How Aerospace and Transportation Design Challenges can be addressed from Simulation-based Virtual Prototyping for Distributed Safety Critical Automotive Applications

Jean-Yves Brunel, Alberto Ferrari, Eliane Fourgeau, Paolo Giusto

How Aerospace and Transportation Design Challenges can be addressed from Simulation-based Virtual Prototyping for Distributed Safety Critical Automotive Applications

Jean-Yves Brunel, Cadence Design Systems, Inc.
Alberto Ferrari, Parades
Eliane Fourgeau, Cadence Design Systems, Inc.
Paolo Giusto, Cadence Design Systems, Inc.

Keywords: Embedded Software, Distributed Safety Critical Automotive Systems, Virtual Prototyping, Simulation, Fault Injection, Flight Control.

Abstract.

The reduction of development and product costs for distributed and software dominated safety-critical automotive applications can only be achieved via novel methodologies and tool sets that address fault injection/analysis and integration testing via simulation-based virtual prototyping. In fact, earlier discovery of design errors and initial proof of safety in critical conditions should be addressed earlier using a system virtual prototype, before hardware and software implementations are available. In this paper, we propose a methodology that allows evaluating fault-tolerant system architectures in the presence of errors caused by faults of hardware elements or interferences. We illustrate how the paradigm shift from physical to virtual integration platforms can be applied to Aerospace and Transportation domains effectively.

Introduction

The automotive industry has been adopting technologies such as by-wire, and new design and implementation paradigms such as distributed embedded software engineering and architectures, that are already familiar to aerospace engineers (e.g. fly-by-wire). On the other hand, we have also observed opposite phenomena such as the adoption by the airspace industry [4] of serial bus protocols and time-triggered architectures, originally being defined for automotive applications. Essentially, the overall goals are common to both segments: from the OEMs standpoint, the need to manage new design complexities due to communicating sub-systems, time to market, and cost reductions, while from the driver/passenger's stand-point, the need for comfort and safety.
Electronics is growing dramatically in both segments. It is expected that up to 90% of future vehicle innovations will be based on electronic embedded systems with the automotive semiconductor market expected to grow to 15.5 billion US dollars by 2010. As far as the aerospace segment is concerned, electronics has increased for on-board long and medium-range airliners. The major need that is emerging is navigation, control and tightly integrated communication resources on board, while providing increasingly sophisticated technologies and features to the passengers. Nowadays, airports and skies are increasingly congested thus airline crews are in major need of those resources. Comfort, just like in cars, plays a major role in determining the airline of choice and electronics enables airlines to provide passengers with improved and more sophisticated in-flight services.
We also observe another interesting and common phenomenon. Both segments are looking for standards that facilitate the plug-and-play of functions and components in a modular architecture in order to favor re-use, better complexity management, streamlining of the design process and thus faster time to market. Automotive initiatives such as AutoOsar [3] and of course, in the past, OSEK [2], are/have been aiming to make software functions inde

pendent from the underlying hardware. As far as communication protocols are concerned, safety critical, fail-safe/fault-tolerant, redundant and dependable protocols are being introduced, such as FlexRay [10], and TTP [11]. These serial protocols provide the needed underlying safety services while hiding un-necessary details to the software developers[1].Likewise, in the aerospace industry, Integrated Modular Electronics (IME), is key [6]. In a nutshell, this means that individual hardware and software modules can be removed and replaced interchangeably. Several projects have been and are under way such as PAMELA [5] and VICTORIA [6]. "VICTORIA's main objective is to design, prepare and validate a new system that would integrate all on-board functions for commercial airliners, including both flight and passenger management systems" [6].
As far as safety analysis is concerned, as described in [12] few methods for safety analysis exist already or have been defined. The methods pertain to: preliminary safety analysis (PSA), preliminary hazard analysis (PHA), detailed safety analysis (DSA), and safety integrity levels (SIL). "An example of a process used in the aerospace industry is DO-178B. The automotive industry is expected to use or develop a similar safety process" [13].

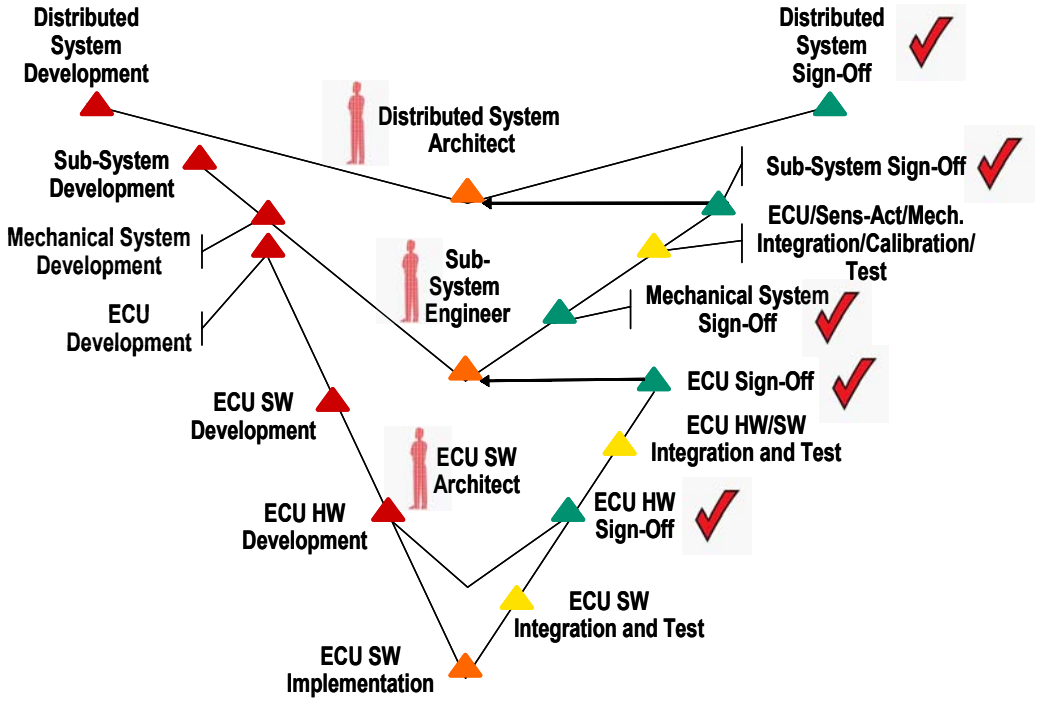


Figure 1: V-Cycles for Distributed Systems

Another important aspect, common to both segments relates to the complex relationships and separation of tasks within the aerospace and automotive supply chains for electronics.
The V-model is a popular formalism used to describe the development process of control software in the automotive domain. In our interpretation of the V-model, as showed in Figure 1, we envision at least 4 V-Cycles! The first top V-Cycle describes the partitioning into sub-systems, their integration into an overall distributed system implementing a complex distributed functionality, and validation of the integration of the distributed system as it is performed by a distributed system architect/integrator (OEM car manufacturer or a Tier1 supplier) – a typical example of such a complex system is an adaptive cruise-control sub-system interacting with a brake-by-wire, and a power train network. The sub-systems components are traditionally provided by the suppliers. At the bottom of the V-Cycle (orange triangle) all sub-systems are available, usually in form of prototyping boards. Next, the distributed system architect (now playing the role of the integrator) can oversee the integration of the sub-systems validating the overall functionality either in the test tracks (if it's safe to do so!) with man in the loop or in the lab with hardware in the loop systems simulating the controlled sub-systems (brakes, engine, etc…) and man in the loop.

[1] Note that in FlexRay the application software makes the final decision in case of critical situations, however, the software developer is using a software API.

The second V-Cycle describes the sub-system development performed by a Tier1 supplier. In this case, the integration test pertains to the validation of the interaction between the electrical/electronic parts (ECUs, sensors, actuators) and the mechanical system being controlled (e.g. the brake). The development usually starts upon reception of functional and non-functional requirements provided by the car manufacturer in form of spreadsheets, and natural language documents.

In the third V-Cycle, a simplified view of the ECU SW development process is described. The HW and SW development are performed in parallel, the SW development realized via emulation boards or instruction set simulators with hardware in the loop systems. Once the HW is available (4th V-Cycle!), HW and SW integration takes place again with hardware in the loop systems simulating, for instance, the engine.

We believe that, the same V-cycle, which in a nutshell, at each stage, pertains to partitioning of functionality and allocation of sub-functions to hardware resources, and verification of the sub-sequent integration, can be used to describe the development process of an integrated system in an aircraft. It is obvious that in an aircraft the sub-systems are computers that are connected together with different redundancy schemes, rather than electronic control units. However, as we will see later in the paper, while many tools support the conventional V-Cycle development process and bring added value when introduced (e.g. automatic code generation), there is a need in both segments to reduce the cost of testing – nowadays estimated as up to 50% of the total development costs. We will describe in the next section how the concept of virtual integration platform can provide the required reduction in testing costs via early fault injection on a virtual prototype.

The ARTIST [7] Project main objective is "Co-ordinate the R&D effort in the area of Advanced Real-time Systems so as to

- Improve awareness of academics and industry in the area, especially about existing innovative results and technologies, standards and regulations
- Define innovative and relevant work directions identify obstacles to scientific and technological progress and propose adequate strategies for circumventing them."

As part of its roadmap, the project focuses on "*Design of Distributed Hard Real-Time Embedded Systems with particular emphasis on Software*" [8]. In [8], a case study of a flight control system is described. In the initial phases of the design, high level system requirements are defined. Dependability issues are obviously part of the design in that fault containment techniques must be defined in order to guarantee the required safety. As described in the case study, this step is carried out via dedicated in-house modeling tools. In a next section of the paper, we will describe how fault injection on a virtual prototype can significantly reduce the cost of expensive tests in labs[2].

In summary, in this paper, we describe how the integration testing, usually taking place at the physical level, should be addressed earlier on a virtual prototype of the system, therefore making early discovery of design errors and, more importantly for safety critical systems, the proof that the system is safe, possible. For safety critical networks, high modeling effort is obviously justified because of safety requirements. This paper is organized as following. In the next section we describe the concept of virtual integration platform. In the section following next, we describe our fault injection and analysis capabilities. Then, we sketch some conclusions.

Virtual Integration Platform

We are continuing the work that has been described in [1]. Few years have past and the VCC technology [1] has taken shape as an automotive focused design environment called Cadence Automotive System Design Platform (hereafter called SysDesign), with foreseeable extensions to aerospace and transportation segments.

The platform is an environment for contract-based design between the (distributed)-system architect and sub-system developer, and the system integrator. The main benefit of the solution is the formalization of the contract between, for instance the OEM system architect and the Tier1 sub-system provider [15]. The environment is based upon the following key points:

- Top-Down constraint refinement from end-to-end constraints (e.g. "the response time from sensor to actuator must be less than 20ms") to finer grain constraints (e.g. SW task level)
- Behavioral Model refinement from coarse grain models (e.g. simple C++ I/O function) to hierarchical and possibly imported models from third party tools such as Matlab/Simulink

[2] The cost for setting up an experiment on a car is about $120-$500 per hour. The time needed to set it up is about 1 hour. The number of tests that can be performed every day is ~2 [14].

- Communication Model refinement from abstract zero-time communications to protocol stack based communication models (e.g. CAN [16]).
- Highly programmable and tunable/configurable architectural models: figures from the implementation world can be back-annotated (e.g. the SW task switching overhead) and/or used to re-assessed a different configuration (e.g. the measured bus traffic figure could be used to change the communication cycle layout of the network bus model). This is taking also into account some possible legacy sub-systems that have been already designed and are re-used within the overall application.
- Design exploration by means of system level profiling and constraint annotation with usage of assertions and constraints to be used:

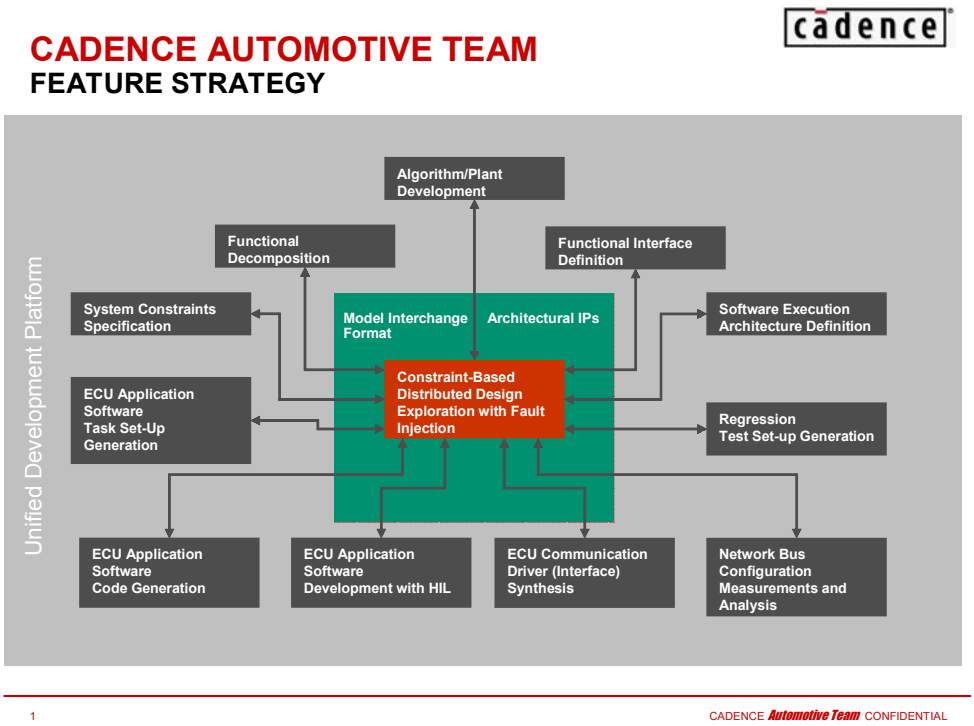


Figure 2

  - Off-Line, as queries on the experiment and results data base (used by the system architect)
  - On-Line, as simulation monitors (used for simulation based verification to check violations during simulation)
  - For Target, meaning to generate non-intrusive monitors for the downstream tools (e.g. SW monitors for the SW developer, bus monitors for the System Integrator performing the bus measurements, etc.)

Therefore, the comprehensive tool strategy for a virtual integration platform includes (Figure 2):

- A linkage with tools for the support of the specification of requirements, constraints and functionality
- A linkage with tools for functional model authoring
- A technology support for contract-based design exploration with provision of Soft Contract [15] capabilities for close loop refinement and interaction between system architect (exploring the system design space) and the ECU software developer and/or system integrator – this is SysDesign.
- A linkage with tools for ECU Application SW generation

- A linkage with tools for ECU Communication SW Layer generation
- A linkage with tools for physical bus configuration, measurement and analysis
- A linkage with tools for regression testing

In [9], the concept of a virtual integration platform is described (Figure 3). In a nutshell, the platform supports a distributed model-based system design process based upon several orthogonal concepts:

- A design shift from physical integration to virtual integration of models
- The extension of a single ECU model-based system design paradigm, in which the transformation from a design representation to its implementation is automatic

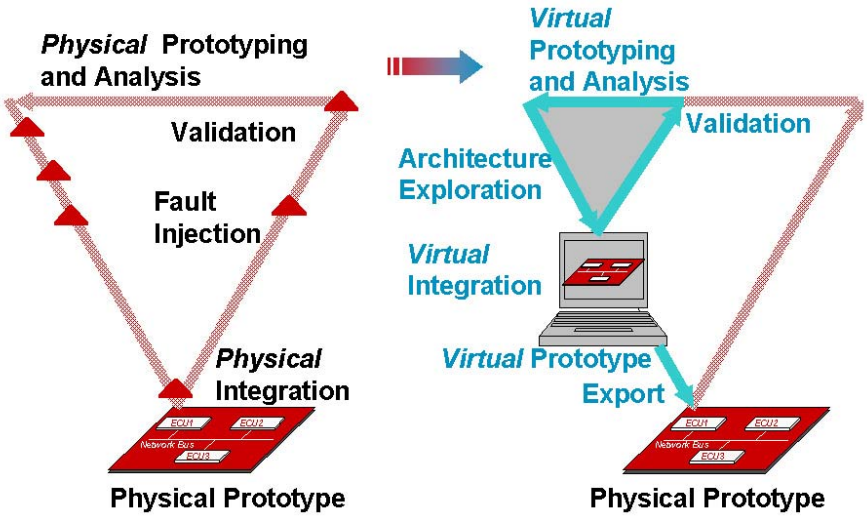


Figure 3: Methodology Shift from Physical to Virtual Integration

- The separation between executable models of control algorithms and the high level/abstract performance/functional models of the architectural resources including the network communication stack
- A binding of control algorithm tasks to HW resources
- A binding of functional abstract communications (signals and shared variables) to network protocol stack models
- The automatic annotation of the control algorithm tasks and communications with timing performance formulae for dynamic computation (at simulation time) of the message transmission latencies over the network bus as well as the SW scheduling execution times due to shared resources (buses, CPUs)
- The simulation of the bound virtual platform (e.g. representing a possible candidate distribution of functionality over the network) for verification purposes under regular conditions and/or with fault injection (e.g. data corruption, abnormal task delay, etc)

Notice that the virtual integration platform relies on the provision of library elements for both functionalities and architectural resources. The platform provides links with the most popular tools for algorithmic development and simulation, thus enabling the seamless import of such models and their composition in the overall distributed complex control function. The platform also provides models of popular communication protocols such as CAN. One important aspect, the XML-based programming capabilities provided by the platform constitutes the core of the linkage between the Cadence Automotive Platform itself and any other environment that utilizes XML-based data dictionaries. Should the schemas be different translations can be performed seamlessly via style sheets.

Simulation Based Fault Injection

Simulation based fault-injection and analysis is key in analyzing system resilience to faults before hardware set-ups are available. Not to mention, fault injection on a virtual prototype enables lower cost and repetitive tests and the system designer to inject faults in any architectural models of the simulated system at any given time, and then observe the performance and signal degradation, via a three-step process:

- Creation of fault models (by instrumenting architectural models)
- Definition of fault scenarios
- Validation of the safety concept of the application via simulation

The following types of fault scenarios can be specified:

- Time Based

At *"Time = 0.5ms"*
Inject Fault *"BusX is Disconnected"*

- State Based with conditional expressions

When *"CpuY is Master"*
Inject Fault *"BusX is Disconnected"*

- Sequence Based (sequence of conditions to be evaluated true before injecting a fault)

After *"VariableX Set To 2 "*
Next *"TaskY is Activated 1 to 3 Times"*
Next *"CpuZ is Master "*
Inject Fault *"BusX is Disconnected"*

Faults can have the following attributes:

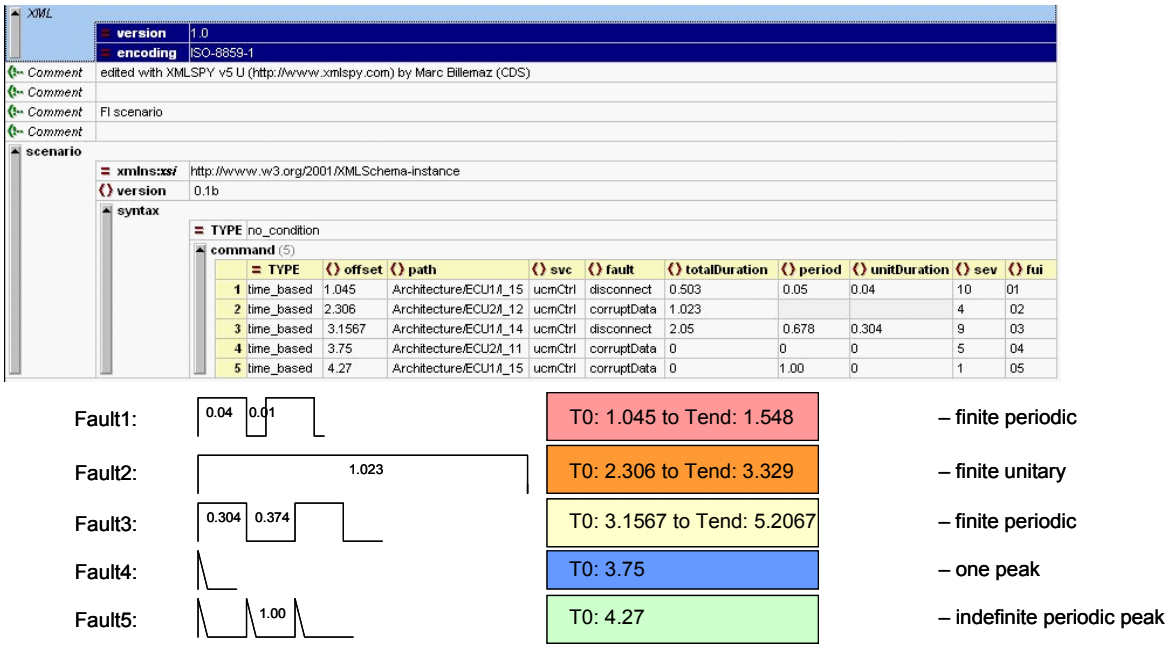
- Fault Type
- Fault Absolute Time
- Fault Duration
- Fault Condition (Optional)
- Resource where the fault occurs (e.g. ECU2/CPU1)

After running a 6 second clock-wall simulation, on a bus redundant ECU architecture for a drive and steer by wire system, the distributed system architect can discover that disconnecting faults have no impact on the signals, while, corrupted data faults have a huge impact, either in periodic or continuous mode, particularly on the brake actuators signals (Figures 4-5).

Conclusions

In this paper, we have described methodology and tool sets that enable the paradigm shift from physical integration testing to virtual model integration. The shift enables early discovery of design errors as well as has the potential to

help designer to provide a first assessment that the application is safe and resilient to catastrophic faults (or at worse prove that it's not!). We do believe that similar techniques and approaches can be applied to other domains such as aerospace, which increasingly share with cutting-edge automotive applications, similar complexities, design challenges and times-to-market.



XML
version 1.0
encoding ISO-8859-1
Comment edited with XMLSPY v5 U (http://www.xmlspy.com) by Marc Billemaz (CDS)
Comment
Comment FI scenario
Comment
scenario
xmlns:xsi http://www.w3.org/2001/XMLSchema-instance
version 0.1b
syntax
TYPE no_condition
command (5)

	TYPE	offset	path	svc	fault	totalDuration	period	unitDuration	sev	fui
1	time_based	1.045	Architecture/ECU1/I_15	ucmCtrl	disconnect	0.503	0.05	0.04	10	01
2	time_based	2.306	Architecture/ECU2/I_12	ucmCtrl	corruptData	1.023			4	02
3	time_based	3.1567	Architecture/ECU1/I_14	ucmCtrl	disconnect	2.05	0.678	0.304	9	03
4	time_based	3.75	Architecture/ECU2/I_11	ucmCtrl	corruptData	0	0	0	5	04
5	time_based	4.27	Architecture/ECU1/I_15	ucmCtrl	corruptData	0	1.00	0	1	05

Figure4: Fault Scenario Definition

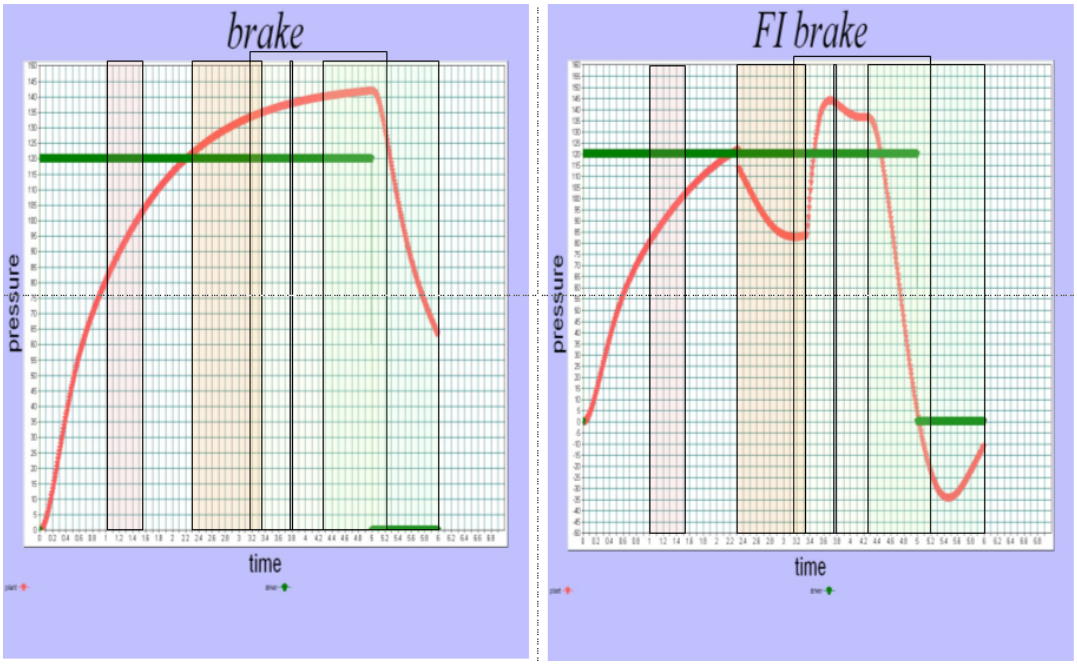


Figure5: Brake Actuator Signals w/o and w/ fault injection

References

[1] P. Schiele, T. Demmeler, E. Fourgeau, P. Giusto. *Rapid Design Exploration for Safety Critical Distributed Automotive Systems*. ERTS 2002, Toulose, January 2002.
[2] OSEK-VDX. *OSEK - Technical report*, http://www-iiit.etec.uni-karlsruhe.de/osek.
[3] AUTOSAR, *Home page*, http://www.autosar.org/find02.php.
[4] TTTECH, *Hamilton Sundstrand's Nord-Micro selects TTP® for Airbus A380 Cabin Pressure Control*, Press Release, http://www.tttech.com/press/docs/pressreleases/PR_2002-07-10-Nord-Micro-TTTech-A380.pdf
[5] PAMELA, *Prospective analysis for modular electronic integration in airborne systems*, http://dbs.cordis.lu/
[6] European Commission: Research: Growth Programme, *Victoria: a new generation of aircraft electronic systems,* Web Page, http://europa.eu.int/comm/research/growth/gcc/projects/in-action-victoria.html
[7] ARTIST, *Advanced Real Time Systems*, Home Page, http://www.artist-embedded.org
[8] ARTIST, *Hard Real-Time Development Environments*, Roadmap, http://www.artist-embedded.org/Roadmaps/A1-roadmap.pdf
[9] Paolo Giusto, Jean-Yves Brunel, Alberto Ferrari, Eliane Fourgeau, Luciano Lavagno, Alberto Sangiovanni Vincentelli. *Virtual Integration Platforms for Automotive Safety Critical Distributed Applications*, Conference International Su les Systemes Temps Reel (RTS), Paris 2003
[10] FLEXRAY Consortium, *Home page*, http://www.flexray-group.com
[11] TTP Forum. *TTP/C specification V0.5*. TTP Forum, 1998
[12] Amberkar, S., D'Ambrosio, J.G., Murray, B.T., Wysocki, J., and Czerny, B.J., *A SystemSafety Process for By-Wire Automotive Systems*, SAE Congress, Detroit, 2002
[13] Juan R. Pimentel, *Designing safety-critical systems: A Convergence of Technologies*, International Workshop on Dependable Embedded Systems, Florence, Italy, 2003
[14] Roland Jutter, *Current Trends in the Design of Automotive Electronic Systems*, DATE Automotive Day, Munich, Germany, 2001
[15] Jean-Yves Brunel, Marco Di Natale, Alberto Ferrari, Paolo Giusto, Luciano Lavagno, *SoftContract: an Assertion-Based Software Development Process that Enables Design-by-Contract*, DATE, Paris, France, 2004
[16] Robert Bosch, *CAN Specification Version 2.0*, Technical Report ISO 11898, Robert Bosch GmbH, 1991