Breaking Down Failures: Elementary Subpart Granularity Being Foundation of Safety Analysis & Verification

Luc Baudoin, Sr Safety Engineer, AMD Austin (Luc.Baudoin@amd.com)

Vedant Garg, Solution Architect, SYNOPSYS Austin (Vedantg@synopsys.com)

Abstract

Functional safety, guided by **ISO 26262** (further referred to as 'the standard'), has become central to the design of complex semiconductors for automotive and other safety-critical domains. The standard [3] extends explicit guidance for semiconductor components, with emphasis on hardware fault analysis, safety mechanisms, and the role of fault injection.

Some practical challenges for semiconductor suppliers are:

- **Coverage Justification** – Ensuring that safety mechanisms comprehensively protect safety-relevant elements.
- **Early Vulnerability Closure** – Identifying architectural weaknesses before RTL simulation campaigns and protecting them using diagnostics ('Safety Mechanisms') with adequate coverage.
- **Efficient Fault Injection** – Injecting faults to validate the safety of the design selectively given its scalability limitations.

Today's flows, however, still rely on discovering vulnerabilities late in the lifecycle, creating costly iterations and introducing risk in safety case preparation. Therefore, a **systematic, automation-driven approach** is required to comply with the standard's guidance [3] while **accelerating closure**.

Figure 1, from the standard [3] provides guidance on how the safety component can be divided into Parts, Subparts and Elementary Subpart. Additionally, a *Fault* in a primitive gate can lead to an *Error* in the Sequential logic and potentially causing a *Failure* if left unprotected. If an *Error* is detected or discovered, the *Failure* can then be mitigated.

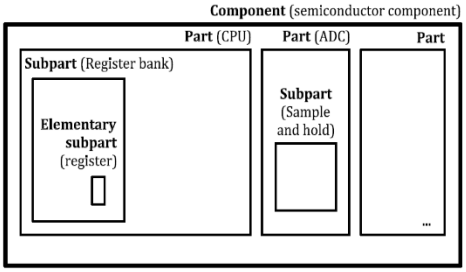


Figure 1: 26262-11 Component breakdown.

This paper introduces elementary subpart extraction (ESE), a methodology providing a true shift-left safety analysis framework aligned with the standard [3]. By decomposing designs into elementary subparts that reflect failure modes and safety relevance, ESE enables systematic gap identification early in the flow. This significantly reduces the costly iterations and volumes of Fault Injection required to demonstrate the safety goal achievement.

As described in the standard [3] the breakdown of a safety atomic unit is done by **decomposing RTL/netlist logic into semantically meaningful elementary subparts.** The subparts represent storage, combinational path and control logic. Each subpart is directly associated with failure modes and mapped to safety mechanisms. In this paper we present a framework that not only provides structural decomposition but also establishes an analytical mapping between RTL elements and FMEDA-level failure contributions, enabling quantitative evaluation well before fault simulation.

Keywords: FMEDA, Fault Injection, Failure Mode, Elementary Subpart, ISO 26262, Diagnostic Coverage, Safety Mechanisms, Design Verification, SPFM, LFM, PMHF

1. Introduction

A. Background and traditional approach

Functional safety in automotive SoCs demands a careful balance between quantitative safety assurance and verification efficiency. The standard establishes a rigorous framework for achieving this balance, prescribing a lifecycle that moves from Hazard Analysis and Risk Assessment (HARA) to safety goal derivation, technical safety concept formulation, and finally verification and validation of implemented safety mechanisms. Within this framework, hardware architectural metrics—notably the single point fault metric (SPFM) and latent fault metric (LFM)—serve as key evidence of compliance. Achieving these metrics requires detailed failure modes, effects and diagnostic analysis (FMEDA) of the semiconductor design. [2]

Early FMEDA approaches were largely manual and qualitative, relying on designer insight to map potential failure modes to their effects. With increasing design complexity, automation emerged leveraging structural netlist analysis and fault injection at the RTL or GLS.

B. Problem statement

The traditional workflows exhibit three major bottlenecks:

1. **Late availability of data:**
   Traditionally fault injections and quantitative FMEDA are performed on netlists. Any architectural change is extremely hard to incorporate and making the step just a final checkpoint.
2. **Simulation and compute overhead:**
   Multiple iterations are spent removing most injected faults which are functionally or structurally redundant thereby adding to computing overhead.
3. **Limited architectural insight:**
   Traditional FMEDA is reactive, focusing on what fails rather than why it is vulnerable. Architectural weaknesses such as unprotected combinational cones of logic (COL), partial coverage of safety mechanisms or inconsistent diagnostics often remain undetected until fault campaigns are driven on the taped out netlist or during silicon validation.

Safety verification in semiconductor designs remains predominantly reactive, addressing safety coverage shortfalls after RTL freeze or during verification closure. This late discovery leads to either multiple design and validation iterations or additional efforts on software diagnostics. This approach significantly increases schedule time, risk and verification cost.

Vulnerabilities stem from:

- Unprotected Cone of logic
- Partial or redundant safety mechanism bindings
- Unaccounted combinational dependencies

Issues in a typical design flow are:

- Overall fault injection overhead
- Dependency on software test libraries to achieve the safety metrics

This paper proposes **elementary subpart extraction (ESE)** an automated safety analysis methodology that decomposes the design's sequential logic into safety-relevant units using **fan-in COL analysis and timing-based overlap correction**. The ESE framework establishes an analytical mapping between the RTL and FMEDA level failure contributions, such enabling quantitative evaluation well before fault simulation.

2. Foundation of Safety Analysis (Proposed Methodology)

A. Elementary subpart Extraction

Elementary Subpart Extraction (ESE) is a **structural abstraction methodology** developed to enable *quantitative FMEDA-level safety analysis directly at the RTL abstraction*. It provides an early, systematic way to correlate architectural assumptions with structural fault models, thereby minimizing the dependency of traditional safety analysis on late design artifacts. The ESE framework not only provides structural decomposition but also establishes an analytical mapping between RTL elements and FMEDA-level failure contributions, enabling quantitative

evaluation well before fault simulation. We are utilizing this concept by decomposing an RTL design into **elementary subparts**, where each ESP is anchored around a **sequential logic element,** typically a flip-flop, latch, or memory bit and its **COL** (i.e., all combinational logic that drives its input).[3] Figure 2, from the standard [3] can be interpreted representing the same concept where the sea of combinational gates are fan-in to a sequential element that is eventually causing a failure.

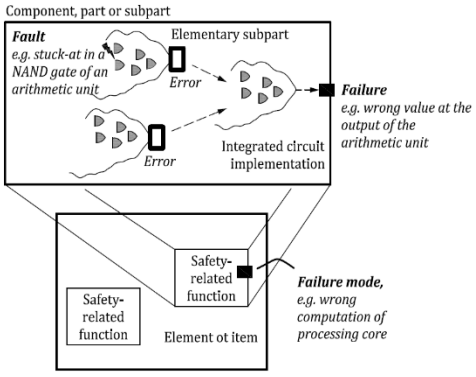


Figure 2: Elementary Subpart definition.

- **Identification of sequential elements**
  All storage elements (e.g., flip-flops, latches, RAMs, FIFOs) are enumerated at RTL (using Quick Synthesis). Each of these elements becomes the **root node** of an elementary subpart. This step creates a one-to-one mapping between sequential nodes and potential fault sources. [8]
- **COL extraction**
  For each sequential element, the **combinational COL**, all logic paths that can influence the element's value is statically traced using dependency analysis. This logical cone effectively represents the *information flow domain* of that element, capturing both direct combinational logic and hierarchical RTL expressions that may contribute to its state. [4]
- **Failure rate attribution**
  Each subpart's failure rate (λ_ESP) is calculated as:

  $$\lambda\_ESP = \lambda\_seq + \lambda\_col$$

  where:
  - λ_seq represents the intrinsic failure rate of the sequential element
  - λ_col represents the cumulative failure contribution from combinational COL.

  The resulting λ values are stored per subpart, forming the *structural backbone* for RTL-based FMEDA.
- **Overlap mitigation via Timing Analysis**
  COLs from different sequential elements often overlap, leading to potential *double counting* of shared combinational logic in aggregated failure rates. ESE resolves this through **timing-based dependency analysis**:
  - **Exclusive path detection:** Identify unique COL segments that are not shared between ESPs.
  - **Shared logic weighting:** When an overlap exists, shared gates are distributed based on **path criticality**, derived from the **static timing analysis (STA)**.
  - **Propagation delay correction:** Gates with negligible timing influence on multiple outputs are de-weighted to prevent overestimation.

The above-mentioned steps were automated using scripts and combination of various EDA tools for design and timing analysis tools to become first step of out methodology.

B. Safety Analysis for FMEA / FMEDA

Safety analysis is the systematic identification and evaluation of faults, failures, and hazards that could compromise the functional safety of an electrical/electronic (E/E) system. Its purpose is to ensure that safety goals and requirements are defined, implemented, and verified such that residual risk is reduced to an acceptable level—as determined by the automotive safety integrity level (ASIL)[3].

The key objectives of the safety analysis, as reinforced by multiple prior studies[8][7] are to:

1. *Identify* faults that can propagate to and violate the safety goal.
2. *Determine* the effects of each fault within the architectural context.
3. Ensure the appropriate *diagnostic coverage* is in place to mitigate the faults.
4. Demonstrate that *residual* risk after applying diagnostics is acceptable for the assigned ASIL.

Given the scale and complexity of modern SoCs—containing millions of sequential elements and interacting Ips the standard [3] introduces a **hierarchical decomposition model** to make semiconductor safety analysis both feasible and traceable. This hierarchical breakdown allows the safety analysis to be performed at **multiple abstraction levels**, each

refining the level of details required and ensuring consistent mapping between safety goals, safety requirements, and verification results. This decomposition enables a **bottom-up compositional methodology** as shown in table 1, where analysis performed at lower levels (sub-part or elementary sub-part) can be **reused** and **aggregated upward** into higher-level safety metrics. This approach provides a scalable and traceable link between **low-level failure data** and **system-level safety assurance**.

Table 1: Bottom up build with safety and verification activities.

Level	Safety Activities	Verification Focus
Component	Full Semiconductor device under analysis	SoC safety plan, FMEDA summary
Part	Safety architecture design, ASIL decomposition, top-level SPFM/LFM computation	Architectural verification, safety goal tracing
Subpart	FMEDA preparation, local safety mechanism definition	Block-level simulation and fault injection
Elementary Subpart	Fault modeling, diagnostic coverage analysis	RTL/Gate Fault injection, formal proofs

This was completed using the extracted data from Elementary subpart and using EDA tools for FMEDA.

C. Integration of ESE FMEDA for Early Verification

Safety verification in semiconductor designs has traditionally been a late-stage activity, performed after the gate-level netlists become available. The introduction of e**lementary subpart extraction (ESE)** shifts this verification earlier in the design cycle, enabling **RTL-based safety validation** that aligns with the standard's "early verification" recommendations for semiconductor safety lifecycle activities [3]. In standard automotive safety flows, FI is used to validate the diagnostic coverage (DC) and confirm the robustness of safety mechanisms. However, performing this at gate level introduces high computational cost and limited agility. Integrating **ESE-derived FMEDA models** into FI addresses both challenges, providing structural precision and functional correlation at RTL. [1] [5] [6]

Targeted FI, this reduces the simulation load while maintaining completeness in safety-relevant regions. We are able to generate precise fault list that reduce the load on fault injection. **Result closure coop,** this creates a **closed verification loop** between FMEDA structural analysis and FI behavioral results, improving both traceability and coverage accountability. **Dynamic update during the design phase,** because ESE operates at RTL (quick synthesis), incremental design updates such as adding safety mechanisms or modifying logic can be done early in the design process.

Figure 3, provides a contrast between the traditional vs the new methodology involving the Elementary subpart extraction and early verification which ultimately provide an overall shift left to the safety verification process.

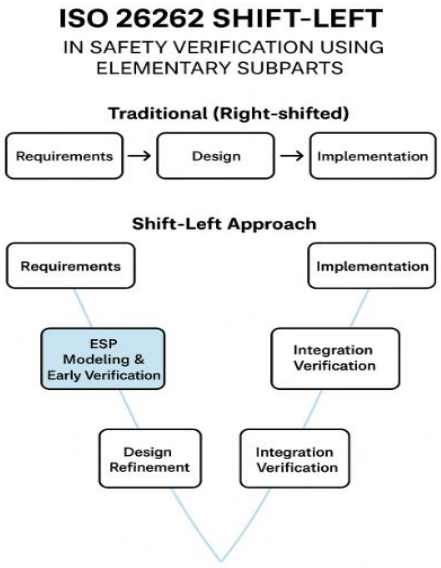


Figure 3: Shift left in safety verification flow.

3. Case Study - ESE applied to an Open Core RISCV CPU

Design under test (DUT) (representative):

The CPU core used in this case study is a small, embedded RISC-V design featuring an in-order, single-issue architecture. The RTL implementation consists of approximately 12,000 sequential elements, such as flip-flops and latches, and about 450,000 combinational logic gates. The initial baseline configuration includes several safety mechanisms, such as parity protection for register file read and write operations, error-correcting code (ECC) on the data RAM, a basic watchdog timer, and sanity checks for control register writes. Figure 4 illustrates the overall methodology and flow that will be used to test the CPU core in this case study.

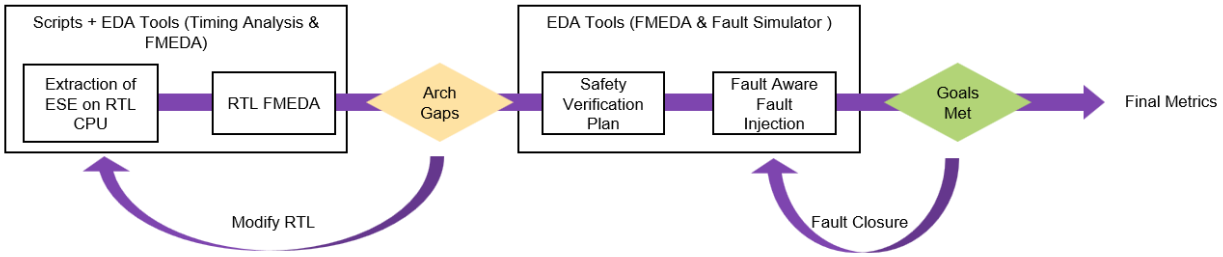


Figure 4: Overall methodology and proposed flow

A. Extraction of ESE on the RTL CPU

- *COL calculations*: For each sequential element, calculate its effective area by summing its own area with the areas of all upstream COL gates.
- *Overlap adjustment*: Use timing analysis to proportionally distribute shared gate areas, preventing double-counting.
- *Final modelling*: This creates weighted ESEs that represent each element's adjusted contribution to system failure exposure.

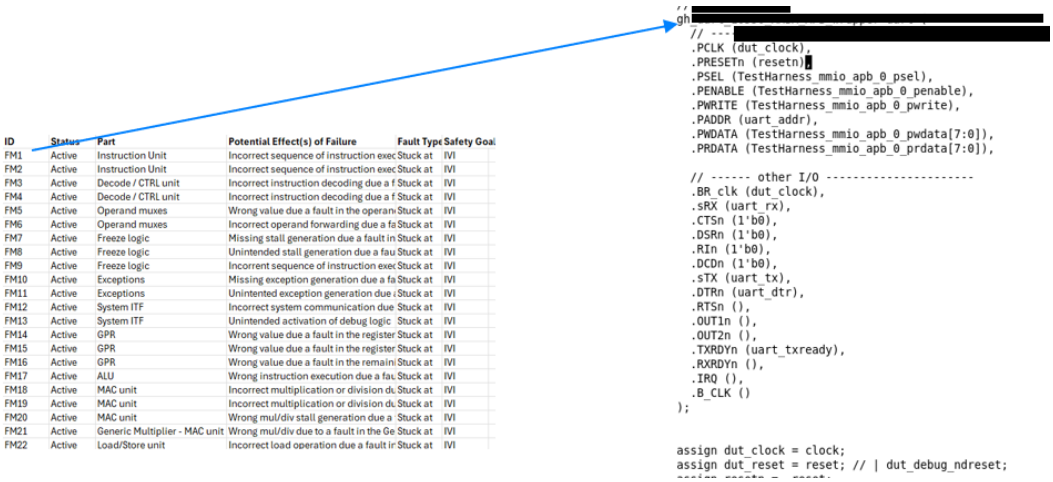
Figure 5, shows how the extracted data is brought into the FMEDA tool. This data will be used for design association for various failure modes.

#	Simple Hierarchy Name	Real Hierarchy Name	Digital Area
-	Technology Total Count	Technology Total Count	20278.00
1	top	or1200_top.or1200_cpu	307.00
2	or1200_ctrl	or1200_top.or1200_c...	62.00
3	or1200_except	or1200_top.or1200_c...	1675.00
4	or1200_freeze	or1200_top.or1200_c...	36.00
5	or1200_gmultu	or1200_top.or1200_c...	456.00
6	or1200_dmmu_tlb	or1200_top.or1200_c...	241.00
7	or1200_genpc	or1200_top.or1200_c...	1050.00
8	or1200_ctrl	or1200_top.or1200_c...	776.00
9	or1200_xprs	or1200_top.or1200_c...	1632.00
10	or1200_if	or1200_top.or1200_c...	324.00
11	or1200_alu	or1200_top.or1200_c...	3173.00
12	or1200_operandmuxes	or1200_top.or1200_c...	423.00
13	or1200_lsu_shadow	or1200_top.or1200_c...	334.00
14	or1200_lsu_shadowdata...	or1200_top.or1200_c...	105.00
15	or1200_lsu_shadowctrl...	or1200_top.or1200_c...	266.00

Figure 5: ESP contributions list for Design mapping.

B. Safety Analysis Using ESE Data

Structurally similar ESEs are aggregated into FMEDA entries, combining effective area and failure rate. Early FMEDA provides the ability to perform What-If analysis at RTL and preview the effectiveness of the overall safety metrics and Architectural vulnerability. Figure 6 shows, the FMEA break down of the IP and how each failure mode is really a representation of HW logic under analysis. Once we have all the failure mode associated with design and mapping of safety mechanisms/diagnostics complete, we are able to get a full FMEDA extracted in the format represented in Figure 7. Use of EDA software really helps in the automation and checking for errors as well as tracking changes that happened during this process.



ID	Status	Part	Potential Effect(s) of Failure	Fault Type	Safety Goal
FM1	Active	Instruction Unit	Incorrect sequence of instruction exec	Stuck at	IVI
FM2	Active	Instruction Unit	Incorrect sequence of instruction exec	Stuck at	IVI
FM3	Active	Decode / CTRL unit	Incorrect instruction decoding due a f	Stuck at	IVI
FM4	Active	Decode / CTRL unit	Incorrect instruction decoding due a f	Stuck at	IVI
FM5	Active	Operand muxes	Wrong value due a fault in the operan	Stuck at	IVI
FM6	Active	Operand muxes	Incorrect operand forwarding due a fa	Stuck at	IVI
FM7	Active	Freeze logic	Missing stall generation due a fault in	Stuck at	IVI
FM8	Active	Freeze logic	Unintended stall generation due a fau	Stuck at	IVI
FM9	Active	Freeze logic	Incorrent sequence of instruction exec	Stuck at	IVI
FM10	Active	Exceptions	Missing exception generation due a fa	Stuck at	IVI
FM11	Active	Exceptions	Unintented exception generation due a	Stuck at	IVI
FM12	Active	System ITF	Incorrect system communication due	Stuck at	IVI
FM13	Active	System ITF	Unintended activation of debug logic	Stuck at	IVI
FM14	Active	GPR	Wrong value due a fault in the register	Stuck at	IVI
FM15	Active	GPR	Wrong value due a fault in the register	Stuck at	IVI
FM16	Active	GPR	Wrong value due a fault in the remaini	Stuck at	IVI
FM17	Active	ALU	Wrong instruction execution due a fau	Stuck at	IVI
FM18	Active	MAC unit	Incorrect multiplication or division du	Stuck at	IVI
FM19	Active	MAC unit	Incorrect multiplication or division du	Stuck at	IVI
FM20	Active	MAC unit	Wrong mul/div stall generation due a	Stuck at	IVI
FM21	Active	Generic Multiplier - MAC unit	Wrong mul/div due to a fault in the Ge	Stuck at	IVI
FM22	Active	Load/Store unit	Incorrect load operation due a fault ir	Stuck at	IVI

```
// ---
.PCLK (dut_clock),
.PRESETn (resetn),
.PSEL (TestHarness_mmio_apb_0_psel),
.PENABLE (TestHarness_mmio_apb_0_penable),
.PWRITE (TestHarness_mmio_apb_0_pwrite),
.PADDR (uart_addr),
.PWDATA (TestHarness_mmio_apb_0_pwdata[7:0]),
.PRDATA (TestHarness_mmio_apb_0_prdata[7:0]),

// ------ other I/O ----------------------
.BR_clk (dut_clock),
.sRX (uart_rx),
.CTSn (1'b0),
.DSRn (1'b0),
.RIn (1'b0),
.DCDn (1'b0),
.sTX (uart_tx),
.DTRn (uart_dtr),
.RTSn (),
.OUT1n (),
.OUT2n (),
.TXRDYn (uart_txready),
.RXRDYn (),
.IRQ (),
.B_CLK ()
);


assign dut_clock = clock;
assign dut_reset = reset; // | dut_debug_ndreset;
```

Figure 6: Failure mode and underlying RTL code.

					Permanent failures								Transient failures				
Part	Sub-part	Elementary sub-parts	Safety Related Component ? Not Safety-Related Component ?	Failure modes	Failure rate (FIT)	Amount of safe faults (see note 1)	Safety mechanism(s) preventing the violation of the safety goal	Failure mode coverage wrt. violation of safety goal	Residual or Single Point Fault failure rate / FIT	Safety mechanism(s) preventing latent faults	Failure mode coverage wrt. Latent failures	Latent Multiple Point Fault failure rate / FIT	Failure rate (FIT)	Amount of safe faults (see note 1)	Safety mechanism(s) preventing the violation of the safety goal	Failure mode coverage wrt. violation of safety goal	Residual or Single Point Fault failure rate / FIT

Figure 7: Overall FMEDA work product using ESE data.

C. Generation of Safety Verification Plan

Through automated extraction, the FMEDA entries with associated design is systematically converted into a comprehensive *Safety Verification Plan*, ensuring complete and traceable coverage.

Fault-aware fault injections are employed, allowing faults to be introduced in relevant contexts, which enhances both the relevance of tests and the effectiveness of safety validation. Key benefits of this process include accelerated test creation and the ability to directly link results to specific failure modes and safety goals, supporting audit readiness and regulatory compliance. This approach reduces manual effort and minimizes the risk of human error, while providing clear documentation for efficient collaboration across teams.

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

# Safety Mechanism Definition
SafetyMechanism PSM_001 {
  Detect {
    " top_testbench.dut_top_i.core_cpu.ipx_rf.safety_err_code"
  }
}
# Failure Mode Observability
FailureMode FM_001 {
  Observe {
    "top_testbench.dut_top_i.core_cpu.ipx_rf"
  }
  SafetyMechanisms (PSM_001)
}
# Safety Aware Fault Generation
FaultGenerate FM_019,FM_021 {
  NA [0, 1] { PORT "top_testbench.dut_top_i.core_cpu.ipx_rf.**" }
  NA [0, 1] { VARI "top_testbench.dut_top_i.core_cpu.ipx_rf.**" }
}
```

Figure 8: Safety Verification Plan Output.

D. Fault Injection Results and Observations

- *Higher accuracy*: Achieved 30–40% better correlation between injected faults and safety goals.
- *Reduced redundancy*: Cut redundant fault simulations by over 60% through more targeted injections.
- *Enhanced traceability*: Improved audit confidence by mapping each test to a specific logical unit (ESE) rather than arbitrary gates

Table 2: Results on IP's data showing ESE flow Improvement.

Metric	Traditional Flow	ESE Fault Aware	Improvement
IP 1			
SPFM	62.3%	91.5%	46.8%
Faults Simulated	11,604	7,809	33%
Unresolved Faults	1,200	600	50%
IP 2			
SPFM	77.8%	90.7%	16.5%
Faults Simulated	72,867	43,632	40%
Unresolved Faults	5,100	870	83%

4. Conclusion

The application of **elementary subpart extraction (ESE)** for semiconductor safety analysis represents a significant advancement with practical, scalable design methodologies. By systematically decomposing the RTL into fine-grained subparts each representing a sequential element and its corresponding fan-in cone, the ESE approach transforms safety analysis from a reactive verification exercise into a *proactive architectural design practice*. From a quantitative standpoint, applying ESE yielded marked improvements in both **SPFM** and **LFM** convergence. The area-weighted subpart modeling provided more accurate fault rate attribution and diagnostic balancing, providing accuracy in the safety analysis. [5][6]

Table 3: Summary of gains in using ESE methodology.

Topics	Elementary Subpart Methodology Contribution/Gains
Shift-Left Enablement (FMEDA)	Enables FMEDA computation at RTL without waiting for final gate-level data.
Architectural Visibility	ESE-based modeling allows detection of hidden architectural vulnerabilities.
Iterative Efficiency	Allows rapid re-evaluation of safety metrics when RTL or safety mechanism bindings change.
Overlap Correction	Eliminates double counting of shared logic via timing and dependency analysis, yielding realistic λ allocation.
Hierarchical Integration	Supports compositional safety analysis from IP to SoC level, enabling reuse of ESP data as verified work products.
Scalability	This enhanced FMEDA approach leverages early RTL data to quantify failure propagation paths thus improving scalability for both FMEDA and Fault injection flow.

In conclusion, the **ESE-based FMEDA and verification flow** provides a measurable *shift-left* in safety verification. It reduces reliance on exhaustive fault simulation, offers earlier and more accurate insight into architectural vulnerabilities, and shortens the overall safety validation cycle. By coupling **analytical precision** with **automation scalability**, ESE transforms FMEDA from a compliance activity into a dynamic design optimization tool, thus improving both efficiency and safety confidence.

This methodology represents a practical step toward *next-generation ISO 26262 workflows*, where safety is not verified after design maturity but continuously engineered throughout the RTL development lifecycle.

5. Reference and Acronyms

[1] Methodology for Efficient closure to Fault injection, Dvcon Europe 2022
[2] ISO 26262 Part 5: Product development at the hardware level, Second edition 2018-12
[3] ISO 26262 Part 11: Guidelines on application of ISO26262 to semiconductors, Second edition 2018-12
[4] Formal Techniques for Optimizing ISO 26262 Fault Analysis, Siemens Verification Academy
[5] Rambus RT-640 road to ISO26262 certification, Rambus white paper
[6] Complex Safety Mechanisms Need Interoperability for Validation and Close Loop for Final Metrics, Protecting Safety and Security, DVCon US 2023
[7] Pragmatic Hardware Safety Analysis for ASIL-D SoCs
[8] Structural FMEDA for Early Safety Correlation, DVCON Europe 2023

Acronym	Full Form / Description
ASIL	Automotive Safety Integrity Level
ESP	Elementary Subpart
ESE	Elementary Subpart Extraction
COL	Cone Of Logic
DC	Diagnostic Coverage
DUT	Design Under Test
EDA	Electronic Design Automation
ESE	Elementary Subpart Extraction
FI	Fault Injection
FMEDA	Failure Modes, Effects and Diagnostic Analysis
GLS	Gate Level Simulation
SM	Safety Mechanism
SPFM	Single Point Fault Metric
LFM	Latent Fault Metric
PMHF	Probabilistic Metric of for random Hardware Failure
RTL	Register Transfer Level
STA	Static Timing Analysis
SoC	System on Chip
STL	Software Test Library