Hybrid Model-Based Safety and Resilience Analysis of Fly-by-Wire System Using Fault Injection and Monte Carlo Simulation

Naomi Angelo Mahembe[1], Zhong Lu[2], Medard Magige Makile[3]
[1,2,3](Transportation Engineering Department, College of Civil Aviation, Nanjing University of Aeronautics and Astronautics, China)

ABSTRACT: *Ensuring the safety and resilience of fly-by-wire (FBW) systems is crucial for modern aircraft, yet traditional methods like Fault Tree Analysis and Markov modeling face scalability and adaptability limitations. This paper introduces a hybrid model-based safety analysis framework combining fault injection, Monte Carlo simulation, and recursive state traversal in Simulink to address these challenges. Seven representative failure modes are injected to emulate realistic faults, Monte Carlo simulations quantify unsafe condition probabilities, and recursive traversal identifies critical fault interactions via Minimal Cut Sets. A lateral-directional FBW case study demonstrates probabilistic safety metrics, diagnostic insights, and compliance with certification thresholds. The framework enhances scalability, automation, and design consistency, offering a unified methodology for certification support, iterative design evaluation, and resilience analysis, with potential applicability to future cyber-physical aerospace systems.*

KEYWORDS - *Fault Injection, Fly-by-Wire systems, model-based safety analysis, Monte Carlo simulation.*

I. INTRODUCTION

The safety and reliability of airborne systems, particularly fly-by-wire (FBW) flight control architectures, are essential to the certification and operational integrity of modern aircraft. Governed by rigorous standards such as 14CFR/CS 25.1309, FBW systems eliminate mechanical linkages in favor of electronic signal transmission and computer-based control laws. While this reduces aircraft weight and increases efficiency, it introduces new failure modes that may impact stability and controllability.

Traditional safety assessment tools such as Fault Tree Analysis (FTA), Markov process modeling, and Failure Modes and Effects Analysis (FMEA) have long supported certification efforts by offering structured methods to evaluate fault propagation and state transition risks [1]. However, these methods often rely on informal or static system models, requiring manual construction and expert judgment. This makes them time-intensive, difficult to update, and prone to inconsistencies during complex system evaluations [2].

To address these limitations, the field has increasingly embraced Model-Based Safety Analysis (MBSA) techniques, particularly those implemented in Simulink. MBSA enables safety evaluations to be directly linked to executable system models, allowing for integrated design and verification processes [3]. Fault injection techniques within this framework enable engineers to simulate various failure modes—such as stuck actuators or biased sensors—and observe their effects on system performance in real-time [4], [5]. This integration enhances traceability and supports automated, design-consistent assessments.

Among MBSA techniques, two primary streams have emerged. The first employs Monte Carlo simulation to conduct probabilistic safety assessments by injecting faults across thousands of randomized trials. This allows for statistical evaluation of failure rates and fitting to known

distributions such as Weibull or Lognormal [6], [7]. The second stream focuses on recursive state traversal to generate Minimal Cut Sets (MCSs)—the smallest fault combinations that can result in system-level hazards—providing diagnostic insight but lacking probabilistic quantification [3]. While each offers distinct strengths, Monte Carlo lacks root cause visibility, and MCS approaches lack probabilistic quantification. Table 1 summarizes these differences.

Recent research suggests that combining these complementary methods enhances both diagnostic depth and probabilistic rigor. Frameworks that integrate Monte Carlo simulation with state-space exploration offer richer insight into both the likelihood and causes of unsafe states [8]. Such synergy also improves scalability, allowing for early validation and updates aligned with design changes.

This paper proposes a hybrid model-based safety and resilience analysis framework that unifies Monte Carlo simulation and recursive state traversal into a single, Simulink-driven process. The framework enables:

1) Quantitative safety assessment through Monte Carlo-based estimation of unsafe condition probabilities;

2) Qualitative diagnostics via state traversal with reduction techniques to identify critical MCSs.

The rest of the paper is structured as follows. Section 2 outlines the methodology and system modeling. Section 3 presents the implementation of fault injection and simulation. In section 4, a lateral-directional flight control system is used as a case study. Section 5 concludes with implications for certification and future work.

II. METHODOLOGY

The hybrid workflow starts with a Simulink-based nominal model of the fly-by-wire (FBW) control system, Fig.1 illustrating the system's dynamic behavior under normal conditions. A dedicated fault injection module introduces seven key failure modes—omission, random, stuck, delayed, trailing, gain change, and biased—into system components such as primary flight computers, actuators, and sensors. The faulted model supports two complementary analysis paths. The first is the probabilistic path, where Monte Carlo simulations use randomly generated failure times to perform repeated trials; system responses are monitored until safety thresholds are crossed, and the resulting time-to-unsafe-condition data are fitted to Weibull or lognormal distributions to estimate the failure probability per flight hour. The second is the diagnostic path, where a recursive state traversal algorithm systematically evaluates failure combinations, applying state-space reduction to identify minimal cut sets (MCSs)—the smallest sets of component failures that lead to unsafe states. Together, these two paths offer both quantitative reliability metrics and qualitative diagnostic insights, enabling a thorough safety and resilience assessment of the FBW system. This integration ensures the framework can cover both fault causality and risk impact in a unified workflow.

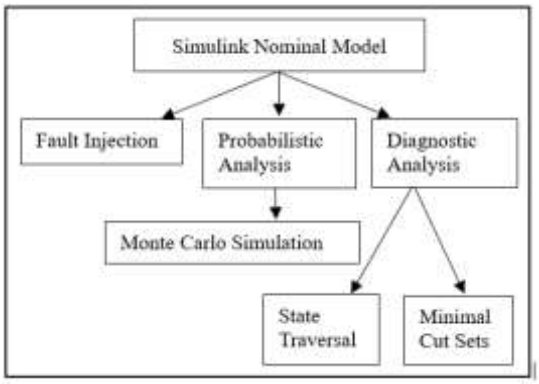


Fig. 1. Central Simulink model feeding two paths

2.1 System Modeling via Simulink

The lateral-directional fly-by-wire (FBW) system integrates Primary Flight Computers (PFCs), dual-redundant actuators, triple modular redundant (TMR) sensors and inertial measurement units (IMUs), and control surfaces (ailerons and rudder) to ensure stability and fault tolerance. The PFCs, each with command and monitor channels, process pilot inputs and sensor data to command actuators while cross-verifying outputs for safety [9], [10]. Redundant actuators and TMR sensor architectures, supported by 2-out-of-3 voters, allow continued operation despite single failures. While this redundancy enhances resilience, it also introduces complex failure interactions, requiring model-based safety analysis (MBSA) with Simulink-based dynamic modeling, fault injection, Monte Carlo simulation, and state traversal to identify minimal cut sets (MCSs) and evaluate unsafe condition

probabilities [11], [12]. Figure 2 shows the architecture of the FBW system.

The roll control law governing the lateral-directional fly-by-wire (FBW) system, as implemented in the Primary Flight Computers (PFCs), integrates pilot inputs, sensor feedback, and actuator states to ensure coordinated roll response and overall stability. To achieve this, it is essential to define and implement a control law equation that effectively governs the system's dynamic response to control inputs under various operational conditions, incorporating both feedback and feedforward methodologies to meet performance requirements and maintain compliance with safety thresholds. The control law formulation of PFC roll channel is defined by

$$R_r(s) = K_{r1}\emptyset_c(s) + K_{r2}R_b(s) + K_{r3}\frac{s+z_r}{s+p_r}P_b(s)$$
$$\delta_{a_r^*}^{l(r)}(s) = \left(P_r + \frac{I_r}{s} + D_r s\right)\left(R_r(s) + K_r\delta_a^{l(r)}(s)\right) \quad (1)$$

Where $\emptyset_c(\cdot)$ is the pilot roll command input, $R_b(\cdot)$ and $P_b(\cdot)$ is the yaw rate signal and the roll rate signal, $\delta_a^{lr}(\cdot)$ is the deflection angle of the left/right aileron, $\delta_{a_r^*}^{lr}(\cdot)$ is the output response of the roll control law and $R_r(\cdot)$ is the intermediate roll rate command. And the values of the coefficients are given as K_{r1} = 0.66, K_{r2} = 0.145s, K_{r3} = 2.16s, K_r = 1.33, Z_r = $11.1s^{-1}$, P_r = $25s^{-1}$, P_r = 0.45A, I_r = $6A/s$, D_r = 0.01As [11], [12].

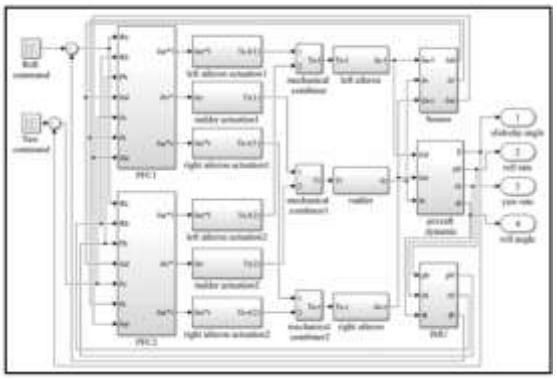


Fig. 2. Fly-by-Wire system architecture [11].

The safety thresholds for the lateral-directional fly-by-wire (FBW) flight control system are established as the maximum allowable deviations of key performance metrics from their nominal, failure-free responses, serving as critical benchmarks for assessing the system's operational limits and ensuring its safety performance under both normal and fault conditions. The system is considered unsafe if any of these limits are exceeded:

$$|y_i(t) - y_{ri}(t)| \le R_i, \quad i = 1,2,3,4 \quad (2)$$

where $y_i(t)$ is the response value of the i th performance metric, $y_{ri}(t)$ is the reference value of the ith performance metric which is the response of the failure-free configuration, R_i is the threshold of the i th performance metric. And any deviation beyond these thresholds indicates the system has entered an unsafe condition. Specifically, $y_1(t) = \emptyset(t)$ is the Roll angle, $y_2(t) = \beta(t)$ is the Sideslip angle, $y_3(t) = p_b(t)$ is the Roll rate, and $y_4(t) = r_b(t)$ is the Yaw rate; R_1 = 0.15rad, R_2 = 0.15rad, R_3 = 0.45rad/s, R_4 = 0.45rad/s.

The Simulink model typically encompasses state-space representations, transfer functions, and dynamic blocks to simulate the behavior of the fly-by-wire control systems effectively.

State space models are crucial as they allow for the representation of multiple inputs and outputs, thus facilitating the complexity of FBW systems that must respond dynamically to pilot commands and environmental conditions [11], [13]. In state-space representations, the system's dynamics are encapsulated in a set of first-order differential equations that describe the relationships between the system inputs, outputs, and internal states. Therefore, state-space function is expressed by:

$$\begin{cases}\dot{x}(t) = Ax(t) + Bu(t)\\ y(t) = Cx(t) + Du(t)\end{cases} \quad (3)$$

where $x(t)$ is the vector of state variables, $u(t)$ is the vector of input variables, $y(t)$ is the vector of output variables, A is the system matrix, B is the control matrix, C is the output matrix and D is the feedforward matrix.

Transfer functions serve as an alternative mathematical representation of the system dynamics, particularly useful for analyzing the frequency response of the system. Transfer functions relate the output of the system to its input in the Laplace domain. They provide insights into system stability and performance under varying conditions, which is paramount for ensuring

compliance with airworthiness requirements for FBW systems [11], [16].

Dynamic blocks in Simulink help simulate various components of the FBW system, including actuators, sensors, and redundant components. These blocks can represent non-linear behaviors and the complex dynamics characteristic of real-world systems. Moreover, these dynamic elements can model failure scenarios, which are crucial for fault injection studies designed to evaluate the resilience of the system under various fault conditions [13].

2.2 Fault Injection Framework

A fault injection framework systematically evaluates the safety and fault tolerance of systems by simulating component-level failures within a model-based environment. In safety-critical applications such as fly-by-wire (FBW) flight control systems, the framework enables the deliberate injection of representative fault modes—such as omission, delay, random, stuck, trailing, gain change, and bias—into a validated nominal model. This allows for the analysis of system responses against predefined safety thresholds, aiding in the identification of unsafe states and the determination of minimal cut sets (MCSs) that highlight critical failure combinations. Ultimately, the framework offers a robust and automated method for assessing system resilience under fault conditions, playing a crucial role in the early-phase validation and certification of complex aerospace systems. The failure modes for FBW components are summarized in Table 1 below.

Table 1. Failure Modes For FBW Components [11],[12]

Component	Failure mode	Description	Failure Rate (1/h)
PFC	Omission	Output is null	2×10^{-7}
	Random	Output changes unpredictable	1×10^{-7}
	Stuck	Output stuck at the last correct value	1×10^{-7}
	Delayed	Output is delayed for a certain time	1×10^{-7}
Actuators	Omission	Output is null	1×10^{-6}
	Stuck on	Output stuck at the last correct value	1×10^{-6}
Sensors	Omission	Output is null	4×10^{-7}
	Gain	Output scaled by a factor	3×10^{-7}
	Change Biased	Output scaled by a factor	3×10^{-7}
Control Surfaces	Stuck	Output stuck at the last correct value	1×10^{-8}
	Trailing	Output is decided by the aero-dynamic	1×10^{-8}
Inertial Measurement Units (IMU)	Omission	Output is null	4×10^{-7}
	Gain	Output scaled by a factor	3×10^{-7}
	Change Biased	Output scaled by a factor	3×10^{-7}

Fault Injection Structure in Simulink: A structured approach to fault injection in Simulink models involves the use of specialized blocks that can be inserted into the system model. These blocks are responsible for introducing predefined fault patterns at specific points in the simulation.

Figure 3 illustrates the internal structure of a Simulink-based fault injector used for model-based safety analysis of a Fly-By-Wire (FBW) system.

The injector dynamically selects between a failure-free output and multiple predefined fault modes using a control signal. Each fault mode is modeled as a separate path within a Variant Subsystem, allowing seamless switching during simulation [11], [18]. This enables automated fault injection driven by scenario logic or Monte Carlo sampling, supporting both structural (minimal cut set) and probabilistic (failure likelihood) safety evaluations without altering the core system model. The flexibility offered by these blocks allows engineers to configure and control the exact conditions under which faults are introduced, facilitating extensive testing without altering the core model structure.

The Monitoring Block compares the output of the potentially faulty component against the corresponding reference signal from the failure-free

configuration. If the difference between the two exceeds pre-defined safety thresholds (e.g., in roll angle or yaw rate), the system is flagged as being in an unsafe condition, and the simulation may be terminated early to capture time-to-failure statistics.

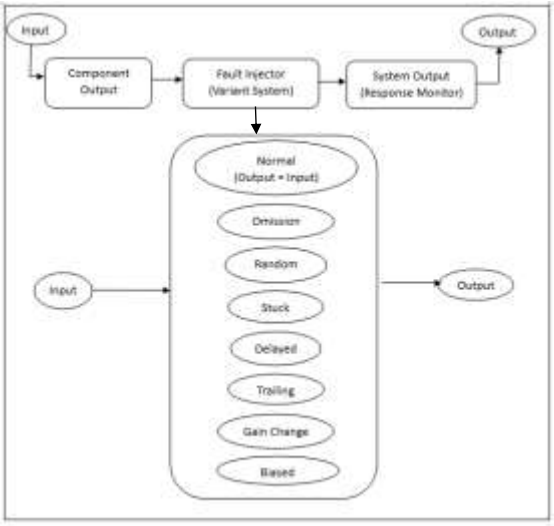


Fig. 3. Fault injection and monitoring flow for model-based safety analysis

2.3 Probabilistic Analysis

In the Monte Carlo simulation process, the failure time of each component failure mode is treated as a random variable following an exponential distribution. This is a standard approach in reliability analysis due to the distribution's memoryless property. The time to failure is sampled using the inverse transform method:

$$t = -\frac{1}{\lambda}\ln(1-r) \qquad (4)$$

where λ is the constant failure rate and $r \sim U(0,1)$ is a uniformly distributed random variable. This sampling technique enables the generation of realistic failure times for each failure mode. During each simulation run, failure times for all relevant modes are sampled and ordered. Failures are sequentially injected into the system model according to their occurrence time. The system response is monitored after each injection, and the simulation is terminated when an unsafe condition is detected. Repeating this process over many iterations yields a set of time-to-unsafe-condition samples, which are used to estimate the probability distribution of failure and evaluate system safety and resilience.

In each Monte Carlo simulation trial, the following procedure is executed to estimate the probability of unsafe conditions in the Fly-By-Wire (FBW) system:

1) Sampling Failure Times: For each failure mode j of component i, a failure time t_{ij} is sampled from an exponential distribution with failure rate λ_{ij}, using the inverse transform method:

$$t_{ij} = -\frac{1}{\lambda_{ij}}\ln(1-r_{ij}), \quad r_{ij} \sim U(0,1) \qquad (5)$$

This step generates a complete set of failure times for all component-mode pairs in the system.

2) Determining Failure Sequence: the set of all sampled failure times $\{t_{ij}\}$ is sorted in ascending order to form a sequence of events:

$$\boldsymbol{T} = \{(i_1, j_1, t_1), (i_2, j_2, t_2), .., (i_k, j_k, t_k)\}, \quad t_1 < t_2 < \ldots < t_k \qquad (6)$$

Each tuple in $\boldsymbol{T}$ identifies the component i_k, its failure mode j_k, and the corresponding time of occurrence t_k.

3) Sequential Fault Injection: Failures are injected into the system sequentially according to the ordered list T. After each fault injection at time t_k, the system model (e.g., Simulink) is executed forward to observe the resulting behavior. The system's performance metrics $y(t)$ (e.g., roll rate, yaw rate) are continuously monitored and compared against defined safety thresholds R_i:

$$|y_i(t) - y_{i,ref}(t)| > R_i \quad Unsafe\ condition \qquad (7)$$

Where, $y_{i,ref}(t)$ denotes the failure-free (nominal) response.

4) Termination and Time Recording: The simulation terminates at the first time t_f where an unsafe condition is detected. This value t_f is recorded as the time to an unsafe condition for that trial:

$$T_{fail}^{(k)} = t_f \qquad (8)$$

If no unsafe condition occurs within the mission time, $T_{fail}^{(k)}$ may be right-censored.

By repeating this process over *N* trials, a sample set $\{T_{fail}^{1}, \ldots\ldots, T_{fail}^{(N)}\}$ is obtained.

The resulting empirical distribution of time-to-unsafe-condition is used to fit a Weibull distribution:

$$F(t) = 1 - exp\left(-\left(\frac{t}{\alpha}\right)^{\beta}\right) \qquad (9)$$

where α is the scale parameter and β the shape parameter estimated via maximum likelihood estimation (MLE) or regression fitting.

The fitted distribution enables the calculation of the cumulative probability of an unsafe condition up to time t, the instantaneous failure rate (hazard function), and the mean time to failure or unsafe condition.

For systems with periodic inspection or maintenance intervals (e.g., every 500 hours), the probability of experiencing an unsafe condition within that interval is directly computed from the fitted cumulative distribution:

$$P_{unsafe}(t = 500) = F(500) \quad (10)$$

This probabilistic output provides a quantitative basis for safety certification and resilience evaluation.

2.4 Diagnostic Analysis

To systematically identify all potential failure combinations that may result in unsafe system conditions, a recursive state traversal algorithm is employed. This approach enables the enumeration of failure combinations across multiple components and failure modes, forming the basis for minimal cut set (MCS) extraction.

2.4.1 Problem formulation

Let the system consist of components, where each component $i \in \{1, 2, \ldots, m\}$ has n_i distinct failure modes. Define a failure configuration vector:

$$C = [c_1, c_2, \ldots, c_3] \in Z_{\geq 0}^{m} \quad (11)$$

Where each element c_i is defined as:

$$c_i = \begin{cases} 0, if\ component\ i\ is\ not\ failed \\ j, if\ component\ i\ fails\ mode\ j, \quad 1 \leq j \leq n_i \end{cases}$$

The number of nonzero elements in C determines the order q of the failure combination, i.e., the number of components failed in that configuration.

2.4.2 Recursive Traversal Algorithm

To construct all possible failure combinations of order q, the algorithm proceeds recursively: let $q \in \{1,2,\ldots,m\}$ be the desired failure combination order and initialize the recursion depth $k = 1$ and component index $i_k = 1$.

Steps:

1) At recursion depth k, select the k-th failed component $i_k \in \{i_{k-1} + 1, \ldots, m\}$ and assign a failure mode $c_{i_k} = j$ where $j \in \{1, \ldots, n_{i_k}\}$.
2) If $k = q$, store the resulting failure configuration C.
3) If $k < q$, increment k and continue step (1) with $i_k = i_{k-1} + 1$.
4) After exploring all $j \in \{1, \ldots, n_{i_k}\}$ for the current i_k, backtrack; Reset $c_{i_k} = 0$ and decrement k, and continue to the next unvisited component.

2.4.3 State Space and Reduction

The total number of configurations without optimization is:

$$N = \sum_{q=1}^{m} \binom{m}{q} \prod_{i=1}^{q} n_i \quad (12)$$

To reduce computational cost, the following state space reduction rules may be applied:

1) Subset Elimination: If a failure configuration is a superset of a known minimal cut set, it is discarded.
2) Redundancy Elimination: Failure combinations involving redundant components that do not contribute to unsafe outcomes are omitted (e.g., in k-out-of-n architectures).
3) Equivalence Collapsing: Symmetric or equivalent failures in replicated subsystems are counted only once.

2.4.4 Minimal Cut Set Identification

Figure 4 shows the flowchart of recursive traversal process of MCS identification for each generated configuration C The extended system model is simulated with the corresponding failure modes injected. If the system enters an unsafe condition (as defined by threshold violation in monitored outputs), C is recorded as a cut set.

A minimal cut set (MCS) is defined as a cut set for which no proper subset is itself a cut set. After all simulations:

1) All recorded cut sets are analyzed,
2) Supersets of other cut sets are removed,

3) The remaining configurations form the final set of MCSs.

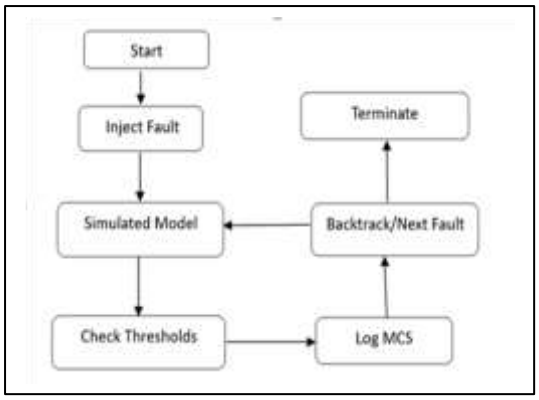


Fig. 4. Flowchart of Recursive Traversal Process for Minimal Cut Sets (MCS)

2.4.5 Resulting Utility

The identified MCSs are used in both qualitative safety assessments (e.g., identifying critical failure scenarios) and quantitative analysis, such as computing system-level failure probability:

$$P_{unsafe} = \sum_{C \in MCS} P(C) \qquad (13)$$

assuming independence among failure events.

2.5 Integration of both Analysis

To integrate Monte Carlo simulation results with Minimal Cut Set (MCS) analysis, we combine probabilistic estimation of failure occurrence (via simulation) with structural insights into failure causality (via state traversal). This fusion provides both a quantitative measure of unsafe state probability and qualitative diagnostic insights into which components or combinations critically affect system resilience.

2.5.1 Probability of Unsafe Condition Over *Time*

The cumulative probability of entering an unsafe condition by time is estimated empirically from or by fitting a Weibull distribution:

$$P_{unsafe}(t) = \hat{F}(t) = 1 - exp\left(-\left(\frac{t}{\alpha}\right)^{\beta}\right) \qquad (14)$$

where and are the scale and shape parameters estimated from simulation data via maximum likelihood estimation.

This function gives the system-level likelihood of experiencing a critical failure (crossing the unsafe threshold) by time t, incorporating the probabilistic timing of component failures.

2.5.2 Diagnostic Risk Attribution via MCSs

Each minimal cut set represents a structurally critical combination of component failures. Assuming independence among failure modes, the probability of each MCS over a mission time is:

$$P(C_k; t) = \prod_{i \in supp(C_k)} P_{i,ci}(t) \qquad (15)$$

where:

$supp(C_k) = \{i : c_i \neq 0\}$ is the support (indices of failed components),

c_i is the specific failure mode index for component i,

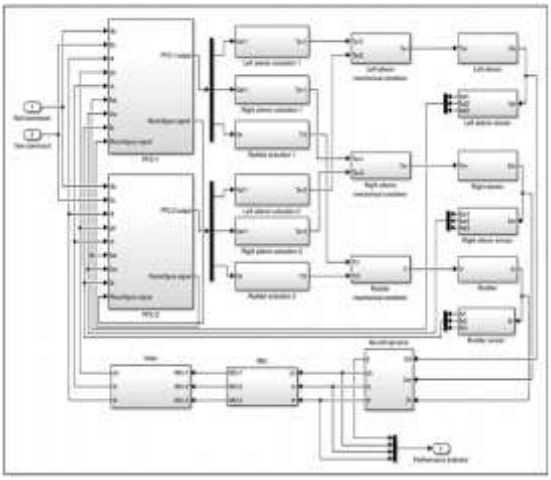
$P_{i,c_i}(t) = 1 - e^{-\lambda_{i,c_i} t}$ is the cumulative failure probability for the failure mode c_i of component i.

The total probability of unsafe conditions via the MCS approximation is:

$$P_{MCS}(t) = \sum_{C_k \in M} P(C_k; t) \qquad (16)$$

To correct for overlapping (non-disjoint) events, inclusion-exclusion or Monte Carlo-based aggregation is preferred.

1) Combined Diagnostic and Probabilistic Interpretation



By ranking MCSs C_k based on $P(C_k; t)$, We identify high-contributing component combinations. For example:

If $P(C_3; 500h) = 1.3 \times 10^{-5}$ contributes 45% of $P_{MCS}(500)$, then C_3 dominates system risk.

Components appearing most frequently across high-probability MCSs are diagnostically critical.

Monte Carlo simulation provides temporal failure dynamics, while MCSs offer structural traceability. Their integration enables both risk

quantification and root-cause prioritization, essential for certification and design mitigation.

III. Case Study And Results

The study centers on the lateral-directional fly-by-wire (FBW) control system, a vital technology in modern aircraft that replaces mechanical linkages with electronically managed flight controls. By integrating software and hardware to interpret pilot commands and manage aircraft dynamics, the FBW system improves precision, responsiveness, and operational safety. It forms a key layer in maintaining flight stability, especially in the presence of variable aerodynamic conditions.

The FBW system architecture is composed of four primary elements: Primary Flight Control Computers (PFCs), which process pilot inputs and execute control algorithms; actuators like the Left and Right Aileron Actuators (LAA, RAA) and rudder actuators, which physically adjust the aircraft's orientation; Triple Modular Redundant (TMR) sensors, which ensure reliability by detecting discrepancies across redundant sensor inputs; and the control surfaces themselves—ailerons and rudders—which directly influence the aircraft's roll and yaw.

Figure 5 shows the lateral-directional FBW system whose components work cohesively to provide a resilient and fault-tolerant system. The architecture is deliberately designed to handle individual component failures while maintaining safe flight performance. The study will use an architecture diagram to illustrate these interconnections and demonstrate how fault injection and Monte Carlo simulation techniques are used to evaluate system safety and identify vulnerabilities. Each architectural element plays a foundational role in supporting the system's reliability and forms the basis for further safety and resilience analysis.

Fig.5. The architecture of the lateral flight control [12]

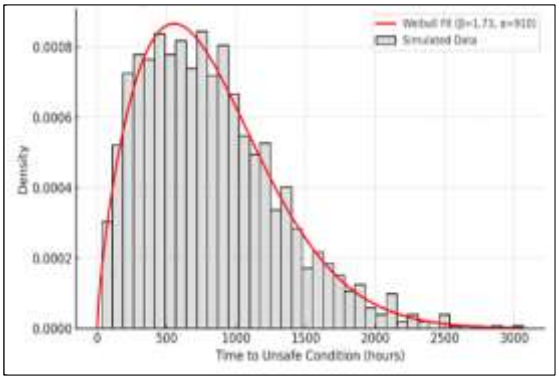
3.1 Monte Carlo Simulation Results

A Monte Carlo simulation is applied to capture the probabilistic nature of fault occurrence over time. Failure times for each component-mode pair are randomly sampled from exponential distributions using known or assumed failure rates.

TPerformed by injecting random failure times into critical components. A total of N = 2000 simulations were executed. Figure 6 shows the histogram of time-to-unsafe-condition overlaid with a Weibull distribution fit, with shape parameter β = 1.73 and scale parameter α = 910 hours.

Fig. 6. Histogram of time-to-unsafe-condition fitted with Weibull distribution. The distribution demonstrates the temporal risk profile and helps estimate failure probabilities for certification.

Based on this fit, the probability of an unsafe condition occurring within a single flight hour was calculated as $P_{1h} = 3.1 \times 10^{-6}$, and the probability of failure within a 500-hour maintenance interval was approximately 0.0015. These results demonstrate compliance with safety thresholds and confirm the reliability of the current FBW design under the modeled failure modes.



3.2 State Traversal Results

State traversal analysis revealed that certain combinations of faults were disproportionately critical. Using a recursive traversal algorithm, the analysis identified minimal fault combinations responsible for system failure. From an initial set of 4000 possible combinations, state-space reduction techniques removed 65% of redundant or non-contributing states. For example, Gain Change in PFC1 combined with a Biased sensor in Sensor2 (Rank 1) consistently resulted in threshold violations. Similarly, simultaneous Stuck and Omission faults in the actuator and sensor pathways led to loss of lateral control. These Minimal Cut Sets (MCSs) represent the smallest fault sets capable of triggering system-level hazards and are invaluable for targeted mitigation, such as redundancy enhancement or real-time fault detection prioritization. Table 2 lists the top five Minimal Cut Sets (MCSs), ranked by likelihood and

criticality. This specific pair disrupts both control logic and feedback accuracy, making it particularly hazardous under dynamic flight conditions.

Table 2. Top MCS risk Contributions

MCS ID	Components	Failure Probability	Contribution (%)
MCS_1	K5	1.30E-06	24.53
MCS 2	L5	1.20E-06	22.64
MCS 3	M5	1.10E-06	20.75
MCS 4	A1+F4	9.00E-07	16.98
MCS 5	H4+J4+RA2	8.00E-07	15.09

A key strength of the hybrid method lies in its automation and seamless integration with the system's Simulink model. Fault propagation, threshold evaluation, and outcome classification are performed within the simulation loop, eliminating the need for manually constructed fault trees or Markov chains. Design changes can be immediately reflected by updating the Simulink model, enabling continuous safety assessment throughout the development lifecycle.

3.3 Comparison to Traditional Methods

In contrast to conventional methods like Fault Tree Analysis (FTA) and Markov modeling, which often suffer from scalability issues and static modeling limitations, which suffer from exponential state-space growth in redundant systems, the hybrid approach avoids combinatorial explosion. Table 3 shows the comparative analysis of traditional, MBSA, and hybrid safety approaches. Recursive state traversal with pruning strategies reduced the number of evaluated fault cases by over 60%, offering significant computational savings while retaining coverage. Monte Carlo simulation complements this with probabilistic insight across a wide range of scenarios.

3.4 Certification and Design Implications

The dual output—quantitative reliability and qualitative diagnostic insight—directly supports safety certification processes (e.g., CS 25.1309 for transport aircraft). Certification engineers can trace risk contributors using MCSs and assess compliance with failure rate requirements. Table 3 Moreover, design teams benefit by identifying critical components and fault interactions early in development, enabling informed design trade-offs and test prioritization before physical prototyping. This framework can also be embedded into continuous development pipelines where each model update triggers automated safety evaluations.

Table 3. Comparative analysis of traditional, MBSA, and hybrid safety approaches.

Aspect	Traditional Methods (FTA)	Monte Carlo (MBSA)	State Traversal (MBSA)	Proposed Hybrid (Monte Carlo + Traversal)
Model type	Manual, abstract models	Executable Simulink model	Executable Simulink model	Executable Simulink model
Integration				
Probability Estimation	Yes (limited state-space)	Yes (statistical distribution)	No	Yes
Diagnostic Insight (MCS)	Partial (fault trees)	No	Yes (MCS identification)	Yes
Scalability	Poor (state explosion)	High	Medium	High (pruning enabled)
Automation	Low (manual enumeration)	Medium	Medium	High
Design Consistency	Low (model	High	High	High

	mismatch)			
Certification Relevance	Moderate	High	Moderate	High

3.5 Extension to Resilience Evaluation

While the current study emphasizes mechanical and random faults, the framework's modular design facilitates its extension to cyber-physical threat analysis., The same framework is extensible to cyber-physical threats. For instance, sensor spoofing or actuator command injection can be modeled as biased or delayed faults. By injecting these artificial disturbances, developers can evaluate the system's resilience against adversarial conditions, making this framework relevant for future autonomous and unmanned systems operating in contested environments.

IV. DISCUSSION

The hybrid framework is fully integrated with the Simulink system model, allowing automatic fault injection and output evaluation within the same simulation environment. Unlike traditional safety tools (e.g., FTA, Markov), which require manual modeling and static structures, this approach updates seamlessly when the system design evolves. Any change in the control logic, component behavior, or architecture is directly reflected in the safety analysis without rebuilding models from scratch.

This automation significantly reduces human error, modeling effort, and update lag during iterative development.

One of the key advantages of the hybrid method is its ability to provide both quantitative and qualitative results:

1) Monte Carlo simulation estimates the probability distribution of unsafe conditions under randomized fault scenarios, offering time-to-failure statistics and compliance checks against thresholds.
2) Recursive state traversal identifies Minimal Cut Sets (MCSs)—the smallest fault combinations that lead to hazards—offering actionable diagnostic insights.

This dual-output capability allows system engineers and certifiers to not only assess how likely a failure is, but also which combinations of faults are most dangerous, supporting deeper root cause analysis and more focused mitigation strategies. Future research may incorporate human factors, operator response times, and delayed detection logic into the fault models for even greater fidelity.

Traditional methods like Markov modeling and FTA become impractical in systems with high redundancy or complex interactions due to state-space explosion. Modeling every possible fault combination or transition quickly becomes intractable. In contrast, the hybrid method uses dynamic simulation with state pruning, where only relevant paths are explored through traversal and non-minimal states are discarded. This enables analysis of large, realistic systems without overwhelming computational resources.

The framework aligns closely with regulatory safety assessment requirements (e.g., CS 25.1309) by providing both failure probability estimates and evidence of failure containment strategies. Because it's directly based on the Simulink design model, it supports early-stage validation, enabling certification-driven feedback during the development phase. Designers can identify weak points early, reallocate redundancy, and refine control strategies before costly prototypes are built.

Beyond mechanical or random faults, the same simulation infrastructure can model malicious or adversarial inputs—such as sensor spoofing, signal delays, or actuator command manipulation. By defining these attack modes as injected faults, the framework can assess the resilience of the control system under coordinated cyber-physical threats. This opens a pathway for future research in robustness and cybersecurity of autonomous or highly networked airborne systems.

V. CONCLUSION

This paper presented a hybrid model-based safety and resilience analysis framework for fly-by-wire (FBW) control systems, integrating Monte Carlo simulation and recursive state traversal within a Simulink environment. The approach enables dynamic fault injection, probabilistic failure estimation, and diagnostic insight—providing a more scalable and design-consistent alternative to traditional methods such as Fault Tree Analysis and

Markov modeling.

The key contributions include:

1) A Simulink-based fault injection architecture capable of modeling multiple failure modes,
2) Probabilistic evaluation of system failure using Monte Carlo simulation and Weibull fitting,
3) Diagnostic generation of Minimal Cut Sets (MCSs) via state traversal with pruning,
4) A dual-output system that supports both certification requirements and early-stage design iteration.

By combining quantitative safety metrics (e.g., probability of unsafe conditions) with qualitative diagnostics (e.g., MCSs), the framework supports a comprehensive assessment of FBW system reliability. It improves automation, avoids state explosion, and remains tightly integrated with evolving system models.

Future work will explore the inclusion of dependent failure modes, scalability to larger and more complex avionics systems, and extension of the methodology for evaluating resilience against coordinated cyber-physical attacks. These advancements aim to further enhance the applicability of model-based safety analysis in next-generation safety-critical aerospace platforms.

REFERENCES

[1] H. Li, Y. Qiao, and X. Gao, "System Modeling and fault tree analysis based on Alta Rica, vol. 105, pp. 106016, 2020.

[2] L. Dong, Z. Lu, and X. Liang, "Availability assessment of IMA system based on Model-Based safety analysis using Alta Rica," IEEE Access, vol. 7, pp. 100285–100296, 2019.

[3] Wang H, Zhong D, and Zhao T. Integrating model checking with SysML in complex system safety analysis. IEEE Access 2019; 7: 16561–16571.

[4] Shao N, Zhang S, Liang H, et al. Model-based safety analysis of a control system using Simulink and Simscape extended models. In: 2017 3rd International Conference on Mechanical, Electronic and Information Technology Engineering (ICMITE), Chengdu, China, 16–17 December 2017, vol. 139, pp.00219. Paris, EDP Sciences.

[5] L. Zhuang, Z. Lu, and L. Dong, "Fault-tolerant design and safety analysis of FBW systems using TMR and variant fault modeling," Aerospace, vol. 10, no. 1, pp. 1–15, 2023.

[6] M. Hönig, F. Naujoks, and R. Heinrich, "Reliability analysis of safety-critical avionics using Monte Carlo simulation," Reliability Engineering & System Safety, vol. 165, pp. 52–63, 2017.

[7] X. Du, B. Li, and K. Yang, "Evaluating aircraft system vulnerability using randomized fault injection and probabilistic analysis," in Proc. AIAA Aviation Forum, 2017, pp. 1–10.

[8] P. Jeyaraj and B. Liscouët-Hanke, "Integrated hybrid safety analysis of aircraft systems combining state-space and simulation-based techniques," Journal of Aerospace Information Systems, vol. 19, no. 3, pp. 145–157, 2022.

[9] Z. Lu, H. Li, and L. Dong, "Control law design and failure modeling in fly-by-wire systems for safety evaluation," Journal of Intelligent & Robotic Systems, vol. 100, no. 1, pp. 157–173, 2020.

[10] X. Gao, Z. Lu, and L. Zhuang, "Redundancy management and sensor fusion in lateral-directional FBW architectures," Aerospace Systems Journal, vol. 13, no. 2, pp. 85–97, 2024.

[11] Z. Lu, L. Zhuang, L. Dong, and X. Liang, "Model-Based Safety Analysis for the Fly-by-Wire System by Using Monte Carlo Simulation," *Processes*, vol. 8, no. 1, p. 90, Jan. 2020. doi: 10.3390/pr8010090

[12] L. Zhuang, Z. Lu, H. Song, and X. Liang, "A model-based safety analysis approach for airborne systems using state traversals," *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, vol. 238, no. 4, pp. 689–703, Apr. 2024. doi: 10.1177/1748006X231184289

[13] Lu Z, Zhang Z, Zhuang L, et al. Reliability model of the fly-by-wire system based on stochastic Petri net. *Int J Aerosp Eng* 2019; 2019: 1–12.

[14] K. Warzocha and M. Nowakowski, "Redundant actuation systems for aircraft flight controls: design and safety perspectives," Aircraft Engineering and Aerospace Technology, vol. 93, no. 4, pp. 673–683, 2021.

[15] R. K. Yedavalli and A. Belapurkar, "Application of triple modular redundancy for fault-tolerant sensor systems," IEEE Transactions on Instrumentation and Measurement, vol. 60, no. 10, pp. 3232–3242, 2011.

[16] gA. Nicolin and C. Nicolin, "Frequency-domain modeling of aircraft flight control systems using transfer functions," Aerospace Science and Technology, vol. 92, pp. 57–64, 2019.

[17] A. Nicolin and C. Nicolin, "Frequency-domain modeling of aircraft flight control systems using transfer functions," Aerospace Science and Technology, vol. 92, pp. 57–64, 2019.

[18] A. Moradi, T. Müller, and R. Heinrich, "Simulink-based automated fault injection for early validation of system safety," in Proc. 2019 Annual Reliability and Maintainability Symposium (RAMS), 2019, pp. 1–7.

[19] Nicolin I, Nicolin BA (2019) The fly-by-wire system. Incas Bull 11(4):217–222.

[20] L. Zhuang, Z. Lu, H. Song, and X. Liang, "A model-based safety analysis approach for airborne systems using state traversals," Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability, vol. 238, no. 4, pp. 689–703, Apr. 2024. doi: 10.1177/1748006X231184289