Multi-Domain Fault Models Covering the Analog Side of a Smart or Cyber-Physical System

Francesco Tosoni, *Student, IEEE,* Nicola Dall'Ora, *Member, IEEE,* Enrico Fraccaroli, *Member, IEEE,* Sara Vinco, *Senior Member, IEEE,* and Franco Fummi, *Member, IEEE*

Abstract—Over the last decade, the industrial world has been involved in a massive revolution guided by the adoption of digital technologies. In this context, complex systems like cyber-physical systems play a fundamental role since they were designed and realized by composing heterogeneous components. The combined simulation of the behavioral models of these components allows to reproduce the nominal behavior of the real system. Similarly, a smart system is a device that integrates heterogeneous components but in a miniaturized form factor. The development of smart or cyber-physical systems, in combination with faulty behaviors modeled for the different physical domains composing the system, enables to support advanced functional safety assessment at the system level. A methodology to create and inject multi-domain fault models in the analog side of these systems has been proposed by exploiting the physical analogy between the electrical and mechanical domains to infer a new mechanical fault taxonomy. Thus, standard electrical fault models are injected into the electrical part, while the derived mechanical fault models are injected directly into the mechanical part. The entire flow has been applied to two case studies: a direct current motor connected with a gear train, and a three-axis accelerometer.

Index Terms—Cyber-physical systems, smart systems, fault taxonomy, fault injection, electrical faults, mechanical faults.

1 Introduction

Nowadays, in every industrial field, the design of complex systems is evolving quickly, as an effect of the adoption of digital technologies, in the form of the current trend named *Industry 4.0*, i.e., the fourth industrial revolution. The new digital technologies allow indeed to closely monitor and mimic the evolution of a Cyber-Physical System (CPS), thus gaining more knowledge of the operating conditions, allowing to foresee future evolution and to observe complex interdependencies [1]. The modeling complexity is, on the other hand, rewarded by the possibilities that it opens, in that it allows to mirror the actual operation of the analog side of a CPS or a smart system. A smart system is designed to integrate heterogeneous components, such as digital, analog, communication modules, Micro Electro Mechanical Systems (MEMS) in one single, miniaturized device [2], [3]. Furthermore, the modeling of such systems allows to perform what-if analysis, estimate the impact of faults on the system or individual components, and better expose complex interdependencies between such heterogeneous aspects in the generation and propagation of faulty behaviors [4]. This procedure, called *fault injection*, supports the creation of reliability mechanisms in the design phases of a smart system or the analog side of a CPS, which underlines weaknesses that could affect the system's safety [5]. However, applying fault injection to the context of CPSs (that are natively multi-domain systems) differs depending on the part of the system and the domain under analysis [6]. Overall, *fault injection* is the state of the practice for ensuring functional safety of both digital and analog circuits, as quoted by the ISO 26262 standard for the functional safety of road vehicles [7]. In the digital domain, the state-of-the-art fault models are stuck-at-0/1 faults [8]. For the analog domain, the first attempt to standardize the analog faults is made by the IEEE P2427 workgroup; the standard is currently under revision [9]. At the state-of-the-art, such techniques do not apply to other domains than the electrical one, for instance, in an electromechanical

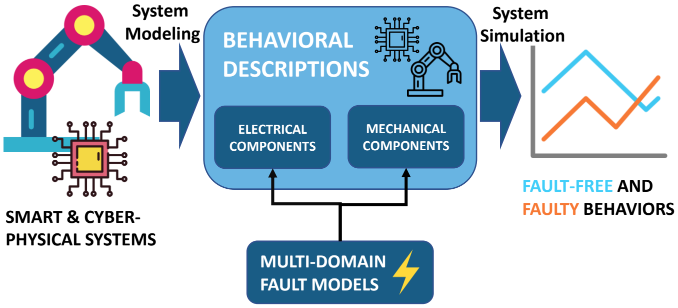


Figure 1. Overview of the proposed methodology to model and inject multi-domain faults into the analog part of smart systems or CPSs described at the behavioral level through differential equations.

- F. Tosoni and N. Dall'Ora and E. Fraccaroli and F. Fummi are with the Department of Engineering for Innovation Medicine, University of Verona, Verona, IT. E-mail: name.surname@univr.it
- S. Vinco is with the Department of Control and Computer Engineering, Politecnico di Torino, Torino, IT. Email: sara.vinco@polito.it

Manuscript received TBD; revised TBD.
This study was carried out within the PNRR research activities of the consortium iNEST (Interconnected North-Est Innovation Ecosystem) funded by the European Union Next-GenerationEU (Piano Nazionale di Ripresa e Resilienza (PNRR) – Missione 4 Componente 2, Investimento 1.5 – D.D. 1058 23/06/2022, ECS_00000043). This manuscript reflects only the Authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them.
Furthermore, this study has received funding from the European Union's Horizon Europe research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 101109243.

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2023.3345135

system, where fault models injected into the electrical part will deeply differ from purely mechanical ones, and this applies even more to other physical domains (e.g., thermal, hydraulic). In the mechanical domain, the usage of fault injection techniques is not as widely adopted as in the electrical one. Thus categorizing mechanical components and machinery working conditions is complex because of a paradigm shift that is too recent to be supported by an established and standardized procedure. Given the impact that mechanical fault injection could have on the evaluation of safety mechanisms, exteding the fault injection techniques to the mechanical domain looks an attractive idea. A key resource comes from the observation that *analogies* exist between the different physical domains. The idea is that different domain-specific behaviors can be mapped onto the same differential equations by interpreting the quantities involved with different domain-specific meanings. The following sections will show how such *analogies* between mechanical and electrical systems can be exploited to perform safety analysis in the mechanical domain by using a new non-electrical fault taxonomy. Moreover, the purpose of the methodology presented in this paper is to model faulty behaviors of a generic system utilizing such analogies, regardless of the underlying physical domain. In this way, it would be possible to apply advanced fault injection techniques, well established in the electrical domain, to other domains, e.g., the mechanical one. The goal, summarized in Figure 1, is to model a smart system (e.g., a car Electronic Control Unit (ECU)) or the analog side of a CPS (e.g., a wind turbine [10]) with the adoption of different types of faults spanning across different domains, to alter its functionality in many ways, and to deeply explore the impact of faults on the different levels of a system. To summarize, the main innovations proposed in this article include:

1) Create a new mechanical fault taxonomy by identifying mechanical fault models and injecting them directly into the mechanical system;
2) Realize a new automatic tool for the automatic fault injection in multi-domain models (electrical and mechanical), where the standard electrical faults are injected in the electrical part, and the proposed mechanical faults are injected in the mechanical part.
3) Verify the behaviors of the proposed system under different faults and operating conditions; this allows to perform functional qualification of the verification testbench.

The structure of this article is the following. Section 2 describes the necessary background. Section 3 shows how to exploit analogies to model mechanical systems and faults. In Section 4, mechanical faults are derived from the electrical domain to build a mechanical fault taxonomy, and Section 5 presents injection techniques for the defined faults. The proposed techniques are applied to complex case studies in Section 6. Finally, conclusions are given in Section 7.

2 Background and state-of-the-art

This section introduces the state of the art of behavioral fault modeling in smart systems and CPSs through physical analogies. Then, it covers the starting point of this paper, summarizing the previous works in this context.

2.1 Behavioral fault modeling

Modeling and simulation of a CPS are key steps for ensuring the functional safety of such systems. At the state-of-the-art, there are two ways to represent and simulate physical systems: graphical tools, e.g., Modelica-based tools and Simulink/Simscape, and adopting description languages, like Verilog-AMS, VHDL-AMS, and SystemC-AMS. With the former solution, the system is built by connecting pre-defined blocks belonging to domain-specific libraries. Those pre-defined blocks hide their internal dynamic, making the internal equations visible only to the solver, which computes the system evolution over time. However, the designer can customize the blocks since they are parametrized. So, the fault injection process is limited in such blocks-only environments [11]. Few works focus on fault modeling in the literature. Several working groups deal with fault detection or diagnosis [12], [13], but fault modeling is still a poorly explored topic. The latter solution requires modeling system evolution explicitly as differential and algebraic equations enriched with the application of energy conservation laws. Although the effort during the model realization is higher, the designer can choose the accuracy of the model description. Thus, injecting faults and faulty behaviors is feasible by manipulating the equations.

A fault represents a wrong response in system behavior because of multiple scenarios, i.e., material aging, strain or breaking of an internal component, a production defect, or a digital failure [14]. Functional safety focuses on ensuring the correct performance of systems in the presence of faults through several procedures. *Fault injection* is one of these procedures, which is formally described for the electrical domain in the ISO 26262 standard [7]. Furthermore, a transistor-level circuit can be faulty by injecting specific fault models and using different fault injection techniques, which describe how and where faults are injected [9]. Instead, fault models and injection techniques are not as advanced in the mechanical domain. Some fault taxonomies exist for mechanical systems [15], [16], but they focus only on the geometry of the system or other physical properties. In literature, the number of works that analyze the multi-domain faults in the mechanical domain is limited. Therefore, classifying fault models suitable for each mechanical model is complex due to the high heterogeneity of the existing mechanical systems, including MEMS. Moreover, all the real-world physical conditions that might vary the behavior of a system with a fault cannot be reproduced in a simulated environment. This article proposes a way to overcome these research gaps.

2.2 Modeling physical systems through analogies

In industry and research fields, the electrical equivalent circuits have been adopted increasingly to model complex systems over the years. Such equivalent models helped to bridge the gap between the electrical and mechanical disciplines, allowing engineers to leverage mathematical tools and concepts developed for electrical circuits to the mechanical domain. These complex systems are built based on analogies that link physical quantities of a specific domain to quantities typical of another domain and exploit the concepts of conservation of energy laws [17] applicable

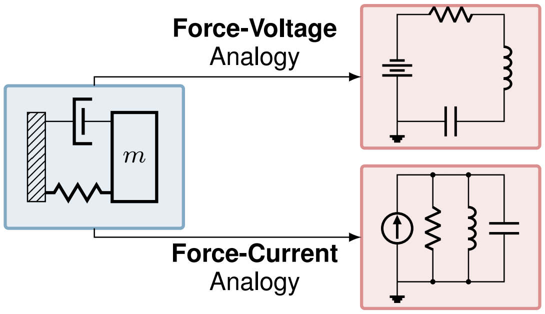


Figure 2. Mapping of a mechanical system to an electrical one by exploiting the physical analogies between mechanical (left) and the electrical equivalent systems (right) according to the force-voltage (top) and the force-current (bottom) analogies.

to different domains like thermal, magnetic, and hydraulic, especially with the electrical one [18]. As an example, electrical circuits are used to model characteristic parameters variability in energy components [19], and ocean wave power takeoffs [20]. Nonlinear dynamical systems can be represented with this methodology by simulating the nonlinear behavior using active electrical components. As such, electrical circuits can be exploited to model translational/rotational mechanical systems through analogies. Using modern simulators (e.g., Spectre and Eldo), simulating the equivalent circuits obtained is feasible in a fast and accurate way. Alternatively, such electrical equivalent circuits can be simulated via multi-physics simulators (e.g., Simulink, Siemens AMEsim, Modelica-based).

2.3 Previous works

The starting points for achieving a complete multi-domain fault simulation are the previous publications [21]. They introduce the use of analogies between the electrical and mechanical domains for fault injection into the mechanical domain. In these earlier papers, the first step of the proposed methodology is accomplished, namely, extending the electromechanical analogy to mechanical faults. Through various case studies, several mechanical systems were translated into electrical circuits. Next, these mechanical models transformed into electrical circuits were faulted, using techniques and fault models belonging exclusively to the electrical domain. Each fault behavior was studied by the authors to see if it could also have meaning in the mechanical domain. Then, the behaviors that were significant in the mechanical domain were added to the taxonomy produced by this analysis. A new taxonomy of mechanical faults was formed through the study of the fault behaviors of equivalent electrical circuits. However, in this paper, the purpose is to test the taxonomy produced directly on mechanical systems and then perform a multi-domain fault analysis. In addition, two very different systems, but both belonging to the analog domain, were chosen as the main case studies for this paper. In fact, what the authors want to prove is that the taxonomy produced is applicable both to fundamental analog components of many CPSs, but also to systems belonging to the MEMS world.

Table 1
Mapping between electrical and mechanical quantities in the force-voltage analogy.

Type	Mechanical translation	Mechanical rotation	Analogous electrical
Effort Variable	Force	Torque	Voltage
Flow Variable	Velocity	Angular velocity	Current
Elements	Damping	Rotational resistance	Resistance
	Mass	Moment of inertia	Inductance
	Compliance	Rotational compliance	Capacitance
	Mechanical impedance	Mechanical impedance	Electrical impedance

The proposed flow is exemplified through the adoption of a typical multi-discipline Hardware Description Language (HDL), i.e., Verilog-AMS, that describes a system as a set of differential equations that can be easily modified to mimic faulty behaviors [22]. An extension of this work concerns the thermal physics domain, which also enjoys the analogy with the electrical discipline [23]. In this paper, the model of one of the case studies proposed in this paper is extended to its thermal part, analyzing how a fault belonging to a specific discipline also affects the other domains that compose the system.

3 Exploiting analogies for modeling mechanical systems and faults

The physical analogies correlating the mechanical and the electrical domains are used to build equivalent systems that share the same behavior [24]. In particular, those analogies are:

- *force-voltage analogy:* mathematical equations of mechanical systems are compared with mesh equations of the electrical system; this analogy is considered the easiest to use (top-right of Figure 2);
- *force-current analogy:* mathematical equations of the mechanical system are compared with the nodal equations of the electrical system, thus being more conservative of the system structure (bottom-right of Figure 2).

From a mathematical perspective, neither analogy is superior since they both lead to valid and consistent results. Therefore, the choice of which one is to be adopted remains arbitrary. The proposed methodology builds upon the *force-voltage* analogy because it is more intuitive than the *force-current* one [25].

3.1 Force-voltage analogy between the mechanical and the electrical domain

Through the *force-voltage* analogy, a mechanical system can be represented as an electrical one by mapping each mechanical component in a corresponding electrical one, as shown in the Table 1:

- force, *effort* variable in the mechanical domain, is represented by voltage in the electrical domain;

- velocity, the mechanical *flow* variable, is related to the current in the electrical domain;
- all the other equivalence relations between physical quantities in the two domains are mathematically derived from the first two equivalencies: e.g., a damper is equivalent to a resistor because both represent energy loss in their domain.

Several mechanical systems and MEMS can be represented through an electrical circuit due to the mathematical analogies between the physical quantities, and they can be handled with electrical methodologies and tools. By translating the mechanical system into an electrical representation, well-established electrical techniques and tools can be directly used to analyze the mechanical system without the need for adaptations. Moreover, when an electromechanical system is transformed into an equivalent circuit, it can be analyzed as a unified electrical entity.

3.1.1 Mechanical system

Let us consider the tuned Mass-Spring-Damper system shown in Figure 3 as an example of the mechanical system. This system is composed of a mass (m_0) connected to the ground reference by a spring (k_0) and a damper (b_0), and to a second mass (m_1) through a spring (k_1) and a damper (b_1). Usually, the first mass (m_0) is bigger than the second mass (m_1). This configuration of the system, composing two Mass-Spring-Dampers, allows the damping of the movement amplitude in one oscillator by installing a second oscillator on it. Thus, if tuned properly, the maximum oscillation amplitude of the first system, with respect to a periodic input signal, will be lowered. The Mass-Spring-Damper behavior is described by the differential equations shown in Equations (1) and (2).

$$m_0\ddot{x_0} = F + k_2(x_1 - x_0) + b_1(\dot{x_1} - \dot{x_0}) - k_0x_0 - b_0\dot{x_0} \quad (1)$$

$$m_1\ddot{x_1} = -k_1(x_1 - x_0) - b_1(\dot{x_1} - \dot{x_0}) \quad (2)$$

3.1.2 Conversion to a mechanical network

In order to convert the mechanical model (Figure 3) to its electrical equivalent, passing through a representation of the dual mechanical network is useful, as the *force-voltage* analogy does not preserve the topology of the mechanical network during the realization of the electrical circuit [25]. In Figure 4 the two nodes x_0 and x_1 express the movements ($x_0(t)$ and $x_1(t)$) of the two masses in the mechanical system. Each node connects the mechanical components, or branches, which are exposed to the same displacement resulting from a force. The state space model of a mechanical system can be derived easily through its mechanical network description. Moreover, such conversion to the mechanical network is a very useful step to obtain the equivalent electrical circuit, especially using the force-voltage analogy. However, the choice of using mechanical networks is mainly because they can be drawn easily when adopting force-voltage analogy and they make the conversion step in the electrical domain simpler [26].

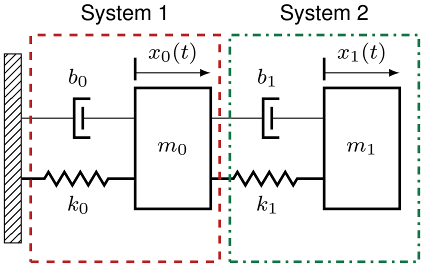


Figure 3. Mechanical representation of a Tuned Mass-Spring-Damper system connected to a fixed reference.

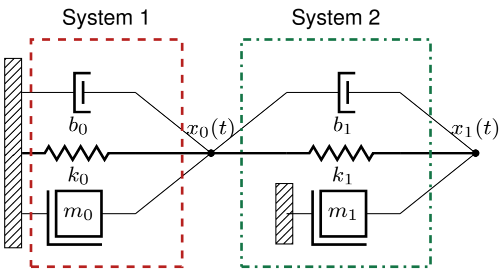


Figure 4. Representation of a Tuned Mass-Spring-Damper system as a mechanical network.

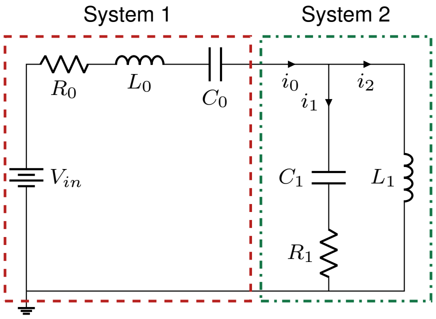


Figure 5. Electrical equivalent representation of the tuned Mass-Spring-Damper system (double-RLC).

3.1.3 Electrical equivalent

The electrical equivalent circuit is a simple electrical system composed of two resistance-inductance-capacitance (RLC) branches shown in Figure 5. Components of the mechanical network connected to the same displacement x_n are connected in series in the electric circuit since they are affected by the same electric current (*force-voltage* analogy consequence). Vice versa, components connected to different displacements will be connected in parallel since they are not affected by the same electric current. The complete behavior of the double-RLC is defined through the differential equations shown in Equations (3) and (4).

$$0 = V_{in} - R_0 i_0 - L_0\frac{di_0}{dt} - \frac{1}{C_0}\int i_0 dt + L_1\frac{di_2}{dt} \quad (3)$$

$$0 = L_1\frac{di_2}{dt} + \frac{1}{C_1}\int i_1 dt + R_1 i_1 \quad (4)$$

3.2 Standardized analog faults

For several years, the set of fault models was limited to open circuit, short circuit, sources, and parameter deviations for the electrical domain. However, what today is known as the IEEE P2427 draft standard [9] has been produced by its working group. The development group of the P2427 standard is working on standardizing the defect modeling, simulation techniques, and coverage metrics for both analog and mixed-signal circuits. In order to determine the behavior of each analog fault, the IEEE P2427 standard specifies that faults need to be injected one model at a time and in a single point of the circuit. In details:

- *short* electrical fault is equivalent to a bridge fault in an analog circuit, where two points of the circuit are joined by a not intended connection: this fault model has been introduced to ensure a connection between two points of the circuit by injecting it in parallel to a circuit component. Usually, the short fault model is added through a small resistance in parallel to a component (see Figure 6(b)).
- an *open* electrical fault represents a lack of a planned connection between two points in the circuit; thus, the electrons flowing in the circuit are not able to pass throuh the original branch anymore. Therefore, the open fault is injected with a large resistor in series before or after a single component (see Figure 6(c)) to stop the current from flowing.
- a *current pulse* fault is related to extra-currents injected in the circuit, usually modeled as pulses and injected in parallel with a component (see Figure 6(d)).
- a *voltage pulse* fault refers to an abnormal voltage difference added in an electrical circuit; it is usually realized as pulses, and it is injected in series with a component (see Figure 6(e)).
- *parametric* faults are related to the variation of a parameter inside a model, e.g., a smaller resistance value in a circuit due to errors in manufacturing.

4 A NOVEL MECHANICAL FAULT TAXONOMY

In literature, mechanical faults are categorized based on the component's physical characteristics [16] in relation to a potential cause (e.g. an overload) and a failure mode that specifies how the component fails. The fault outcomes mainly involve changes in materials, component geometries, shapes, and dimensions.

The *behavior* of mechanical systems is the main focus of this paper; therefore, a methodology relying entirely on the *behavioral-level* of abstraction has been investigated. Representing mechanical systems as mechanical networks and describing their behavior using equations allows inferring a fault taxonomy closer to this level. The fault modeling and fault injection operations of the electrical domain are also suitable for mechanical networks representing our mechanical system since the injection is performed into single branches in both cases. Moreover, building a mechanical network is structurally very similar to electrical circuits since both descriptions consist of nodes and branches, which have their own equations. For this reason, modeling a mechanical system through an electrical circuit seems to be a good approach for simulating faulty mechanical behaviors.

It is important to note that physical analogies do not provide any information about faults. The behavior is mathematically equal across the domains, but this does not mean that they are functionally equal, and thus, the correspondence of electrical faults with mechanical ones is not obvious. If the behavior of a faulty system makes sense from both an electrical and a mechanical perspective, then the fault models belonging to different domains are equivalent. This correlation can be established by simulating equivalent faulty electrical circuits using electrical fault injection techniques and then by studying the obtained behavior from the mechanical point of view. Some mechanical faults can be assimilated to the faulty behaviors obtained through electrical fault simulation, as defined in [15], [16]. Table 2 shows how mechanical fault behaviors can be derived by simulating faulty electrical circuits equivalent to mechanical systems. Now let us see how from electrical faults, the corresponding mechanical faults can be inferred.

4.1 Open circuit fault

The open fault simulates a break in the circuit line, resulting in a significant reduction of the current flow. Conceptually, the introduction of a braking agent in a mechanical system can be correlated to the open fault, as both results in a significant reduction in the flow variable (electric current or velocity, respectively). The fault can be thought of as increasing friction on the system's surface caused by many factors, such as temperature, wear, or debris.

4.2 Short circuit fault

The short circuit fault consists of an unintended connection between two points in the circuit that are not meant to be connected. This fault represents a disconnection of mechanical components from the remaining part of the system, such as springs or dampers. This fault can occur if the component is damaged by excessive backlash or a rupture.

4.3 Voltage and current sources faults

Voltage and current sources change the voltage and the current at a given point in the circuit in an unexpected trend. These sources of interference can be caused by surrounding electrical circuits or by alpha particles coming from outer space impacting the circuit. Considering the analogy, the voltage source is analogous to an unexpected external force on the faulty component. On the other hand, a current source can be considered as a changing factor of the velocity. Therefore, a current source changes the displacement or rotational velocity of the component affected by the fault.

4.4 Parametric faults

Parametric faults are equivalent in both domains since they consist of simple parameter variations.

The taxonomy shown in Table 2 is derived essentially from the simulation of several mechanical systems described through the electrical analogy: a mass-spring-damper, a tuned mass-spring-damper, a double pendulum, and a DC motor [21], [27]. Thus, the presented fault taxonomy can be extended by building equivalent circuits of more complex mechanical systems and simulating their behavior. After forming the taxonomy, fault analysis becomes multi-domain: faults are injected and then simulated in their

Table 2
Mechanical fault taxonomy derived from the analysis of electrical faults injected in the equivalent mechanical circuit.

Mechanical Behavior	Mechanical Effect	Mechanical Fault	Electrical Fault Equivalent
Brake/ friction	Added/increased braking force on failed component	Galling/Seizure, Creep, Spalling, Wear/-Corrosion	Open
Disconnection	The failed component detaches from the system	Rupture, excess of Backlash	Short
External force	An abnormal (external) force affects the component	It can cause deformations or cracks or ruptures (from impact of fatigue)	Voltage source
Limited movement	The direction of displacement of a component is abnormally modified	Rupture, Deformation, Wear	Current source
Parametric	Intrinsic characteristics of the failed component altered	Wear (component aging) and all the parameters changes	Parametric

own domain of belonging. For example, the mechanical faults produced are injected into mechanical systems or into the purely mechanical parts of the tested systems. In the following sections, the procedures of injection and fault simulation become multi-domain and they are illustrated and then exemplified on two complex systems: a DC motor connected to a gear train, which is the analog part of many CPSs, and a 3-axis accelerometer, which is a typical MEMS sensor.

5 Multi-domain fault injection

In this section, we present a methodology to inject the fault models introduced in Section 4 through the Verilog-AMS language.

5.1 Modeling multi-domain systems via Verilog-AMS

Verilog-AMS is the latest extension of the Verilog language created for combining the *digital* and the *analog* part. The communication between the *digital* and the *analog* part is possible through pre-defined language functions, e.g., `timer()`, activating a timer inside an analog design, and `cross()`, describing a crossing routine when the monitored analog function crosses the zero value of magnitude.

The Verilog-AMS language also defines constructs to model systems belonging to different physical domains, in particular, electrical, mechanical, and thermal. Complex models can be defined by combining various physical domains, e.g., by modeling electro-mechanical systems such as a direct current motor. A *discipline* represents a physical domain (e.g. electrical or mechanical), and it is composed by *natures*. For instance, the electrical domain is represented by the `electrical` discipline. The natures composing the discipline are the voltage and the current, and they are accessible respectively through the functions `V()` and `I()`. Conservative systems are defined by introducing *potential* and *flow* variables. Moreover, it is possible to define custom disciplines by changing the pre-defined *potential* and *flow* natures. The behavior of the conservative systems is modeled by the evolution of the natures of each domain over time, which needs to be specified. Building an electrical circuit is feasible by using the `branch` statement, which creates a connection between two nodes. Simulating behavioral Verilog-AMS models can be achieved in an efficient trend by using commercial SPICE-based simulators. The simulation can be handled by custom testbenches modules developed in SPICE-based code, e.g., Eldo and Spectre languages.

5.2 Behavioral fault injection

Fault injection in an analog system treats separately the injection of faults in the different physical domains. In the case of an electromechanical system, the injection process follows different rules for electrical equations and mechanical equations. Thus, let us introduce the procedures for injecting faults in the electrical and mechanical disciplines.

5.2.1 *Faults in the electrical discipline*

Figure 6 shows the location of the faults injected in a typical electrical circuit: Let us take as reference the short circuit fault between nodes `p` and `n1` shown in Figure 6(b). We can model that fault in Verilog-AMS with the following statement:

```
I(p, n1) <+ V(p, n1) / rshort
```

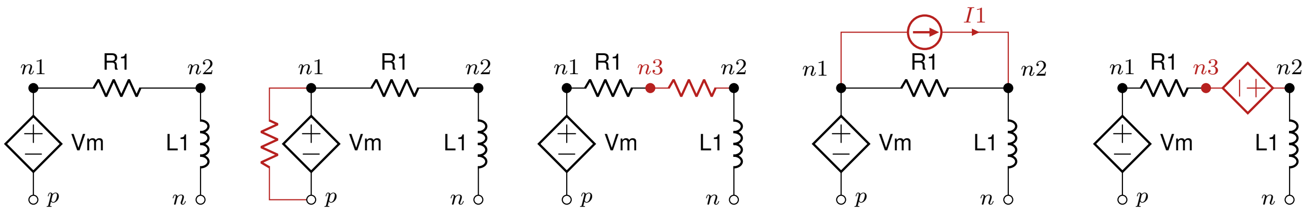
modeling a small resistor inserted between two nodes `p` and `n`, which is in parallel to the voltage source Vm.

We can model an open-circuit fault in Verilog-AMS, like the one shown in Figure 6(c), with the following statement:

```
V(n3, n2) <+ I(n3, n2) * ropen
```

which is a high resistive contribution to the series of branches that goes from node `n1` to node `n2` of the circuit. Conversely to what we did with the short circuit, here we need to add a new node `n3` between resistor `R1` and node `n2`, to inject our open fault.

Similarly to the short and open fault, voltage and current sources are also injected by following the same guidelines. These sources are unwanted external contributions in the branch in which they are injected, affecting its behavior. Adding a current source to a branch is done by injecting the equation in parallel on a specific branch, as described for the short fault. A voltage source is injected by adding the fault equation in series to the specific branch, as described for the open fault. Furthermore, parametric faults can be injected to alter the values of inductors and resistors.



(a) Fault-free circuit. (b) Short circuit. (c) Open circuit. (d) Current pulse. (e) Voltage pulse.

Figure 6. Representation of a simple electrical circuit in the fault-free configuration and in four faulty configurations.

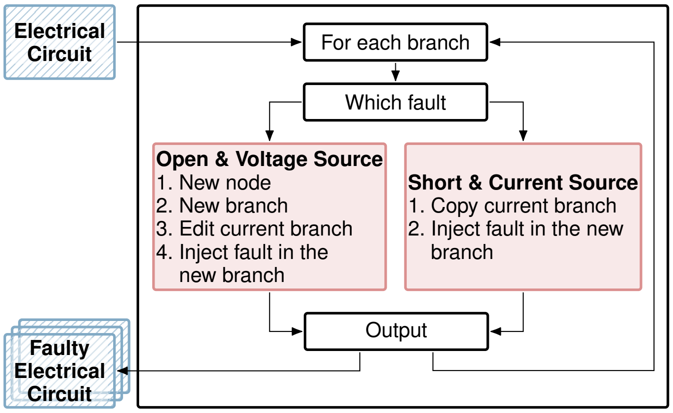


Figure 7. Structure of the fault injection framework for altering electrical descriptions. Red boxes identify the injection process, while blue striped boxes the input and output circuits.

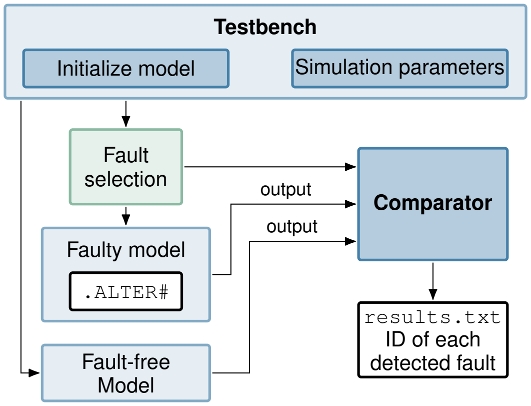


Figure 8. Structure of the simulation flow used to test the different faulty models.

5.2.2 Faults in the mechanical disciplines

In the mechanical domain, including both translational and rotational mechanics, faults are injected as a direct contribution to the branch that composes a mechanical system. The mechanical disciplines of Verilog-AMS are non-conservative; thus, the system is not modeled by a composition of mechanical nodes. The following statement represents a damper fault modeled in Verilog-AMS:

```
Tau(p, n) <+ Omega(p, n) * value
```

where *value* represents the value of resistance to motion. The higher this value, the greater the braking power will be to interrupt the component motion. The damper is injected in the same way into a translational mechanic model by replacing `Tau` with `F` and `Omega` with `Pos` (`kinematic` discipline) or `Vel` (`kinematic_v` discipline). Similarly, an external torque exerted on the rotational component included between the nodes `p` and `n` is modeled as:

```
Tau(p, n) <+ transition(100, 0, 0.1, 0.1)
```

where `transition` is a function that characterizes a transient contribution, i.e., not constant during the time with an associated rise and fall time.

5.3 Fault injection framework

The faults presented in Section 5.2 are injected automatically into Verilog-AMS descriptions by a developed tool. Currently, the framework supports the manipulation of the electrical and mechanical discipline of Verilog-AMS. As mentioned before, only one fault model is injected at a time and at a single point in the system, according to the fault taxonomies shown in Section 4. The fault injection framework is built as an additional tool of HIFSuite [28], which allows manipulation of various HDL descriptions, including Verilog/A-MS. Figure 7 explains how the fault injection process takes place for the electrical part: the tool converts any circuit described in Verilog-AMS into XML as an intermediate modeling representation for simpler fault injection. Depending on the fault model being injected, there is a procedure for faults injected in series to a branch (i.e. open and voltage source faults) and one for faults injected in parallel (i.e. short and current source faults). Finally, the tool returns all the faulty circuits, based on the electrical fault models, in their original HDL language.

The same procedure is followed for the injection of mechanical faults into Verilog-AMS models. The injection tool is able to generate multiple faulty descriptions of the original mechanical model with the same steps as with electrical circuits. Referring to Table 2, the braking agent and the external force faults will be injected once in each mechanical branch. The disconnection and the limited movement are injected between each couple of mechanical nodes in the Verilog-AMS module. The simulation is run by a testbench module, which instantiates both the fault-free model and the faulty models using the `alter` command (see Figure 8). The instantiated modules provide their simulation output to the comparator, which will detect which fault pattern has been injected and at what point in the simulation time. This testbench module simulates every module only once for all comparisons, obtaining all results in a dedicated file.

Table 3
DC motor with gear train parameters.

Variable	Name	Value	SI Unit
Input voltage	V_1	120	V
Back-EMF coefficient	K_E	0.1785	$\mathrm{V{\cdot}s/rad}$
Torque coefficient	K_T	25.2756	$\mathrm{N{\cdot}m/A}$
Armature resistance	R_A	8.4	Ω
Armature inductance	L_A	0.0084	H
Motor inertia	J_1	0.0035	$\mathrm{kg{\cdot}m^2}$
Motor friction	B_1	0.064	-
Gear inertia	J_2	0.035	$\mathrm{kg{\cdot}m^2}$
Gear friction	B_2	2.64	-
Motor shaft radius	r_1	0.02	m
Gear shaft radius	r_2	0.16	m
Gear ratio	N	r_2/r_1	-

5.4 Functional qualification of the verification testbench

The behavioral fault models presented in the previous sections are defined to be applied in analog descriptions that allow the representation of faulty continuous behaviors. This situation could present many variabilities due to the model itself and due to the stimulus provided to the model by the testbench. Thus the testbench quality is fundamental to adequately stimulate the model in the presence of nominal and faulty conditions [29]. These stimuli can be applied at different points of the model, depending on the physical system under analysis. For example, a direct current motor can be stimulated by applying different source waveforms to the electrical part enabling a different response of the motor dynamic. This implies that different input waveforms can stimulate different faults injected into the model by changing the diagnostic coverage metrics calculated to asses the functional safety of complex systems. These variabilities due to the input waveforms are described in the literature for the analog domain as *testbench qualification* [30], [31].

The same concepts can be applied to systems described at the behavioral level, as proposed in this article. Consequently, an in-depth analysis should be performed to retrieve the range of values in which the input waveforms need to be positioned to stimulate the system correctly, e.g., by applying a specific waveform frequency range as input. By combining ad-hoc testbenches with systematical fault injection campaigns in multi-domain systems, correct diagnostic coverage metrics can be calculated to guarantee the functional safety of the overall system. In this article, the proposed case study (see Section 6) has been analyzed under different faulty conditions (see Section 4) and various operating conditions. The latter scenario has been created by applying different stimuli to the system under test. These stimuli change the behavior of the system, and they are obtained through a refinement of the testbench module. Refining means building the testbench that stimulates the widest amount of the system's components, namely activating the largest number of faults among all the injected into the system.

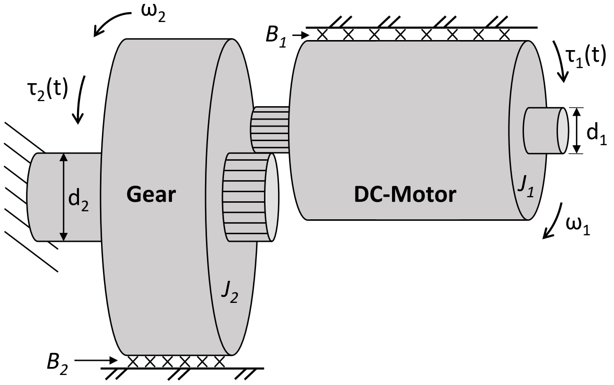


Figure 9. DC motor with a gear train: the DC motor (on the right top of the figure) is connected with the gear train (center of the figure).

6 Application to complex systems

The fault taxonomies presented in Section 4 are validated on two complex cases of study. The first one is a Direct Current (DC) motor with a gear train shown in Figure 9. The second one is a 3-axis accelerometer realized as a MEMS sensor. Every physical domain is subject to variations due to faults. Consequently, the different faults (adapted for specific physical domains) can be applied to different classes of systems, ranging from smart systems to the analog side of CPSs. If considered an open fault applied on a subcomponent of a smart system, e.g., an RF switch MEMS [32], the movement of the internal switch will be blocked to one of its configurations, forcing the internal resistance in the on or off-state limiting the functionalities of the smart system. The presented results are exemplified on a DC motor due to its extensive application in every industrial field, from miniaturization scale [3] to a bigger scale [33]. The same consideration holds for the accelerometer model, which is a very common sensor in multiple working environments [22]. The two models are initially modeled with differential equations, then written in Verilog-AMS and simulated with a SPICE-based simulator.

6.1 DC motor with gear train

The DC motor is an electromechanical component; consequently, it is modeled with electrical and mechanical differential equations, while the gear train component receives the torque from the connection with the DC motor and is modeled only with mechanical equations. The following constitutive relations model the dynamics of the motor connected to the gear train:

$$V_1 = K_E \cdot \omega_1 + R_A \cdot I_A + L_A \cdot \frac{d(I_A)}{dt}\,, \tag{5}$$

$$\tau_1 = K_T \cdot I_A - J_1 \cdot \frac{d(\omega_1)}{dt} - B_1 \cdot \omega_1\,, \tag{6}$$

$$\tau_2 = - N \cdot \tau_1 + J_2 \cdot \frac{d(\omega_2)}{dt} + B_2 \cdot \omega_2\,, \tag{7}$$

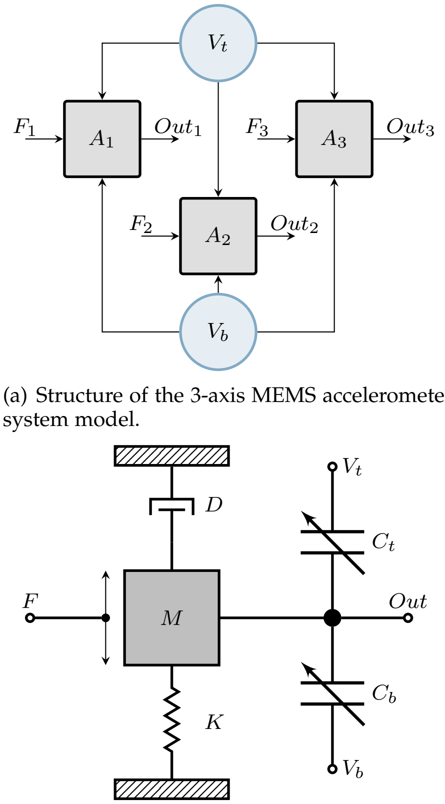
where V_1 represents the input voltage source for controlling the velocity of the rotor; the variable I_A represents the current flowing in the motor's windings; the angular velocity of the shaft is given by the variable ω, while the torque on the shaft by the variable τ. K_T and K_E are motor coefficients used to specify the size of the motor. Table 3 lists the value of each parameter used in this test case. Equation (5) and Equation (6) represent the electrical and mechanical rotational dynamics of the DC motor, while Equation (7) represents the dynamic of the gear train that receives the opposite torque (i.e., it turns in the opposite directions of the motor) from the motor shaft reduced by factor N representing the reduction factor. This mechanical system is used in many applications to increase the torque on the gear train shaft, e.g., as the driver of the joints of an anthropomorphic manipulator [34].

```
`include "disciplines.vams"
`include "constants.vams"
`timescale 1us / 1us

module dcmotor(tau_drive, p, n, tau_load);
  // Parameters
  ...
  // Ports
  output tau_drive;
  input p, n, tau_load;
  // Nodes
  electrical p, n, n1, n2;
  rotational_omega tau_drive, tau_load, rgnd;
  // Reference nodes
  ground rgnd;
  // Branches
  branch (p, n1)  vm;     // motor voltage
  branch (n1, n2) ra;     // motor resistance
  branch (n2, n)  la;     // motor inductance
  branch (tau_drive, rgnd) motor; // DC motor
  // Behavior
  analog begin
    // Electrical motor internal dynamic
    V(vm)  <+ Ke * Omega(motor);
    V(ra)  <+ Ra * I(ra);
    V(la)  <+ La * ddt(I(la));
    // Motor dynamic
    Tau(motor) <+ + (Kt * I(vm));
    Tau(motor) <+ - (B1 * Omega(motor));
    Tau(motor) <+ - (J1 * ddt(Omega(motor)));
    // Gear contribution
    Tau(motor) <+ + (coef) * (Tau(tau_load));
  end
endmodule

module reductor(tau_drive, tau_load);
  // Parameters
  ...
  // Ports
  output tau_load;
  input tau_drive;
  // Nodes
  rotational_omega tau_drive, tau_load, gnd_mec;
  // Reference nodes
  ground gnd_mec;
  // Branches
  branch (tau_load, gnd_mec) gear;
  // Behavior
  analog begin
    N = r2 / r1;
    // Gear train dynamic
    Tau(gear) <+ - (N * Tau(tau_drive));
    Tau(gear) <+ + (B2 * Omega(gear));
    Tau(gear) <+ + (J2 * ddt(Omega(gear)));
  end
endmodule
```

Listing 1. Verilog-AMS modules implementing the DC motor and the gear train.

6.2 MEMS 3-axis accelerometer

As the second case of study, we chose the MEMS model of a 3-axis accelerometer, shown in Figure 10. Until a few years ago, accelerometers were intended mainly for scientific, military, or civilian uses. However, due to the recent evolution of electronics, reduction of costs, and development of applications, accelerometers are widely used on everyday objects [35]. The sensor considered in this article was made by combining three individual accelerometers [22], orienting them in three different directions perpendicular to each other.



(a) Structure of the 3-axis MEMS accelerometer system model.

(b) Schematic of the internal components of a single accelerometer model [22].

Figure 10. The overall structure of the second case study.

The structure of the 3-axis accelerometer is depicted in Figure 10(a). The three accelerometers, identified as A1, A2 and A3, are identical and, for this reason, powered by the same voltage sources V1 and V2. Each accelerometer is stimulated by different external forces, F1, F2 and F3, since they are positioned orthogonally, thus producing three different output voltages. The internal structure of each single-axis accelerometer is shown in Figure 10(b). Like the motor, this model is also a system composed of mechanical and electrical parts. First, the acceleration exerted on the seismic mass is caused by the force F exerted from outside. The displacement of the mass, limited by the spring and the damper, is directly proportional to the intensity of the acceleration experienced by the system. This purely mechanical behavior is expressed by Equation (8),

$$F(t) = Kx + D\frac{dx}{dt} + M\frac{d^2(x)}{dt^2} \qquad (8)$$

where x is the displacement of the mass, M is the mass weight, D is the damping factor, and K is the spring stiffness. Thus, the displacement exerted by the mass has an impact on the electrical part of the accelerometer. It consists of two capacitors, both connected to a dedicated voltage source. The mass forms the middle plate of the differential capacitive circuit: the intervals between the plates change when the mass is not in equilibrium anymore, thus changing the capacitive value of the whole circuit. These changes are expressed by Equation (9),

$$C_t = \frac{\varepsilon A}{d_0 - x}, \qquad C_b = \frac{\varepsilon A}{d_0 + x} \tag{9}$$

where A is the area of the capacitor plate, d_0 the gap between the plates at equilibrium, and ε is the vacuum permittivity constant ($8.85\mathrm{e}{-12}$ F/m). Finally, Equation (10) shows the behavior of the currents flowing through the two physical capacitors, producing the output voltage value of acceleration.

$$I_t = C_t \frac{dV_t}{dt}, \qquad I_b = C_b \frac{dV_b}{dt} \tag{10}$$

6.3 Multi-domain fault injection and simulation

In this section, the fault taxonomy produced by applying the methodology expressed in this article is used to perform a fault analysis on the two case studies presented. Both case studies were modeled using Verilog-AMS through multiple modules. The simulation environment is set up on a CentOS machine built of an i7-9700 with 3.0 GHz frequency and 16 GB RAM memory. The Verilog-AMS code is simulated with the Questa-ADMS tool (that uses Eldo as an analog simulator and Questa as a digital simulator) by Siemens EDA. The faults of the two domains composing both the systems have been injected through the automatic fault injection tool presented in the previous sections.

6.3.1 DC Motor

The DC motor with gear train has been modeled with two different modules described with the Verilog-AMS language and shown in Listing 1. The first module contains the DC motor equations (Equation (5) and Equation (6)), while the second module includes the gear train equation (Equation (7)). The two modules are interconnected by a `rotational_omega` port that allows the torque exchange between the two components. Regarding fault injection in the electrical part, consisting of three branches, the tool injected 3 open, 12 shorts, 3 voltage sources, and 12 current sources. On the other hand, the mechanical faults of Table 2, except the parametric, have been injected once per mechanical branch, then 2 faults. The entire model has been simulated in different operating conditions, and the faults from Table 2 have been tested on this system. Moreover, the system has been stimulated with different input curves, e.g., step or sine curve, in order to analyze the different faulty behaviors. Figure 11 shows the simulation results for this first case study. The plots are related to the angular velocity of the DC motor with the gear train simulated for 10 seconds: they report fault-free simulation (dotted blue line) and four different fault models:

- one from the electrical domain: an open fault in the electrical part of the motor (open-m-100, green line);
- three mechanical domain faults: an external torque source on the motor with a value of $120N \cdot m$ (τ-source-m-120, red line), a damper on the motor with a value of 1e03 (damper-m-1e03, magenta line), and a torque source on the gear train with a value of $40N \cdot m$ (τ-source-r-40, yellow line).

Note that Figure 11(b) and Figure 11(c) are related to the simulation of the system stimulated with a voltage step of 120 V for 5 seconds. Furthermore, Figure 11(d) and Figure 11(e) are related to the simulation of the system stimulated with a sin curve of 120 V with frequency 0.1 Hz for 5 seconds. The plots show the fault-free response of the system reflecting system equations (blue dotted line) and the effect of faults: the external torque accelerates the two components drastically (red line); the damper stops motor rotation (magenta line), and the open fault reduces the voltage feed for the motor, allowing the system to rotate less than the normal response (green line). The presented simulation results have been obtained by feeding the system with stimuli that allow the detection of all the injected fault models. We can notice how, by feeding the system with different stimuli, faulty behaviors act in a different way.

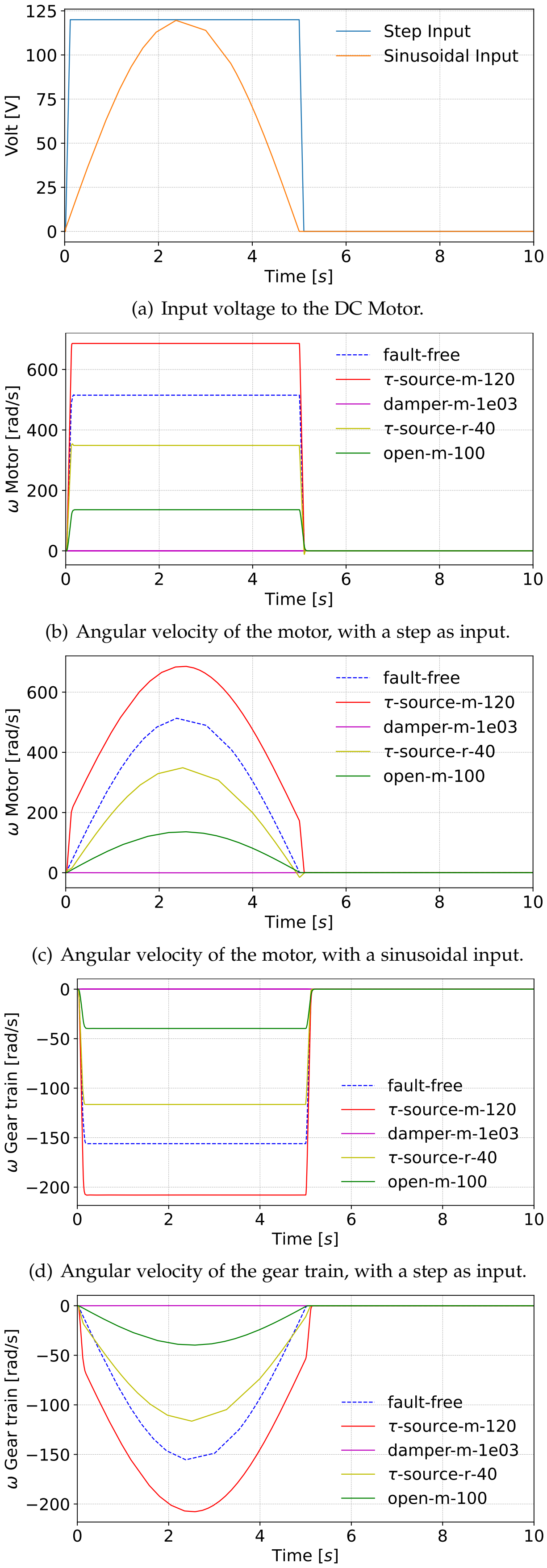
For example, with an insufficient supply voltage source to the motor, the effect of a damper on the motor cannot be visible because the motor would not move anyway. However, with a torque source on the gearbox, it is still possible to detect the faulty behavior of this fault (because everything should be stopped). Instead, a high input voltage source implies that the motor works at high velocity: a torque source on the gearbox cannot be noticed, whereas the effects of a damper on the motor will be detected. This type of analysis highlights how the quality of the testbench module affects the system behavior during the fault campaign simulation.

These considerations take place in the digital domain as *functional qualification of the testbench verification* analysis [36]. In the analog field, this approach needs to be considered to verify the correctness of the design by using the multi-domain fault modeling presented in this article. Consequently, an in-depth analysis should be performed to retrieve the range of values in which the input waveforms need to be positioned to stimulate the system correctly, e.g., by applying a specific waveform frequency range as input [30].

6.3.2 MEMS Accelerometer

Moving to the 3-axis accelerometer, the faults have been injected into both physical disciplines that form the system. In the electrical part, we injected 6 open circuits, 27 short circuits, 6 voltage sources, and 27 current sources were injected. In the mechanical part, we injected 12 faults were injected, 3 for each present in the taxonomy, see Table 2. The model was stimulated under several scenarios through different forces applied to the system, e.g., shocks through force pulses or continuous acceleration through sine waves. Figure 12 shows the simulation results for this second case study. In particular, all mass displacements related to the different scenarios, fault-free and faulted ones, have been included in Figure 12(a). The colors represent different working conditions and each color is related to its own acceleration output in the next plots. For example, the fault-free displacement is represented with a blue line, and the related acceleration value, depicted in Figure 12(b), is drawn with the same line format. The same rule lasts for all the

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2023.3345135



(a) Input voltage to the DC Motor.

(b) Angular velocity of the motor, with a step as input.

(c) Angular velocity of the motor, with a sinusoidal input.

(d) Angular velocity of the gear train, with a step as input.

(e) Angular velocity of the gear train, with a sinusoidal input.

Figure 11. Fault-free and faulty DC Motor waveforms.

faulty behaviors shown as well. The model was simulated for 200 µs and it was fed with a sinusoidal acceleration. Figure 12(a) and Figure 12(b) highlight that the measured acceleration is greater when the mass movement is more intense.

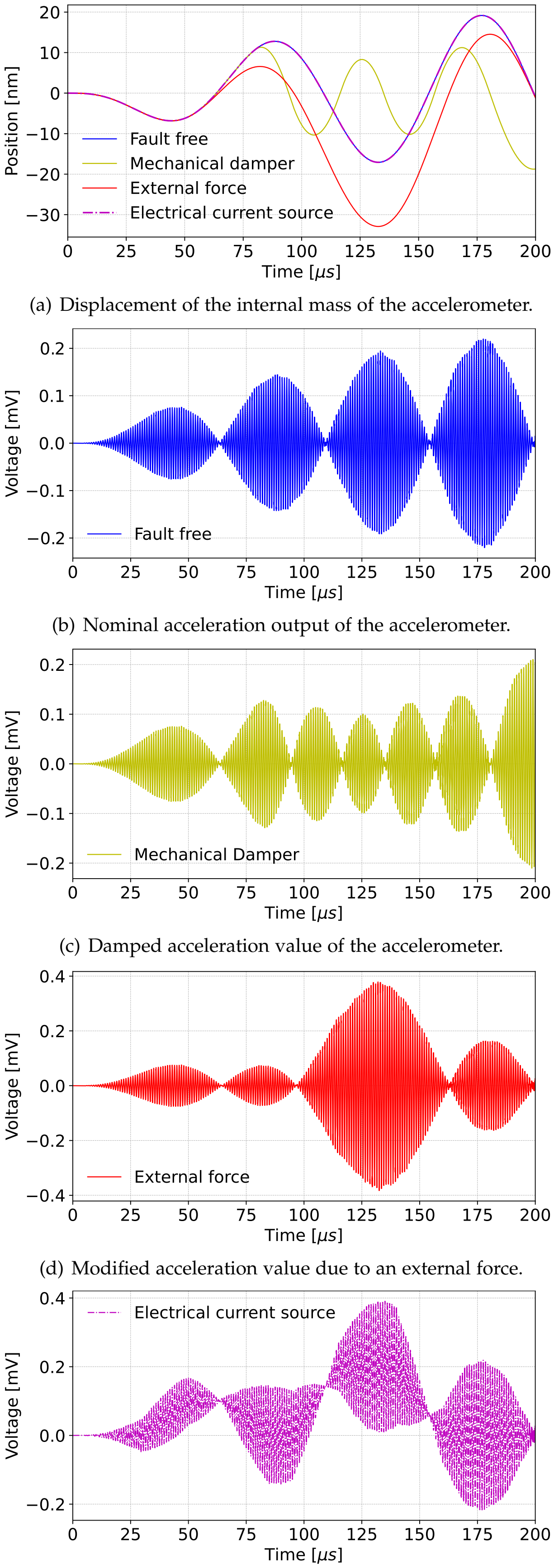
Let us now analyze some examples of faulty behaviors in the MEMS accelerometer. Figure 12(c) represents the model's response when suffering a damping fault, which can be caused by many factors, such as an increased friction value or an object that slows the movement of the mass. This fault has been represented both in Figure 12(a) and Figure 12(c) with a yellow line. This time, the fault is transient, meaning that it is not active for the entire simulation duration but only for a limited period. The motivation behind this choice is just to model a more realistic fault scenario. The duration of the effect of this fault is 100 µs, specifically, from 80 µs to 180 µs. While this fault is active, the oscillation of the mass is limited (see Figure 12(a)). Consequently, the acceleration measured by the accelerometer will be incorrectly lower (see Figure 12(c)).

Another fault model tested on the system is the unexpected external force drawn with a red line. In particular, the system has been subjected to a force that is not part of the input provided to the system. In this scenario, the mass suffers a displacement that is not due to the force exerted on the whole system, as can be seen in Figure 12(a). As a result, the accelerometer measures an acceleration that does not fully reflect the force given as input to the system, but the one indicated by the erroneous displacement of the mass. This transient fault is active from 50 µs to 130 µs, for a total duration of 80 µs. For this reason, the acceleration measured after the fault disappears reflects the behavior of the fault-free model.

Finally, Figure 12(e) shows the effect of an electrical fault injected on the upper capacitor of the sensor, C_t. The fault represents an anomalous current source connected to the branch, thus adding additional current to the capacitor's current. Again, this is a transient fault: the source is active from 30 µs to 65 µs, and from 95 µs to 135 µs, but with doubled intensity. Figure 12(e) depicts clearly how the current source changes the intensity of the acceleration measured by the accelerometer, changing the amplitudes in the first three oscillations.

7 Conclusions

This study proposes techniques to support the *functional safety* in the analog side of a smart system or CPS by defining an automatic flow to inject faults into multi-domain models and create an efficient simulation environment to analyze faulty behaviors. Starting from the electrical faults described in state-of-the-art, a mechanical fault taxonomy is derived from analogy. The fault taxonomy is derived by analyzing the faulty behaviors of equivalent mechanical systems represented as an electrical circuit extended with electrical faults. Then, a framework that allows injecting systematically multi-domain faults into Verilog-AMS descriptions is presented and validated through different case studies. The faulty models produced by using the framework are then simulated through an automated simulation flow that exploits the SPICE-simulators' potentialities. The entire



(a) Displacement of the internal mass of the accelerometer.

(b) Nominal acceleration output of the accelerometer.

(c) Damped acceleration value of the accelerometer.

(d) Modified acceleration value due to an external force.

(e) Modified acceleration value due to an external current source.

Figure 12. Fault-free and faulty accelerometer waveforms.

methodology helps to understand the failure mechanisms of multi-domain systems and propose mitigation solutions in the design phases allowing to test accurately the correct functionality of the safety mechanisms. Future works will explore how a fault injected in one physical domain composing a smart system or CPSs will impact the overall functionality of these systems. Moreover, we will explore how to test the correct functionality of functional safety mechanisms when multi-domain faults occur in the system.

References

[1] E. Fraccaroli and S. Vinco, "Modeling cyber-physical production systems with systemc-ams," *IEEE Transactions on Computers*, pp. 1–13, 2022.

[2] M. Lora, S. Vinco, and F. Fummi, "Translation, abstraction and integration for effective smart system design," *IEEE Transactions on Computers*, vol. 68, no. 10, pp. 1525–1538, 2019.

[3] A. C. Fernandez-Pello, A. P. Pisano, K. Fu, D. C. Walther, A. Knobloch, F. Martinez, M. Senesky, C. Stoldt, R. Maboudian, S. Sanders, and D. Liepmann, "Mems rotary engine power system," *IEEJ Transactions on Sensors and Micromachines*, vol. 123, no. 9, pp. 326–330, 2003.

[4] Z. Huang, C. Wang, M. Stojmenovic, and A. Nayak, "Characterization of cascading failures in interdependent cyber-physical systems," *IEEE Transactions on Computers*, vol. 64, no. 8, pp. 2158–2168, 2015.

[5] C. Bolchini and A. Miele, "Reliability-driven system-level synthesis for mixed-critical embedded systems," *IEEE Transactions on Computers*, vol. 62, no. 12, pp. 2489–2502, 2013.

[6] M. Moradi, B. Van Acker, and J. Denil, "Failure identification using model-implemented fault injection with domain knowledge-guided reinforcement learning," *Sensors*, vol. 23, no. 4, 2023. [Online]. Available: https://www.mdpi.com/1424-8220/23/4/2166

[7] *ISO 26262 – Road vehicles – Functional safety*, ISO, 2011.

[8] I. Pomeranz and S. Reddy, "Location of stuck-at faults and bridging faults based on circuit partitioning," *IEEE Transactions on Computers*, vol. 47, no. 10, pp. 1124–1135, 1998.

[9] I. S. Association, *IEEE-SA Standards Board P2427/D0.13 Draft Standard for Analog Defect Modeling and Coverage*. IEEE, 2019. [Online]. Available: https://standards.ieee.org/project/2427.html

[10] R. Stetter, "Approaches for modelling the physical behavior of technical systems on the example of wind turbines," *Energies*, vol. 13, no. 8, p. 2087, Apr. 2020. [Online]. Available: https://doi.org/10.3390/en13082087

[11] X. Yang, C. Yang, T. Peng, Z. Chen, B. Liu, and W. Gui, "Hardware-in-the-loop fault injection for traction control system," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 6, no. 2, pp. 696–706, 2018.

[12] M. Fernandes, J. M. Corchado, and G. Marreiros, "Machine learning techniques applied to mechanical fault diagnosis and fault prognosis in the context of real industrial manufacturing use-cases: a systematic literature review," *Applied Intelligence*, vol. 52, no. 12, pp. 14 246–14 280, 2022.

[13] L. Tang, H. Tian, H. Huang, S. Shi, and Q. Ji, "A survey of mechanical fault diagnosis based on audio signal analysis," *Measurement*, p. 113294, 2023.

[14] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE transactions on dependable and secure computing*, vol. 1, no. 1, pp. 11–33, 2004.

[15] E. Balaban, P. Bansal, P. Stoelting, A. Saxena, K. F. Goebel, and S. Curran, "A diagnostic approach for electro-mechanical actuators in aerospace systems," in *2009 IEEE Aerospace conference*. IEEE, 2009, pp. 1–13.

[16] S. J. Uder, R. B. Stone, and I. Y. Tumer, "Failure analysis in subsystem design for space missions," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 46962, 2004, pp. 201–217.

[17] J. Bougie and A. Gangopadhyaya, "Conservation laws and energy transformations in a class of common physics problems," *American Journal of Physics*, vol. 87, no. 11, pp. 868–874, nov 2019.

[18] N. Hogan and P. C. Breedveld, “The physical basis of analogies in physical system models,” in *The mechatronics handbook*. CRC Press/Balkema, 2002.

[19] R. R. Thakkar, “Electrical equivalent circuit models of lithium-ion battery,” *Management and Applications of Energy Storage Devices*, 2021.

[20] C.-A. Chen, X. Li, L. Zuo, and K. D. Ngo, “Circuit modeling of the mechanical-motion rectifier for electrical simulation of ocean wave power takeoff,” *IEEE Transactions on Industrial Electronics*, vol. 68, no. 4, pp. 3262–3272, 2020.

[21] F. Tosoni, N. Dall’Ora, E. Fraccaroli, and F. Fummi, “A Framework for Modeling and Concurrently Simulating Mechanical and Electrical Faults in Verilog-AMS,” in *2022 Forum on Specification & Design Languages (FDL)*, 2022, pp. 1–8.

[22] F. Pecheux, C. Lallement, and A. Vachoux, “VHDL-AMS and verilog-AMS as alternative hardware description languages for efficient modeling of multidiscipline systems,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 2, pp. 204–225, 2005.

[23] F. Tosoni, N. Dall’Ora, E. Fraccaroli, S. Vinco, and F. Fummi, “Thermal Digital Twin of a Multi-Domain System for Discovering Mechanical Faulty Behaviors,” in *2023 IEEE 21st International Conference on Industrial Informatics (INDIN)*, 2023, pp. 1–8.

[24] P. Gardonio and M. J. Brennan, “On the origins and development of mobility and impedance methods in structural dynamics,” *Journal of Sound and vibration*, vol. 249, no. 3, pp. 557–573, 2002.

[25] J. López-Martínez, J. C. Martínez, D. García-Vallejo, A. Alcayde, and F. G. Montoya, “A new electromechanical analogy approach based on electrostatic coupling for vertical dynamic analysis of planar vehicle models,” *IEEE Access*, vol. 9, pp. 119 492–119 502, 2021.

[26] M. Smith, “Synthesis of mechanical networks: the inerter,” *IEEE Transactions on Automatic Control*, vol. 47, no. 10, pp. 1648–1662, 2002.

[27] N. Dall’Ora, F. Tosoni, E. Fraccaroli, and F. Fummi, “Inferring mechanical fault models from the electrical domain,” in *2022 IEEE 5th International Conference on Industrial Cyber-Physical Systems (ICPS)*. IEEE, 2022, pp. 01–08.

[28] N. Bombieri, G. D. Guglielmo, L. D. Guglielmo, M. Ferrari, F. Fummi, G. Pravadelli, F. Stefanni, and A. Venturelli, “HIFSuite: Tools for HDL code conversion and manipulation,” in *2010 IEEE International High Level Design Validation and Test Workshop (HLDVT)*. IEEE, Jun. 2010.

[29] H.-Y. Lin, C.-Y. Wang, S.-C. Chang, Y.-C. Chen, H.-M. Chou, C.-Y. Huang, Y.-C. Yang, and C.-C. Shen, “A probabilistic analysis method for functional qualification under mutation analysis,” in *2012 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2012, pp. 147–152.

[30] M. Hassan, D. Große, H. M. Le, T. Vörtler, K. Einwich, and R. Drechsler, “Testbench qualification for SystemC-AMS timed data flow models,” in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2018, pp. 857–860.

[31] N. Bombieri, F. Fummi, V. Guarnieri, and G. Pravadelli, “Testbench Qualification of SystemC TLM Protocols through Mutation Analysis,” *IEEE Transactions on Computers*, vol. 63, no. 5, pp. 1248–1261, 2014.

[32] G. Zorpette, “Rf mems deliver the "ideal switch": After two decades of development, mems-based rf switches are finally finding real-world uses,” *IEEE Spectrum*, vol. 57, no. 8, pp. 8–9, 2020.

[33] D. A. Barkas, G. C. Ioannidis, C. S. Psomopoulos, S. D. Kaminaris, and G. A. Vokas, “Brushed DC motor drives for industrial and automobile applications with emphasis on control techniques: A comprehensive review,” *Electronics*, vol. 9, no. 6, p. 887, May 2020.

[34] S.-H. Yen, P.-C. Tang, Y.-C. Lin, and C.-Y. Lin, “A sensorless and low-gain brushless DC motor controller using a simplified dynamic force compensator for robot arm application,” *Sensors*, vol. 19, no. 14, p. 3171, Jul. 2019.

[35] A. D’Alessandro, S. Scudero, and G. Vitale, “A review of the capacitive mems for seismology,” *Sensors*, vol. 19, no. 14, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/14/3093

[36] “Ensuring functional safety for self-driving cars,” https://semiengineering.com/ensuring-functional-safety-for-self-driving-cars/, 2019.

Francesco Tosoni (Student, IEEE) is currently a Ph.D. student at the University of Verona. He received his M.Sc. degree in Computer Science and Engineering at the University of Verona, in March 2022, with a thesis on Multi-Domain Fault Injection in Cyber-Physical Systems. His main research interest focuses on system simulation and fault injection techniques to ensure the functional safety in the context of Industrial Cyber-Physical Systems.

Nicola Dall’Ora (Member, IEEE) is a postdoctoral researcher at the Department of Engineering for Innovation Medicine of the University of Verona (Italy). He received his Ph.D. degree in Computer Science from the University of Verona, Italy, in May 2023. His primary research focuses on defect/fault injection and simulation for different modeling levels, from transistor-level to multi-domain systems described at the behavioral level, to extract fault modes useful to guarantee functional safety at the system level.

Enrico Fraccaroli (Member, IEEE) is a postdoctoral researcher at the Department of Computer Science of the University of Verona (Italy). He received his Ph.D. degree in computer science from the University of Verona, Italy, in May 2019. His research interests are the development of new methodologies for the efficient simulation and functional safety evaluation of embedded platforms composed of analog, digital, and network components.

Sara Vinco (Senior Member, IEEE) is Associate Professor at Politecnico di Torino, Turin, Italy, since 2021. She received the Ph.D. degree in Computer Science from the University of Verona, Verona, Italy, in 2013. Her current research interests include digital twins, energy efficient electronic design automation, efficient simulation and optimization of energy systems, and techniques for simulation and validation of heterogeneous embedded systems.

Franco Fummi (Member, IEEE) received the Laurea degree in Electronic Engineering at the Polytechnic of Milan in 1990 and the Ph.D.in Electronic and Communication Engineering in 1994 at the Polytechnic of Milan. Since March 2001 he is a Full Professor in Computer Architecture at Università di Verona. He is leading the Cyber-physical and IoT Systems Design (CISD) group of the Università di Verona, currently composed of more than 20 people, and working on hardware description languages and electronic design automation methodologies for modeling, verification, testing, and optimization of cyber-physical systems. He is also a co-founder of two spin-off companies: EDALab, focused on networked embedded systems design; and the automation control software company FACTORYAL.