Evaluating Real Checkability for FPGA-based Components of Safety-Related Systems

Oleksandr Drozd[a], Kostiantyn Zashcholkin[a], Maciej Dobrowolski[b], Anatoliy Sachenko[b,c], Oleksandr Martynyuk[a], Olena Ivanova[a] and Julia Drozd[a]

[a] *Odessa National Polytechnic University, 1, Shevchenko Ave., Odesa, 65044, Ukraine*
[b] *Kazimierz Pułaski Technology and Humanitarian University, 29, Malczewskiego Str., Radom, 26-600, Poland*
[c] *West Ukrainian National University, 11, Lvivska Str., Ternopil, 46009, Ukraine*

Abstract

The paper focuses on the study of the checkability of digital circuits in relation to FPGA (Field Programmable Gate Array) components of safety-related systems that serve high-risk facilities, maintaining their functional safety in synergy with its own. Functional safety breaches are associated with failures that stimulate the use of fault-tolerant solutions. However, the possibilities of these solutions are limited by the number of failures which can be countered. As a result, functional safety, based only on circuit fault tolerance, faces the problem of multiple failures. This problem manifests itself in the example of hidden faults, which can be accumulated in significant quantities during extended normal operation of the system. The multiple manifestations of these faults in emergency mode call into question the fail-safety of fault-tolerant circuits, including FPGA components, which can accumulate faults in the memory of the LUT units. Ensuring the fail-safety of circuits requires taking into account their checkability, which depends on the data arriving at the inputs of the circuit in normal and emergency modes. A method for assessing checkability, which is important for the fail-safety of FPGA components, is proposed. Checkability is assessed on real input data, the change of which often extends only over a part of the range of values related to the normal functioning of the system. The method makes it possible to evaluate the change in the checkability of the circuit depending on the change in its input data.

Keywords

Safety-related system, FPGA component, LUT memory, normal and emergency modes, multiple failures, hidden faults, fault tolerance, fail-safety, checkability

1. Introduction

The activity of mankind in production and consumption is scaled up at an increasing pace and is accompanied by the creation of powerful infrastructures for the generation and distribution of electricity and other resources of mass demand. These manufacturing and transport infrastructures are gradually becoming safety-related and transforming into high-risk facilities. This trend can be traced, since these objects are associated with the largest number of accidents that occurred with significant negative consequences [1, 2].

The development of high-risk objects can be attributed to objectively occurring processes that cannot be stopped. The rapid development of the energy sector is accompanied by an increase in the number and capacity of power plants and power grids. Growing cargo flows stimulate the development of transport infrastructures for land, air and water communications. Chemical and

COLINS-2021: 5th International Conference Computational Linguistics and Intelligent Systems, April 22–23, 2021, Kharkiv, Ukraine
EMAIL: drozd@ukr.net (O. Drozd); const-z@te.net.ua (K. Zashcholkin); m.dobrowolski@uthrad.pl (M. Dobrowolski); sachenkoa@yahoo.com (A. Sachenko); anmartynyuk@ukr.net (O. Martynyuk); en.ivanova.ua@gmail.com (O. Ivanova); yuliia.drozd@opu.ua (J. Drozd)
ORCID: 0000-0003-2191-6758 (O. Drozd); 0000-0003-0427-9005 (K. Zashcholkin); 0000-0003-0296-9651 (M. Dobrowolski); 0000-0002-0907-3682 (A. Sachenko); 0000-0003-1461-2000 (O. Martynyuk); 0000-0002-4743-6931 (O. Ivanova); 0000-0001-5880-7526 (J. Drozd)

biological production creates their own risks associated with toxic and no less hazardous products, which also need to be ensured for safe storage, transportationб use and disposal. The accumulation of various types of weapons and ammunition supplements, but does not exhaust, many of today's sources of risk. Risk assessment is based on taking into account two factors related to the probability of an accident and the cost of its possible consequences [3, 4].

The described processes demonstrate an increase in the total power of high-risk objects, which humanity is unable to refuse. This feature of our development is manifested in the constant increase in the cost factor of the consequences caused by accidents. In the current circumstances, containing the growth of risks becomes a necessary condition for survival, and the implementation of this urgent requirement is only possible by reducing the factor associated with the probability of an accident. This problem is being solved through the development of information technologies implemented in computer systems for managing high-risk facilities. With this specificity of use, computer systems exhibit features that transform them into a special type of instrumentation and control safety-related systems. These systems are aimed at ensuring functional safety, which is considered in a complex both in relation to the control object and in relation to the system itself, and serve to prevent accidents, as well as reduce losses in the event of an accident. The purpose of safety-related systems determines the features of their design, which provide for the organization of their operation in two modes: normal and emergency [5, 6].

Requirements for the functional safety of critical systems are regulated by international standards, which pay considerable attention to the challenges associated with failures [7].

Safety-related systems must resist failure, i.e., continue to properly perform their functions to ensure their own safety and the safety of control objects, even in the event of failures. Such capabilities are based on the use of fault-tolerant solutions in the design of systems and their components [8]. International standards pay the greatest attention to failures for a common cause, which arise as a result of copying circuit solutions into duplicate channels of fault-tolerant structures and can replicate failures, for example, as a result of a design error [9].

The emphasis on common cause failures is due to the limited capabilities of fault-tolerant circuits, which are designed to withstand a specified number of failures, and typically provide a response to one or two failures. A significant limitation in the number of failures to be parried is dictated both by the desire for simple solutions and by the expectation of one failure, as the most probable in relation to their greater number. Therefore, the development of fault-tolerant schemes is mainly due to the restrictions imposed on copying solutions using multi-version technologies to eliminate the common causes of replicating failures. The main direction in the development of multi-version technologies is associated with the expansion of the types of diversity for schemes that duplicate functions in fault-tolerant structures [10, 11].

Thus, the main challenge for fault-tolerant solutions used in safety-related systems is multiple failures, since in the event of multiple faults, these solutions no longer guarantee functional safety, i.e., they do not become fail-safe. For this reason, one of the most important issues in ensuring the functional safety of critical systems is to study the sources of multiple failures.

Such failures can occur due to vulnerabilities manifested in the field of information security by deliberate malicious actions that occur, for example, as a result of cyber-attacks by botnets that violate the integrity of a computer system [12–14].

For safety-related systems and control objects, emergency modes are not only the most critical, but also the least studied. The main contribution to their study is made by the ongoing accidents. Erroneous actions of the maintenance personnel lead to new unforeseen scenarios for the development of events with unexpected results, including the manifestation of hidden defects, which, if properly maintained, would not have emergency consequences [15, 16].

One of the most significant sources of multiple failures, which is inherent in fault-tolerant solutions used in critical applications [17], manifests itself in the problem of hidden faults. Such faults can accumulate in digital circuits during an extended normal mode under conditions where the input data manifesting them is not used in this mode. The emergency mode can manifest accumulated faults with the arrival of the corresponding input data in an amount exceeding the possibility of their parrying by fault-tolerant circuits [18].

Thus, the source of multiple failures is the accumulation of faults that cannot be detected in normal mode. Indeed, the detection of faults is performed by methods and means of on-line testing using

logical control. In this case, a fault can only be detected if it causes an error in the monitored result [19, 20].

The problem of hidden faults is outside the scope of international standards related to functional safety. At the same time, this source of multiple failures is directly due to the peculiarity of safety-related systems to operate in two modes. In conventional computers, hidden faults do not create the problem of multiple failures, since they remain hidden and do not have any effect on the computational process throughout the entire operating mode.

The problem of hidden faults is better known from the practice of using simulation modes, which recreate emergency conditions to control the functioning of safety-related systems and their components. Such modes are used in modern critical systems to detect hidden faults that do not appear as errors on the input data of normal mode. However, simulation modes themselves pose a risk to functional safety, as they are generators of emergency conditions. The practice of their application includes cases of both planned and unauthorized switching on, which have repeatedly led to emergency consequences. The planned activation of the simulation mode is a dangerous procedure, since it is accompanied by the shutdown of emergency protections, which was one of the reasons for the Chernobyl tragedy. Unauthorized switching on of simulation modes, which repeatedly occurs through the fault of a person or an arising malfunction, is distinguished by a factor of surprise and is also fraught with emergency consequences [21, 22].

The use of simulation modes indicates the importance of hidden faults, which are feared to a greater extent than the recreation of emergency conditions. In addition, the presence of dangerous simulation modes indicates a distrust experienced in relation to fault-tolerant solutions that do not always translate into fail-safe solutions that are important for ensuring the functional safety of critical systems. The reason for such an unwarranted transformation of fault-tolerant circuits into fail-safe is associated with their limited checkability [18].

The checkability of a digital circuit prevents the accumulation of faults, and its limitation can be inherited by the fault tolerance of circuit solutions and have a significant impact on the processes important for ensuring the functional safety of safety-related systems.

Safety-related systems are developed using the most effective technologies, proven in practice, according to the component approach [23, 24]. The development of digital components is gaining priority in FPGA designing (Field Programmable Gate Array), which combines the versatility of a hardware solution with the flexibility of programmable logic [25, 26].

Digital circuit design is performed by programming FPGA chips within their LUT-oriented (Look-Up Table) architecture. During programming, the program code is written into the memory of the LUT units. Therefore, the fail-safety of fault-tolerant FPGA projects depends on their checkability in terms of the memory of LUT units [27].

In this regard, the assessment of the checkability of FPGA projects is of great importance. The division of the operating mode of the critical system into normal and emergency is inherited by the input data. Taking into account such a division makes it possible to evaluate the checkability of the memory of LUT units [28]. However, this assessment can be significantly different from the real one and can be misleading in terms of the safety of fault-tolerant schemes.

This paper aims to draw attention to the real checkability of digital circuits in FPGA components of safety-related systems. Section 2 identifies the problem associated with evaluating the real checkability of digital circuits in FPGA components of safety-related systems. Section 3 describes a method for assessing the real checkability of digital circuits in critical applications. Section 4 presents the results of experiments to assess the real checkability of digital circuits using the example of a 4-bit mantissa multiplier, implemented in an FPGA project as a component of a safety-related system.

2. Related works and definition of problem

The suitability both natural and man-made processes and objects for checking is an important condition for survival and development. In computer technology, this trend began to develop in the testability of digital circuits, the increasing complexity of which began to impede the effective development of tests for detecting faults [29, 30].

The testability of a digital circuit is assessed using its controllability and observability, which are determined on all input data. Thus, testability is completely determined by the schema structure and therefore can be considered as structural checkability.

In the operating mode, the checkability of digital circuits is their important characteristic in relation to the implementation of on-line testing [31, 32]. Logic checking methods aimed at detecting errors in computed results are limited in their capabilities by logical checkability, which determines the ability of a digital circuit to exhibit faults in the form of errors on the input data of the operating mode. For this reason, the logic checkability of digital circuits gains an additional dependence on the input data in the operating mode. A fault for which there is no input data, manifesting it in the form of an error, is hidden to any logical checking.

However, this lack of logical checking has no consequences, since such a fault is also hidden for the entire operating mode and, therefore, harmless for the functioning of the circuit. The tolerance for hidden faults changes in safety-related systems due to the diversification of the operating mode, which is divided into normal and emergency. Diversification of the operating mode has a significant impact on many aspects in the functioning of critical systems. In particular, it corrects the purpose of on-line testing, which in emergency mode is aimed at checking the trustworthiness of the calculated results, and in normal mode it is used to clear the circuit from faults. In addition, the diversification of the operating mode is inherited by the input data, which becomes different in these modes and makes the checkability of the digital circuit correspondingly different depending on them.

So far, the checkability of a digital circuit has been viewed in terms of its positive role in developing tests and enhancing on-line testing. At the same time, checkability should be considered from the standpoint of its features, which, depending on the conditions, can have both positive and negative effects. Indeed, the checkability of a digital circuit consists in the manifestation of malfunctions in the form of an error in the calculated results. However, errors reduce the trustworthiness of the results, which, as a rule, are the main purpose of the calculations performed.

In addition, it should be noted that most often the results are distorted by transient faults, which are much more likely events than permanent faults [33]. Therefore, the high checkability of the digital circuit leads to a decrease in the trustworthiness of the results, mainly due to transient faults, the detection of which does not help to clear the circuit from malfunctions in the normal mode. Most of all, the emergency mode needs to ensure high trustworthiness of the calculated results, for which the manifestation of faults in the form of errors, due to high checkability, is undesirable.

For this reason, different checkability in normal and emergency mode is also a feature that can play both a positive and a negative role in these modes.

The best combination of the checkability of the scheme and the trustworthiness of the results is achieved when these indicators are provided, respectively, at a high and low level in normal mode and in the opposite ratio in emergency mode.

In the case where the checkability of the circuit in the normal mode includes the checkability that the circuit exhibits in the emergency mode, the problem of hidden faults does not arise since the hidden faults remain so in both modes. Otherwise, the various checkability of a digital circuit contributes to the accumulation of hidden faults that pose a threat to fault tolerance, preventing the transformation of fault-tolerant circuits into fail-safe ones.

The checkability of FPGA projects has its own characteristics associated with their LUT-oriented architecture. Computations are organized by decomposing them into logical functions performed by LUT units. The description of these logical functions is stored in the form of program code in the configuration file of the FPGA project and in the process of its programming is written into the memory of the LUT units. Inputs of the LUT units take the values of arguments on which logical functions are defined. The arguments form the address at which the function value is read from the LUT memory to the unit's output. The number of inputs of LUT units can vary from 4 to 8 for different LUT-oriented architectures. The most widespread are LUT units containing 4 inputs: A, B, C, D for addressing 16-bit memory [34, 35].

The checkability of the LUT unit is evaluated taking into account its possible faults. The structure of the LUT unit includes a register memory in which the program code is written and a circuit for selecting the bit read from this memory. The register memory of each LUT unit is regularly tested using a checksum that is generated and validated for the entire program code of the FPGA design.

The scheme for selecting the read bit of the LUT memory consists of switches, which are controlled by the address code formed at the inputs of the LUT unit. Regular testing of register memory makes it checkable in relation to all its possible faults, excluding their accumulation in normal mode. Switches of the selection circuit are not covered by program code checking and can accumulate faults on their information and control inputs.

Malfunctions, which appear in the form of errors at the information inputs of the switches, distort the values of the bits read from the register memory. Malfunctions of the control inputs lead to addressing errors, which are manifested in the case of a mismatch in the values in the bits addressed to the correct and corrupted address. Thus, the checkability of the register memory does not ensure the checkability of the LUT memory as a whole, including the circuit for selecting the bits to be read. The values of the read bits of the LUT memory can be corrupted by faults accumulated during normal operation in the switches.

The bits of the LUT memory are checkable for faults that cause an error in the monitored results. This checkability of the LUT memory plays a positive and negative role in normal mode and emergency mode, respectively. In critical systems, checkability is valuable, first of all, from the standpoint of withstanding multiple faults to transform a fault-tolerant circuit into a fail-safe one.

The checkability of an FPGA project with a LUT-oriented architecture will be estimated by the ratio of the number of checkable bits of LUT memory to their total number.

The plurality of checkable bits of a LUT memory should be determined taking into account the following features inherent in checkability of circuits in critical applications:

1. Significance of checkability of schemes for ensuring the fail-safety of a fault-tolerant solution.
2. Dependence of the checkability of the circuit on the real input data of the normal mode.

In the first feature, the checkability of the FPGA project is characterized from the standpoint of its main role, which it should play in safety-related systems to ensure the fail-safety of fault-tolerant solutions.

Circuit checkability is important to system safety in those bits of the LUT memory that are used in emergency mode and are therefore addressed in both modes. Therefore, in assessing the checkability of a circuit, not all checkable bits of the LUT memory should be considered, but only those that are used in both normal and emergency modes.

The second feature of checkability in safety-related systems is associated with the nature of the input data entering the inputs of the circuits in the normal mode.

For circuits used in critical applications, it is usually known to separate the input data according to their normal and emergency use. For example, the input data can be separated by a threshold, reaching and exceeding which marks the beginning and continuation of the emergency mode.

Raising the threshold increases the amount of input data used in normal mode and reduces the amount of input data for emergency mode. Such changes contribute to an increase in the checkability of the circuit, as well as an increase in the number of checkable memory bits in normal mode and a decrease in the number of memory bits addressed only at the beginning of the emergency mode. An increase in the checkability of schemes with an increase in the threshold is confirmed by experimental data [36].

At the same time, such an assessment of checkability has a significant drawback associated with the nature of the data arriving at the inputs of circuits in critical applications. The safety-related systems are characterized by the functions of monitoring various parameters of the control object and the system itself. Due to the high technology prioritized in critical applications, the parameters generally exhibit high stability during normal operation. The input data for digital components of critical systems is formed from the results of measurements of these parameters and inherits a high stability of values, which is characterized by minor changes in noise level. The change in the input data does not cover the entire range of values of the normal mode and, as a rule, extends to its insignificant part. This fact leads to a significant difference between the real checkability of digital circuits and checkability, estimated relative to the threshold separating the input data of normal and emergency modes.

Real checkability may turn out to be significantly lower than estimated and create a false idea of information sufficiency [37, 38] in assessing the fail-safety of a fault-tolerant FPGA component for a safety-related system.

Thus, the problem of erroneous judgment about the real checkability of circuit solutions can have a significant impact on ensuring the functional safety of safety-related systems and control objects.

3. A method of real checkability assessment for circuits with LUT-oriented architecture

The initial data for assessing the real checkability are the real input data of the circuit, its description and the threshold S, which distinguishes between normal and emergency modes.

Typically, the input data for component circuits in safety-related systems is recorded. The continuous nature of change, inherent in many parameters, is reflected in the change in numerical data with a step of 1 at the inputs of digital circuits. In this case, the change in the input data is fully characterized by the range, indicating only its boundaries. The range of input data variation can be estimated in the circuit itself and be operatively available in FPGA components, for example, by using JTAG technology. A description of a circuit with a LUT-oriented architecture can be obtained from a CAD database, for example, Compiler DB CAD Intel Quartus Prime 20.1 Lite Edition using an application in TCL language [39, 40].

The description of the circuit contains a list of LUT units with an indication of the program codes, as well as the circuit inputs and outputs of the LUT units connected to their inputs. In addition, the circuit inputs and LUT units connected to its outputs are described. The threshold S can impose constraints directly on the input data or the result calculated by the FPGA component of the system. The proposed method for assessing the real checkability of an FPGA component is based on the study of the controllability and observability of the memory LUT bits, taking into account the range of input data changes that occurs in normal mode. Checkability is considered in that part of it, which provides or is not able to provide the fail-safety of a fault tolerant solution on the real input data of the normal mode.

LUT units in the circuit description are arranged in the order in which their functions are performed, and then these functions in this order are executed during the simulation of computations.

Simulation is performed on the input data in the range of their real change in normal mode. In addition, the computations are simulated on the input data of emergency mode, which start at the S threshold value. The method simulates a circuit with a LUT-oriented architecture in two stages aimed at assessing the controllability and observability of the memory LUT bits, respectively.

At the first stage, the correct operation of the circuit is simulated. For each LUT unit, the simulation results determine two sets of P_N and P_E, which contain the bit numbers of the LUT memory addressed in normal and emergency mode, respectively. These sets are used to define the sets $P_{N\&E} = P_N \cap P_E$ and $P_{E\setminus N} = P_E \setminus P_N$ of memory LUT bits, addressed in both modes and only in emergency mode, respectively.

The use of memory LUT bits makes them controllable in the appropriate mode. The definition of the P_N and P_E sets is performed with different simulation costs.

The P_N set depends on the normal mode inputs and grows with their actual range. Therefore, this set needs to be adjusted when the ranges of the input data change. The set P_E is completely determined by the threshold S, from which the emergency mode begins, and remains constant when the input data range is changed in normal mode.

In the second stage, the input data, on which the memory LUT bits of the P_E set are defined, are used to analyze the observability of these bits. The output of the LUT unit is inverted and the circuit is simulated based on the introduced fault. An error at the output of the circuit determines the analyzed bit of the LUT memory as observable.

It should be noted that $P_{N\&E} \subset P_E$. Therefore, the results of the observability analysis also apply to the memory LUT bits of the $P_{N\&E}$ set and define two sets P_{NE} and P_{EN} of the memory LUT bits, which are controllable and observable in both modes and only in emergency mode, respectively.

The P_{NE} and P_{EN} sets of the memory LUT bits are dependent on the actual input data of normal mode. Ignoring the actual change in the input data makes it possible to judge these sets based only on the value of the threshold S. In this case, the simulation is performed taking into account all the input data of the normal mode and determines these sets as $P_{NE\,S}$ and $P_{EN\,S}$, respectively.

The memory LUT bits of the P_{NE} plurality are checkable in normal mode and thus eliminate the accumulation of hidden faults, as well as the problem of their manifestation in emergency mode. Therefore, the fail-safe circuit of the FPGA component is checkable and fail-safe in these memory LUT bits.

The degree of fail-safety of a fail-safe FPGA component can be evaluated by checkability as $C = B_{NE} / B_{EN\_S}$, where $B_{NE} = |P_{NE}|$; $B_{EN\_S} = |P_{EN\_S}|$.

The degree of fail-safety violation of the fail-safe FPGA component can be estimated by the checkability deficit using the formula $D = B_{EN} / B_{EN\_S}$, where $B_{EN} = |P_{EN}|$.

4. Case study and discussion of the method

The method for assessing the real checkability of an FPGA component is represented by its software implementation, executed in the Delphi 10 Seattle demo version [41].

The checkability of a circuit with a LUT-oriented architecture is determined for eight different values of the Sr threshold, which limits the actual change of the input data in normal mode. The Sr threshold imposes a limitation on the calculation result. The largest value of the Sr threshold coincides with the S threshold separating the normal and emergency input data.

The software implementation of the method was tested on FPGA circuits of arithmetic devices from the Library of parameterized modules [42]. The studies were carried out using CAD Quartus Prime 20.1 Lite Edition [43]. The device circuits were implemented in the Intel Cyclone 10 LP FPGA chip: 10CL025YU256I7G [44].

As an example, the study results are presented for a 4-bit iterative array multiplier that calculates the product of normalized mantissas. The circuit is implemented on 30 LUT units. The binary codes of the normalized mantissa of the factors vary from 8 to 15, and the binary code of the product ranges from Pr = 64 to 225.

The main program bar shown in Fig. 1, contains the control keys, a memory image for the selected LUT unit, the main simulation results and explanations to them. The “START” and “EXIT” keys allow you to start the simulation and end the program. Before the start of the simulation, the S threshold value is set, which determines 8 Sr threshold values, evenly distributed over the interval from Pr to S. In this case, the threshold is set by S = 136. Keys used to determine the Sr threshold values are replaced with information these values: “Sr: 64 - 73… 136”.

The “LUT # 9” key selects LUT node 9 to view its memory at all values of the Sr threshold. Pressing this key moves to the next LUT unit in a circle. The LUT memory is shown as a matrix of squares containing the bit values. The numbering of bits from 0 to 15 is shown in binary code using the values of the arguments at the inputs D, C and B, A. These values are specified in binary codes from 00_2 to 11_2 A memory bit located at the intersection of a row and a column, indicated by binary codes, has a number equal to the address at which it is fetched from the LUT memory. For example, the bit located in the third row and third column, i.e., at the intersection of codes 10_2, located at 1010_2 and has the number 10.

The memory LUT bits take on the same values in all eight matrices, but they are colored in different colors: green, yellow and blue if they are used only in normal mode, only in emergency mode, or in both modes, respectively. Below the matrices, the total number of bits used and the number of bits colored in different colors are indicated.

With an increase in the Sr threshold, bits addressed only in emergency mode are replaced by bits that are used in both modes, and then colored into bits that are used only in normal mode. This process improves the checkability of the LUT memory and helps transform fault-tolerant FPGA components into fail-safe circuits. Bits with values highlighted in red are observable only in emergency mode. They refer to non-checkable bits that can accumulate hidden faults that reduce the fail-safety of FPGA components. As the real Sr threshold decreases, LUT 9 increases their number from 2 to 11.

The main results show the number of B_N and $B_{N\&E}$ bits addressed only in normal mode and both modes, as well as the number B_{NE} and B_{EN} of checkable and non-checkable bits important to safety. In addition, the checkability C and its deficit D, expressed as a percentage, were assessed.

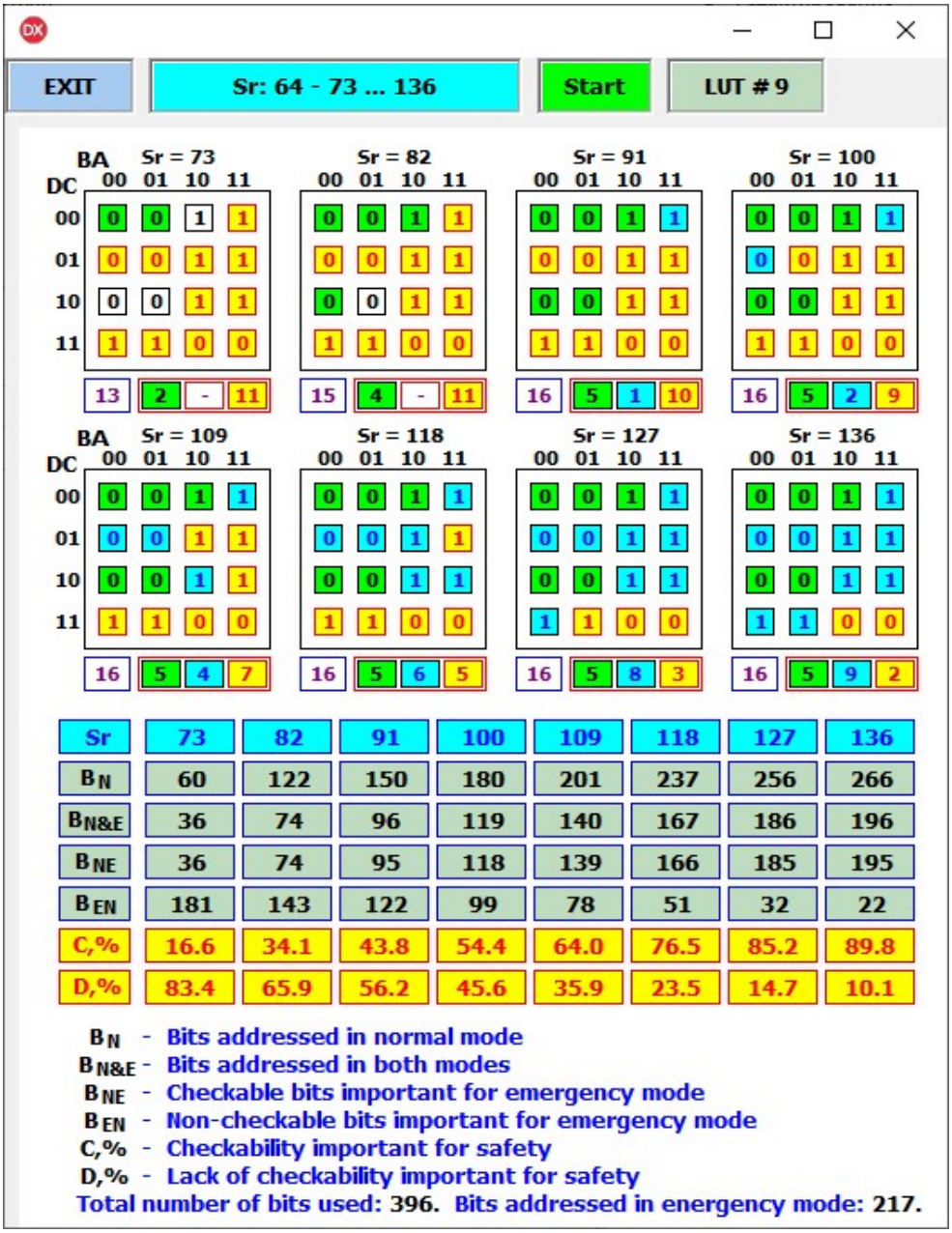

Figure 1: Main results of the method implementation

These results show a significant decrease in checkability with lowering the Sr threshold, reflecting the actual change of the input data in the normal mode. A decrease in the real Sr threshold from 136 to 73 (less than 2 times) reduces the checkability from 89.8% to 16.6%, i.e., by 5.4 times.

5. Conclusions

Fault-tolerant circuit solutions play an important role in ensuring the functional safety of critical systems and high-risk objects controlled by them. However, the fault tolerance of the circuits is ensured with respect to a limited number of faults and therefore is vulnerable to multiple failures that may arise in connection with the existing problem of hidden faults. This problem appeared along with safety-related systems, a feature of which is their designation for operation in two modes: normal and emergency. In normal mode, faults have the ability to accumulate due to the lack of input data that could manifest them. With the onset of the emergency mode, the accumulated faults create a problem

of multiple failures for fail-safe circuits, preventing their implementation in the main mission of ensuring fail-safety.

The accumulation of faults occurs due to the low checkability of digital circuits, which in normal mode, as a rule, is limited to insignificant changes in the input data. The primary sources of these data are sensors of various parameters of the control object. Thanks to high technologies, which are used as a priority in critical domains, the measurement results are highly stable and, after digitization, are transformed into a sequence of slightly varying data arriving at the inputs of digital circuits. As a result, the normal mode input range is only used up to a small fraction of it. The checkability of the schemes, assessed over the entire range of the normal mode, ceases to reflect its real level.

The development of critical systems using modern CAD systems makes the analysis of this problem relevant for FPGA components, focusing it on the checkability of the memory of LUT units. Experiments carried out with FPGA circuits have shown that their real checkability can be many times lower than the values that can be expected, taking into account the entire range of input data changes in normal mode. Under such circumstances, the capabilities of fault-tolerant circuits are significantly limited in ensuring functional safety, for which one of the challenges is the misconception concerning its indicators.

Unfortunately, the envisaged scenarios for the development of accidents are mainly replenished as a result of the analysis of new accidents. Such an expensive path to improving safety, in particular, is due to the limited depth of analysis, where the objective processes that generate these details are not traced behind individual details.

One of future research direction is exploring neural networks [45, 46] for assessing the checkability of digital circuits.

6. References

[1] A. J. Masys, Black swans to grey swans: Revealing the uncertainty, Disaster Prevention and Management 21 (3) (2012), 320–335.

[2] A. Hopkins, Issues in safety science, Safety Science 67 (2014), 6–4.

[3] T. Aven, Risk assessment and risk management: Review of recent advances on their foundation, European Journal of Operational Research 253(1) (2016) 1–13.

[4] L. Cox, Confronting deep uncertainties in risk analysis, Risk Analysis 32 (2012) 1607–1629.

[5] A. Hale, Foundations of safety science: A postscript, Safety Science 67 (2014) 64-69.

[6] R. Ouache, M.N. Kabir, A. Adham, A reliability model for safety instrumented system, Safety Science 80 (2015) 264–273. URL: https://doi.org/10.1016/j.ssci.2015.08.004.

[7] F. Brissaud, L.F. Oliveira, Average probability of a dangerous failure on demand: Different modelling methods, similar results, in: 11th International Probabilistic Safety Assessment and Management Conference and the Annual European Safety and Reliability Conference, PSAM11 ESREL, 2012, pp. 6073–6082.

[8] A. Romankevich, A. Feseniuk, V. Romankevich, T. Sapsai, About a fault-tolerant multiprocessor control system in a pre-dangerous state, in: Proceedings of the 9th IEEE International Conference DESSERT, Kyiv, Ukraine, 2018, pp. 215–219.

[9] S. Alizadeh, S. Sriramula, Impact of common cause failure on reliability performance of redundant safety related systems subject to process demand, Reliability Engineering & System Safety 172 (2018) 129–150.

[10] K. Salako, L. Strigini, When does 'Diversity' in development reduce common failures? insights from probabilistic modelling, IEEE Transactions on Dependable and Secure Computing 11(2) (2014) 193–206. doi: http://doi.ieeecomputersociety.org/10.1109/TDSC.2013.32.

[11] A. A. Gashi, L. Povyakalo, M. Strigini et. al., Diversity for safety and security in embedded systems, in: Faste Abstracts of the IEEE International Conference on Dependable Systems and Networks, Atlanta, GA, USA, 2014.

[12] S. Lysenko, K. Bobrovnikova, S. Matiukh et. al., Detection of the botnets' low-rate DDoS attacks based on self-similarity, International Journal of Electrical and Computer Engineering 10(4) (2020) 3651–3659.

[13] S. Lysenko, K. Bobrovnikova, O. Savenko, A. Kryshchuk, BotGRABBER: SVM-Based Self-Adaptive System for the Network Resilience Against the Botnets' Cyberattacks, Communications in Computer and Information Science 1039 (2019) 127–143.
[14] W. Young, N. Leveson, Systems thinking for safety and security, in: Proceedings of the 29th Annual Computer Security Applications Conference, New Orleans, LA, USA, 2013, pp. 1–8.
[15] R. Semenas, M. Kaijanen, European Clearinghouse analysis of events related to NPP digital instrumentation and control systems, JRC Science and Policy Report, European Commission, Joint Research Center, 2014.
[16] P. V. Miguel, Operating experience with digital I&C systems at nuclear power plants, JRC Technical Report, European Commission, Joint Research Center, 2018.
[17] A. O. El-Rayis, A. Melnyk, Localized Payload Management Approach to Payload Control and Data Acquisition Architecture for Space Applications, in: Second NASA/ESA Conference on Adaptive Hardware and Systems (AHS 2007), Edinburgh, UK, 2007, pp. 263–272. doi: 10.1109/AHS.2007.70.
[18] O. Drozd, V. Antoniuk, V. Nikul, M. Drozd, Hidden faults in FPGA-built digital components of safety-related systems, in: Proceedings of the IEEE International Conference TCSET, Lviv-Slavsko, Ukraine, 2018, pp. 805–809. doi:10.1109/TCSET.2018.8336320.
[19] M. Abramovichi, C. Stroud, C. Hamiliton, S. Wijesuriya, V. Verma, Using roving STARs for on-line testing and diagnosis of FPGAs in fault-tolerant applications, in: Proceedings of the IEEE International Test Conference, 1999, pp. 973–982.
[20] A. Drozd, S. Antoshchuk, New on-line testing methods for approximate data processing in the computing circuits, in: Proceedings of the IEEE International Conference IDAACS, Prague, Czech Republic, 2011, pp. 291–294. doi:10.1109/IDAACS.2011.6072759.
[21] Y. Hussain, A. Rehalia, A. Dhyan, Case Study: Chernobyl Disaster, International Journal of Advanced Research in Computer Science and Software Engineering 8(2) (2018) 76–78.
[22] D. Gillis, The Apocalypses that Might Have Been, 2007. URL: https://www.damninteresting.com/the-apocalypses-that-might-have-been/
[23] N. Antunes, F. Brancati, A. Ceccarelli et al.: A monitoring and testing framework for critical off-the-shelf applications and services, in: Proceedings of IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), 2013, pp. 371–374.
[24] F. Duchi, N. Antunes, A. Ceccarelli et. al., Cost-Effective Testing for Critical Off-the-Shelf Services, in: Proceedings of International Conference on Computer Safety, Reliability, and Security, Delft, The Netherlands, 2014, pp 231–242.
[25] S. Verma, P. Srivastava, D. Ramavat, N. Srivastava, A Review Paper on Comparative Study of FPGA Implementation of Adhoc Security Algorithms, International Journal of Management & Information Technology 7(1) (2013).
[26] J. Jung, I. Ahmed, Development of field programmable gate array-based reactor trip functions using systems engineering approach. Nuclear Engineering and Technology 48(4) (2016) 1047–1057.
[27] O. Drozd, K. Zashcholkin, R. Shaporin, J. Drozd, Y. Sulima, Development of ICT Models in Area of Safety Education, in: Proceedings of the IEEE EWDT Symposium, Varna, Bulgaria, 2020, pp. 212–217. doi: 10.1109/EWDTS50664.2020.9224861.
[28] O. Drozd, I. Perebeinos, O. Martynyuk et. al., Hidden fault analysis of FPGA projects for critical applications, in: Proceedings of the IEEE International Conference TCSET, Lviv–Slavsko, Ukraine, 2020. doi:10.1109/TCSET49122.2020.235591.
[29] IEEE Std1500-2005 Standard Testability Method for Embedded Core-based IC, 2005. doi:10.1109/IEEESTD.2005.
[30] V. Hahanov, A. Hahanova, S. Chumachenko, S. Galagan, Diagnosis and repair method of SoC memory, WSEAS Transactions on Circuits and Systems 7(7) (2008) 698–707.
[31] A. Drozd, M. Lobachev, J. Drozd, The problem of on-line testing methods in approximate data processing, in: Proceedings of the 12th IEEE International On-Line Testing Symposium, Como, Italy, 2006, pp. 251–256. doi: 10.1109/IOLTS.2006.61.
[32] J. Drozd, A. Drozd, M. Al-Dhabi, A resource approach to on-line testing of computing circuits, in: Proceedings of the IEEE EWDT Symposium, Batumi, Georgia, 2015, pp. 276–281. doi: 10.1109/EWDTS.2015.7493122.

[33] C. Metra, L. Schiano, M. Favalli, B. Ricco, Self-checking scheme for the on-line testing of power supply noise, in: Proceedings of the Design, Automation and Test in Europe Conference, Paris, France, 2002, pp. 832–836.
[34] Cyclone Architecture. Cyclone Device Handbook, Volume 1. Altera Corporation, 2008. URL: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/cyc/cyc_c51002.pdf.
[35] M. Farias, R.H.S. Martins, P.I.N. Teixeira, P.V. Carvalho, FPGA-based I&C systems in nuclear plants, Chemical, Engineering Transactions 53 (2016) 283–288. doi: 10.3303/CET1653048.
[36] O. Drozd, K. Zashcholkin, O. Martynyuk, O. Ivanova, J. Drozd, Development of Checkability in FPGA Components of Safety-Related Systems, CEUR-WS 2762 (2020) 30–42. URL: http://ceur-ws.org/Vol-2762/paper1.pdf.
[37] T. Hovorushchenko, O. Pomorova, Information Technology of Evaluating the Sufficiency of Information on Quality in the Software Requirements Specifications, CEUR-WS (2104) 555–570.
[38] O. Pomorova, T. Hovorushchenko, The Way to Detection of Software Emergent Properties, in: Proccedings of the IEEE International Conference IDAACS, Warsaw, Poland, 2015, pp. 779–784.
[39] Ashok P. Nadkarni, The TCL Programming Language: A Comprehensive Guide, CreateSpace Publishing, 2017.
[40] Intel Quartus Prime Standard Edition User Guide Scripting, 2020. URL: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/ug-qps-scripting.pdf
[41] Delphi 10 Seattle: Embarcadero https://www.embarcadero.com/docs/datasheet.pdf.
[42] Intel FPGA Integer Arithmetic IP Cores User Guide, 2020. URL: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/ug_lpm_alt_mfug.pdf.
[43] Intel Quartus Prime Standard Edition User Guide, 2020. URL: https://www.intel.com/content/dam/alterawww/global/en_US/pdfs/literature/ug/ug-qps-getting-started.pdf.
[44] Intel Cyclone 10 LP Core Fabric and General Purpose I/Os Handbook, 2020. URL: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/cyclone-10/c10lp-51003.pdf.
[45] A. Sachenko, V. Kochan, V. Turchenko, Instrumentation for gathering data, IEEE Instrumentation and Measurement Magazine 6(3) (2003) 34–40.
[46] V. Golovko, Y. Savitsky, T. Laopoulos, A. Sachenko, L. Grandinetti, Technique of learning rate estimation for efficient training of MLP, in: Proceedings of the International Joint Conference on Neural Networks, 2000, pp. 323–328.