x}) \leftrightarrow \boldsymbol{F}(\boldsymbol{y}), \quad (8)$$

where $\leftrightarrow$ is the symbol of logical biconditional and it can be interpreted in terms of vector logic functions as follows:

$$\boldsymbol{F}_o(\boldsymbol{y}; \boldsymbol{x}) \leftrightarrow \boldsymbol{F}(\boldsymbol{y}) = [F_{1,o}(\boldsymbol{y}; \boldsymbol{x}) \leftrightarrow F_1(\boldsymbol{y})] \wedge [F_{2,o}(\boldsymbol{y}; \boldsymbol{x}) \leftrightarrow F_2(\boldsymbol{y})] \wedge ... \wedge [F_{m,o}(\boldsymbol{y}; \boldsymbol{x}) \leftrightarrow F_m(\boldsymbol{y})], \quad (9)$$

where $F_t(\boldsymbol{y})$ is the t-th element of the vector logic function $\boldsymbol{F}(\boldsymbol{y})$, i.e. $F_t(\boldsymbol{y})$ represents the expected value of the t-th output signal, and $F_{t,o}(\boldsymbol{y}; \boldsymbol{x})$ is the t-th element of the vector logic function $\boldsymbol{F}_o(\boldsymbol{y}; \boldsymbol{x})$ that defines the real value of the t-th output signal ($t = 1,2,...,m$).

B. Substructure Functions of Logic Network with Multiple Outputs

The structure function (8) and (9) allows analyzing the correlation between values of input signals and the correct work of the network (i.e. correct work of all outputs) on one side and correlation between activity of logic gates and the correct work of all outputs on the other side. However, it can also be useful to analyze these correlations with respect to only one output. Therefore, using notation (9), we can define m substructure functions:

$$\phi_t(\boldsymbol{x}; \boldsymbol{y}) = [F_{t,o}(\boldsymbol{y}; \boldsymbol{x}) \leftrightarrow F_t(\boldsymbol{y})], \text{ for } t = 1,2,...,m, \quad (10)$$

that describe the relation between the correct work of the t-th output and values of input signals and operability of individual logic gates.

Clearly, according to the previous paragraphs, the structure function of a general logic network can also be defined in the following way:

$$\phi(\boldsymbol{x}; \boldsymbol{y}) = \phi_1(\boldsymbol{x}; \boldsymbol{y}) \wedge \phi_2(\boldsymbol{x}; \boldsymbol{y}) \wedge ... \wedge \phi_m(\boldsymbol{x}; \boldsymbol{y}). \quad (11)$$

According to the previous formula, the structure function of a logic network with multiple outputs can be simply derived from substructure functions of individual outputs.

C. Unreliable Logic Gates

Every logic gate realizes some logic function (e.g. AND, OR, etc.). However, this is true if the gate is functional. Now, assume that the failed gate in mathematical interpretation generates signals that can also be interpreted as values 0 or 1. This assumption implies that the broken gate realizes a logic function too, but it is different from the original one. Therefore, the failure

of a logic gate can be modeled as a change of function realized by the gate [6 and 12].

Consider a logic network of n logic gates. The i-th gate of the network realizes two different functions depending on the state (functional/failed) of the gate: (i) when the gate is functional, then it implements function $f_i(\boldsymbol{y})$, (ii) when it is broken, then it realizes function $f_{i,u}(\boldsymbol{y})$. Therefore, the unreliable logic gate is a realization of function $f_{i,o}(\boldsymbol{y}; x_i)$ [12]:

$$f_{i,o}(\boldsymbol{y};x_1) = x_i f_i(\boldsymbol{y}) \vee \overline{x}_i f_{i,u}(\boldsymbol{y}). \quad (12)$$

When functions $f_{i,u}(\boldsymbol{y})$ are known for all logic gates of the network, function $F_{t,o}(\boldsymbol{y}; \boldsymbol{x})$, which defines the real value of the t-th output of the network, can be obtained simply by replacement of every logic gate in the scheme of the network by functions $f_{i,o}(\boldsymbol{y})$ of individual gates and then the substructure functions (10) and the structure function (11) can be identified.

D. Direct Partial Logic Derivatives

Direct Partial Logic Derivatives (DPLDs) are part of logical differential calculus that has been developed to analyze dynamic properties of Boolean functions [13]. The structure function can also be interpreted as a Boolean function and, therefore, DPLDs can be used in reliability analysis [8].

In paper [12], two types of DPLDs of the structure function have been considered in the analysis of a logic network. The first one is defined as follows:

$$\frac{\partial \phi(j \to \overline{j})}{\partial x_i(a \to \bar{a})} = \{\phi(a_i, \boldsymbol{x}; \boldsymbol{y}) \leftrightarrow j\} \wedge \{\phi(\overline{a}_i, \boldsymbol{x}; \boldsymbol{y}) \leftrightarrow \overline{j}\}, \quad (13)$$

where $\phi(a_i, \boldsymbol{x}; \boldsymbol{y}) = \phi(x_1, x_2, ..., x_{i-1}, a, x_{i+1}, ..., x_n; \boldsymbol{y})$, $a, j \in \{0, 1\}$.

DPLD (13) identifies situations when the failure/repair of given component results system failure/repair [8, 11 and 12]. However, the structure function of a logic network depends not only on states of individual components (logic gates) but also on the values of individual input signals. Therefore, we can define another logic derivative that analyzes situations in which the change of given input results the failure/repair of the logic network:

$$\frac{\partial \phi(j \to \overline{j})}{\partial y_l(a \to \bar{a})} = \{\phi(\boldsymbol{x}, a_l; \boldsymbol{y}) \leftrightarrow j\} \wedge \{\phi(\boldsymbol{x}, \overline{a}_l; \boldsymbol{y}) \leftrightarrow \overline{j}\}, \quad (14)$$

where $\phi(\boldsymbol{x}; a_l, \boldsymbol{y}) = \phi(\boldsymbol{y}; y_1, y_2, ..., y_{l-1}, a, y_{l+1}, ..., y_n)$, $a, j \in \{0, 1\}$.

These two DPLDs can also be defined for individual substructure functions of the system. In this case, DPLD (13) has the following form:

$$\frac{\partial \phi_t(j \to \overline{j})}{\partial x_i(a \to \bar{a})} = \{\phi(a_i, \boldsymbol{x}; \boldsymbol{y}) \leftrightarrow j\} \wedge \{\phi(\overline{a}_i, \boldsymbol{x}; \boldsymbol{y}) \leftrightarrow \overline{j}\}, \quad (15)$$

and it can be used to detect situations in which the failure/repair of given logic gate results in the failure of the t-th output, i.e. situations when the real value of the t-th output is different from the expected one.

Similarly, DPLD (14) can be defined for system substructure function as follows:

$$\frac{\partial \phi_t(j \to \overline{j})}{\partial y_l(a \to \bar{a})} = \{\phi_t(\boldsymbol{x}, a_l; \boldsymbol{y}) \leftrightarrow j\} \wedge \{\phi(\boldsymbol{x}, \overline{a}_l; \boldsymbol{y}) \leftrightarrow \overline{j}\}. \quad (16)$$

Derivatives (13) – (16) are very similar, but there is a principal difference in their meaning and use. DPLDs (13) and (15) analyze the impact of the failure/repair of given component and, therefore, they can be used in importance analysis to find components with the most influence on the system proper work. On the other hand, DPLDs (14) and (16) are useful in the creation of test cases for detection of failed logic gates, because, when some gates are failed, then these DPLDs reveal situations in which the change of given input signal causes the change of the value of the structure function [12 and 14].

E. Structural Importance Measure

System availability (3) is very important measure that defines the probability that the system is working. However, it does not allow us to find the influence of individual system components on the system activity, i.e. to identify which components are the most important for proper work of the system. For this purpose, there exist other measures that are known as *Importance Measures* (IMs) [15].

One of the basic IMs is *Structural Importance Measure* (SIM) that estimates the topological influence of given component on the system work. For noncoherent systems, it is defined as the relative number of situations in which the change of system component state (component failure/repair) results in the system failure [11].

In paper [12], two types of the SIM for logic network with one output have been considered. The first one estimates the topological importance of given component when the exact values of input signals are known. This SIM is defined for given logic gate and given vector $\boldsymbol{s}$ of input signals as follows:

$$SIM_{i,y=s} = \frac{\rho(\partial \phi_{y=s}(1 \to 0) / \partial x_i(1 \to 0))}{2^{n-1}} + \\ + \frac{\rho(\partial \phi_{y=s}(1 \to 0) / \partial x_i(1 \to 0))}{2^{n-1}}, \quad (17)$$

where $\phi_{y=s}(\boldsymbol{x}) = \phi(\boldsymbol{x}; \boldsymbol{y} = \boldsymbol{s}) = \phi(\boldsymbol{x}; s_1, s_2, ..., s_k)$ and $\rho(.)$ is the function that returns the number of state vectors for which the argument has nonzero, i.e. true value. For example $\rho(x_1 \vee x_2) = 3$, because the logic function $x_1 \vee x_2$ is true for 3 state vectors, i.e. (0,1), (1,0) and (1,1); $\rho(x_1 x_2) = 1$, because the logic function $x_1 x_2$ has true value only for state vector (1,1).

The SIM (17) permits to identify which gates are the most important for given values of input signals, but, it does not allow analyzing the overall influence of given gate on the proper work of the network. For this purpose, another type of the SIM has been defined in paper [12]. Using the SIM (17), this IM is defined in the following way:

$$SIM_i = \sum_{s\in\{0,1\}^k} \Pr\{y = s\} SIM_{i,y=s}, \tag{18}$$

where $[0, 1]^k$ is the space of all possible input signals.

In the case of a logic network with multiple outputs, definitions (17) and (18) can be used in two ways. Firstly, they can be used with the structure function (8) of the network. In this case, they have the same meaning as those proposed in paper [12] because they analyze the influence of given gate on the whole logic network. On the other hand, the structure function $\phi_{y=s}(\boldsymbol{x})$ in (17) can be replaced by the substructure function $\phi_{t,y=s}(\boldsymbol{x}) = \phi_t(\boldsymbol{x}; \boldsymbol{y} = \boldsymbol{s}) = \phi_t(\boldsymbol{x}; s_1, s_2, ..., s_k)$. In this case, SIM (17) will identify the influence of gate i on the correct value of the t-th output when individual input signals have values $s_1, s_2, ..., s_k$ and SIM (18) will estimate the total topological influence of component i on the correct value of the t-th output.

4. Case Study: One-bit Full Adder

F. Structure and Substructure Functions

Consider a one-bit full adder that is implemented according to the scheme depicted in Fig. 1. It has three input signals where y_1 and y_2 represent bit operands and y_3 represents a bit carried from the previous less significant stage, and 2 outputs whose behavior is defined by functions $F_1(\boldsymbol{y})$ (an output bit) and $F_2(\boldsymbol{y})$ (a carry out bit).

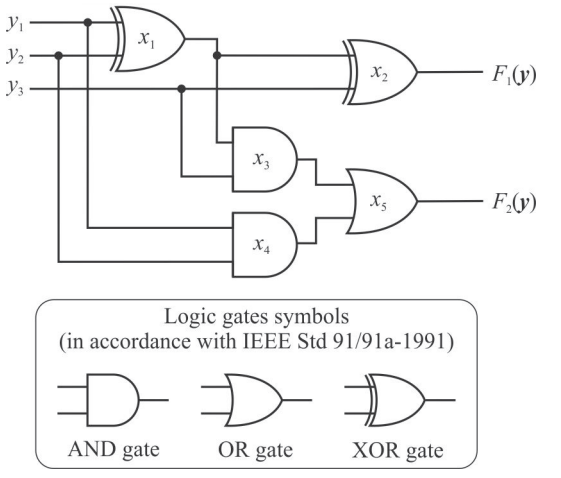

Fig. 1. A one-bit full adder

According to the scheme in Fig. 1, the outputs of the adder are realized as follows:

$$\begin{aligned} F_1(\boldsymbol{y}) &= \mathrm{XOR}(\mathrm{XOR}(y_1, y_2), y_3) = (y_1 \oplus y_2) \oplus y_3, \\ F_2(\boldsymbol{y}) &= \mathrm{OR}(\mathrm{AND}(\mathrm{XOR}(y_1, y_2), y_3), \\ AND &= (y_1, y_2)) = (y_1 \oplus y_2) y_3 \vee y_1 y_2, \end{aligned} \tag{19}$$

where symbol $\oplus$ denotes exclusive or (XOR).

Equations (19) define the expected values of the output signals. Now, we need to find their real values. This can be done simply by replacement of every logical operator using (12). So, we get the next formula for the first output:

$$\begin{aligned} F_{1,o}(\boldsymbol{y};\boldsymbol{x}) &= x_2 \mathrm{XOR}(x_1 \mathrm{XOR}(y_1, y_2) \vee \overline{x_1} f_{1,u}(y_1, y_2) y_3) \\ &\vee \overline{x_2} f_{2,u}(x_1 \mathrm{XOR}(y_1, y_2) \vee \overline{x_1} f_{1,u}(y_1, y_2) y_3), \end{aligned} \tag{20}$$

where $f_{1,u}(\boldsymbol{y})$ represents the logic function that is realized by the 1-st logic gate when it is failed and $f_{2,u}(\boldsymbol{y})$ represents the logic function realized by the failed 2-nd gate. Similarly, we can identify the real value of the second output.

There are different types of failures in logic networks [5 and 16]. For the simplicity, assume that the failure of the XOR gates results that they will generate only value 1 regardless of the values of input signals while the failure of the AND and OR gates causes that they will generate only 0-signal on the output. Using this assumption, the real outputs of the adder are:

$$\begin{aligned} F_{1,o}(\boldsymbol{y};\boldsymbol{x}) &= x_2((x_1(y_1 \oplus y_2) \vee \overline{x_1}) \oplus y_3) \vee \overline{x_2}, \\ F_{2,o}(\boldsymbol{y};\boldsymbol{x}) &= x_5(x_3(x_1(y_1 \oplus y_2) \vee \overline{x_1}) y_3 \vee x_4 y_1 y_2). \end{aligned} \tag{21}$$

In the next step, the substructure functions $\phi_1(\boldsymbol{x}; \boldsymbol{y})$ and $\phi_2(\boldsymbol{x}; \boldsymbol{y})$ can be identified by comparing functions $F_{1,o}(\boldsymbol{y}; \boldsymbol{x})$ and $F_{2,o}(\boldsymbol{y}; \boldsymbol{x})$ from (21) with functions $F_1(\boldsymbol{y})$ and $F_2(\boldsymbol{y})$ from (19) according to definition (10) of the substructure function. After using some rules of Boolean algebra, the following equations can be obtained:

$$\begin{aligned} \phi_1(\boldsymbol{x};\boldsymbol{y}) &= x_1 x_2 \vee x_2 y_1 \overline{y_2} \vee x_2 \overline{y_1} y_2 \vee \overline{x_2} y_1 y_2 y_3 \vee \\ &\overline{x_2}\,\overline{y_1}\,\overline{y_2} y_3 \vee y_1 \overline{y_2}\,\overline{y_3} \vee \overline{y_1} y_2 \overline{y_3}, \\ \phi_1(\boldsymbol{x};\boldsymbol{y}) &= x_1 x_3 x_5 \overline{y_1} \vee \overline{x_1} x_3 x_5 y_2 y_3 \vee x_3 x_5 y_1 \overline{y_2} \vee \\ &\overline{x_3}\,\overline{y_1}\,\overline{y_2} \vee x_4 x_5 y_1 y_2 \vee \overline{x_5}\,\overline{y_1}\,\overline{y_2} \vee \overline{y_1}\,\overline{y_3} \vee \overline{y_2}\,\overline{y_3}. \end{aligned} \tag{22}$$

The first (second) equation identifies all conditions that ensure that the first (second) output will generate the correct value. For example, according to the first equation, the first output is correct when both XOR gates are working $(x_1 x_2)$, or the second XOR gate is working and values of the 1-st and 2-nd input are 1 and 0 respectively $(x_2 y_1 \overline{y_2})$, or the second XOR gate is working and values of the 1-st and 2-nd input are 0 and 1 respectively $(x_2 \overline{y_1} y_2)$, or the second XOR gate is failed and all input signals have value 1 $(\overline{x_2} y_1 y_2 y)$, etc.

The previous equations define the relations between states of individual logic gates, values of input signals and correctness of individual outputs and, therefore, they are useful for analysis of this kind of dependencies. However, they do not allow us to evaluate the reliability of the network as a whole. For this purpose, the structure function has to be found. This can be done using definition (11) of the structure function:

$$\phi(\boldsymbol{x};\boldsymbol{y}) = x_1x_2x_3x_5\overline{y_1} \vee x_1x_2x_3x_5y_1\overline{y_2} \vee x_1x_2\overline{x_3}\,\overline{y_1}\,\overline{y_2} \vee x_1x_2x_4x_5y_1y_2 \vee x_1x_2\overline{x_5}\,\overline{y_1}\,\overline{y_2} \vee x_1x_2\overline{y_1}\,\overline{y_3} \vee x_1x_2\overline{y_2}\,\overline{y_3} \vee x_2x_3x_5y_1\overline{y_2} \vee \overline{x_1}x_2x_3x_5\overline{y_1}y_2y_3 \vee \overline{x_1}\,\overline{x_2}x_3x_5y_1y_2y_3 \vee \overline{x_2}x_4x_5y_1y_2y_3 \vee x_1\overline{x_2}x_3x_5\overline{y_1}y_2y_3 \vee \overline{x_2}\,\overline{x_3}\,\overline{y_1}y_2y_3 \vee \overline{x_2}x_5\overline{y_1}y_2y_3 \vee y_1\overline{y_2}\,\overline{y_3} \vee \overline{y_1}y_2\overline{y_3} \quad (23)$$

According to definition (3), the overall network availability, i.e. the probability that both outputs are correct, can be computed from formula (23) if the availabilities of individual logic gates and probabilities of individual values of input signals are known.

G. Topological Analysis of the One-bit Full Adder

Consider the one-bit full adder in Fig. 1. Using equations (17) and (18), the topological importance of every gate can be computed if the probabilities of values of input signals are known. For this purpose assume that the input signals have probabilities defined in Table 1.

When we want to analyze the topological importance of individual logic gates, the values of SIMs (17) have to be computed for every combination of input signals for every component. This implies that DPLDs (13) should be computed and, then, numbers of state vectors, for which the DPLDs are true, have to be identified. For illustration, these numbers are presented in Table 2 for the first 3 components.

The Probabilities of Individual Values of Input Signals of the One bit-Adder — Table 1

Input signal	Probability of value 0	Probability of value 1
y_1	0.5	0.5
y_2	0.5	0.5
y_3	0.75	0.25

When the DPLDs and numbers of their nonzero elements are computed, then we can calculate the SIMs (17) for individual logic gates (Table 3).

Finally, using the probabilities in Table 1 and the SIMs in Table 3, the overall SIMs (18) of individual logic gates can be computed. These values are computed in Table 4.

SIMs of Logic Gates for Individual Input Signals — Table 3

Vectors of Input Signals	$SIM_{1,y=s}$	$SIM_{2,y=s}$	$SIM_{3,y=s}$	$SIM_{4,y=s}$	$SIM_{5,y=s}$
(0,0,0)	0.5	0.5	0	0	0
(0,0,1)	0.625	0.375	0.125	0	0.125
(0,1,0)	0	0	0	0	0
(0,1,1)	0	0.25	0.25	0	0.25
(1,0,0)	0	0	0	0	0
(1,0,1)	0	0.25	0.25	0	0.25
(1,1,0)	0.125	0.125	0	0.125	0.125
(1,1,1)	0.1875	0.1875	0.0625	0.3125	0.4375

The Overall SIMs of Logic Gates in the One-bit Adder — Table 4

Logic Gate	SIM_i
x_1	0.167969
x_2	0.183594
x_3	0.042969
x_4	0.042969
x_5	0.089844

According to Table 4 the most important components of the one-bit full adder are the first and second XOR gate, i.e. gates 1 and 2, while gates 3 and 4 (AND gates) have the smallest

Numbers of Nonzero Elements of Individual DPLDs — Table 2

Vectors of Input Signals	$\rho\left(\frac{\partial \phi_{y=s}(1\to 0)}{\partial x_1(1\to 0)}\right)$	$\rho\left(\frac{\partial \phi_{y=s}(1\to 0)}{\partial x_1(1\to 0)}\right)$	$\rho\left(\frac{\partial \phi_{y=s}(1\to 0)}{\partial x_2(1\to 0)}\right)$	$\rho\left(\frac{\partial \phi_{y=s}(1\to 0)}{\partial x_2(1\to 0)}\right)$	$\rho\left(\frac{\partial \phi_{y=s}(1\to 0)}{\partial x_3(1\to 0)}\right)$	$\rho\left(\frac{\partial \phi_{y=s}(1\to 0)}{\partial x_3(1\to 0)}\right)$
(0,0,0)	8	0	8	0	0	0
(0,0,1)	10	0	0	6	0	2
(0,1,0)	0	0	0	0	0	0
(0,1,1)	0	0	4	0	4	0
(1,0,0)	0	0	0	0	0	0
(1,0,1)	0	0	4	0	4	0
(1,1,0)	2	0	2	0	0	0
(1,1,1)	2	1	0	3	1	0

influence. Therefore, we should focus on the XOR gates in other phases of reliability analysis.

5. Conclusion

Logic networks are special type of systems from reliability point of view because their structure function depends not only on states of their components (logic gates) but also on other characteristics that can be identified as the environment influence. This influence is included in input signals whose values do not depend on the network properties but on the environment in which the network is situated.

Most techniques of reliability engineering assume that the studied system is coherent. However, this assumption is not valid for logic networks because there can exist situations when the failure of some logic gate results that the network begin generate the correct output signal, while, before the failure, the output signal has been incorrect [12]. Therefore, the reliability analysis of logic networks is more complicated than analysis of other types of systems.

One of the principal steps of reliability analysis is identification of the system structure function. In paper [12], there has been proposed a method for this task when the analyzed system is a logic network with one output. That method is based on the assumption that a real logic gate is unreliable and, therefore, its output depends on whether it is working or failed. In this paper, we generalized this concept on logic networks with multiple outputs.

Two types of structure functions can be identified in a logic network with multiple outputs. The first one is the structure function of the whole network that characterizes the network as a whole because it defines the dependency between proper work of logic gates and the correct values of all outputs. This structure function can be used to estimate network availability or to analyze the importance of individual logic gates for the proper work of the network.

The second ones are the substructure functions of individual network outputs. They define the correlation between the proper work of individual logic gates and the correct value of one output signal. These functions can be used to evaluate the influence of logic gates on the value of studied output signal and, therefore, they can be used in importance analysis or in creating scenarios for identification of failed logic gates.

In the last part of this paper, we focused on the use of the structure and substructure functions in importance analysis. We proposed the definitions of the SIM for logic networks with multiple outputs. Our definitions are based on logical differential calculus and they allow identify which components have the most influence on the proper work of the whole network or on the correct value of one concrete output signal from topological point of view. Although results of this paper have more theoretical significance, a case study considered at the end of this paper indicates that our approach is useful and its further development and practical implementation could be beneficial for reliability analysis of logic networks.

Acknowledgment

This work was partially supported by the research grants of Slovak Research and Development Agency SK-PL-0023-12 and VEGA 1/0498/14.

References

[1] SHOOMAN, M. L.: *Reliability of Computer Systems and Networks: Fault Tolerance, Analysis, and Design*. New York, : John Wiley & Sons, 2002.

[2] RADOS, I., SCHWARTZ, L.: The Worst Availability as a Parameter for Designing and Reporting on the Network Performances, *Communications - Scientific Letters of the University of Zilina*, vol. 13, No. 1, pp. 60-66, 2011.

[3] BRIS, R.: Assessment of the Availability of an Offshore Installation by Stochastic Petri Net Modeling, *Communications - Scientific Letters of the University of Zilina*, vol. 16, No. 1, pp. 90-96, 2014.

[4] TAYLOR, E., FORTES, J.: *Faults, Error Bounds and Reliability of Nanoelectronic Circuits,* 2005 IEEE Int. Conf. Appl. Syst. Archit. Process., pp. 247-253, Jul. 2005.

[5] FRANCO, D. T., VASCONCELOS, M. C., NAVINER, L., NAVINER, J.-F.: Signal Probability for Reliability Evaluation of Logic Circuits, *Microelectron. Reliab.*, vol. 48, No. 8-9, pp. 1586-1591, Aug. 2008.

[6] HAN, J., CHEN, H., BOYKIN, E., FORTES, J.: Reliability Evaluation of Logic Circuits Using Probabilistic Gate Models, *Microelectron. Reliab.*, vol. 51, No. 2, pp. 468-476, Feb. 2011.

[7] RAUSAND, M., HØYLAND, A.: *System Reliability Theory: Models, Statistical Methods, and Applications*. Haboken, : John Wiley & Sons, Inc., 2004.

[8] ZAITSEVA, E. N., LEVASHENKO, V. G.: Importance Analysis by Logical Differential Calculus, *Autom. Remote Control*, vol. 74, No. 2, pp. 171-182, Feb. 2013.

[9] ANDREWS, J. D., BEESON, S.: Birnbaum's Measure of Component Importance for Noncoherent Systems, *IEEE Trans. Reliab.*, vol. 52, No. 2, pp. 213-219, Jun. 2003.

[10] BEESON, S., ANDREWS, J. D.: Importance Measures for Non-coherent-system Analysis, *Reliab. IEEE Trans.*, vol. 52, No. 3, pp. 301-310, Sep. 2003.

[11] KOSTOLNY, J., KVASSAY M., KOVALIK, S.: Reliability Analysis of Noncoherent Systems by Logical Differential Calculus and Binary Decision Diagrams, *Communications - Scientific Letters of the University of Zilina*, vol.16, No. 1, pp. 114-120, Feb. 2014.

[12] ZAITSEVA, E., KVASSAY, M., LEVASHENKO, V., KOSTOLNY, J.: Reliability Analysis of Logic Network by Logical Differential Calculus, *ELEKTRO, 2014*, May 2014, pp. 245-250.

[13] YANUSHKEVICH, S. N., MILLER, D. M., SHMERKO, V. P., STANKOVIC, R. S.: *Decision Diagram Techniques for Micro- and Nanoelectronic Design. Handbook*. Boca Raton, FL: CRC Press, 2006.

[14] CHANGQIAN, W., CHENGHUA, W.: *A Method for Logic Circuit Test Generation Based on Boolean Partial Derivative and BDD,* World Congress on Computer Science and Information Engineering, pp. 499-504, Mar. – Apr. 2009.

[15] KUO, W., ZHU, X.: *Importance Measures in Reliability, Risk, and Optimization: Principles and Applications*. Chichester,: John Wiley & Sons, Ltd., 2012.

[16] LEVASHENKO, V., MORAGA, C., SHMERKO, V., KHOLOVINSKI, G., YANUSHKEVICH, S.: Test Algorithm for MVL Combinational Circuits, *Autom. Remote Control*, vol. 61, No. 5, pp. 844-857, May 2000.