Decoding the Unknown: A Synergy of Formal and Simulation Methods for Unclassified Faults

Siri Rajanedi, Prashantkumar Ravindra
siri.rajanedi@analog.com, prashantkumar.ravindra@analog.com

***Abstract*- During fault injection simulations, injected faults can be detected by the Safety Mechanism (SM) and/or can be observed to impact design functionality. Diagnostic Coverage (DC) serves as a key measure to assess the effectiveness of the SM. While achieving the target DC is the ultimate objective, ensuring this is accomplished with a high degree of confidence is essential. Unclassified faults—those that are Unobserved functionally and Undetected by SM (UU)—must be carefully analyzed and reclassified to confidently finalize the achieved DC. This paper presents simulation-based Fault Barrier Analysis as an innovative approach, emphasizing its advantages over conventional formal methods in specific design contexts. It addresses the challenges faced in formal-based UU analysis. It also offers a comprehensive analysis contrasting formal methods with simulation techniques. The results provided from both the methods offer valuable insights to help users select the most suitable methodology for UU fault analysis based on their design needs. Following the saturation of formal-based analysis, a simulation-based approach is currently being implemented in project 2, where it is yielding promising results.**

I. INTRODUCTION

Functional Safety (FuSa) ensures that a system operates correctly in response to its inputs and safely handles potential failures to prevent risks or hazards. ISO 26262 is the recognized standard for Functional Safety in automotive Electrical and Electronic Systems. It offers a FuSa development process for the entire automotive supply chain. It also provides an automotive-specific approach for determining risk classes known as Automotive Safety Integrity Levels (ASILs) as shown in fig. 1.

ASIL assessment is based on three parameters:

- Severity - Potential severity of harm
- Exposure - Potential exposure to the operational situation
- Controllability - Potential for controllability to avoid harm

Figure 1. ASIL assessment

ISO 26262 recommends fault injection as a method for

- Supporting the evaluation of the hardware architectural metrics.
  - Evaluating the DC of a Safety Mechanism
  - Evaluating the diagnostic time interval and the fault reaction time interval
  - Confirming the fault effect
- Supporting pre-silicon verification of an SM, including its ability to detect faults and manage their effects

The Failure Modes Effects and Diagnostic Analysis (FMEDA) verification flow, shown in fig.2 starts from the FMEDA document with estimated DC and terminates with the annotation of the verified DC back into the FMEDA. Unclassified faults block the DC sign-off even when the target DC is met, as the confidence in such DC is low.

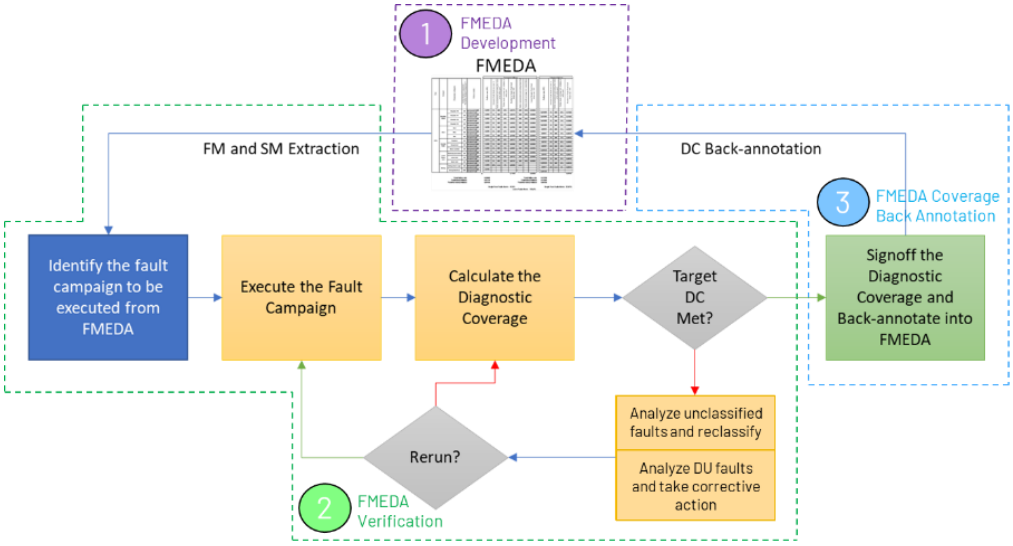

Figure 2. FMEDA Verification Flow

FMEDA is a systematic analysis technique in which the FuSa design is represented as part, sub-part, and elementary sub-part and their respective estimated DC and Single Point Fault Metric (SPFM) are determined. This forms the basis for the fault campaigns as shown in fig. 3.

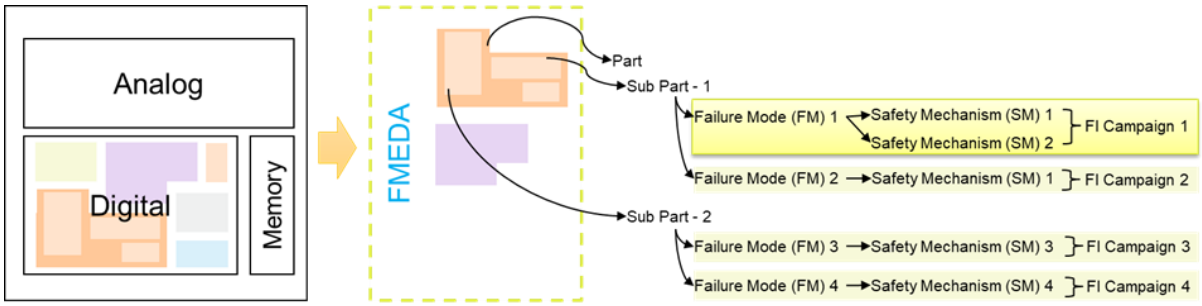

Figure 3. Planning fault campaigns from FMEDA

During a fault injection campaign, faults are classified based on their impact and detection. If a fault affects the functional outputs (FO), it is deemed dangerous. If it is detected on the checker outputs (CO), it is marked as detected. Faults that do not affect either output are labeled as unobserved or undetected. Consequently, faults are categorized as Safe (S), Dangerous Detected (DD), Dangerous Undetected (DU), Unobserved Undetected (UU), and Unobserved Detected (UD). Diagnostic Coverage (DC) is determined by analyzing DD and DU faults, while SPFM and Latent Fault Metric (LFM) calculations incorporate S, DD, and DU faults, as shown in fig. 4,

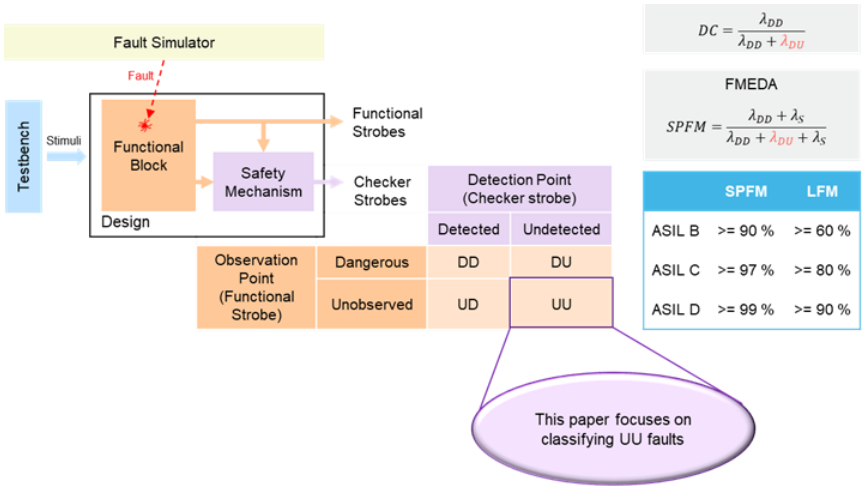

Figure 4. Fault categorization matrix

A safe fault is one that lies structurally outside the Cone-of-Influence (COI) of the functional outputs (FO) or is unactivatable and/or unpropagatable. Faults whose effects are not observed on the FO are termed as unclassified, which forms the central focus of this paper. It is both noteworthy and concerning that a significant portion of the fault nodes often remain unclassified in initial fault injection campaigns, a phenomenon that is routinely observed across the industry.

II. DESCRIPTION

Fault Injection simulations on a design yields fault classifications, including UU faults. Primary causes of UU fault are Insufficient stimulus/workload [1], structurally Out of COI (OOCOI) of the FO, structurally in and functionally out of COI of the FO as shown in fig. 5,

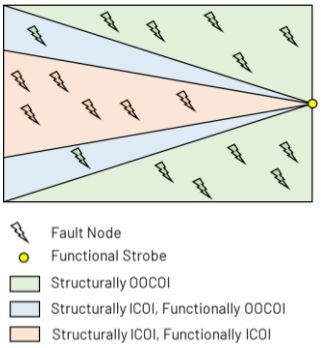

Figure 5. Cone of Influence of FO

A. Formal Analysis

Solutions include employing formal tools to filter out faults that are structurally out of COI of the FO [1] and classify them as safe. Additionally, applying constraints to exclude non-safety-critical modules, such as scan logic, can further enhance the safe fault count. Ensuring that the DV test suite used for fault campaigns achieves 100% functional coverage is another, albeit labor-intensive, solution. A key innovation in leveraging formal methods is the identification of stimuli capable of propagating UU faults to the FO. This is accomplished through propagatability and detectability analysis of UU faults [1]. The resulting stimuli traces can then be used to develop tests that classify UU faults within the simulation environment. These various formal solutions will aid in categorizing UU faults arising from diverse underlying causes.

Let us understand precisely how each of these causes could be addressed with above mentioned formal solutions. Table 1 shows different reasons for unclassified faults and their respective solutions [2]

TABLE 1
UU CAUSES AND THEIR FORMAL SOLUTIONS

Reason for a fault to be unclassified	Various solutions for each reason of unclassified fault
Insufficient stimulus/workload	• Use Test-Constant analysis – This will identify all the test-based constants • Identify the modules/sub-modules with more unclassified faults and create the stimulus/tests to activate that module/sub-module • Create an RTL regression suite to check the fault injection tests have relevant functional and code coverage. • Use formal propagatability and detectability analysis to identify the input port values that can stimulate a particular node and create simulation test(s).
Structurally OOCOI of the FO	• Use formal structural analysis to identify structurally safe faults
Structurally ICOI and functionally OOCOI of the FO	• Provide basic 'assume' in the structural analysis to identify functionally safe faults. • Further, improve the 'assumes' based on the design understanding to improve the safe fault result. • Based on the expert understanding of the design add stopats/barriers to exploit unactivatable and unpropagatable nodes and classify them as safe.

The following table 2 consolidates various solutions for UU analysis, the level of understanding of design and formal concepts needed and the overall impact [2]

TABLE 2
FORMAL SOLUTIONS AND THEIR IMPACT

Technique	Design Understanding	Formal Understanding	Impact
Constraint design inputs (assume)	Basic	Basic	Targets all functionally safe nodes
Forcing internal nodes (stopats / barrier)	Expert	Medium	Targets unactivatable and unpropagatable nodes
Test based fault pruning	Basic	Basic	Test-based constants and marked as UU
Identifying stimuli	Expert	Expert	Targets difficult-to-activate type of faults

It is a well-recognized challenge within the industry that formal analysis methods can encounter convergence issues, which often require substantial manual effort to address. The one of the mentioned formal analyses of identifying stimulus with propagatability and detectability analysis is no exception, where it was proven that increasing resources—such as licenses, time, and computational power—could accelerate convergence in digital heavy designs and block level analysis as shown in results. However, even with extensive resource allocation, full convergence was not achieved in high sequential depth and mixed signal designs. Due to the inherent limitations of formal tools, the process faced significant delays and various challenges, resulting in a prolonged execution time. Consequently, the Return On Investment (ROI) was not satisfactory. This prompted an exploration of alternative methods to address UU faults, leading to the development and discovery of the Fault Barrier Analysis method.

B. Fault Barrier Analysis

Understanding the factors that prevent fault propagation to the FO led to the identification of chokepoints or barriers in fault propagation. A fault barrier is defined as the first design signal where fault propagation ceases as shown in fig. 6.

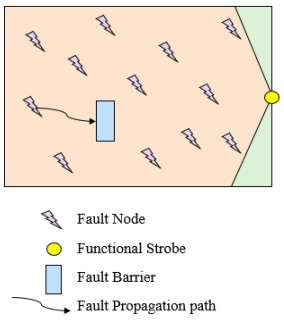

Figure 6. Fault Barriers in the design

Identifying these barriers is crucial for introducing design constraints or creating stimuli to either disable the barrier or enable fault propagation. In barrier analysis, barriers are ranked in descending order based on their contribution to fault non-propagation, from those causing the highest to the lowest number of UU faults. This ranking facilitates the prioritization of efforts in creating constraints or stimuli for the most significant barriers. This

simulation-based analysis requires no additional setup, as it leverages existing fault simulation runs to identify fault barriers, thereby minimizing setup time. Upon completion of the simulation, the tool generates a list of faults and their corresponding chokepoints, outputting one barrier database file per test run. This data can significantly aid in addressing a large volume of UU faults by identifying design areas responsible for fault non-propagation and non-detection.

The generated barrier to fault mapping file contains the information such as barrier ID, barrier node, fanin strength, impacted fault(s) ID's as shown in fig. 7,

Figure 7. Snippet of barriers.csv (barrier to fault relation) on an example design

The generated fault to barrier mapping file contains the information such as fault ID, fault node, fault type, fault injection time, fanout strength, barrier ID's as shown in fig. 8,

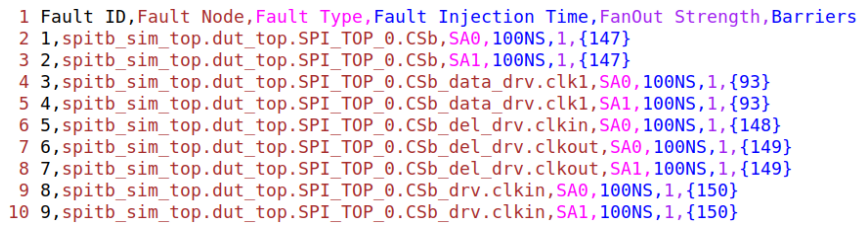

Figure 8. Snippet of faults.csv (fault to barrier relation) on an example design

Snippets from actual project are not shared because of project confidentiality. In the following section, results are shared from the actual projects.

III. RESULTS AND TAKE-AWAYS

Table 3 below presents project details, including gate and flip-flop counts, to illustrate the design scale and complexities for formal method across various projects for unclassified fault analysis. These insights were gathered to ensure a comprehensive understanding of the diverse design types evaluated prior to forming a detailed recommendation.

TABLE 3
PROJECT DETAILS

S. No	Parameter	Project 1		Project 2	Project 3	Project 4
		Block 1	Block 2			
1	Design Type	Digital	Digital	Mixed Signal	Mixed Signal	Mixed Signal
2	Gate count	76125	4571	616621	24784	16074
3	FF count	1125	82	14816	1398	956
4	Complexity	High sequential depth	None	Non-synthesizable models	Counters	Real Number Models

Fig. 9, demonstrates that increased resource allocation led to less convergence time i.e., reduction of 85% and 50% in block 1 and block 2 respectively for project 1. Fig. 10, shows that this increase in resource also led to more convergence i.e., less unknown faults in block 1 and block 2 of project 1 which is a digitally intensive design.

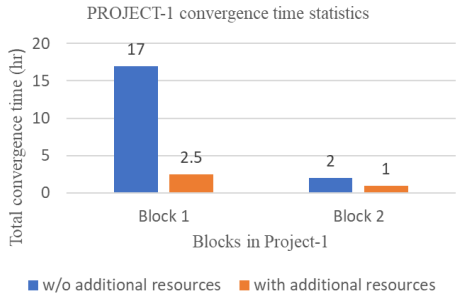

Figure 9. Convergence time in Project 1

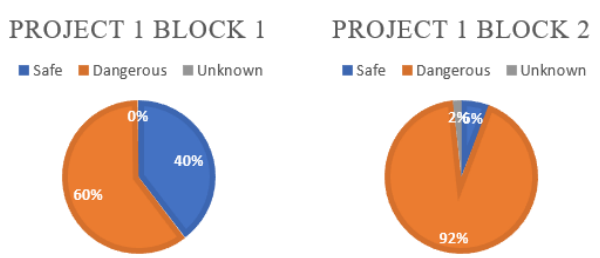

Figure 10. Propagatability analysis in Project 1 Block 1 and 2

However, fig. 11, reveals that, despite high resource usage, a significant portion of faults remained unclassified in formal-based analysis in project 2, 3 and 4.

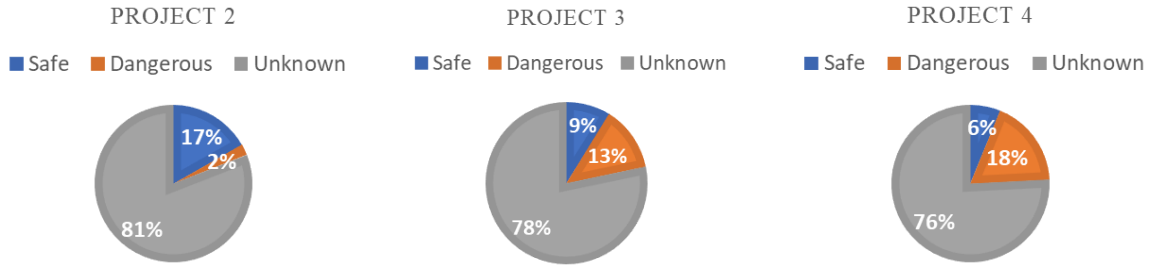

Figure 11. Propagatabaility analysis in Project 2, Project 3 and Project 4 respectively

Fig. 12, illustrates the timeline of challenges encountered across project 2, 3 and 4 when applying formal methods in mixed-signal designs. In fig. 12, the bubble size represents the severity of each issue, while the solid line denotes the resolution time for each. Combinational loops emerged as the most critical issue across all projects, requiring substantial time to resolve, which had a significant impact on the overall project timeline. After the resolution of combinational loops, another major challenge arose—convergence issues, which remain unresolved to this day. Where formal methods proved insufficient in such mixed signal projects, simulation-based fault barrier analysis was introduced. The final presentation will present the comparative analysis of results between formal and simulation methods.

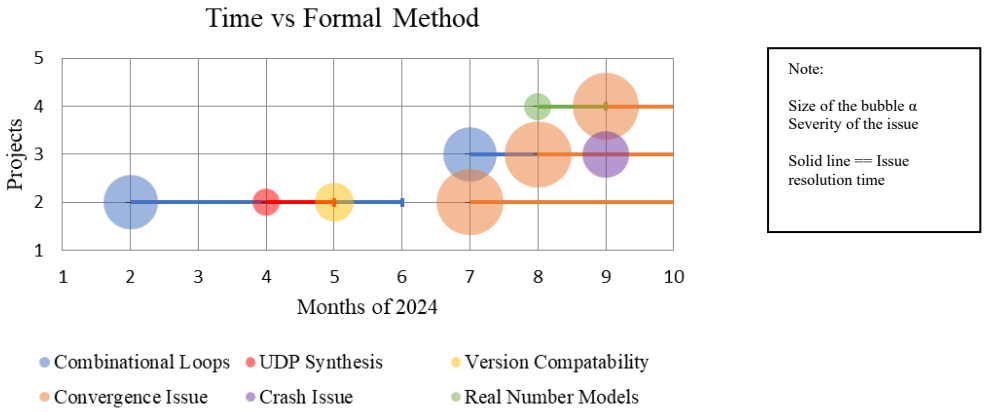

Figure 12. Challaenges faced in Projects 2, 3, 4 in Formal-based analysis

One of the challenges encountered in barrier analysis is that, after barriers are initially identified and removed, certain faults persist as UU, with new barriers emerging in the propagation path. This necessitates an iterative process of identifying and eliminating fault barriers to progressively reduce the UU fault count.

Fig. 13, presents the outcomes of the fault barrier analysis, highlighting that certain barrier nodes were analysed to result in safe faults, while others remained inconclusive, requiring further investigation to determine their classification.

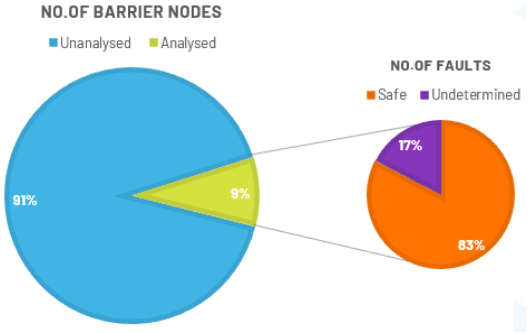

Figure 13. Results of fault barrier analysis in Project 2

Table 4 presents the fault barrier analysis statistics for Project 2. Despite the extensive number of nodes analyzed, 66.67% remained unprocessed. To address the challenges associated with iterative process inherent in the simulation method as mentioned above, an alternative approach was adopted for this particular project as an experiment—analyzing all barrier nodes at once instead of incrementally processing subsets and rerunning simulations. That being said, this approach posed to be cumbersome, requiring considerable time and manpower.

TABLE 4
SIMULATION METHOD STATISTICS IN PROJECT 2

S. No	Category	Sub-Category	Value	Percentage	Sub-category	Value	Percentage
1	Barrier Nodes	Analyzed	10500	9%	Safe	2460	27%
					Undetermined	540	9%
		Unanalyzed	109500	91%	Unprocessed	6000	66.67%
2	Total number of barrier nodes		120000	100%	Total Fault Nodes	9000	100%

However, most of the barrier nodes exhibited similar characteristics, effectively alleviating the issue. These nodes of a comparable type can be categorized, allowing for a detailed analysis of a select subset within a specific category, while the remaining nodes can be designated as reviewed to facilitate fault classification.

By utilizing the barrier-to-fault relation and fault-to-barrier relation mapping as shown in fig.7 and fig. 8 respectively, the nodes that are directly mapped in a one-to-one fashion between barriers and faults are analyzed first to immediately determine the fault classification. In the case of Project 2, a substantial number of barriers are attributed to DFT logic, and the corresponding faults can be classified as safe through expert judgment. However, the remaining faults, following the removal of the barrier, must undergo simulation to establish the final classification. Further analysis is being conducted on Projects 3 and 4 to develop a streamlined process that can be applied to any project. This approach aims to minimize the iterative nature of fault barrier analysis, ultimately enhancing efficiency.

A key takeaway from this analysis in project 2 is that the simulation method limits debuggability and does not support reactive, co-simulation, or GLS SDF annotation testbenches. Both formal and barrier analysis have their respective advantages and limitations. To leverage the strengths of both approaches, based on various experiments conducted in the production design, it is recommended to apply formal analysis to modules with lower sequential depth for faster results, while using barrier analysis for the other modules. Comparative results on UU faults will be presented in the final presentation.

Below table 5 shows the comparison between formal and simulation methods which highlights distinct strengths and limitations for each approach. Formal methods are particularly well-suited for digital-heavy designs, typically applied at the block level, though they demand significant computational resources and often encounter convergence issues. On the other hand, simulation methods excel in mixed-signal environments, are more commonly used at the chip-top level, and require moderate resources. However, they follow an iterative process, which can prolong the analysis. This contrast underscores the need to select the appropriate method based on the design type and available resources, balancing efficiency with complexity.

TABLE 5
FORMAL VS SIMULATION METHODS

S. No	Parameters	Formal Method	Simulation Method
1	Design Type	Digital Heavy	Mixed Signal
2	Hierarchy	Block Level	Chip Top
3	Resource	Intensive	Moderate
4	Drawbacks	Convergence Issues	Iterative Process

IV. CONCLUSION

A universal solution is rarely effective in complex verification scenarios. This paper examines the inherent limitations of formal methods and demonstrates how simulation-based fault barrier analysis effectively mitigates unclassified (UU) faults when formal techniques reach their boundaries. By selecting the optimal approach aligned with specific design requirements, significant time savings can be achieved. Additionally, combining both methods, where applicable, enhances confidence in Diagnostic Coverage (DC) during fault injection campaigns in Functional Safety (FuSa) designs.

ACKNOWLEDGEMENT

Authors would like to thank the members of ADI Automotive BU and Engineering Enablement for their support and collaborative work. Authors would also like to thank the Cadence FuSa team for their collaboration.

REFERENCES

[1] Praneeth Uddagiri, Veera Satya Sai Gavirni and Prashantkumar Ravindra, “Fault Injection Strategy to Validate ASIL-D Requirements of BMS Components” DVCON, India, September 2022.

[2] Siri Rajanedi, Prashantkumar Ravindra, “Target Diagnostic Coverage is Achieved! What about Unclassified Faults?” CDNLive, India, August 2023.