Drones and Autonomous Vehicles

SCIEPublish

Article

Model-Based Formal Verification and Hybrid Fault-Tolerance for Safety-Critical Autonomous Vehicle Systems

Fayhaa Khulaif and Shawkat Sabah Khairullah *

Computer Engineering Department, College of Engineering, University of Mosul, Mosul 41002, Iraq; fayhaa.20enp14@student.uomosul.edu.iq (F.K.)

* Corresponding author. E-mail: shawkat.sabah@uomosul.edu.iq (S.S.K.)

Received: 11 June 2026; Revised: 24 August 2026; Accepted: 22 October 2026; Available online: 5 November 2026

ABSTRACT: Reliability and safety assurance in autonomous vehicle systems and related cyber-physical systems (CPSs) deployed in life-critical domains—such as automotive safety, avionics, industrial automation, and transportation—require rigorous verification and validation (V&V). Design errors in system specifications and hardware/software (HW/SW) faults within architectural models can compromise functionality and lead to catastrophic outcomes. This paper introduces a model-based verification framework that integrates formal verification with hybrid fault-tolerance mechanisms, demonstrated through a safety-critical Seatbelt Reminder (SBR) subsystem as part of intelligent vehicle safety architectures. The proposed design employs Simulink Design Verifier (SLDV) as a formal model checker to validate safety properties, while an embedded Triple Modular Redundancy (TMR) scheme detects and masks both transient and permanent faults. Simulation results show that safety property proofs were achieved with preprocessing and analysis times of 10 and 12 s, respectively. Runtime execution of the verifiable model demonstrated high dependability across diverse safety-critical properties, and the TMR system meets high reliability equal to 99.3096×10^{-2} with a failure rate of 0.005 failure per hour. Validation on a processor operating at 1500 MHz with 16 GB memory confirmed the effectiveness of the approach in enhancing reliability, fault resilience, and overall dependability in autonomous vehicle CPS design.

Keywords: Safety; Cyber-physical; Industrial automation; Dependability; Formal verification

1. Introduction

Reliable digital embedded devices are increasingly integral to autonomous vehicle systems and components in cyber-physical systems (CPSs) applications, where higher levels of reliability, availability, safety, and formal verification and validation (V&V) are essential [1–3]. Examples of such life-critical digital systems include safety-critical instrumentation and control (I&C) systems in industrial automation, medical CPS devices, avionics electronics, and intelligent transportation systems [4]. Within autonomous vehicles, the correct execution of hardware and software components is particularly critical, as these subsystems susceptible to the effects of different classes of software defects and natural hardware faults that can comprise safety [5]. Verification of the embedded software and hardware in CPS requires quality

© 2026 The authors. This is an open access article under the Creative Commons Attribution 4.0 International License (https://creativecommons.org/licenses/by/4.0/).

assurance of both desired and undesired properties to confirm that systems operate safely and effectively [6]. Formal verification and validation (V&V) techniques are therefore strongly advised in the design and modeling of autonomous vehicle architectures and cyber-digital systems [7] to be flawless, effective, robust, and meet user-defined specifications [8]. Moreover, safety-critical CPSs in autonomous vehicles must achieve a high level of dependability (reliability, testing, and safety) through embedding traditional fault-tolerance techniques to be precisely architected [9]. Recent developments in autonomous vehicle safety research highlight the importance of standards such as ISO 26262 [10], which mandate systematic approaches to functional safety in automotive electronics. In parallel, intelligent fault detection, runtime monitoring, and AI-driven safety mechanisms are increasingly being adopted to enhance resilience against both transient and permanent faults. These trends underscore the need for architectures that combine formal verification methods with fault-tolerance techniques, ensuring that autonomous vehicles can operate dependably under diverse and unpredictable conditions. Figure 1 illustrates the two foundational concepts: formal verification methods and fault-tolerance techniques being used as a foundation for building highly dependable and safety-critical digital embedded devices that underpin the development of highly dependable and safety-critical digital embedded devices in autonomous vehicle safety architectures. Together, these approaches provide the structural basis for resilient systems capable of mitigating faults while maintaining compliance with stringent safety requirements.

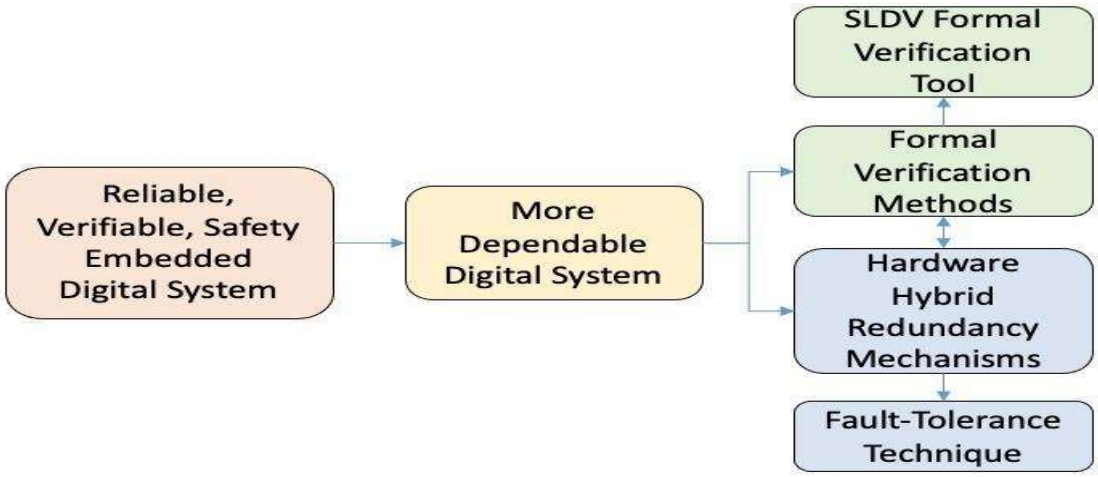


Figure 1. Traditional approaches in safety-critical and embedded CPS systems.

The architectural design of a safety-critical embedded system is typically prone to different types of defects and software errors that can attack its internal structure and defeat the embedded traditional simulation, testing, and fault mitigation techniques. Traditional simulation methods, such as software-based simulation and hardware-based testing, can detect several types of simple apparent errors [11]. The hardware-based testing method checks the behavior of all inputs of the digital device, while the software-based simulation method checks the behavior of the overall model of the design [12]. Traditional verification and validation (V&V) techniques are usually used to detect invisible software-based errors, such as division-by-zero errors, overflow flags, dead logic, and two-dimensional array-access violations. Hence, it is essential to define the set of requirements and specifications for developing a safety-critical model-based design during the early stage of the digital CPS design [13], which will achieve a quick mitigation against errors. The behavior of the digital system can be analyzed using a mathematical approach called formal verification and validation, which is performed formally and more precisely. In addition, it is observed that using formal verification tools enables the system to be examined in the whole space of possible simulations and validation during the CPS system design procedure [14].

2. Related Work

This section presents a discussion on the previous research methods in verifying digital cyber physical systems and the main contribution of the paper.

2.1. Overview of the Previous Work

The verification methods of digital cyber-CPS systems and many papers on the topical areas of verification, model-based digital design, and fault-tolerance that have been presented in the literature, as it is shown in Figure 2, are discussed in detail in this section. Askari Hemmat et al. (2015) [15] modeled and implemented a train-control CPS system as a complex system design to verify its correct functionality and behavior. They have used the NuSMV symbolic model checking tool, which was built based on binary decision diagrams, for detecting the predicted bugs during the implementation stage of the design. An advanced RISC Machine-based processor has been used in the realization of their proposed model. The authors in ref. [16] have used the state-flow design using UPPAL as a model checker event driven tool for verification, validation, and modeling of a real-time fuel digital control CPS system. As a comparison with the simulation results of Simulink-based stateflow that do not find the invalid properties, their approach demonstrated its effective design in generating counterexamples for the invalid properties and identifying the incompatibility between the automation functions. Although Kang et al. in ref. [17], the authors in ref. [18], and Wisniewski et al. in ref. [19] have dealt with verifying the correct operation of the cyber-physical systems (CPSs), each research group has its method of verification and validation. In ref. [17], the authors have used two formal verification tools, the Simulink design verifier and the SMC-UPPAAL model checker, to formally analyze the desired nonfunctional properties. The verification and validation process has been performed on the behavior of an automated traffic sign identification vehicle application. Additionally, the simulation results showed that the counterexamples generated by the tools have been used to analyze the invalid properties and refine the system model depending on the traces of the errors.

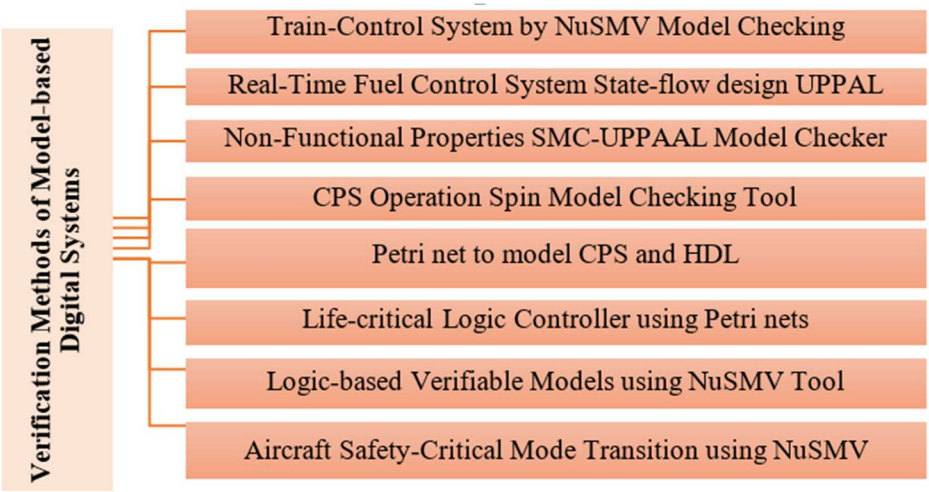


Figure 2. The different verification tools that have been used in the literature to architect model-based digital CPS system.

While the authors in ref. [18] built a validation system design and used Spin as a model checking tool to verify the architectural operation of a CPS, they attempted to model a timed automotive and hybrid traffic control system consisting of four sub-controllers. Furthermore, their proposed verification approach has been recommended to be used in the implementation of timing protocols for automated based systems. In ref. [20], the authors used the Petri net to model the operation of cyber physical systems (CPSs) and realize its hardware architecture using hardware description language (HDL) as a hardware modeling language.

In contrast to traditional model-based design, the proposed software-based approach was highly verifiable and demonstrated on a direct matrix-converter circuit through various phases of verification before the hardware implementation. In ref. [9], the authors simulated and verified the design of life-critical logic controllers that must be working to meet the predefined requirements and desired specifications. The Petri nets were utilized as an interpretant modeling tool in the design and verification of a smart-home application. Furthermore, the designed system was divided into submodules, and each submodule was verified and validated separately. The authors in Ref. [21] presented a new approach to combine hardware-based failures and communication delays in safety-critical instrumentation and control systems with logic-based verifiable models using the NuSMV tool. Finally, in Ref. [22], the authors employed a formal verification method to verify the model-based design for safety-critical Mode Transition Logic in aircraft using the NuSMV model checker and proved its complete correctness, which has been led to improving the system dependability.

2.2. The Main Contribution and the Outline

The scientific novelty of the proposed approach lies in integrating model-based formal verification and runtime fault tolerance within a unified safety-critical CPS architecture. In the proposed SBR system, SLDV is employed at the design stage to formally verify user-defined safety-critical properties, while a hybrid TMR mechanism is incorporated to detect and tolerate permanent and transient faults during system operation. Thus, rather than introducing SLDV or TMR as individual techniques, the proposed framework combines property-level formal verification with runtime fault-tolerance mechanisms within the same model-based SBR architecture.

Many contributions in the topics of formal verification and validation, and fault tolerance of model-based digital embedded CPS devices are presented in this paper, and they can be summarized as follows:

- Design, simulation, and verification of a safety-critical and CPS-based Seatbelt Reminder (SBR) architecture using formal verification and validation, and fault-tolerance methods.
- Experimental study on the use of the Simulink Design Verifier (SLDV) model checker to verify user defined properties of safety-critical CPS-based SBR architecture diagrams.
- Verifying safety-critical properties in the proposed SBR architecture and tolerating permanent and transient runtime faults using a hybrid fault-tolerant Triple Modular Redundancy (TMR) technique ensures high reliability in the final design.
- The developed Seatbelt Reminder (SBR) architecture was used to demonstrate that the Simulink design verifier can check the verification of user-defined safety-critical CPS properties. Whenever a falsified user-defined property is presented the SLDV can detect the erroneous property, trace its error, and eliminate it.

The remainder of the paper is organized into three sections. Section 3 presents the experimental results and methods used to design a highly reliable and verifiable digital SBR CPS system. Section 4 describes the architectural concept of the Seat Belt Reminder digital system and its specification using Simulink SLDV design tool. This section also discusses the model-based design, its simulation constraints, and the description of safety-critical properties proof for the developed SBR architecture. The paper concludes by summarizing the experimental simulation results and the ongoing future works for this research in Section 5.

3. Methods and Experimental Results

To increase the dependability of safety-critical digital CPS systems and make it highly reliable and verifiable, using traditional fault tolerance techniques [13] it is essential to be combined with formal verification techniques to make the controllable model-based system resilient and work under any potential fault [23,24]. This paper presents a new method for obtaining highly reliable digital CPS systems, given the growth of such systems and applications whose failure is critical and dangerous, whether to human lives

or to economic losses, as in medical, aerospace, transportation, industrial, and other applications. The proposed approach supports the formal verification, which was represented in SLDV [25], which applied to the seatbelt reminder (SBR) architecture as a case study. SLDV was selected for the proposed CPS architecture because it is directly integrated with the MATLAB/Simulink R2026b and Stateflow model-based design environment used to develop the SBR system. This integration enables the verification of user-defined safety properties on the same executable model without requiring the system to be reformulated in a separate modeling language. Compared with general-purpose model-checking alternatives such as NuSMV, UPPAAL, and SPIN, which have been successfully employed in previous CPS studies, SLDV is particularly suitable for the present framework because it supports property proving, counterexample generation, automatic test-case generation, design-error detection, and traceability within the Simulink verification workflow. Therefore, its selection facilitates a direct connection between model development, simulation, formal verification, and debugging of the proposed safety-critical CPS architecture. The Simulink Design Verifier toolbox supports a diverse set of analyses, such as generating text, proving properties, and detecting errors [26,27]. Generating the test cases automatically and proofing the critical properties are embedded in the SLDV tool. A fixed-step solver was used in the proposed work since there is not compatibility for the Simulink tool with all design models.

4. The Architecture of the Seat Belt Reminder Application Specified by SLDV Tool

The architectural model-based design for the seatbelt reminder CPS system developed in this work is presented in Figure 3. The architecture reads the status of a diverse set of sensing units such as the speed-sensing unit, rain-sensing unit, fog-sensing unit, temperature-sensing unit, and pressure sensing unit embedded in the developed system. All the sensors measure the physical measurements continuously in order to alert the driver whenever the seatbelt is unfastened. Developing a new verification approach based on the Simulink model for the safety-critical seatbelt reminder application is essential to analyze its robust behavior.

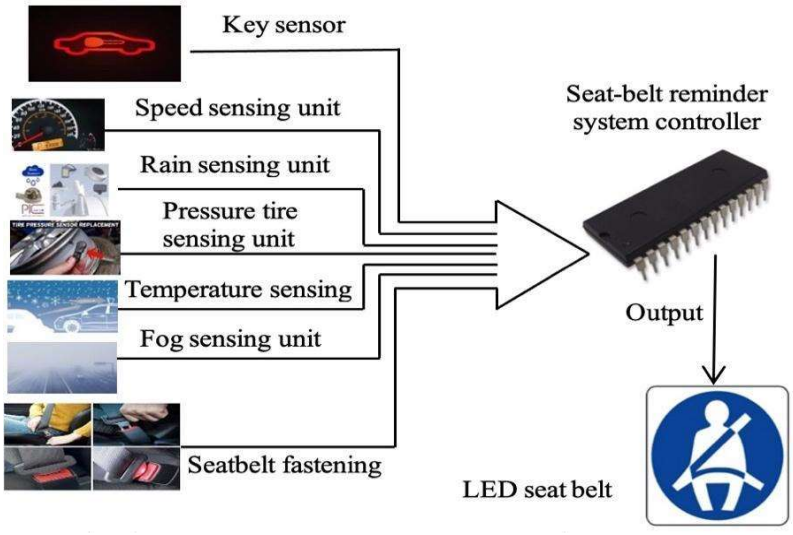
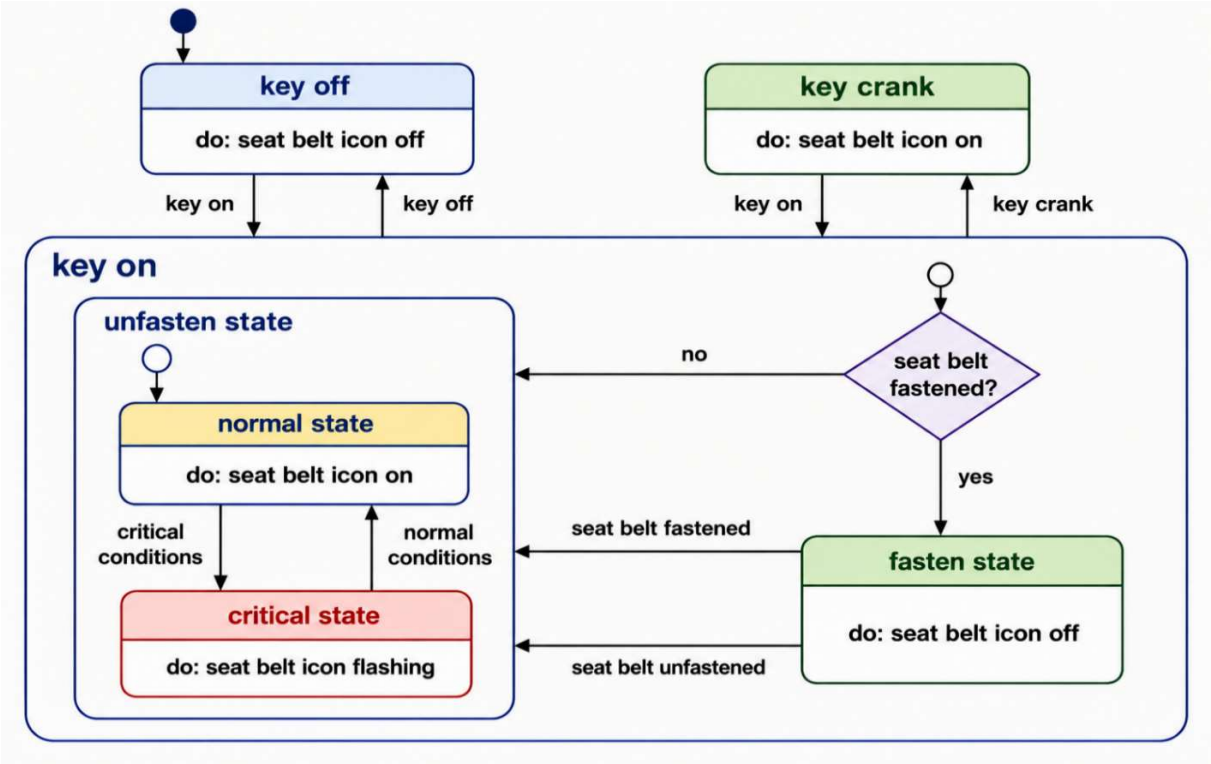


Figure 3. A schema of the developed seatbelt reminder CPS system based on a model-based architecture.

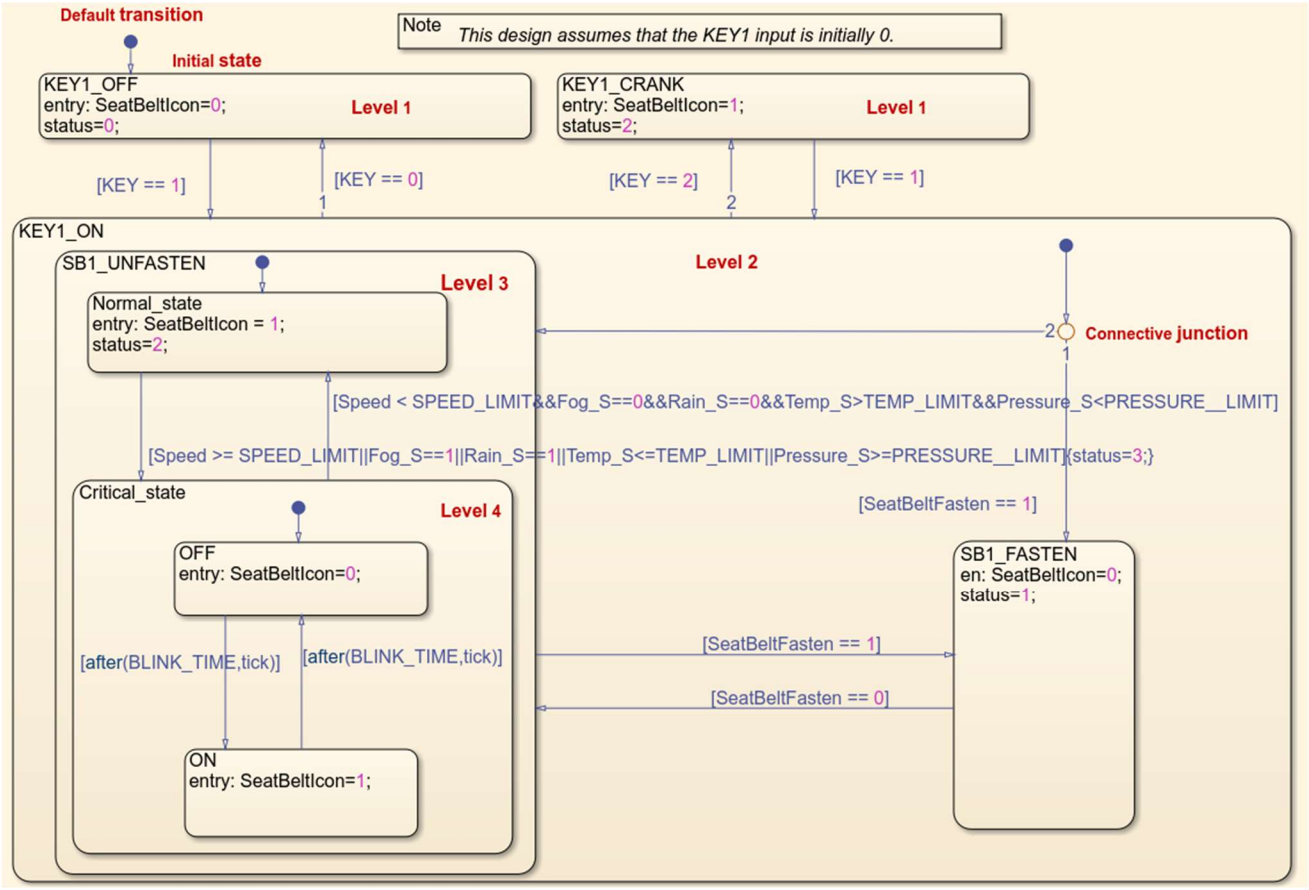
According to the literature, the safety-critical seatbelt controller usage has effectively reduced the death cases caused by the collision's occurrence in vehicles on highway roads. As an example, when the driver wears the seatbelt correctly, the front seat fatality rate can be reduced approximately by 60% [27]. In ref. [28], the authors designed a vehicle control system using NuSMV to formalize and check the safety-critical CTL and LTL properties. In addition, the high risk of a collision accident may increase significantly in case of traffic conditions such as rainy days, snow, and icy conditions, according to ref. [29]. Furthermore, ref.

[30] shows that the pressure of the vehicle tire is a significant component that impacts vehicle and human safety. As a result, a seat belt CPS system model design was created and utilized as a case study to demonstrate how the proposed approach fits the relevant criteria and can be evaluated. The development of the CPS system behavior model represents the first level of the system design, while the unified modeling language (UML) represents the graphical programming language that can be used for the description of the architectural behavior of the systems [31].

The developed block diagram for the state transition of the seatbelt reminder application as a case study is presented in Figure 4a. In addition, all the states of the proposed model-based system are expressed using the principle of hierarchical levels approach, as it is illustrated in Figure 4b. The MathWorks Simulink toolkit was used to model the proposed CPS system architecture and verify its safety-critical and functional properties [28]. A block diagram for our verification technique employing the SLDV formal verification tool is given in Figure 5. The verification model-based design VMBD process involved three models. When designing a verification model for the developed seat belt, the system model should be produced separately from the VMBD and referenced in the VMBD. The goal is that when the system attributes must be shown, the design model will not change. As a result, the VMBD incorporates a model block that references the seatbelt model-based design and verification model, which includes a constrained subsystem and a safety characteristics subsystem. To apply the model-based design (MBD) and simulate it, it is necessary to produce robust, fault tolerance-neutral systems and higher-quality designs. MBD uses simulation to analyze the behavior of a physical system being built or already in existence [32]. At each stage of the model-based design technique, including requirements analysis, design, implementation, and testing, a system model is at the heart of the entire development process.



(**a**)



(**b**)

Figure 4. (**a**) The developed state transition block diagram for the seatbelt reminder CPS application; (**b**) State level of the seatbelt reminder CPS model.

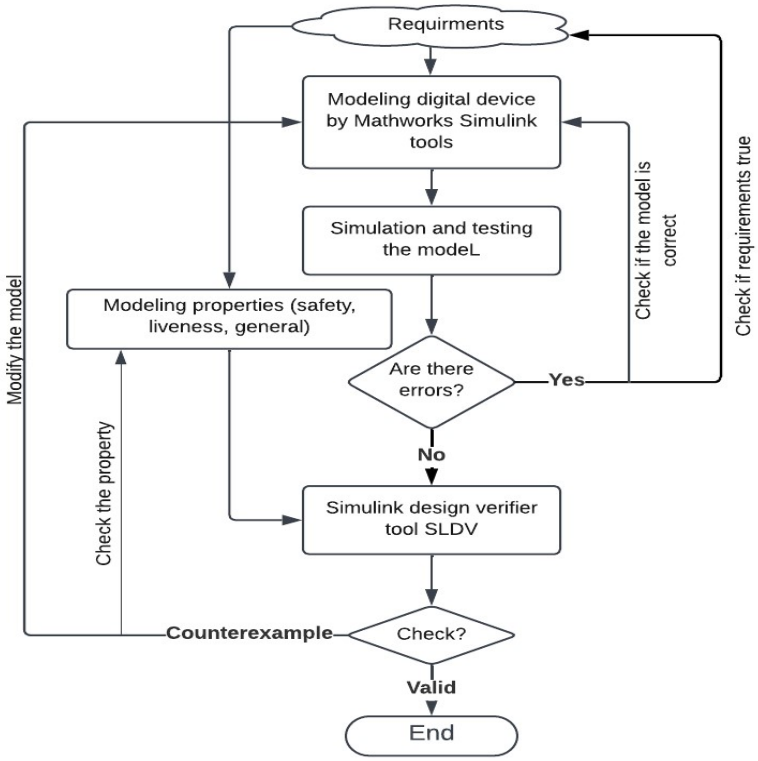


Figure 5. Algorithmic flowchart for the CPS verification approach using SLDV.

4.1. Model-Based Design

Model-based design typically decreases the development time and costs by detecting defects early on the design stages [33,34]. An abstract and condensed description of a CPS system is presented by a model, which uses executable graphics and mathematical equations. Modeling is the process of employing block diagrams to create a simulated virtual representation of a real-world environment. As result, modeling and simulation are helpful for testing in the early phases of development when hardware setup is not accessible. In the modeling stage, it depended on the graphical language, which is called the Stateflow toolbox, that contains the following blocks: state transition diagrams, flow charts, state transition tables, and truth tables. Figure 5 shows flow chart of the seatbelt system architecture, the model includes the following three states: off state, crank state, and super state, which contain the seat belt fastened state and seat belt unfastened super state. The input signals of this model are represented by the input bus external data, which contains the following elements: Key, SeatBeltFasten, Speed, Fog_S, Rain_S, Temp_S, and Pressure_S. As output signal, there are two output signals, the first is the SeatBeltIcon, which represent as a lamp output, and the second output is a status, which represents three statuses as shown in Figure 6. The triplicated modules process the same input data concurrently, and their outputs are evaluated by the majority voter on the corresponding clock boundaries. When a transient or permanent fault is detected in one of the active modules, the fault-detection mechanism identifies the faulty module without interrupting the operation of the remaining two healthy modules. The faulty module is then isolated from the voting path, while the spare module is initialized or loaded with the required configuration and synchronized with the current system state.

After completing the model base design stage for the seat belt system and simulating it to confirm that the model is free of errors, the verification model-based design (VMBD) stage was used. A formal model-based verification approach was proposed to check the correct operation of the seat belt system. In our approach a formal notation for specifying desired and undesired behaviors of the safety critical system was used. In this approach, it is very important to constrain the decision of the input signal; therefore, (VMBD) involves a constraint model. In this stage, the seat belt MBD was adopted as the reference model to verify that the system operation met the required safety properties using the Simulink Design Verifier toolbox; then, two verification subsystems that define the properties and any required constraints were designed. Figure 6 shows highlighted assertion debugging to trace violations of the safety property object using a harness model generated automatically by the SLDV verifier tool.

Cyber-physical systems (CPSs) embedded in safety-critical applications such as medical instrumentation and control (I&C) devices, avionics electronics, industrial automation, and transportation systems require high levels of dependability, electronic testing, and formal verification to cope with different classes of hardware faults and software flaws. Fault occurrences in these life-critical systems can change their correct functionality and cause catastrophic consequences in the environment. This paper presents a novel model-based CPS verification approach that uses formal verification and fault-tolerant techniques targeting the design and modeling of a safety-critical Seatbelt Reminder (SBR) architecture. The proposed fault-tolerant, verifiable architecture leverages the Simulink Design Verifier (SLDV) as a formal verification and model-checking tool to make the CPS system highly reliable and verifiable. A hybrid fault-tolerance TMR (triple module redundancy) technique was embedded in the model design to detect transient and permanent faults and mask their effects. The experimental simulation results prove that the analysis and preprocessing times for proofing the safety properties were 12 s and 10 s in duration. A runtime execution of the verifiable model-based approach achieved high levels of verification and dependability against various classes of safety-critical properties. The proposed approach was simulated and validated on a processor with a frequency of 1500 MHz and 16 GB of memory.

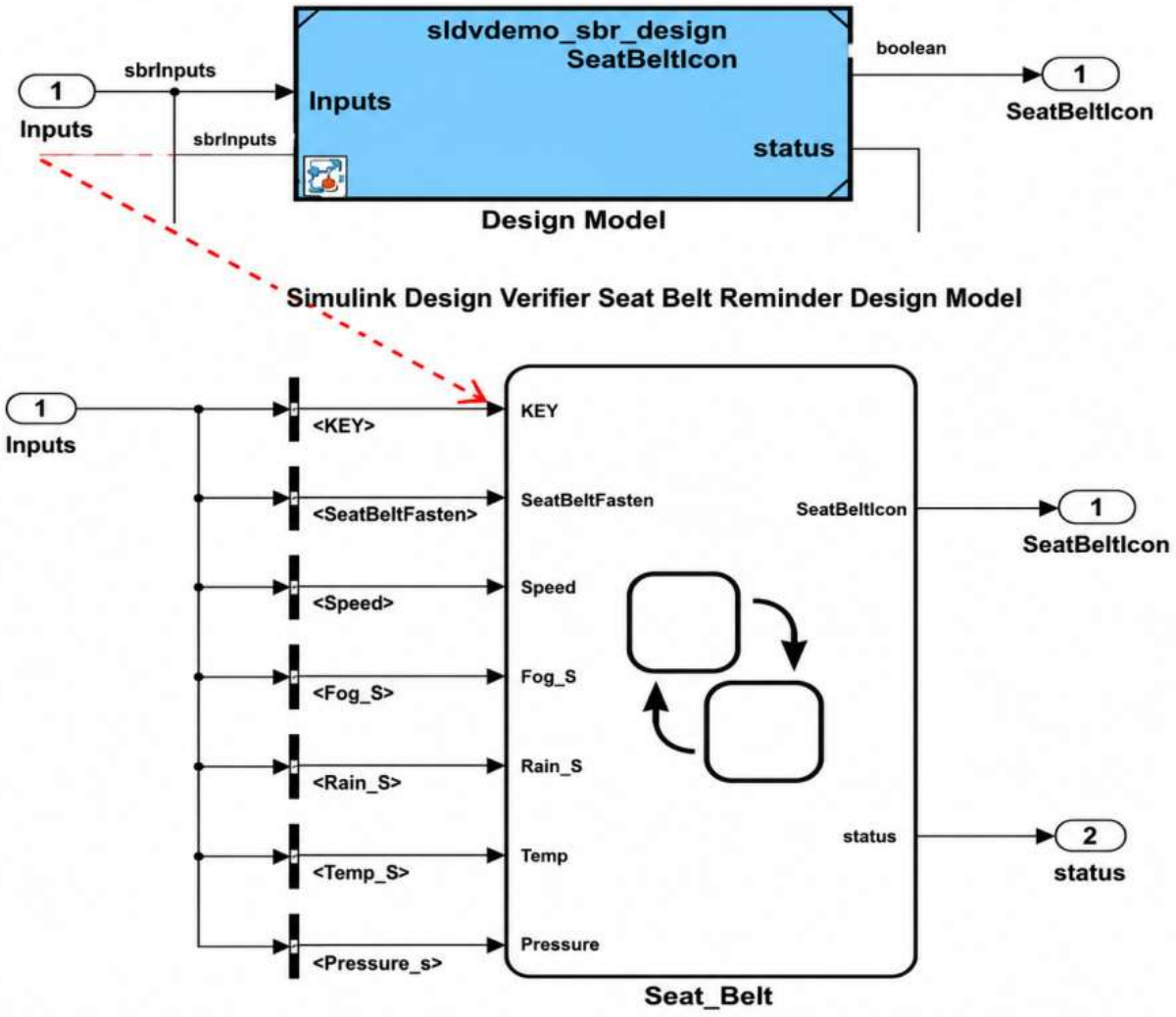


Figure 6. Block diagram for the CPS verification approach using SLDV.

To assess the quantitative improvement in dependability metrics that is resulting from the design of TMR model for the SBR-based safety-critical autonomous vehicle system, a simple markov chain model was constructed. The Markov model consists of one operational state defined as state_O, one failed state defined as state, and one failed unsafe state defined as state_FU, as they are described in Table 1 shown below. R(t) represents the reliability that can be calculated in a quantitative approach as shown in reliability Equation (1), in which the reliability of a three-modules system is mathematically calculated. Where λ represents the failure rate for the SBR system, and the factor T represents the time consumed during the SBR operation.

$$\mathrm{R(t)} = 3e^{-2\lambda T} - 2e^{-3\lambda T} \quad (1)$$

Table 1. Description of the Markov reliability model three states.

States Names	Description of State Status
State_O	All components/circuits are working correctly (SBR system is fully operational).
State_F	One of the triplicated SBR systems is affected by a transient fault. The failure effect is tolerated using a majority voting circuit (SBR system is failed-operational.
State_FU	Two SBR devices are shutdown and affected by permanent faults or three SBR systems are down and defected by faults (SBR system is failed).

Figure 7 Presents the three state that describe the behavior of the TMR-based SBR system in which three operational states are executed. As it is shown in Figure 8 and in Table 2, the level of the quantitative reliability decreases as the time in hours and the rate of failures increased. In addition, the break-even point or critical-intersection point (R-simples (t) = 49.659/100, R-TMR (t) = 49.488/100) is shown when comparing the simplex system to triple modular redundancy (TMR). The system breaks-even at 14 h because the mechanism of TMR relies on the presense of three modules working together and the system is considered operational when it has at least 2 out of 3 operational modules. The TMR system meets high reliability equal to 99.3096×10^{-2} with failure rate 0.005 failure per hour.

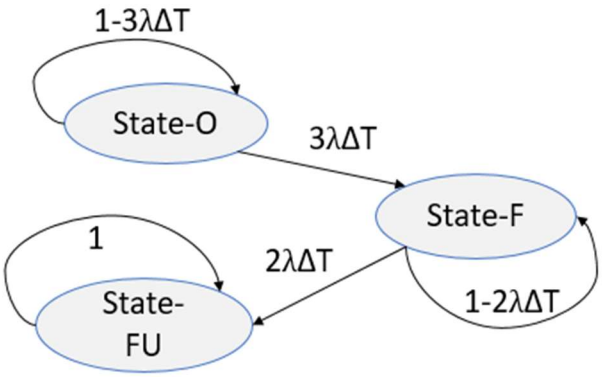


Figure 7. Analysis Model for the SBR-based TMR system.

Table 2. Reliability decreases as time increases with failure rates.

Time_in Hours (t)	$\lambda 1 = 0.05$ f/h	$\lambda 2 = 0.005$ f/h
10	0.65738	0.993096
11	0.61451	0.99171
12	0.57298	0.99022
13	0.533205	0.98862
14	0.49488	0.98691
15	0.458592	0.98509

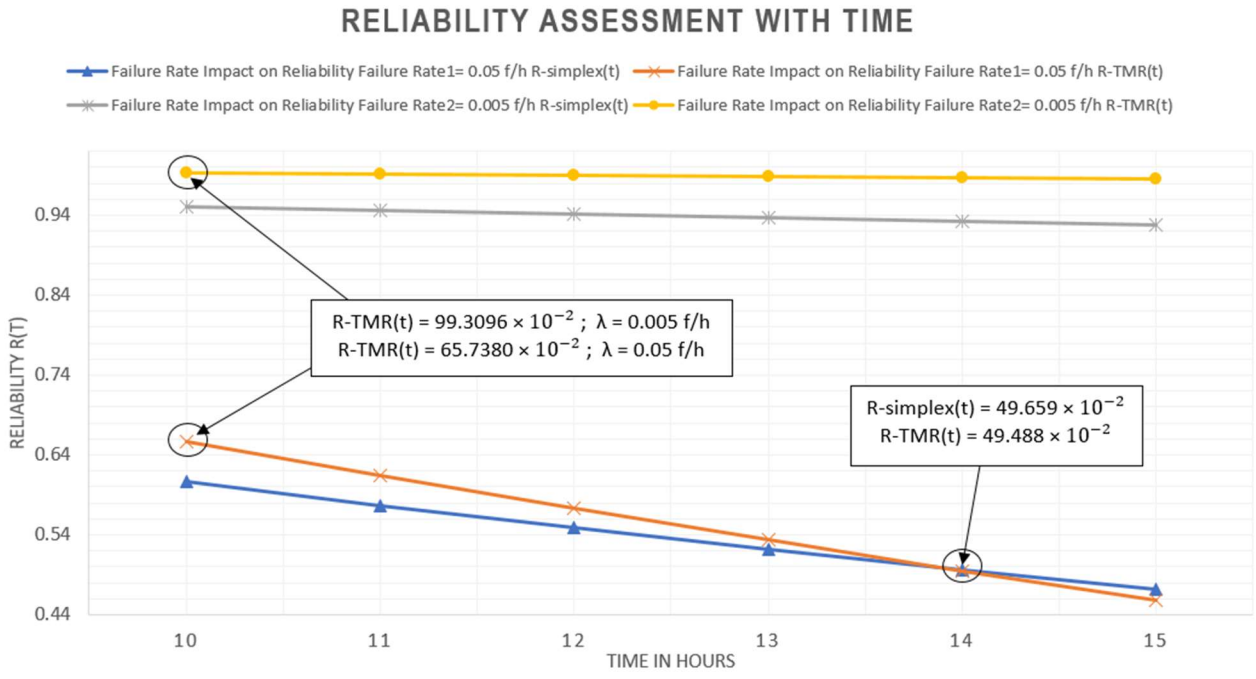


Figure 8. Reliability Assessment for the SBR-based TMR system.

4.2. Constraints of the Model-Based Sybsystem

Input signals for the safety-critical seatbelt architecture can be specified using constraints of the simulation model-based subsystem to specify its output decision [35]. The subsystem decision for the input signal is constrained by the smallest and greatest range value limits, which represent the simulated input block. To design the subsystem, two methods that contain the truth table blocks and Simulink blocks were used as building blocks. The truth table includes two tables (condition and action tables). After inserting the constraint conditions in the condition table, the proof assumption and test condition functions were used in the action table to constrain the decision of the input signal.

4.3. Safety Properties Subsystem

In this subsystem, the safety properties, which are required to prove them, were specified, and by this process, the system was proven to work safely under any circumstances. These safety properties were formulated in two ways: Simulink blocks and a truth table. The safety properties, which were inserted in the truth table and Simulink blocks, were illustrated in Table 3. Then the proof of properties was run in the Simulink Design Model Verification window to start the analysis. When the analysis is completed, the Simulink Design Verifier log window indicates that the analysis proved the validity of all safety properties. However, whenever a change in one of the properties is made, or the model does not meet the required properties, the analysis results will contain a falsified property that can be traced by the created detailed report, which involves pdf and HTML, a counterexample, and, among the results, the highlighted analysis results on the model, which indicate the location of the error as shown in Figure 9. Table 4 summarizes the verification results of the specified safety-critical properties and their corresponding counterexamples. In real-world driving environments, fog and rain sensors can generate fluctuating and noisy measurements that affect the correct operation of the SBR system. The proposed Triple Modular Redundancy (TMR) architecture enhances system resilience to these transient inconsistencies by processing sensor information through three redundant channels and applying majority voting to their outputs. When a transient error affects only one sensing module, the remaining sensing modules generate consistent outputs, enabling the majority circuit to mask the defective module. This type of error in the proposed work can be known by clicking the debug, as well as the Harness model, by which the counterexample can be simulated and tracking the simulation to know the reason for the falsified object whether it is a defect in the design model or is a formulation of the property. There are a diverse number of hidden defects could be identified with the Simulink design through the SLDV Simulink verifier.

Examples of such software errors and faults [30] include dead logic, out-of-bounds array access, division by zero, overflow, floating-point values, and data-storage violations. In our approach, the error detection that has been with respect to division by zero, overflow, and dead logic analysis for the specified block input range violations.

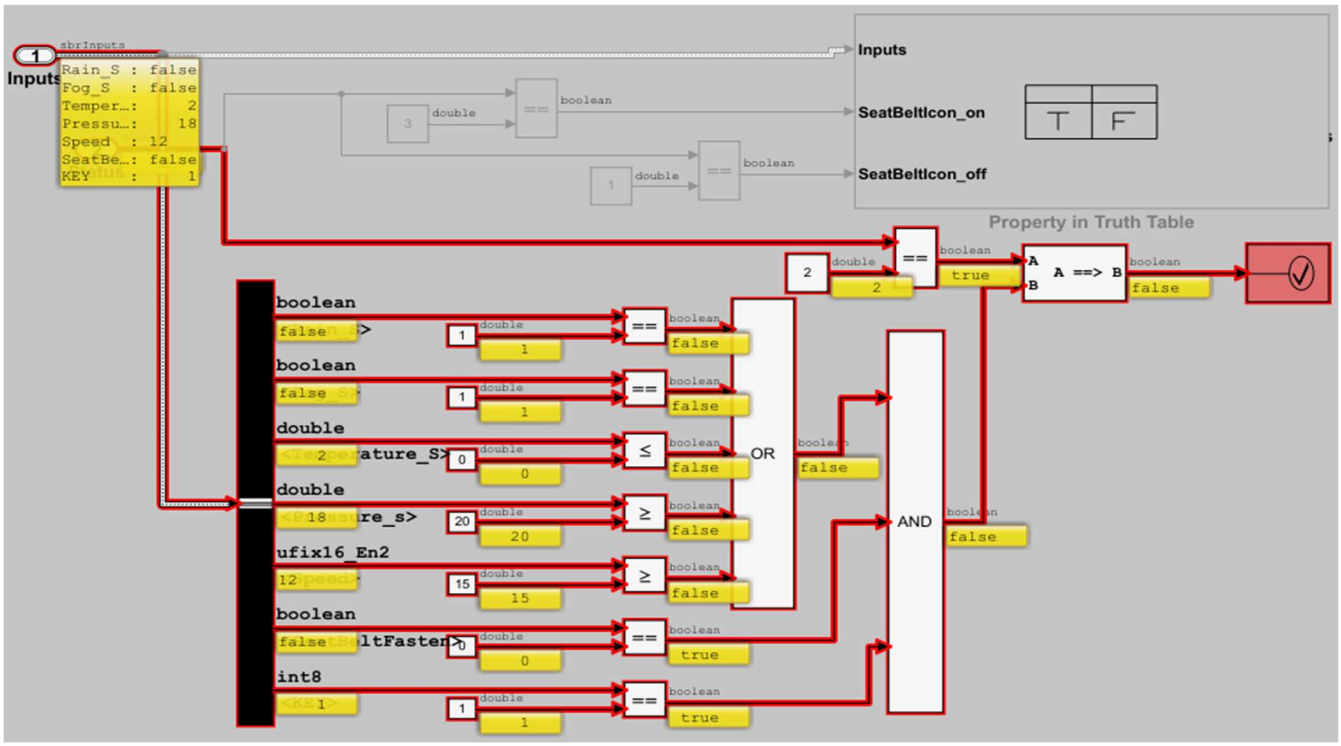


Figure 9. Block diagram for the SLDV verification approach.

Table 3. Safety-critical properties proofs.

Property	Property Method	Description of the Property Proof	Action
Property 1 (P1)	Truth Table	The light of the safety seatbelt reminder application turns off whenever the seat belt is fastened, and the vehicle is in either a critical or normal state.	sldv.prove(SeatBeltIcon_off)
Property 2 (P2)	Truth Table	The light of the safety seatbelt reminder application turns on whenever the seat belt is unfastened, the status of the safety-critical sensing units is within the safe, limited, and acceptable range, and the vehicle is in either a critical or normal state.	sldv.prove(SeatBeltIcon_on)
Property 3 (P3)	Simulink Block	The light of the safety seatbelt reminder application flashes whenever the seat belt is unfastened, the status of the safety-critical sensing units is outside the safe, limited, and acceptable range, and the vehicle is in a critical state.	Assert

Table 4. Summary of Safety-Critical Property Verification Results and Counterexample.

Property Type	Model-Based Design Item	Verification Action	Verification Result	Counterexample/Reason	Analysis Time
Property1(P1) ObjectiveProving	Safety Property/Property in Truth Table	sldv.prove (SeatBeltIcon_off)	Valid (Proved)	None	11 s
Property2 (P2) ObjectiveProving	Safety Property/Property in Truth Table	sldv.prove (SeatBeltIcon_on)	Valid (Proved)	None	11 s
Property3 (P3) Assertion	Safety Property/Assertion	Assert	Falsified	A counterexample was generated because the model did not satisfy the specified safety property under the analyzed input conditions (see Figure 9).	12 s

4.4. Triple Modular Redundant Fault-Tolerance Approach

Fault-tolerance is an essential concept that has been utilized in the design of dependable hardware and software embedded systems. In this proposed verification and fault-tolerant approach, three modules of the seatbelt model-based design were built and implemented as a Triple-Modular Redundancy (TMR) technique [36–38]. In ref. [39], a zonotopic-observer framework was introduced to estimate reliable state bounds and assess abnormal system behavior through a quantitative anomaly indicator. The simulation results demonstrated the effectiveness of the approach in capturing state uncertainty and identifying anomalous conditions. Unlike this estimation-oriented approach, the present study combines SLDV-based formal verification with TMR-based fault tolerance to support safety verification and fault handling within an integrated model-based framework. These modules execute similar functions, operate synchronously, and connect to a majority-voting circuit that can detect failures.

The TMR fault-tolerant technique was implemented in two levels of model hierarchy. The first level, which includes three submodules, and one spare module reads the data of the input bus and sends the commands to two outputs, which are the icon and the status. Figure 10 shows the state flow diagram for the majority voter circuit and the second level of the TMR circuit, which consists of four submodules. All the variables have been used to describe the inputs and outputs of the proposed TMR fault tolerant approach are defined in Table 5. Any fault occurrence in the system was masked, and the location of the failed module was identified by a majority voting circuit. As shown in Figure 10, the proposed voting circuit operates on the input data by using one of the three identical submodules, represented by the signal In2, as the reference module. To identify the failure, the outputs of the other two submodules represented by the signals S0 and S1 were used. If all TMR submodules fail or the reference submodule fails, the fault is quickly detected through the Fault detection and correction simulated logic circuit represented by the flowchart shown in Figure 11.

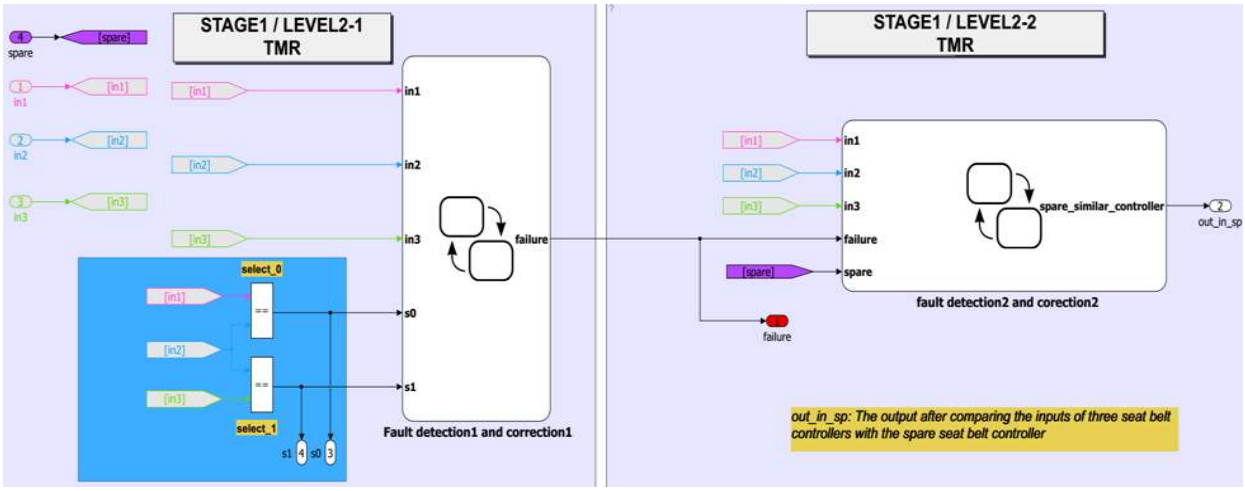


Figure 10. TMR and Voter of phase1/stage1/level2.

Table 5. Operation of level2 for the TMR fault-tolerance.

Signal	Brief Description
In1_signal	"Status-1": represents the output of submodule1
In2_signal	"Status-2": represents the output of submodule2
in3_signal	"Status-3": represents the output of submodule3
Spare_submodule	"Status_spare": represents the output of spare_module
S0_signal	The output of the select box, entered to the flow-chart
S1_signal	The output the select box, entered to the flow-chart
Failure	Output of the flow chart, which detects the 1st fault-detection.

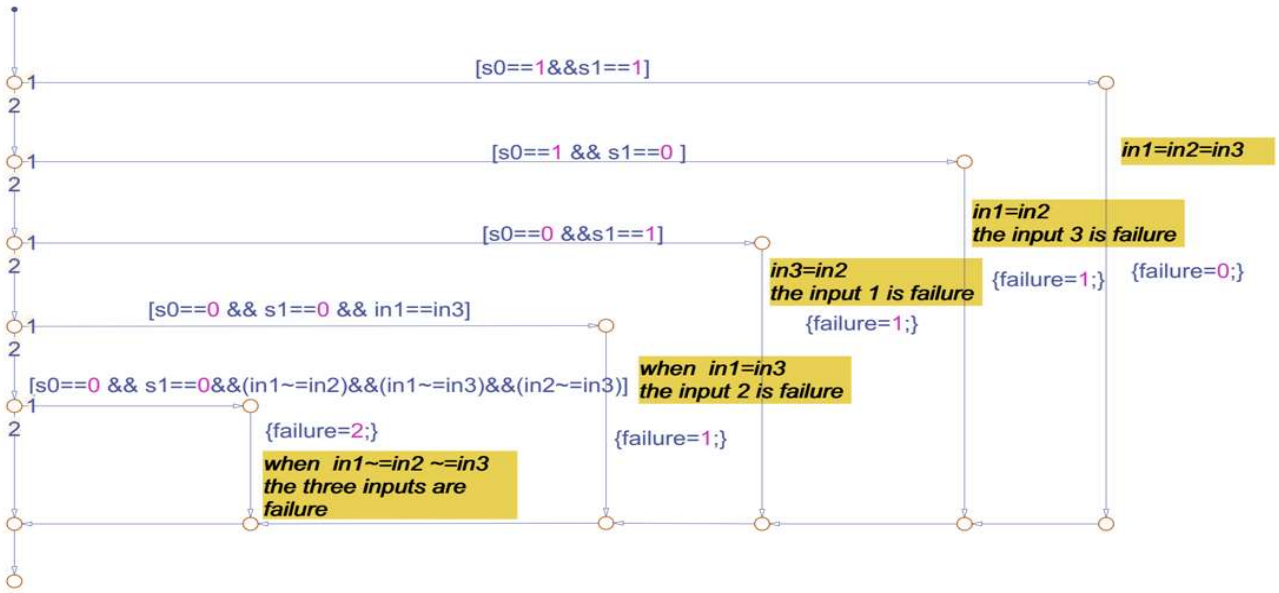


Figure 11. Fault detection and correction flowchart.

In addition, the three replicated modules operate synchronously and in parallel during system execution, allowing their outputs to be continuously evaluated by the TMR majority voting mechanism. The output of the TMR circuit is determined by the majority voter circuit to be one of the triplicated submodules outputs or transit into a warning state in which all the submodules are detected to be failed. When a discrepancy among the triplicated module outputs is detected, the majority voting circuit identifies the faulty module, while the Failure signal generated by the fault detection and correction logic circuit is used to activate the spare module. Consequently, the system detects the failed module, deactivates it, and replaces it with a spare module using duplication with a comparison fault-detection technique embedded in the majority voting circuit. This parallel operation and fault-triggered spare-module activation allow the TMR architecture to maintain system operation during the fault detection and module replacement process, as illustrated in Figure 11.

The two formal verification mechanisms, e.g., error detection in the digital design and test case generation for the Simulink SLDV design, were used to diagnose concealed design errors and faults in the proposed SBR system. The experimental simulation results showed that the SBR system implementation contains an overflow as a realization error. As a result, the data type of the control key variable was changed from integer to double, which overcame the generation of hidden errors. Figures 12 and 13 present the simulation execution results for detecting design errors in the proposed SBR model and its harness model timing diagrams.

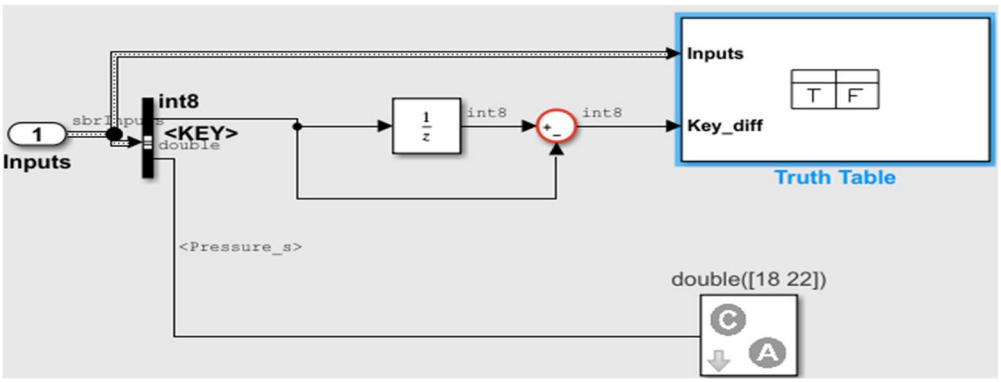


Figure 12. Detecting Design Errors Results for the SBR Model.

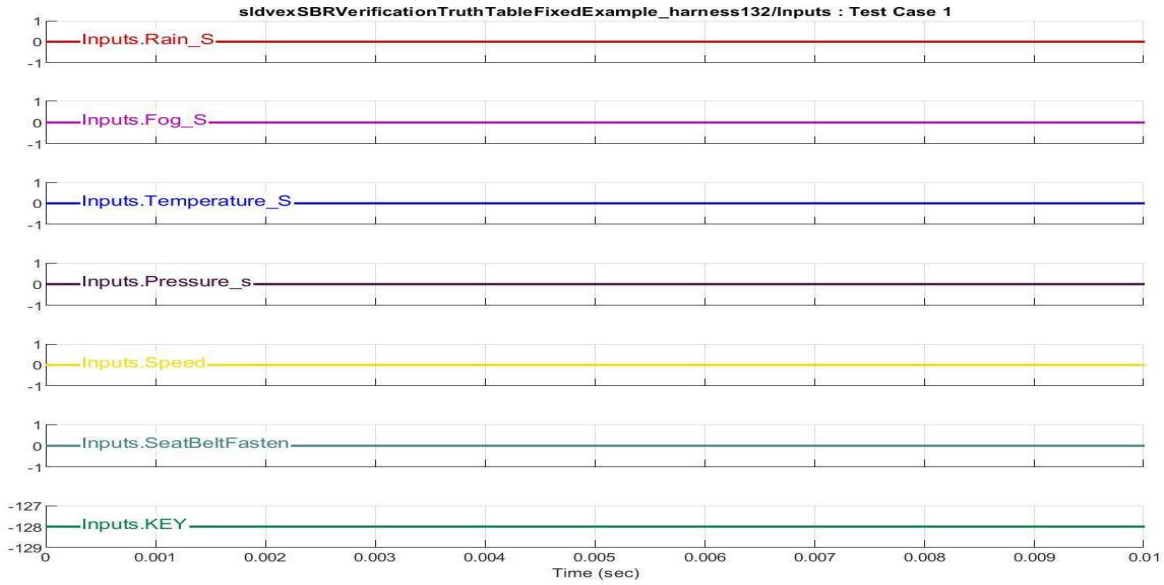


Figure 13. Harness Model for Design Error Detection Results for the SBR Model.

5. Conclusions and Future Work

This paper has presented a model-based approach for the design, verification, and simulation of fault-tolerant and safety-critical architectures in autonomous vehicle systems. The proposed framework integrates formal verification using Simulink Design Verifier (SLDV) with a hybrid Triple Modular Redundancy (TMR) mechanism to detect and mask transient and permanent faults, thereby enhancing the dependability of safety-critical subsystems such as the Seatbelt Reminder (SBR). The safety properties of the digitally embedded SBR model were formally proved against system specifications and requirements prior to hardware and software realization, ensuring compliance with stringent reliability standards. Experimental simulations demonstrated the effectiveness of the approach, with efficient preprocessing and analysis times and robust runtime dependability across diverse safety-critical scenarios. These results validate the potential of combining formal verification and fault-tolerance techniques to strengthen the reliability of autonomous vehicle safety architectures.

Future work will extend this framework toward runtime verification of hybrid fault-tolerant properties, as well as the integration of safety monitors and watchdog timers for data, address, and control buses. These mechanisms will be implemented using hardware description languages and deployed on FPGA-based platforms, further advancing resilience and compliance with emerging standards such as ISO 26262. Ultimately, this research contributes to the development of autonomous vehicle systems that are not only intelligent but also demonstrably safe, reliable, and fault-resilient.

The present SBR case study, although suitable for clearly demonstrating the proposed verification and fault-tolerance workflow, does not represent the full functional complexity of perception, control, and decision-making architectures in autonomous vehicles. Therefore, the results of this study demonstrate the applicability of the proposed SLDV–TMR framework to the considered SBR system under the defined assumptions and specified safety properties and should not be interpreted as full compliance with autonomous-vehicle safety requirements. Future work will extend the proposed framework to more representative autonomous-driving systems and target embedded hardware platforms. In addition, the developed fault-tolerance architecture incorporates an automated fault-management mechanism capable of fault detection, classification as transient or permanent, isolation of the faulty module, and automatic replacement with an available spare module when required. Because of the complexity and multi-stage

nature of this architecture, a detailed implementation and comprehensive evaluation require a dedicated study and are therefore reserved for future work.

Author Contributions

All authors contributed to the design, modelling, and simulation of the proposed model-based verification approach. The developed design seat belt reminder system, data collection, and analysis of the experimental results were performed by F.K. The first draft of the manuscript was written by S.S.K., and all authors commented on previous versions of the manuscript. All authors have read and approved the final version of the manuscript.

Ethics Statement

Not applicable.

Informed Consent Statement

Not applicable.

Data Availability Statement

The model-based verification design that was used in this paper is available from the authors upon request.

Funding

The authors confirm that this research work has not received any funding or grants.

Declaration of Competing Interest

The authors declare no competing interests or financial or non-financial conflicts of interest to disclose.

References

1. Khairullah SS, Qassabbashi FN, Kareem JA. Design and analysis of fault-tolerant sequential logic circuits for safety-critical applications. *Bull. Electr. Eng. Inform.* **2024**, *13*, 413–421. DOI:10.11591/eei.v13i1.5713
2. Khairullah SS, Mostfa AA. Reliability and Safety Modeling of a Digital Feed-Water Control System. *J. Univ. Babylon Pure Appl. Sci.* **2020**, *28*, 284–304. Available online: https://www.journalofbabylon.com/index.php/JUBPAS/article/view/3001 (accessed on 8 January 2024).
3. Khairullah SS. Formal Verification of a Dependable State Machine-Based Hardware Architecture for Safety-Critical Cyber-Physical Systems: Analysis, Design, and Implementation. *J. Electron. Test.* **2024**, *40*, 509–523. DOI:10.1007/s10836-024-06126-6
4. Khairullah SS, Elks CR. A Bio-Inspired, Self-Healing, Resilient Architecture for Digital Instrumentation and Control Systems and Embedded Devices. *Nucl. Technol.* **2018**, *202*, 141–152. DOI:10.1080/00295450.2018.1450014
5. Kumar V, Singh LK, Singh P, Singh KV, Maurya AK, Tripathi AK. Parameter Estimation for Quantitative Dependability Analysis of Safety-Critical and Control Systems of NPP. *IEEE Trans. Nucl. Sci.* **2018**, *65*, 1080–1090. DOI:10.1109/TNS.2018.2827106
6. Wojszczyk R. Verification of Accuracy and Cost of Use Methods of Quality Assessment of Implementation of Design Patterns. *Appl. Comput. Sci.* **2019**, *15*, 5–17. DOI:10.35784/acs-2019-01
7. Kaliappan PS, Kaliappan VK. Deriving the behavioral properties from UML designs as LTL for model checking. In Proceedings of the 2015 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES), Kozhikode, India, 19–21 February 2015; pp. 1–5.
8. Kumar P, Singh LK, Kumar C. Model Based Verification of Safety-Critical Systems. In Proceedings of the 2021 2nd International Conference for Emerging Technology (INCET), Belagavi, India, 21–23 May 2021; pp. 1–9.

9. Grobelna I, Wisniewski R, Grobelny M, Wisniewska M. Design and Verification of Real-Life Processes with Application of Petri Nets. *IEEE Trans. Syst. Man Cybern. Syst.* **2017**, *47*, 2856–2869. DOI:10.1109/TSMC.2016.2531673
10. Soleimani M, Sari AA. A unified safety framework for automated vehicle development: integrating ISO 26262, SOTIF, and UL 4600. *Transp. Res. Interdiscip. Perspect.* **2026**, *36*, 101831. DOI:10.1016/j.trip.2025.101831
11. Khairullah SS, Elks CR. Self-repairing hardware architecture for safety-critical cyber-physical-systems. *IET Cyber-Phys. Syst. Theory Appl.* **2020**, *5*, 92–99. DOI:10.1049/iet-cps.2019.0022
12. Ali AT, Alneema DAF. Design Analysis of Turbo Decoder Based on One MAP Decoder Using High Level Synthesis Tool. *Al-Rafidain Eng. J.* **2020**, *25*, 70–77. DOI:10.33899/rengj.2020.126801.1022
13. Kassab Bashi FN, Khairullah SS, Elks CR. Realization of Dependable Digital Systems for Safety-Critical Computer Systems using FPGAs. *IOP Conf. Ser. Mater. Sci. Eng.* **2021**, *1152*, 012020. DOI:10.1088/1757-899X/1152/1/012020
14. Khlaif FH, Khairullah SS. A Survey on Formal Verification Approaches for Dependable Systems. *Int. J. Adv. Comput. Electron. Eng.* **2022**, *7*, 13. DOI:10.48550/arXiv.2204.12913
15. Askari Hemmat MH, Ait Mohamed O, Boukadoum M. Formal modeling, verification and implementation of a train control system. In Proceedings of the 2015 27th International Conference on Microelectronics (ICM), Casablanca, Morocco, 20–23 December 2015; pp. 134–137.
16. Kang EY, Liu K, Hua MZ, Wang YX. Verifying Automotive Systems in EAST-ADL/Stateflow Using UPPAAL. In Proceedings of the 2015 Asia-Pacific Software Engineering Conference (APSEC), New Delhi, India, 1–4 December 2015; pp. 143–150.
17. Kang EY, Mu D, Huang L, Lan Q. Verification and Validation of a Cyber-Physical System in the Automotive Domain. In Proceedings of the 2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C), Prague, Czech Republic, 25–29 July 2017; pp. 326–333.
18. Kumar NS, Kumar GS. Modeling and verification of timed automaton based hybrid systems using spin model checker. In Proceedings of the 2016 IEEE Annual India Conference (INDICON), Bangalore, India, 16–18 December 2016; pp. 1–8.
19. Wisniewski R, Bazydło G, Szcześniak P, Grobelna I, Wojnakowski M. Design and Verification of Cyber-Physical Systems Specified by Petri Nets—A Case Study of a Direct Matrix Converter. *Mathematics* **2019**, *7*, 812. DOI:10.3390/math7090812
20. Pakonen A, Buzhinsky I. Verification of fault-tolerant safety I&C systems using model checking. In Proceedings of the 2019 IEEE International Conference on Industrial Technology (ICIT), Melbourne, Australia, 13–15 February 2019; pp. 969–974.
21. Shreya V, Nanda M. Analysing MTL properties using NuSMV model checker. In Proceedings of the 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, India, 20–21 May 2016; pp. 817–820.
22. Saleh HA, Salim TM. Design and Implementation of Model Predictive Controller. *Al-Rafidain Eng. J.* **2022**, *27*, 219–230. DOI:10.33899/rengj.2022.130477.1108
23. Khairullah SS, Sharkawy AN. Design and Implementation of a Reliable and Secure Controller for Smart Home Applications Based on PLC. *J. Robot. Control* **2022**, *3*, 614–621. DOI:10.18196/jrc.v3i5.15972
24. The MathWorks, Inc. *Simulink Design Verifier User's Guide*; The MathWorks, Inc: Natick, MA, USA, 2026.
25. Delebarre V, Etienne JF. Proving Global Properties with the Aid of the SIMULINK Design Verifier Proof Tool. In *Formal Methods*; Boulanger JL, Ed.; Wiley: Hoboken, NJ, USA, 2013; pp. 183–223.
26. Hamon G. Simulink Design Verifier: Applying Automated Formal Methods to Simulink and Stateflow. In Proceedings of the Third Workshop on Automated Formal Methods (AFM 2008), Princeton, NJ, USA, 14 July 2008.
27. Rezapour M, Ksaiabti K. Factors Impacting the Choice of Seatbelt Use, Accounting for Complexity of Travelers' Behaviors. *Future Transp.* **2022**, *2*, 237–248. DOI:10.3390/futuretransp2010012
28. Fritzsch J, Schmid T, Wagner S. Experiences from Large-Scale Model Checking: Verifying a Vehicle Control System with NuSMV. In Proceedings of the 2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST), Porto de Galinhas, Brazil, 12–16 April 2021; pp. 372–382.
29. Yang Y, Ke W, Yang J, Li X. Integrating UML With Service Refinement for Requirements Modeling and Analysis. *IEEE Access* **2019**, *7*, 11599–11612. DOI:10.1109/ACCESS.2019.2892082
30. Afzaal U, Lee JA. A Self-Checking TMR Voter for Increased Reliability Consensus Voting in FPGAs. *IEEE Trans. Nucl. Sci.* **2018**, *65*, 1133–1139. DOI:10.1109/TNS.2018.2824821
31. MATLAB and Simulink In-Depth: Model-based Design with Simulink and Stateflow, User Interface, Scripting, Simulation, Visualization and Debugging—MATLAB & Simulink Books. Available online: https://www.mathworks.com/academia/books/matlab-and-simulink-in-depth-patankar.html (accessed on 8 January 2024).

32. Conrad M, Mosterman PJ. Model-Based Design Using Simulink—Modeling, Code Generation, Verification, and Validation. In *Formal Methods*; Boulanger JL, Ed.; Wiley: Hoboken, NJ, USA, 2013; pp. 159–181.
33. Gautham S, Jayakumar A, Hite R, Deloglos C, Tantawy A, Gibson M, Rajagopala A, Elks C. Model-Based Design Assurance and Verification in the Context of IEC-61508 SIL-4 Standard. In Proceedings of the 12th Nuclear Plant Instrumentation, Control and Human-Machine Interface Technologies (NPIC&HMIT 2021), Virtual Meeting, 14–17 June 2021; pp. 824–836.
34. *IEEE Std 1850-2010*; IEEE Standard for Property Specification Language (PSL). IEEE: Piscataway, NJ, USA, 2010; pp. 1–182.
35. Madhana Mohan K, Balavikash K, Harish S, Jayabalaji M, Vignesh B. Accident Prevention by Monitoring and Control of Vehicle Tyre Pressure using Wear & Tear and Pressure Sensor. *Int. J. Recent Technol. Eng.* **2019**, *8*, 5281–5284. DOI:10.35940/ijrte.B1061.078219
36. Appels K, Prinzie J. Novel Full TMR Placement Techniques for High-Speed Radiation Tolerant Digital Integrated Circuits. *Electronics* **2020**, *9*, 1936. DOI:10.3390/electronics9111936
37. Khairullah SS, Bakker T, Elks CR. Toward Biologically Inspired Self Healing Digital Embedded Devices: Bio-Symple. In Proceedings of the 10th International Topical Meeting on Nuclear Plant Instrumentation, Control and Human-Machine Interface Technologies (NPIC&HMIT 2017), San Francisco, CA, USA, 11–15 June 2017.
38. Mahmood HA, Khairullah SS. Design and Simulation of a Dependable Architecture Using Triple Modular Redundancy for Embedded Cyber-Physical Systems. *J. Electron. Test.* **2025**, *41*, 63–74. DOI:10.1007/s10836-025-06159-5
39. Wang X, Ning Z, Dong H, Chen W, Wang Y. Zonotopic-Observer Based Deterministic State Estimation and Risk Evaluation for Unmanned Surface Vehicles. *IEEE Trans. Veh. Technol.* **2026**, *75*, 9535–9544. DOI:10.1109/TVT.2025.3649417