

Intelligent Fault Injection Scenario Generation Using Level 4 Virtual Electronic Controller Unit and Verification Efficiency Enhancement

- Systematic Fault Injection and Coverage Evaluation on a Level 4 AUTOSAR Virtual ECU -

Young-Mo Lee¹⁾ Dae-ki Kim¹⁾ Sung-hoon Woo¹⁾

¹⁾ Hyundai-Autoever, 12, Teheran-ro 113-gil, Gangnam-gu, Seoul, Republic of Korea

KEY WORDS: Virtual ECU; Fault Injection; AUTOSAR; ISO 26262; FDC; Software-in-the-Loop

1. Introduction

ISO 26262 [1] requires systematic fault injection verification of automotive safety mechanisms; however, physical ECU-based injection is constrained by hardware inaccessibility, timing imprecision, permanent damage risk, and limited observability of internal software state [4,9]. A Level 4 virtual ECU (vECU), executing the unmodified production binary on an instruction-set simulator with AUTOSAR-compliant peripheral models [6,10], eliminates all of these constraints. This paper presents a category-based fault injection framework on a Level 4 vECU with contributions in: (1) 14-category fault classification across four AUTOSAR layers with physical ECU feasibility assessment; (2) a systematic campaign covering all fault classes and architectural layers; (3) quantitative FDC evaluation stratified by feasibility; and (4) campaign throughput characterization demonstrating CI/CD-ready scalability.

2. Fault Classification and Catalog

Following ISO 26262 Part 4/5 terminology, 14 categories span six fault classes (Memory, Communication, Timing, I/O, Safety Monitor, SW Interface) across Hardware, MCAL, BSW, and OS layers. Feasibility is rated High (bus/signal injectable), Medium (JTAG required), or Low (vECU only). The catalog — derived from AUTOSAR R44 BSW source analysis and TC397 configuration — totals 730 fault candidates: Hardware 166 (RAM bit-flip 144, DMA 22), MCAL 244 (CAN 12, LIN 6, ADC 20, DIO 150, SPI 56), BSW 315 (API stub 175, DEM 77, NvM 9, WdgM 23, SMU 14, E2E 17), OS 5.

3. Experimental Setup

The framework was applied to a battery electric vehicle drive controller (AUTOSAR, TC397, ASIL-B). The production binary executes on a Level 4 vECU hosted on a Windows x86-64 workstation, connected via TCP to a Python-based fault injection runner supporting memory write, software stub, SFR register write, analog port override, CAN/LIN frame injection, and listener-based verdict collection without halting simulation.

4. Results and Verification Efficiency

Table 1 presents FDC results across all 730 test cases. All three feasibility groups achieved 100% FDC, confirming complete safety mechanism coverage across all four AUTOSAR layers and six fault classes. The Low feasibility group (378 TCs, 51.8% of total) — executable only on the vECU — verified that software-internal faults including RAM bit-flips, BSW API errors, NvM corruption, and watchdog supervision failures are correctly detected.

Feasibility	Description	TCs	FDC
High	Bus/signal injectable	157	100.0%
Medium	JTAG/debug interface	167	100.0%
Low	vECU only (not feasible on HW)	378	100.0%
Overall		730	100.0%

Table 1. FDC stratified by physical ECU feasibility (730 TCs, all PASS).

The complete 730-case campaign executed in approximately six minutes on a single host instance (~430 ms/TC average). Throughput scales linearly with concurrent vECU instances, enabling CI/CD integration. Physical ECU campaigns incur additional overhead from hardware allocation, flashing cycles, and probe attachment; 51.8% of test cases are inaccessible on physical hardware regardless of tooling, requiring source-level code modification per scenario.

5. Conclusion

A systematic Level 4 vECU-based fault injection framework achieved 100% FDC across 730 test cases spanning 14 categories and four AUTOSAR layers. The 51.8% Low-feasibility group — physically inaccessible on ECU hardware — was fully verified on the vECU in six minutes. The approach complements HIL testing by covering an inherently inaccessible fault space, enabling more complete ISO 26262 evidence at lower cost and earlier in the development cycle.

References

- (1) ISO 26262:2018, Road Vehicles — Functional Safety, ISO, 2018.
- (3) AUTOSAR Consortium, "Virtual Functional Bus," Release 23-11, 2023.
- (4) Natella et al., ACM Comput. Surv., 2016. DOI:10.1145/2841425
- (6) Park, Choi, Electronics, 2020. DOI:10.3390/electronics9050850
- (7) Bringmann, Krämer, ICST 2008. DOI:10.1109/ICST.2008.45
- (9) Hsueh et al., Computer Vol.30, 1997. DOI:10.1109/2.585157
- (10) Kim et al., Electronics Vol.13, 2024. DOI:10.3390/electronics13183704