t-WISE-BASED MULTI-FAULT INJECTION TECHNIQUE FOR THE VERIFICATION OF SAFETY CRITICAL I&C SYSTEMS

Vyacheslav Kharchenko
National Aerospace University KhAI
17 Chkalov St, Kharkiv, Ukraine 61070
v_s_kharchenko@ukr.net

Oleksandr Gordieiev
University of Banking of the National Bank of Ukraine
1 Andriivska St, Kyiv, Ukraine 04070
alex.gordeyev@gmail.com

Sergiy Vilkomir
East Carolina University
1001 E Fifth St, Greenville, NC, USA 27858
vilkomirs@ECU.EDU

Oleg Odarushchenko
Research and Production Corporation Radiy
29 Geroyev Stalingrada St, Kirovograd, Ukraine 25006
skifs2002@ukr.net

ABSTRACT

One well-known method for the verification and certification of NPP I&C and other safety critical systems is the fault injection technique (FIT). FIT is based on a design fault injection and vulnerability injection into the software code and the field-programmable gate array (FPGA) design or a physical fault injection into hardware modules. The requirements for FIT for safety critical applications are described in detail in the standard NUREG/CR-7151. In addition to the injection of single faults, NUREG/CR-7151 recommends employing a multi-fault injection technique (MFIT). The application of MFIT increases the verification time, but it has the potential to significantly improve the quality of the system. This paper reviews different MFITs for FPGA- and software-oriented systems, with special consideration to NPP I&C systems. To increase the effectiveness of the fault injection, an MFIT procedure that applies the t-wise combinatorial method is proposed. This method allows combinations of multiple faults to be found and provides full coverage of all combinations of any t-types of faults with a minimum number of checks. While the t-wise method is well known in software testing, to the best of our knowledge, this is its first application to the fault injection. Different types of coverage are considered. A case study of MFIT is considered based on a project of FPGA-based modules for NPP I&C systems.

Key Words: Safety, fault injection, multi-faults, I&C system, t-wise method

1 INTRODUCTION

1.1 Motivation

The implementation of a rigorous approach to verification and validation (V&V) of I&C systems for nuclear and other safety critical domains is an important measure to ensure compliance with standard
requirements and the requirements of regulators and customers. There are a lot of different V&V methods, in particular [1, 2]:

- Reviews and inspections.
- Formal verification based on mathematical methods, model checking, modeling using emulators.
- Analysis techniques: design rules and constraints checking, static code analysis, timing analysis, FMEA-based analysis.
- Testing: functional testing, load and validation testing, etc.

One well-known method for the verification and certification of NPP I&C and other safety critical systems is a fault injection technique (FIT). The FITs are based on the injection of design, physical, and interaction faults [3-6]: fault injection into software (SW) code and FPGA design (design faults); fault injection into chips and hardware/FPGA-based modules (physical faults); vulnerability injection into software and hardware (interaction faults and anomalies). The goals of FIT are the following [7]:

- To assess the test quality and efficiency of online testing considering test coverage issues (design and physical faults).
- To analyze fault-tolerance (design and physical faults).
- To analyze intrusion-tolerance (vulnerabilities).

Input data for the implementation of FIT are sets of injected faults. The faults can be received with special techniques, for example, Failure Modes Effects and Diagnostic Analysis (FMEDA) for hardware, Software Failure Modes and Effects Analysis (SFMEA) for software, and Intrusion Modes and Effects Analysis (IMEA) for security assessment. During FIT performing, single faults are injected, testing is executed, and results are analyzed. Requirements for FIT for safety critical applications are described in detail in the standard NUREG/CR-7151 “Development of a Fault Injection-Based Dependability Assessment Methodology for Digital I&C Systems” [8].

This standard recommends applying a multi-fault injection, together with the injecting of single faults. This takes into account the increasing complexity of software and FPGA-based systems and the nature of chip faults, multiplicity of design faults, and vulnerabilities. The application of MFIT significantly increases verification time. Moreover, the application of MFIT causes a few limitations related to the mutation of faults and blockage of verifiable system performance. Hence, development, enhancement, and implementation of MFIT are the challenges in V&V engineering. Besides, the application of known and the development of new MFITs should be performed while considering the specific features of the SW- and FPGA-based NPP I&C systems and actual practice-oriented restrictions.

1.2 Related Works

There are a lot of techniques and tools for the FIT applied for dependability assessment and verification of software- and FPGA-based I&C systems. Surveys of the techniques and tools for FIT are given in [1, 5, 9]. In particular, the techniques and tools for assessing the critical systems developed by the use of OTC-components are described in [1]. Papers [5, 9] provide an overview of the techniques and tools for fault injection. These techniques and tools do not support multi-fault injection in an explicit form. To provide MFIT using these tools, it is necessary to integrate them into the general environment.

The paper [10] discusses alternative approaches for performing transient fault injection in circuits described in a high-level language (VHDL for FPGA design). A model of behavior is generated, allowing the designer to identify the detailed error propagation paths in the FPGA chip. In [11], a simulation is done to prove that the proposed fault injection method can inject multiple faults into the different reconfigurations and to individually observe the effects on the entire system of each of the faults.

The paper [12] presents a simulation-based tool VHDLSFI that aims at conducting a multi-bit fault injection campaign to analyze the dependability of the VHDL model. This tool utilizes the Force and
Release statements that are features in the VHDL 2008 standard to inject transient and permanent single and multi-bit faults. Authors of the [13] propose an approach to assess the intrusion tolerance of ASIC and simulate attacks more representative than random multi-bit fault injections. It allows for the designing and validating of countermeasures against laser attacks on ASICs implementing cryptographic algorithms. Metrics and techniques to assess the security of OTS-based systems are described in [6], considering components vulnerabilities and simulating attacks into injected vulnerabilities.

The paper [14] introduces a new in-system multi-fault injection strategy for automatic test pattern injection. The proposed abstraction supports the task separation of the design and test-engineers and enables the emulation of physical attacks at the circuit level. The high level of abstraction is combinable with the advantage of high-performance autonomous emulations on high-end FPGA-platforms. The tools for fault injection into multi-version systems and the features of selecting of representative faults to inject using different techniques are described in [15, 16].

Analysis of the references allows concluding that there is a lack of techniques for the selection of multi-faults that can solve practical problems such as verification time minimization, faults self-masking, and blocking of systems caused by fault injections.

1.3 Main idea and objectives

Faults of different types are usually used during multi-fault injection. It is desirable to inject a maximum workable number of the various combinations of fault types. However, it is not practically possible to cover all combinations. Even when numbers of injected faults and their types are relatively small, the number of combinations is usually too large. For example, when only three faults are injected at the same time and each fault has 10 different types, the total number of all 3-tuple combinations of types is \(10^3=1000\). To fulfill this number of injections is not realistic. One of the possible approaches in such situations is to provide combinatorial coverage of the combinations of the smaller size (for example, cover pairs of types of faults instead of 3-tuples). In other words, when \(n\) faults are injected, the task is to cover (inside selected \(n\)-tuples) all possible combinations of fault types for any \(t\) faults where \(t<n\). Such an approach is known as \(t\)-wise coverage [17] and is used in different areas, for example, the experiment planning and software testing [18, 19]. We propose the MFIT procedure based on the application of the \(t\)-wise method. To the best of our knowledge, this is its first application to fault injection.

The rest of the paper is organized as follows. Section 2 discusses the general MFIT approach based on the \(t\)-wise method. Section 3 presents a case study of MFIT development using the \(t\)-wise method for the logic module on the FPGA-based platform. Finally, Section 4 contains conclusions and briefly outlines the future work.

2 GENERAL APPROACH TO MULTI-FAULT INJECTION

We represented the multi-fault injection processes model in the form of Integrated Computer Aided Manufacturing DEFinition for Function Modeling (IDEF0) diagram [20]. The context IDEF0 diagram (Fig. 1) is the diagram of the top level that defines the input data (Module), output data (Assessment results), control (Requirements to FMEDA and standards, Internal regulatory documents), and mechanisms (Verifier, Developer). For a more detail presentation of the processes, the model context IDEF0 diagram has been decomposed into seven interconnected stages. The model was developed on the basis of the FIT life cycle model [21]. Brief characteristics of each stage are represented in Fig. 2:

Stage 1. FMEDA [22] analysis. A result of this analysis is the nomenclature of types and subtypes. Such data are a base for forming fault type profiles.
Stage 2. Defect profile forming. Input data for this stage are the nomenclature of faults, which are received as a result of FMEDA analysis. Profiles of faults can correspond to FMEDA nomenclature of faults or can correspond to subset types of faults as represented by the compliant taxonomy.
Stage 3. The t-wise procedure’s application. The injection of multi-faults demands time and human resources. Such cost is directly proportional to amount of fault injections. The t-wise procedure allows for decreasing the number of fault injections and provides good coverage of the different combinations of faults. Forming such sets of fault injections can reduce the costs of their implementation.

Stage 4. Analysis of t-wise procedure results. For some generated fault sequences, it is not possible to inject them into the system. The reason is that these fault sequences can:

- The main goal this stage is the detection of such fault sequences and to exclude them from the list of injections.

Stage 5. MFIT technique. The goal of this stage is the choice and realization of MFIT techniques. The two variants of the realization of such techniques can be used by single injections without preliminary reinjections and by sequential injections.

Stage 6. Testing. The goal of this stage is testing with injected faults. The self-control assessment and internal control should be applied.

Stage 7. Analysis of assessment results. This stage is oriented on system testing results assessment. The basis of this stage is a comparison between the set of faults.

Figure 1. IDEF0 diagram of MFIT. Initial scheme.

Figure 2. IDEF0 diagram of MFIT. Decomposition.
3 T-WISE-BASED MFIT FOR FPGA-BASED MODULE

3.1 Faults of the FPGA-based logic module

In this section, we present the industrial realization of a few elements from the t-wise-based, multi-fault injection technique. For this purpose, we observed the fault injection procedure over the Logic Module (LM), which was performed during the SIL-3 certification, according to the requirements of the standard IEC 61508 [23]. LM is a part of the FPGA-based digital RadICS platform [24].

The FMEDA approach is applied at the first stage of the t-wise-based, multi-fault injection technique. The list of injected fault types and subtypes presented in the form of FIT-table is a result of FMEDA. This table contains the complete list of injected faults with their detailed description. Such a table can comprise the following information:

- Definite names of the detectable symptoms.
- Modules implicated of FPGA-based platform; name of units with injected faults.
- The list of modules that are presented in the chassis for the test.
- Designers’ recommendation how to make this symptom appear.
- Module’s mode for the test.
- Importance of module or software modification for testing.

The obtained FIT-table is analyzed to determine the number of injected fault types and subtypes. Table I shows fragments of the analysis. LM is analyzed to find the points for injection of all faults subtypes. A few limitations are taken into account: element parameters are not changed under temperature/mechanical influences; fault injection for elements does not cause a failure or an unacceptable parameter changing of other ones; any element must be acceptable for fault injection.

Table I. Description of types and subtypes of injected faults for Logic Module

<table>
<thead>
<tr>
<th>Types of injected faults</th>
<th>Subtypes of injected faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>Lost V (LV)</td>
<td>Loss of voltage</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>V change (VC)</td>
<td>Change of voltage</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Stuck on (SO)</td>
<td>Stuck on in high or low level</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Communication and clock faults (CC)</td>
<td>Loss or short of circuit</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Configuration and memory faults (CM)</td>
<td>Configuration error of FPGA unit</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3.2 Combinatorial coverage of different types of injected faults

At the next step, we apply the t-wise approach to reduce the number of fault injections, but we still have a proper coverage of different combinations of types of faults. Let \( k \) be the number of possible types/subtypes of faults and \( n \) be the number of faults that are simultaneously injected. The value of \( k \) does not depend on \( n \) and could be much bigger than \( n \). One \( n \)-tuple of values of fault types represents one simultaneous injection of \( n \) faults. Our approach is to generate a set of \( n \)-tuples in such a way that all combinations of fault types of any \( t \) faults \((t\leq n)\) are covered.

The t-wise coverage is usually used in practice for \( t \) from 1 to 6. However, the number of requested combinations can be too large for fault injection when \( t>2 \). That is why the most practical approach is the 2-wise (pair-wise) coverage with three injected faults. The aim is a selection of the minimum number of injections of three faults which cover all possible pairs of types for all pairs of faults. In this case study, values \( n=3, t=2, k=5, \) and \( k=10 \) are considered. To generate a set of fault injections, tools can be used.

3.3 ACTS tool

We use the Advanced Combinatorial Testing System (ACTS) tool [25, 26] to generate different variants of fault injections according to the t-wise (pair-wise) approach. ACTS has been developed by the U.S. National Institute of Standards and Technology (NIST) and the University of Texas at Arlington for combinatorial t-wise software test generation for \( 1\leq t\leq 6 \). The tool is based on the mathematical covering array technique and uses several different algorithms, including various versions of In-Parameter-Order-General (IPOG) algorithm [27]. Though ACTS is a software testing tool, it can be directly applied without any changes for the generation of the different variants of multi-fault injection.

Input data of ACTS are a list of testing parameters (in our case, faults for injection) and their values (in our case, types of the faults). ACTS allows using a big number of parameters and their values and, in the case of fault injection, this number is more than sufficient for any practical application. The output value of the tool is a list of tests (in our case, variants of multi-fault injection) that guarantees the t-wise coverage. Users can establish some logical constrains (restrictions) that must be satisfied by output values. It is very important in our case because, as mentioned above, some combinations of faults may not be valid for injection. The tool has a simple and convenient GUI interface as well as a command line interface and generates output results very fast.

Figure 3. Description of possible faults in ACTS
3.4 t-wise coverage for different approaches of fault injection

Approaches to the injections of multi-faults could be different in different circumstances and depend on the following:

- Time, cost, and human resources.
- Importance of specific combinations of fault types/subtypes.

Available resources affect the number of multi-fault injections that can be allowed. To understand the importance and select which combinations of fault types/subtypes should be covered, the level of coverage should be determined (only the most important types of faults, all types, types and subtypes, repetitions of types for different faults, etc.).

The several approaches are considered below. The same r-wise method is applied in all approaches. Also, in all approaches:

- The number n of simultaneously injected faults equals 3. This number is selected as the most reasonable and practically important. However, the approaches can be easily extended for n>3.
- The level of coverage in the t-wise method equals 2, i.e., the pair-wise method is applied. Because t<n and n=3, t=2 is the only option. However, in case n>3, the approaches can be easily extended for t>2.
- The order of injection is important, i.e., injections (Fault_1, Fault_2, Fault_3) and (Fault_2, Fault_1, Fault_3) are considered as different.

3.4.1 Approach 1—only important types of fault are covered

The number n of simultaneously injected faults equals three, so the most important types of faults should be selected (let it be LV, VC, and, SO). Each type has two subtypes and the total number of all possible combinations (injections) is 2x2x2 = 2³ = 8. The number of pair of fault subtypes is 12: four pairs for (Fault_1, Fault_2), four pairs for (Fault_1, Fault_3), and four pairs for (Fault_2, Fault_3). Each injection of three faults covers three pairs, so to cover all pairs of subtypes according to the pair-wise method, four injections are sufficient. Injections generated by ACTS are shown in Table II.

<table>
<thead>
<tr>
<th>Injections</th>
<th>Covered pairs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 LV-P</td>
<td>(LV-P, VC-H), (LV-P, SO-S), (VC-H, SO-S)</td>
</tr>
<tr>
<td>2 LV-P</td>
<td>(LV-P, VC-L), (LV-P, SO-D), (VC-L, SO-D)</td>
</tr>
<tr>
<td>3 LV-D</td>
<td>(LV-D, VC-H), (LV-D, SO-D), (VC-H, SO-D)</td>
</tr>
<tr>
<td>4 LV-D</td>
<td>(LV-D, VC-L), (LV-D, SO-S), (VC-L, SO-S)</td>
</tr>
</tbody>
</table>

3.4.2 Approach 2—combinations of all types of faults (without of subtitles) are covered; repetitions of types are allowed

Each fault has five different types, so the total number of all possible combinations (injections) is 5x5x5 = 5³ = 125. Coverage of subtypes is not considered. It is easy to see that the minimum number of injections that necessary to cover all pairs of types is 25. Indeed, among all pairs, pairs of types for first and second faults should be covered. The number of such pairs is 5x5 = 25 and one injection covers one
such pair. It is interesting that the maximum number that is sufficient to cover all pairs according to the pair-wise method coincides with the minimal number and equals 25. These 25 combinations (injections) generated by ACTS tool are shown in Fig. 4.

### 3.4.3 Approach 3—combinations of all types of faults (without of subtitles) are covered; repetitions of types are not allowed

This is the same approach as in Section 3.4.2, but only different types of defects are allowed. The total number of all possible injections is $5 \times 4 \times 3 = 60$. The minimum number of injections to cover all pair of types is $5 \times 4 = 20$. To generate variants of injections, the constraints are described in ACTS: Fault_1 ≠ Fault_2; Fault_1 ≠ Fault_3; Fault_2 ≠ Fault_3. Total 25 combinations, generated by ACTS tool, cover all pairs of types for any pair of faults.

### 3.4.4 Approaches 4 and 5—combinations of all types of faults with subtypes are covered

In Approach 4, everything is the same as in Section 3.4.2 but the coverage of subtypes is considered now. Each defect has 10 options (5 types x 2 subtypes). The total number of all possible combinations (injections) is $10 \times 10 \times 10 = 10^3 = 1000$. The necessary and sufficient number of injections to cover all pairs of subtypes is 100.

Approach 5 is similar to Approach 4, but all injected subtypes should be different. The total number of all possible injections is $10 \times 9 \times 8 = 720$. The minimum number of injections to cover all pair of types is $10 \times 9 \times 8 = 90$. To cover all pairs, ACTS tool generates 97 injections, a number that is slightly higher than the minimum number but is significantly less than the total number of possible injections.

![Figure 4. Injections generated by ACTS for Approach 2.](image)

### 3.4.5 The last step

On the last step, the developed fault combinations are injected in the module and feedback is analyzed. Module reaction is compared with the expected one and the conclusions about the success of the passing certain stage of the testing are made. Types and subtypes of faults, which can switch the entire module into a faulted mode, are defined. Faults can be injected in serial and in parallel. During the testing, we found, for example, that after injecting the Clock faults, some combinations of the faults switch the module into the faulted mode. If such combinations are recognized in advance, it is possible to avoid their generation by including corresponding constraints in ACTS.
4 CONCLUSIONS AND FUTURE STEPS

The multi-fault injection technique is becoming one of the important and modern variants of the well-known FIT procedure. This technique allows researching the behavior and verifying a lot additional properties of safety critical I&C systems such as a tolerance to multiple faults of different components and different types (design, physical, interaction). The standard NUREG/CR-7151 recommends employing a multi-fault injection technique, but it does not describe procedures of injecting. In this paper, we propose the MFIT procedure that applies the \( t \)-wise combinatorial method. The application of MFIT may be time-consuming. In this paper, the proposed technique assures a decreasing time of verification due to the optimal number of generated fault injections.

The future steps can concentrate on the development of techniques and tools that take into account the possibilities of injecting different fault types for different components of I&C systems. For FPGA-based NPP I&C systems, it may be physical faults injecting at the module and chip levels, design faults, and vulnerabilities injecting into VHDL code. Besides, an interesting direction for research is the development of methods to ensure the ability to design multi-fault injections similar to the FIT-ability of FPGA-based systems [28].

5 REFERENCES


