Closing the safety Verification Loop, FMEDA-Driven Fault Simulation and Advanced Debug for Efficient Fault Classification

Sagar Hema Vidya, Sr.Manager Silicon Design Engineering – AMD Hyderabad
(hema.vidya.sagar@amd.com)
Vedant Garg, Solutions Architect – Synopsys Austin
(vedantg@synopsys.com)
Sudhakar Chappidi, SMTS Silicon Design Engineer – AMD Hyderabad
(Sudhakar.Chappidi@amd.com)
Sharon Monica Kurmana, Sr. Silicon Design Engineer – AMD Hyderabad
(sharon.monica.kurmana@amd.com)

1. Abstract

The rapid advancement toward autonomous mobility is reshaping the automotive landscape and accelerating the evolution of semiconductor technologies. As manufacturers pursue higher integration densities and lower operating voltages, modern semiconductors become increasingly susceptible to random hardware failures that can occur unpredictably over a device's lifetime. Advanced Driver Assistance Systems (ADAS) and Autonomous Vehicle (AV) features depend on these digital and analog systems to execute critical, real-time applications raising essential questions about their validation and safety.

Functional safety standards such as ISO 26262 and IEC 61508 require comprehensive analysis of hardware safety mechanisms, typically conducted through Failure Modes, Effects, and Diagnostic Analysis (FMEDA). However, traditional verification and fault simulation processes often operate independently of FMEDA activities, leading to disconnects. Faults identified without FMEDA may not be directly mapped to simulation campaigns, and without this safety context it can lead to multiple iterations, lack of tracking and pose challenges for achieving robust safety verification closure.
This paper introduces a methodology that directly bridges FMEDA outputs with fault simulation execution, enabling a verification-driven approach where simulation campaigns are precisely tailored to FMEDA findings. By leveraging FMEDA data, the methodology defines exactly which faults must be simulated, specifies relevant detection mechanisms, and establishes safety-relevant coverage metrics.

A further contribution of this work is an advanced debug methodology for resolving ambiguities in fault classification an increasingly critical need in large-scale fault simulation campaigns. The paper details:

- FMEDA Data extraction and Automated generation of fault simulation verification plan. [section 3a]
- Fault simulation execution including automated random fault sampling. [section 3b]
- Faults analysis results illustrating gains in efficiency. [section 3c, 3d]
- Advanced debug strategies for comprehensive fault classification closure. [section 4]

This work will benefit verification engineers, functional safety managers, and EDA tool developers, offering both methodological advancements and practical insights adaptable to safety-critical designs across automotive, aerospace, and industrial domains.

Keywords: ISO 26262, Functional Safety, FMEDA, Fault Injection, Fault Debug, Fault Sampling

2. Introduction and Problem Statement

Functional safety standards such as ISO 26262 require hardware designs to demonstrate that safety mechanisms can reliably detect or control faults that may lead to hazardous behavior. As SoC complexity increases, it becomes challenging to show that safety coverage targets are met without excessive manual efforts or simulation cost .[1][5].

In practice, safety verification relies on two activities:

FMEDA defines what needs to be protected and its safety mechanism and fault simulation measures the DC coverage achieved by the safety mechanism. Although both activities target the same safety goals, they are often executed independently, leading to duplicated effort, inconsistent assumptions, and gaps in traceability. This motivates a unified method that connects FMEDA intent directly to simulation evidence.[4].

A. Faults, Errors, and Failures

Failures occur when a componenet no longer performs its intended function. They categorised as below:

- **Systematic failures** design or implementation mistakes.[1]
- **Random hardware failures** events such as bit flips or transient stuck-at conditions.[1]

Both categories require different mitigation strategies, but only random hardware faults contribute to ISO 26262 diagnostic coverage. Our focus is therefore on modeling and validating random hardware faults at gate level.

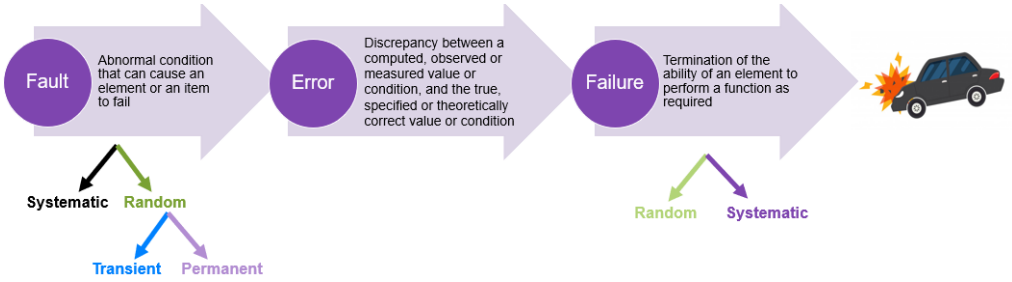

Figure 1 Fault, Error, Failure relationship

B. Safety Mechanisms and ISO 26262 Metrics

ISO 26262 requires designers to show that the implemented safety mechanisms such as parity/ECC, watchdogs, lockstep, or reset monitors can detect or control relevant hardware faults. The standard quantifies this using SPFM and LFM metrics, which depend on how many faults are:

- Detected by a safety mechanism
- Results in a safe reaction
- Remain latent or undetected

Achieving ASIL-D requires evidence that undetected single-point faults remain below very small thresholds. This drives the need for accurate and traceable fault analysis linked to FMEDA.

C. FMEDA and Fault Simulation

FMEDA identifies which elements require safety mechanisms and what diagnostic coverage can be achieved. Fault simulation provides experimental evidence by injecting faults and checking whether the safety mechanism responds correctly.

However, in current industrial flow these two activities are disconnected:

- *Non-targeted fault campaigns*: Simulators often inject faults across the entire Netlist, including blocks not listed in FMEDA, resulting in unnecessary compute overhead.

- *Weak traceability*: Simulated fault IDs do not directly map to FMEDA entries, complicating coverage justifications and audits.
- *Manual classification effort:* Safety teams must manually accommodate FMEDA spreadsheets with simulation reports.

These gaps create inefficiencies and risk msialignment between safety intent and verification results. A methodology that automatically links FMEDA entires to targeted fault simulation avoids these issues.

3. Proposed Methodology

The methodology presented in this work establishes a direct, data-driven linkage between FMEDA outputs and the fault simulation environment, effectively closing the verification loop and ensuring traceable, efficient, and safety-focused fault analysis. Unlike traditional flows where FMEDA and fault simulation operate independently, this integrated approach aligns simulation campaigns directly with safety-relevant design elements, reducing redundancy and improving coverage confidence.

A. FMEDA Data Extraction (Elementary Subpart Level)

The first step in the flow involves parsing FMEDA data at the elementary subpart level, extracting information such as failure modes, affected blocks, diagnostic coverage targets, and associated safety goals. This structured extraction ensures that every fault injected during simulation has a clear rationale, traceable back to the original FMEDA, thereby strengthening compliance documentation and audit readiness. By working at the elementary subpart level, the flow also allows fine-grained control of safety verification, avoiding unnecessary simulation of non-safety-critical signals [5][6]. The definition of elementary subpart as defined in ISO 26262, part 11 as shown in figure 2. A stuck at fault originating in the gates that are fan-in to the elementary subpart (sequential element), eventually this error have a potential to cause a failure.

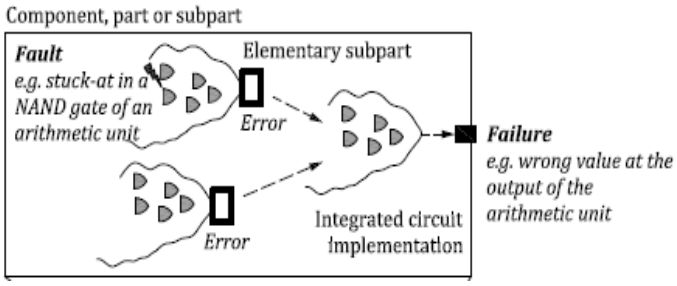

Figure 2 Elementary subpart definition

B. Fault Mapping and Safety Verification Plan Generation

Once FMEDA targets are extracted, a fault simulation plan is generated. The fault simulation verification plan is achieved through automation using the FMEDA tool from Synopsys (VC FSM). This plan maps each FMEDA-defined failure mode to one or more simulatable faut model and area in RTL or gate-level representations. Key features include:

- *Safety-focused filtering*: Faults that do not contribute to SPFM, LFM, or latent fault metrics are automatically excluded, significantly reducing simulation scope.
- *Prioritization based on ASIL and coverage relevance*: High-ASIL blocks or mechanisms with weak diagnostic coverage are simulated first.
- *Traceable mapping*: Every fault in the simulation plan is linked to its corresponding FMEDA entry, enabling automated reporting for audits or safety cases.

```
#### Safety Verification Plan ####

FailureMode FM_001 {
    Observe {
        "top_testbench.dut_top_i.core"
    }
    SafetyMechanisms [PSM_001]
}

FaultGenerate FM_019, FM_021 {
    NA [0, 1] { PORT "top_testbench.d" ... }
    NA [0, 1] { VARI "top_testbench.d" ... }
}
```

Figure 3 Safety verification plan definition

This step is critical for **reducing verification effort** while preserving full alignment with safety targets, a major improvement over traditional methods where fault lists are generic and unfiltered.

C. Fault Simulation Execution

A precise understanding of safety-relevant faults and their mapping to the design is essential for an efficient fault simulation campaign. If the fault set is not clearly defined, simulation runs may become incomplete or unnecessarily large. To avoid this, the execution process begins with extracting fine fine-grained FMEDA data and generating a targeted and prioritized fault simulation plan. This ensures that simulation resources are used efficiently, diagnostic responses are interpreted correctly, and the resulting coverage metrics accurately reflect ISO 26262 safety goals.[2]

The fault simulation stage leverages multiple optimization and automation techniques to maximize efficiency and diagnostic coverage. Selected faults are injected at gate-level, and diagnostic responses are monitored based on the safety mechanisms. Simulation campaigns are prioritized by safety relevance, ensuring that ASIL-critical elements are verified first.

Several advanced techniques are applied to improve fault simulation execution:

- *Fault pruning and optimization:* pruning is the process of eliminating the faults scenarios that do not contribute to meaningful analysis. In large systems, the number of possible faults can be enormous, but many of them:
    - Do not affect system safety or functionality (e.g., fault in unused logic paths)
    - Are already coveted by other faults (redundant scenarios)
    - Have negligible probability or impact

    .
- *Fault sampling*: Instead of simulating every possible fault exhaustively, a statistically representative subset is selected as shown in Figure 4. For instance, in one execution scenario, more than one million potential faults were efficiently reduced through strategic sampling. This approach reduced simulation duration from five days to less than one day, resulting in nearly 10× improvement in fault analysis efficiency while preserving block-level safety integrity.[2]. Refer to table 2 and table 3

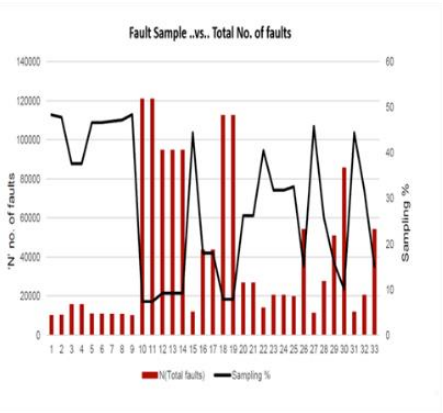

Figure 4 Fault Sample versus Total No. of faults

- *Automation for fault debug*: Scripts and automated analysis classify diagnostic responses, detect latent fault interactions, and generate residual fault break-up reports like Figure 9, without manual intervention.

D. Iterative FSIM flow to achieve target DC.

Figure 5 shows the overall process starting from the fault extraction and how it integrates elementary subpart FMEDA with fault simulation flow. Figure 6 illustrates the iterative FSIM (Fault Simulation) flow used to achieve the target diagnostic coverage (DC) defined in the FMEDA. The process begins with fault extraction, as described in Section 3, where safety-relevant faults are identified based on FMEDA data.

- *Workload Definition*:

After fault extraction, appropriate test stimuli are defined to activate the safety mechanisms associated with each FMEDA partition. Software Test Libraries (STLs) are used to emulate realistic in-field diagnostic behavior and trigger fault responses.

- *Strobe Point Selection*:

Critical observation points (strobes) are added to the FSIM environment based in the FMEDA safety mechanisms. Strobes ensure that fault effects are correctly captured and correlated to the expected diagnostic responses. [2][5]

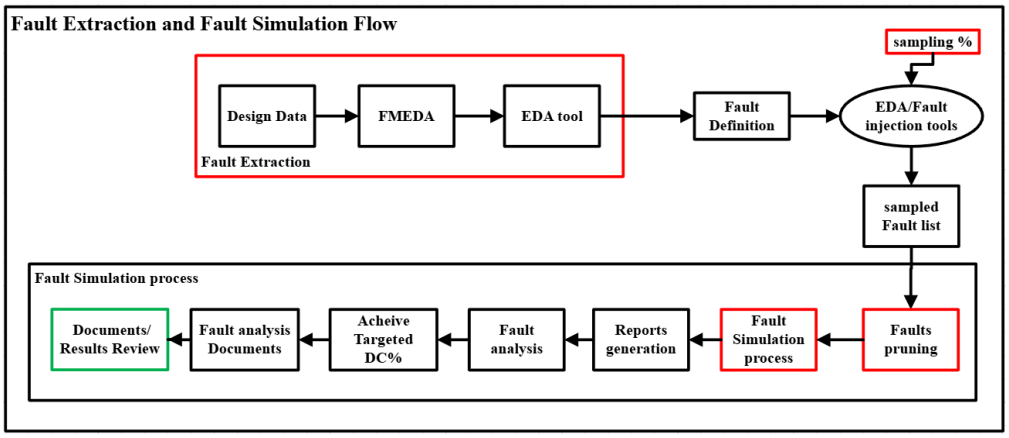

Figure 5 Fault Extraction and Fault Simulation Process

.

- *Fault Simulation and Analysis*:

Fault simulation is performed using the selected STLs to validate diagnostic coverage across the FMEDA scope. Post-simulation, and any undetected faults are analyzed to identify potential issues such as blocked propagation paths, missing strobe points, or inadequate test stimuli. Using this feedback, workloads and observation logic are iteratively refined to improve diagnostic coverage and eliminate verification gaps. [2]

- *Advanced Fault Debug and Classification*:

For faults that remain unclassified or ambiguous, AI is used to perform deeper structural analysis, recognize fault behavior patterns, and predict root causes. Machine learning models help resolve detection uncertainties by dynamically classifying faults into clear categories and suggesting targeted fault injection for improved coverage, this phase is in progress.

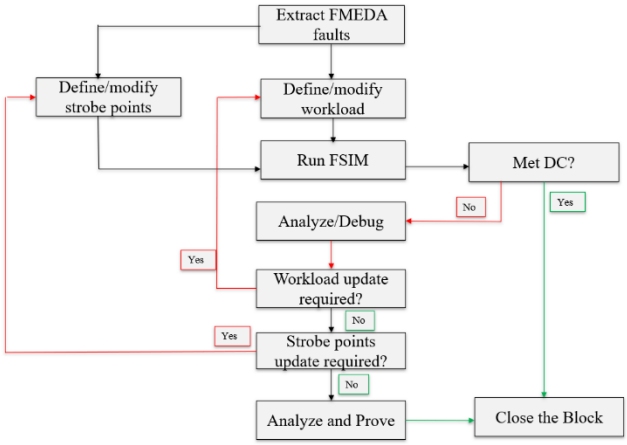

Figure 6 Iterative Fault Simulation process

- *Final Validation:*

Once DC targets are met the results are analyzed, formally proven, and the block is closed.

4. Faults Analysis

After fault simulation, there are some unresolved faults that needs to be analyzed using debug techniques. This section summarizes how each fault is evaluated for observability, controllability, and diagnostic visibility to determine its safety relevance. The analysis identifies which faults are safe and which many affect safety goals by reviewing signal propagation and masking at both structural and functional level as described in Table 1. Overall, the process ensures only safety-impacting faults are flagged for further diagnostic handling.

Table 1 Fault analysis

Reference	Type	Block/Example	Safety Reason	Diagnostic Logic
Figure 7	Safe	Scan FF (fault on CP / D / Q, SE=0).	Fault cannot reach functional logic	Scan disabled, path masked
Figure 8	Safe	Clock Gate (EN S@0).	No impact on functional data/control	Clock toggles but logic unaffected
Figure 8	Detect by Analysis	RAM ECC (other banks)	Behavior same across all the banks	ECC guaranteed detect/correct

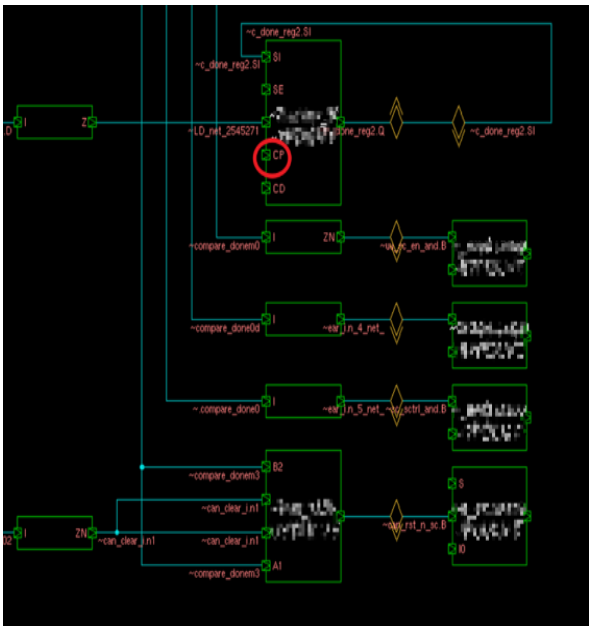

Figure 7 Safe fault, Fault injected on Scan logic

Figure 8 Fault injected on Clock Gates

5. FSIM Results and Conclusion

The FMEDA-driven fault simulation methodology introduces a structured, ISO 26262-compliant approach to safety verification for automotive SoCs. Each detection result from FSIM is mapped to its corresponding FMEDA entry, creating a one-to-one link between simulated fault behavior and its associated safety goal or diagnostic mechanism. This process establishes a closed-loop system that ensures traceability, accuracy, and audit readiness. This alignment eliminates redundant fault campaigns and improves the key safety metrics such as SPFM and LFM.[3]

Table 2 Comparison between traditional vs FMEDA driven flow

Stage	Flop count	Faults Count	Simulation Time (days)
Design subblock Scale	26939	67,780	15
For the target FMEDA Design subblock	3562	11,604	4
After Sampling	3562	7,809	2

The process also enables automated progress tracking across iterations allowing engineers to clearly identify coverage gaps, residual fault trends, and areas requiring workload refinement. Ultimately, this approach ensures that every simulated fault contributes meaningfully to a safety objective, transforming fault simulation from a statistical exercise into a structured, standards-compliant verification methodology that supports ISO 26262 safety case development with quantifiable evidence.

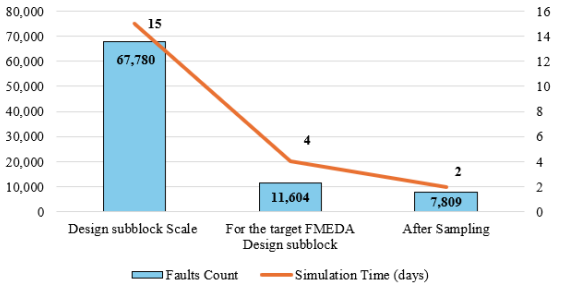

Figure 9 Faults Count Vs Simulation Time

Compared to traditional exhaustive fault injection methods, the proposed flow delivers measurable efficiency gains reducing fault population by over 33%, manual debug effort by nearly 66%, and compute resource utilization by 3–5×. These improvements were validated on a production automotive SoC block equipped with multiple safety mechanisms, demonstrating scalability and industrial applicability for next-generation designs with thousands of safety-relevant faults.[2]

Table 3 Comparison between traditional vs FMEDA driven flow

Metric	Traditional Flow	Proposed-Flow	Improvement
Faults Simulated	11,604	7,809	↓ 33%
Traceable Faults	65%	92%	↑ 42%
Unresolved Faults	~4000	~600	↓ 85%
Manual Debug Time	6 weeks	2 weeks	↓ 66%
Audit readiness	Partial	Full	✓

```
#-----------------------------------------------------------------------
# Fault Coverage Summary for "FMEA__001"
#
#                                                    Total
#-----------------------------------------------------------------------
# Number of Faults:                                  7809 100.00%
#
# Testable Faults:                                   7809 100.00%  100.00%
#       Obs &     Det                        OD      4581  58.66%   58.66%
#   Detected by Analysis - Expert Judgement  DA       596   7.63%    7.63%
#   Safe by Analysis - Expert Judgement      SA      2632  33.70%   33.70%
#
# Status Groups -------------------------------------------------------
#   Dangerous Detected                       DD      4581  58.66%
#
# Coverage ------------------------------------------------------------
#   Fsafe                                             33.70%
#   DC                                               100.00%
#   SM Faults                                             0
#   Errors                                                0
#-----------------------------------------------------------------------
```

Figure 10 Fault simulation result

Overall, this data-driven methodology not only achieves the same or better diagnostic coverage (SPFM/LFM) as the traditional approach but does so with far greater scalability and audit readiness. The flow effectively transforms fault simulation from a resource-intensive exercise into a repeatable, traceable, and standards-compliant verification methodology, closing the safety verification loop with quantifiable gains in both accuracy and efficiency for ASIL coverage.

6. Acronyms & References

Acronym	Full Form / Description
ASIL	Automotive Safety Integrity Level
DC	Diagnostic Coverage
ESE	Elementary Subpart Extraction
FMEDA	Failure Modes, Effects and Diagnostic Analysis
Fsafe	Safe Fault (Faults with no safety impact)
LFM	Latent Fault Metric
S@1 / S@0	Stuck-at-1 / Stuck-at-0 Fault Model
SM	Safety Mechanism
SPFM	Single Point Fault Metric
STL	Software Test Library

[1] Rambus RT-640 Road to ISO26262 certification, Rambus white paper.
[2] Complex Safety Mechanisms Need Interoperability for Validation and Close Loop for Final Metrics, Protecting Safety and Security, DVCon US 2023.
[3] Statistical Fault Injection: Quantified Error and Confidence, R. Leveugle, A. Calvez, P. Maistri, P. Vanhauwaert TIMA Laboratory (Grenoble INP, UJF, CNRS).
[4] ISO 26262: Road Vehicles – Functional Safety (especially Part 5: Product development at the hardware level, Part 9: ASIL-oriented and safety-oriented analyses, Part 11). ISO, 2018.
[5] Brown, L. D., Cai, T. T., and DasGupta, A. "Interval Estimation for a Binomial Proportion." Statistical Science, 2001. Recommended methods (Wilson/Agresti–Coull) for reporting diagnostic coverage with confidence useful for audited claims.
[6] Agrawal, V. D., and Seth, S. C. "Fault Simulation for Combinational Circuits." IEEE Design & Test of Computers, 1984. Core techniques for efficient fault simulation that underpin modern flows.