# Hidden fault analysis of FPGA projects for critical applications

Oleksandr Drozd
Institute of Computer Systems
Odesa National Polytechnic University
Odesa, Ukraine
drozd@ukr.net

Ihor Perebeinos
Institute of Computer Systems
Odesa National Polytechnic University
Odesa, Ukraine
perebeinos92@gmail.com

Oleksandr Martynyuk
Institute of Computer Systems
Odesa National Polytechnic University
Odesa, Ukraine
anmartynyuk@ukr.net

Kostiantyn Zashcholkin
Institute of Computer Systems
Odesa National Polytechnic University
Odesa, Ukraine
const-z@te.net.ua

Olena Ivanova
Institute of Computer Systems
Odesa National Polytechnic University
Odesa, Ukraine
en.ivanova.ua@gmail.com

Myroslav Drozd
Institute of Computer Systems
Odesa National Polytechnic University
Odesa, Ukraine
myroslav.drozd@opu.ua

## I. Introduction

The development of computer systems in critical applications takes place in conditions of increased decision-making responsibility, as it involves the requirements of safe operation of high-risk facilities, which include power plants and power grids, high-speed transport and their infrastructure, space and military equipment.

Computer systems are converted into instrumentation and control safety-related systems directed to ensure functional safety of both the system and the control object for preventing accidents and reducing losses from accidents in case of their occurrence.

Functional safety requirements are regulated by many international standards, which provide for verification of the instrumentation and control safety-related system and its components at all phases of the safety life cycle, including routine actions and operations to detect implicit failures of the system.

The most dangerous among implicit failures are hidden faults. The circuits can accumulate them in the absence of input data showing these failures as an error of the monitored result. To withstand failures, the safety-related systems and their components are being designed using fault-tolerant solutions which provide for different types of reservations and reconfiguration.

However, we can see a lack of confidence in the effectiveness of fault-tolerant solutions. First of all, this mistrust manifests itself in the use of imitation modes for recreating emergency conditions to detect hidden faults, which can only manifest themselves with the onset of an accident. Thus, the threat from hidden faults is considered to be more significant than the presence of simulation modes, which were repeatedly activated by an unauthorized person or failure and resulted in real accidents.

The deliberate use of imitation modes is also very dangerous, as it provides for the shutdown of emergency protections preventing the start and development of the accident, as it was in the case of the Chernobyl disaster.

We can note an important trend in development of critical systems and their components. It is related to the use of modern CAD (Computer-Aided Design), focused on Field Programmable Gate Array (FPGA) design, the features of which affect the hidden fault problem.

This paper aims at examining the hidden fault problem inherent in critical applications and proposes a method of analyzing digital circuits for the possibility of hidden failures in FPGA components of safety-related systems. Section 2 discloses the hidden fault problem as a growth one associated with transition of safety-related systems to a higher level of development than conventional computer systems. This transition is limited at the design level of components that inherit traditional parallel code processing from conventional computer systems using matrix structures. Section 3 considers the method of analyzing digital circuits to identify potentially dangerous points where problematic hidden faults may occur. Case study of the method is represented on example of an iterative array multiplier implemented in an FPGA project with LUT-oriented architecture (Look-Up Table).

## II. Analysis of of the Hidden Fault Problem

### *A. Resource approach*

The hidden fault problem is inherent in modern safety-related systems due to their two features. An important integral feature of critical systems is splitting an operating mode into normal and emergency one according to their use in critical applications. Another feature characteristic of all modern computer systems, including safety-related systems, is the design of their digital components using matrix structures that perform data processing in parallel codes.

According to the resource approach, these two features become two conditions for the emergence of growth problems, which include the hidden fault problem. This problem is the accumulation of hidden failures over a long-term normal mode in the absence of input data that show them. They appear in emergency mode and activate failures which reduce fault tolerance of circuit solutions and functional safety of the system that is based on these solutions.

The resource approach analyzes the integration of models, methods, and means that make up human resources into the natural world. Models are our ideas, our understandings about this world. The methods aim to develop and evaluate resources. Means (materials and tools) are material carriers of the information part of resources: models and methods, and represent them in their structure and functioning. Human mission is to read models and methods from the material carriers of the natural world, to develop them, to verify by the creation of livelihood means and to record on computer carriers in open code.

The resource approach identifies three levels of resource development: replication, diversification, and as a goal, self-sufficiency. The lower level is replication, which is known in natural world with the slogan "more to give birth than will die." For example, bacteria integrate into the natural world in this way. Replication is characterized by open resource niches (environmental, technological, market), i.e. no barriers to free stamping. Successful integration into the natural world is achieved by increasing productivity, which with the closure of resource niches results in overproduction crises, stopping the integration process at the replication level. When closing resource niches, clones survive only in case particularity development. They rise to the level of diversification and become individuals, versions. Integration at this level is achieved through increased trustworthiness, which consists in adequacy to the natural world, including aspects of safety.

All levels of resource development are represented in the process of their improvement in the modern computer world, but replication dominates. Software is stamped with the addition of replicable redundant finished modules of the program products in conditions of resource niches opened in ever-increasing performance and computer system memory capacity.

Mobile systems restricted in power niche have to develop green technologies that reflect diversification level in resource development and aim at increasing self-sufficiency in energy consumption.

Hardware is stamped by designing circuits using matrix structures formed from sets of uniform elements. Such solutions are prepared for commercial and critical applications according to the OTS (Off-The-Shelf) component approach and add libraries to modern CAD systems.

Hardware and software replication is widespread in networking technologies.

### *B. Checkability of circuits in critical applications*

Lack of confidence in fault-tolerant solutions is due to the low level of checkability of circuit solutions. Test developers for schematics use testability, which is structural checkability, i.e. depending only on the circuit structure. In the operating mode, the checkability becomes structurally-functional, that is, dependent also on the input data on which the circuit operates. Methods of on-line testing are limited in detection of faults and errors by structurally-functional checkability of circuits.

Safety-related systems designed for operation in two modes convert structurally-functional checkability into a dual-mode one, which can differ in normal and emergency mode in case of different input data.

The deficit of dual-mode checkability is determined by the difference of structurally-functional checkability of the circuit in emergency and normal modes. This deficit creates the problem of hidden faults when the fault manifests itself in emergency mode, being hidden in normal mode. Note that in conventional computer systems working in only one operating mode, hidden faults do not cause problems because they do not appear as errors throughout the mode.

It should be noted that the lack in checkability of the digital component circuits is achieved when two conditions are met at the same time. The first condition matures with the transformation of the conventional computers into safety-related systems, i.e. with the rise to the level of resource diversification, that manifests itself in the division of the operating mode into two: normal and emergency. This is an objective process involving the field of critical application, where most resource niches are closed, since security is provided within the framework of trustworthiness at the diversification level.

The second condition is determined by the level in development of computer system components. In critical applications, they continue to be designed at the replication level using matrix structures to handle data in parallel codes. This makes input data different in normal and emergency mode. In this case, structurally-functional checkability inherits the difference in these modes and generates a dual-mode checkability deficit for critical applications.

Hence, the hidden fault problem is a growth one, as it is created by the transition of safety-related systems to a level of diversification in conditions where their digital components continue to be stamped at the replication level of resource development.

## III. Analysis of the Multiplier Circuit Implemented in FPGA Project

### *A. Features of the analysis of the circuit*

It should be noted that the checkability of the circuit decreases with the transition from structural form to structurally-functional and further dual-mode. Therefore, the most effective detection of hidden faults can be expected at the level of structural checkability. Such detection of hidden faults is carried out using imitation modes, the danger of which is contrary to their use. We offer a method of safely analyzing a digital circuit within its structural checkability considered at all input data of normal and emergency mode. This analysis aims to find digital circuit points that are potentially dangerous because they can create a hidden fault problem, i.e. they can be carriers of the hidden faults that manifest themselves in emergency mode.

Analysis is focused on digital circuits implemented in FPGA projects with LUT-oriented architecture. The LUT unit is a logic function generator. The LUT unit generates a logic function from four variables in case of four inputs A, B, C and D. The description of the function is stored in 16-bit LUT memory.

The circuit points are the LUT unit memory bits. LUT operation is characterized by sets of $N$ and $E$ memory bits addressed in normal and emergency modes, respectively.

The circuit analysis is performed by obtaining $N$ and $E$ sets for each LUT unit and comparing them to each other. A point is potentially dangerous if the corresponding memory bit of the LUT unit is addressed only in emergency mode. A set of potentially dangerous points in the LUT unit memory are determined by the difference $\Delta = E \setminus N$.

### *B. Case study of the method*

The proposed method is illustrated by analyzing the iterative array multiplier circuit that is implemented in the Intel Max 10 FPGA 10M50DAF672I7G chip using the Quartus Prime 18.1 CAD system. For 4-bit operands, the iterative array multiplier circuit is designed using 30 LUT units.

We have developed a program model of this multiplier using the freely distributed Delphi 10 Seattle demo-version. The program simulates the operation of the obtained circuit in normal and emergency mode for different values of threshold $S$ separating input data in these modes. Both factors take all possible values from 0 to $S – 1$ and from $S$ to 15 in normal and emergency mode, respectively. Circuit operation is investigated for threshold values $S$ varying from 5 to 12 (8 experiments). All memory address values of each LUT unit are fixed in each mode to obtain sets $N$ and $E$.

### *C. Results of analysis*

The main panel offers EXIT and RUN modes respectively to exit the program and execute it, as well as LUT # 1 mode of sequential viewing of all LUT units, starting with LUT # 1. Fig. 1 shows the main panel for LUT # 2. The LUT unit memory is shown for all experiments determined by the threshold value $S$.

The LUT unit memory bits are shown as a matrix of squares. The matrix columns are numbered 00, 01, 10, and 11 of inputs B and A. Similarly, the matrix rows are denoted by inputs D and C. The matrix squares contain a bit value and can be colored in one of three colors: Aqua and Yellow, if the bit is addressed only in normal or emergency mode. Lime color shows addressing in both modes. The LUT unit memory bits painted yellow belong to the potentially dangerous points of the iterative array multiplier circuit. As the threshold $S$ rises, the number of such bits decreases. They are replaced by bits addressed only in normal or both modes. The lower part of the main panel shows the values of threshold $S$ and corresponding values $H$ of the total number of potentially dangerous points of the whole circuit.

In Table 1, this data is supplemented by the percentage $H^*$ of potentially dangerous points in the entire circuit in relation to their total number (280), the number $Z$ of input words received in each of the experiments in normal mode, and their percentage $Z^*$ concerning all input words (256).

The table shows a reduction in the number of potentially dangerous points from 78 to 13 with an increase in the number of input words in normal mode from 25 to 144.

The diagram shows the dependence of the relative estimates $H^*$ and $Z^*$ on the value of the threshold $S$. A set of potentially dangerous points decreases as the set of input words increases. However, this non-empty set retains the danger of hidden faults even when one half of the set of all input words is exceeded and reaches nearly 28% when 10% of normal mode input words are used.

## IV. Conclusions

The analysis of the problem of hidden faults shows two of its sources, two conditions of its occurrence. The first is related to the introduction of safety-related systems for servicing high-risk objects and consists in the peculiarities of designing these systems for operation in two modes: normal and emergency. Proof of the materiality of this condition is the absence of the problem in a conventional computer used only in one operating mode.

The second condition is also essential and consists in the traditional use of matrix structures to design safety-related system components. These structures process input data in parallel codes, making them different in normal and emergency mode. The structurally-functional checkability of the circuits, which depends on the input data, also becomes different. This difference creates a shortage of dual-mode checkability in components of critical systems and, as a result, creates the hidden fault problem.

This problem occurred at the intersection of two conditions. Had this problem been predetermined? To what extent is the crossing process of both conditions objective? The answer to these questions becomes clear when considering the hidden faults problem as a growth one: the system has risen to the level of diversification to solve security problems (which cannot be solved at the replication level), and the components have remained at the lower level – replication.

Growth problems are characteristic of complex systems. On the one hand, systems are being developed in critical applications that require diversification. On the other hand, the dominance of matrix structures keeps system components at the replication level. At the same time, it should be noted that matrix structures have dominated for decades and forced the intelligence of an entire generation of researchers and developers who have created a powerful infrastructure to support matrix parallelism: models, methods and means, such as modern CAD.

An important step in solving the problem of hidden faults is its demonstration, which is presented in the proposed analysis of the circuit within its structural checkability. The analysis method showed the presence of potentially dangerous points where a problem can occur, in a simple circuit of a 4-bit iterative array multiplier, implemented in an FPGA project with LUT-oriented architecture.

As the normal mode input word set increases, the number of these points decreases, but is not completely eliminated even at 50% of the input words. A set of potentially dangerous points reaches 28% when the number of input words is reduced to 10%. However, in normal mode, actual components can operate at the noise level, i.e., use significantly fewer input words.