

# HiL Firmware Prototyping Using an COSIDE<sup>®</sup> Asynchronous Motor Model

**Author: Paul Ehrlich**

**Fraunhofer Institute for Integrated Circuits**

This work has been developed in the project Effektiv. Effektiv (reference number: 01IS13022) is funded by the German ministry of education and research (BMBF) within the research programme ICT 2020. The authors are responsible for the content of this publication.



## 1. Introduction

- Effektiv Project
- HiL Intro
- Example System
- Motivation: Why HiL?

## 2. Tooling

## 3. Failure Injection

## 4. Demonstrator

# Research project Effektiv



- Virtual stress tests for intelligent motion control systems
- German BMBF research project
- Fraunhofer as subcontractor of SIEMENS
- Consortium

**Motor**  
Überhitzung durch  
Lagerschaden  
Fehler simuliert ✓

**Steuergerät**  
Ausfall eines  
Chips  
Fehler simuliert ✓

1	5
2	6
3	7
4	8

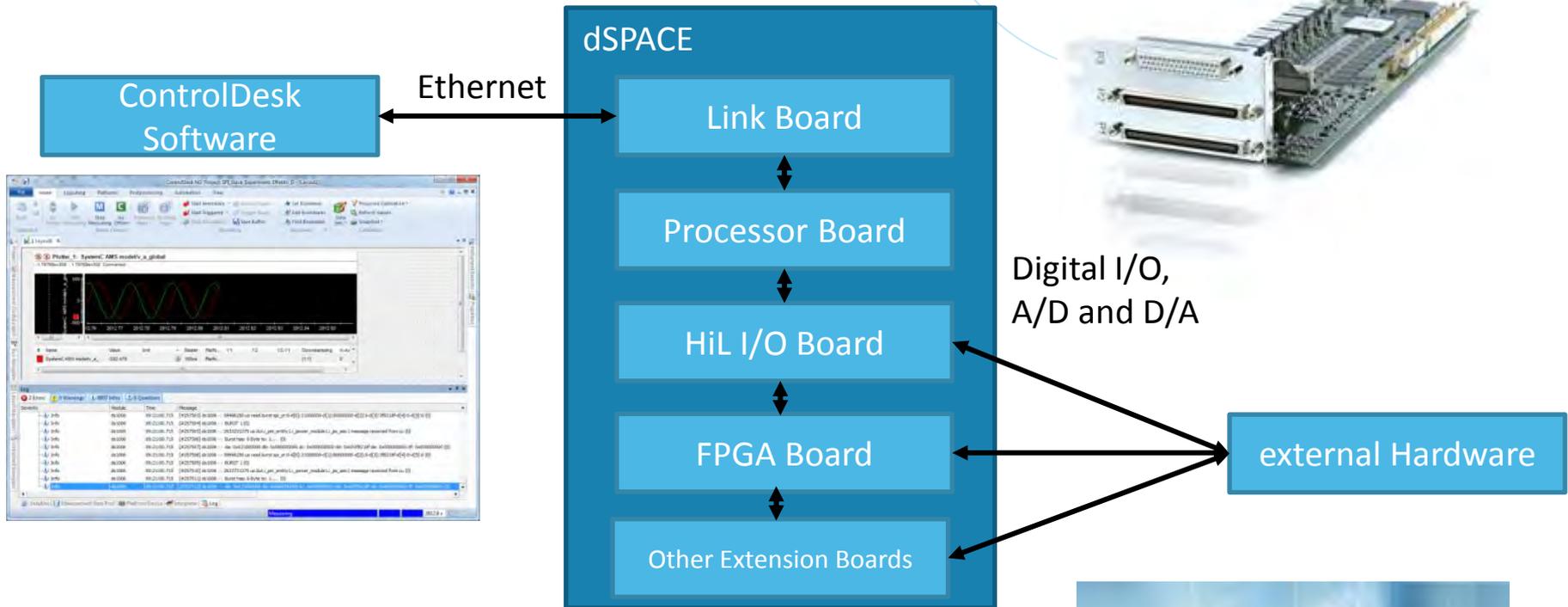
**Sensor**  
Fehlerhafte  
Beschleunigungswerte  
Fehler simuliert ✓

**Software**  
Sicher im Fehlerfall ✓



- Our Focus: Hardware in the Loop Simulations with SystemC AMS

# HiL Introduction



- **Other Extension Boards for specific needs (high speed or precision) or additional interfaces (LIN ... ) available**



Source: dSPACE

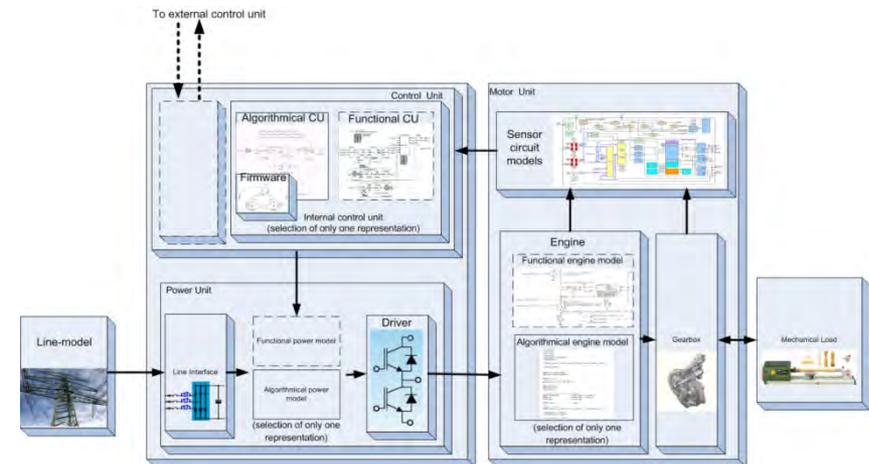
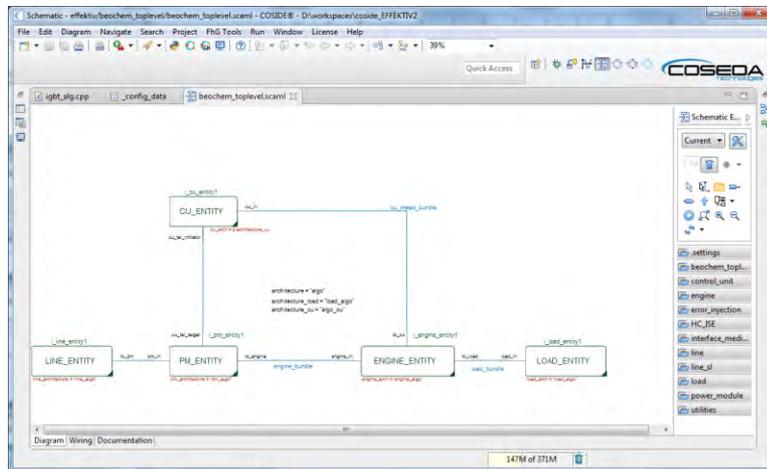
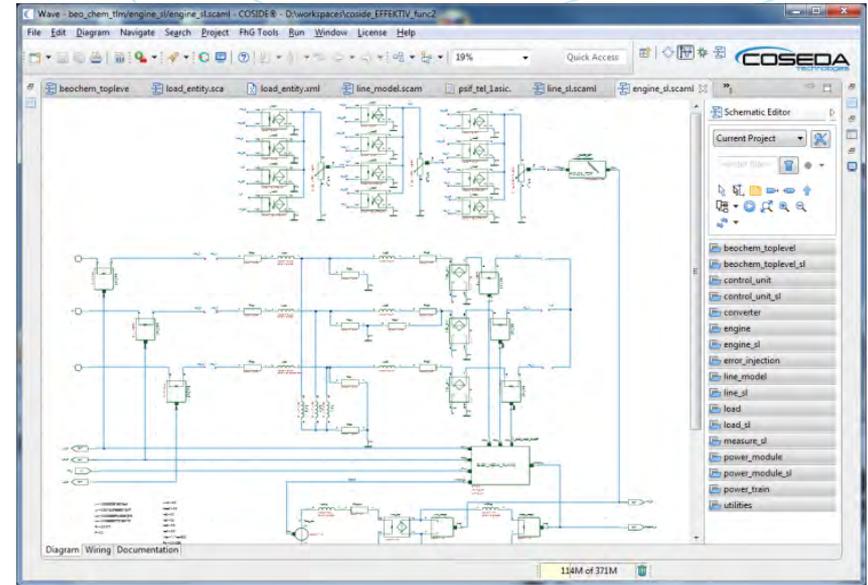
- **Debug** – Accessibility and Traceability from Signals for error root cause analysis
- **Speed/Detail** – Tradeoff between model speed vs detail level
- **Failure Injection** – Possibility to manipulate expected behavior to evaluate system robustness

Pure Virtual	Real Hardware	Mixed (HiL)
<ul style="list-style-type: none"><li>• Debug ++</li><li>• Environment model (+)<ul style="list-style-type: none"><li>- Virtual Sensors</li></ul></li><li>• Speed/Detail --</li><li>• Failure Injection ++</li></ul>	<ul style="list-style-type: none"><li>• Debug --</li><li>• Environment model (+)<ul style="list-style-type: none"><li>- Hardware Sensor</li></ul></li><li>• Speed/Detail ++</li><li>• Failure Injection --</li></ul>	<ul style="list-style-type: none"><li>• Debug +</li><li>• Environment model (++)<ul style="list-style-type: none"><li>• Both possible</li></ul></li><li>• Speed/Detail +</li><li>• Failure Injection +</li></ul>

# Example System

## ■ Motion-Control-System

- Control Unit (CU)
- Power Module (PM)
- Engine
- Load
- Line-Module



- **Task: Firmware development on the Control Unit (CU) for an asynchronous Motor**
- **Setup:**
  - CU as prototypes or hardware revision
  - SystemC AMS of PM, engine, load, line model running on dSPACE
- **Advantages:**
  - Realistic Firmware environment for CU through the use of real hardware
  - Use of proven failure injection models within SystemC AMS
    - Enables also the exploration of critical corner cases, which would destroy the real hardware

## 1. Introduction

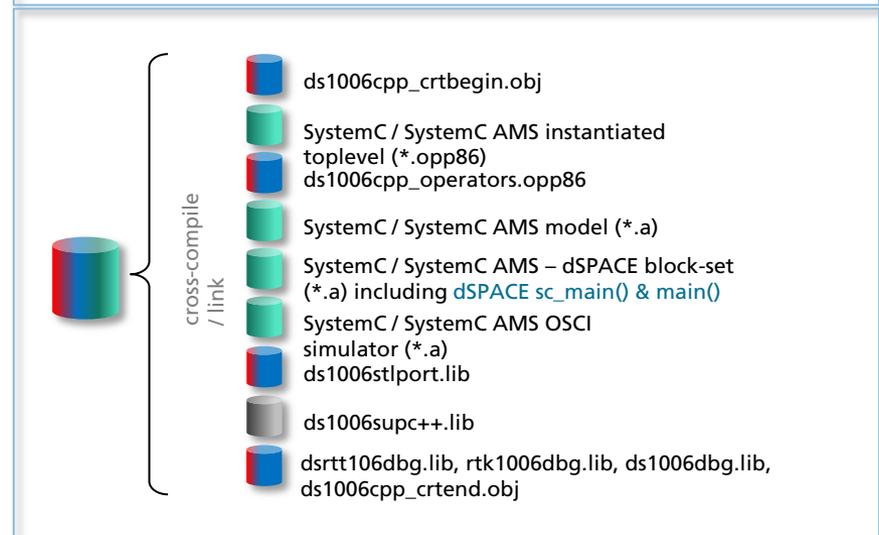
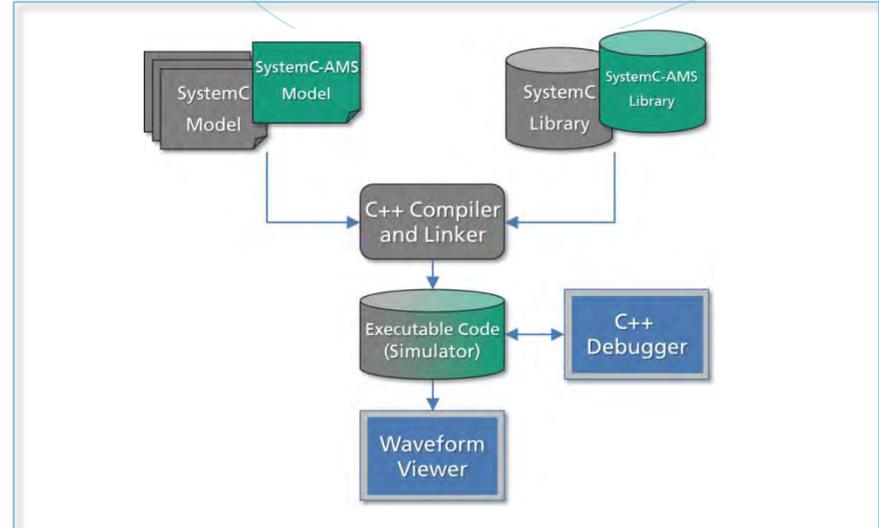
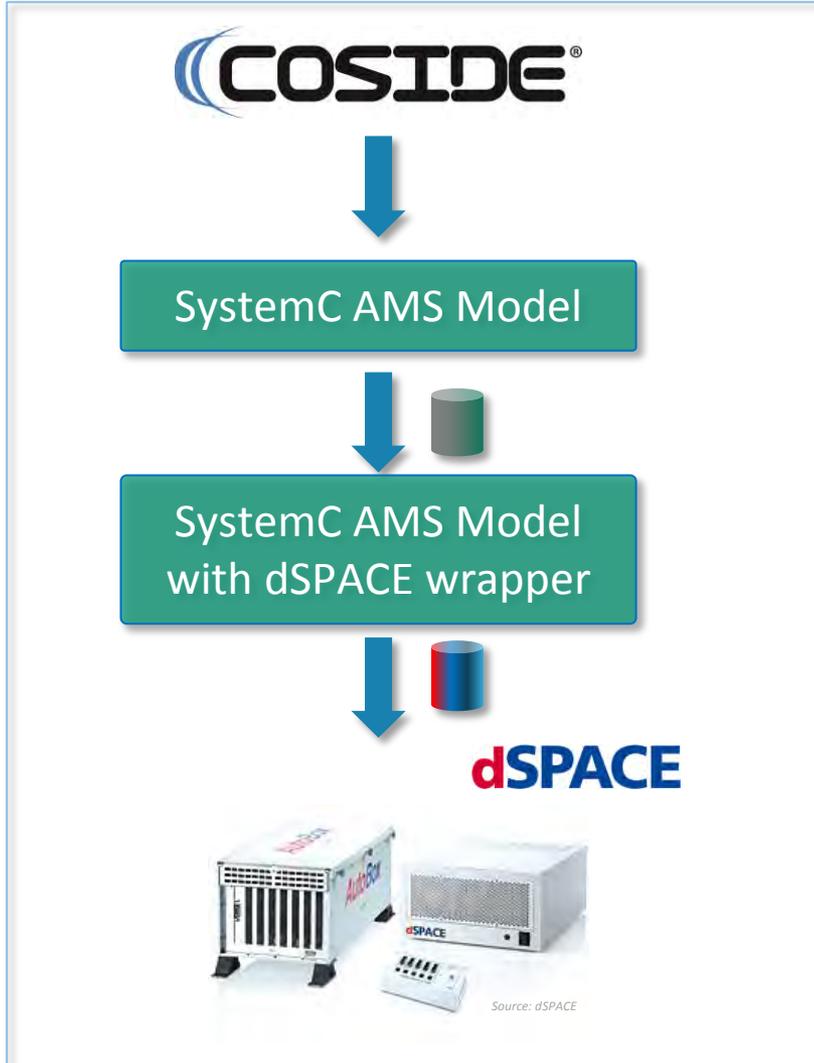
## 2. Tooling

- SystemC AMS on dSPACE
- Tooling Environment

## 3. Failure Injection

## 4. Demonstrator

# SystemC-AMS Executable



# Development Environment

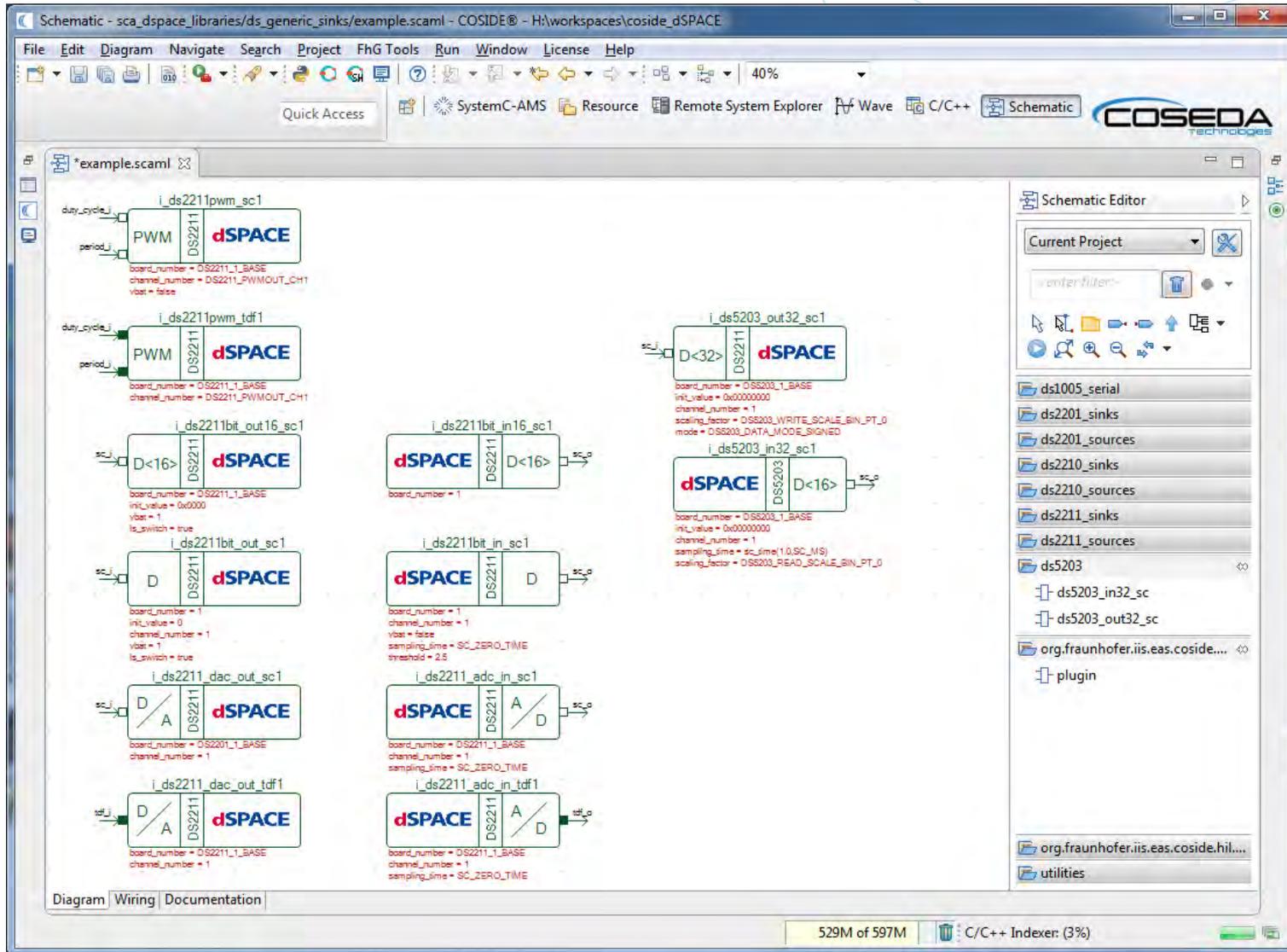


The screenshot displays the COSIDE development environment. The top window shows a schematic diagram of a control unit with entities: LINE\_ENTITY, PM\_ENTITY, ENGINE\_ENTITY, and LOAD\_ENTITY, all connected to a central CU\_ENTITY. The bottom window shows the ControlDesk NG Project interface, including a plotter displaying a sine wave graph and a console window with log messages.

The screenshot displays the COSIDE development environment. The top window shows a schematic diagram of a control unit with entities: LINE\_ENTITY, PM\_ENTITY, ENGINE\_ENTITY, and LOAD\_ENTITY, all connected to a central CU\_ENTITY. A context menu is open over the diagram, showing options like 'New', 'Open', 'Copy', and 'Paste'. The 'dSPACE' menu item is highlighted, showing sub-options like 'DS1005 create', 'DS1005 compile', and 'DS1006 create and download'.

- COSIDE® model entry
- dSPACE specific Menus
- ControlDesk Waveforms & Console

# Interface Library



1. Introduction
2. Tooling
3. **Failure Injection**
4. Demonstrator

## ■ Properties

- Non intrusive failure injection, without model changes
  - Failure cases are connected into the netlist dynamically during runtime
  - Thread based for independent de/-activation
- Failure scenario are build up hierarchically
  - Reusability (e.g. failure structures and scenarios)
  - Unified interfaces (for e.g. initialization, de/-activation)
- Error location and time can be scattered statistically
  - Use of extensive statistic library

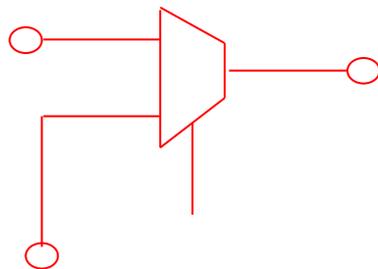
## ■ Hierarchical test structure

- Testbench
- Testcase/Stimuli without failure injection ← includes additional fault stimuli

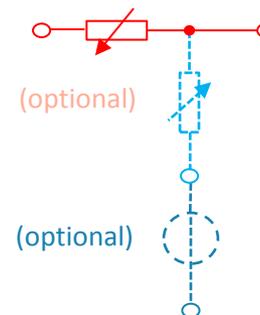
## ■ Extended hierarchical test structure with failure injection

- Failure stimuli (describes location and time behavior)
- Fault scenario – combination of Fault models e.g. SPI
- Fault models – e.g. Stuck at, Open-Short, Cross Talk ...
- Low-Level-Fault models (e.g. SC/TDF: MUX, ELN: resistor...)

For sc/ tdf connections



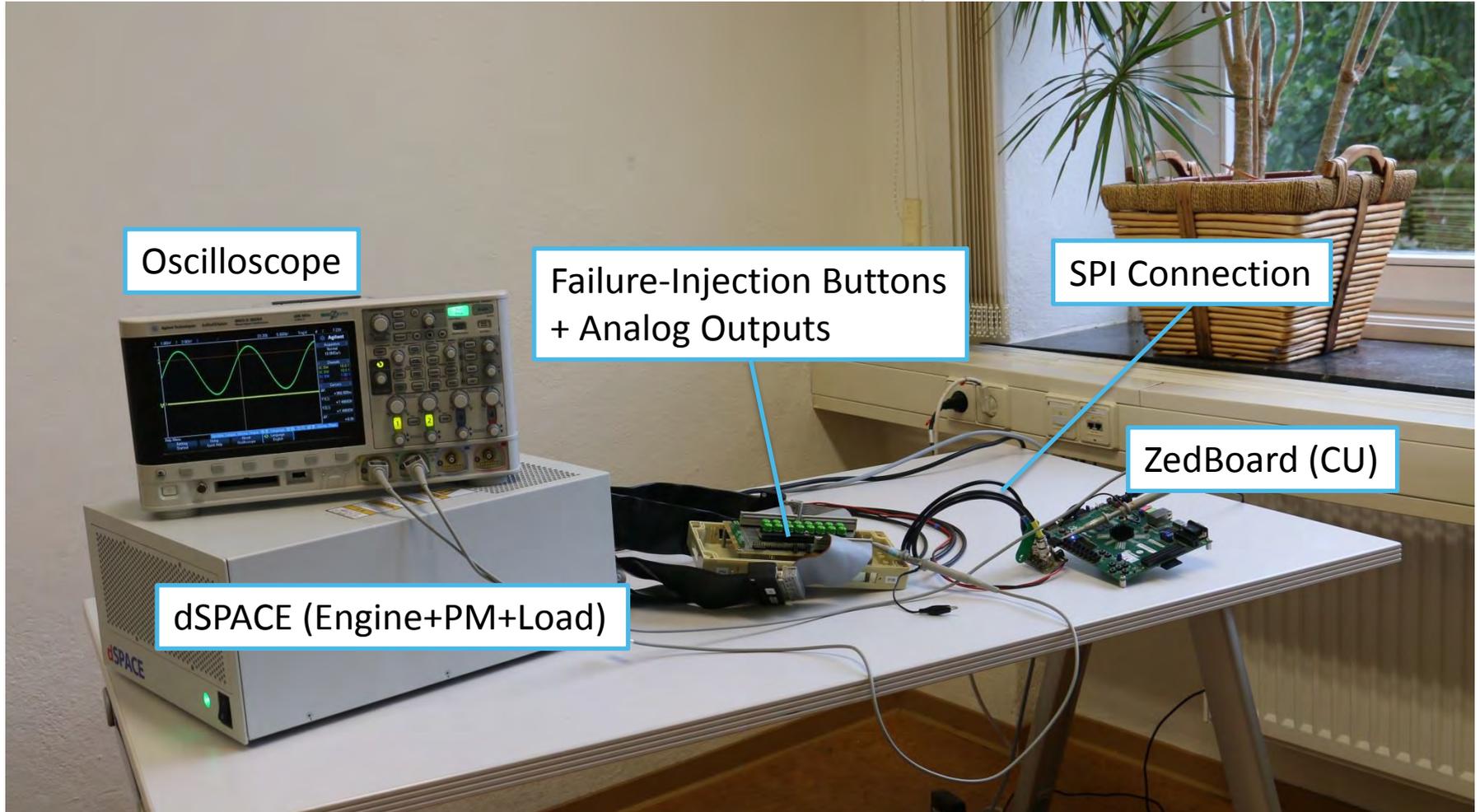
for ELN networks



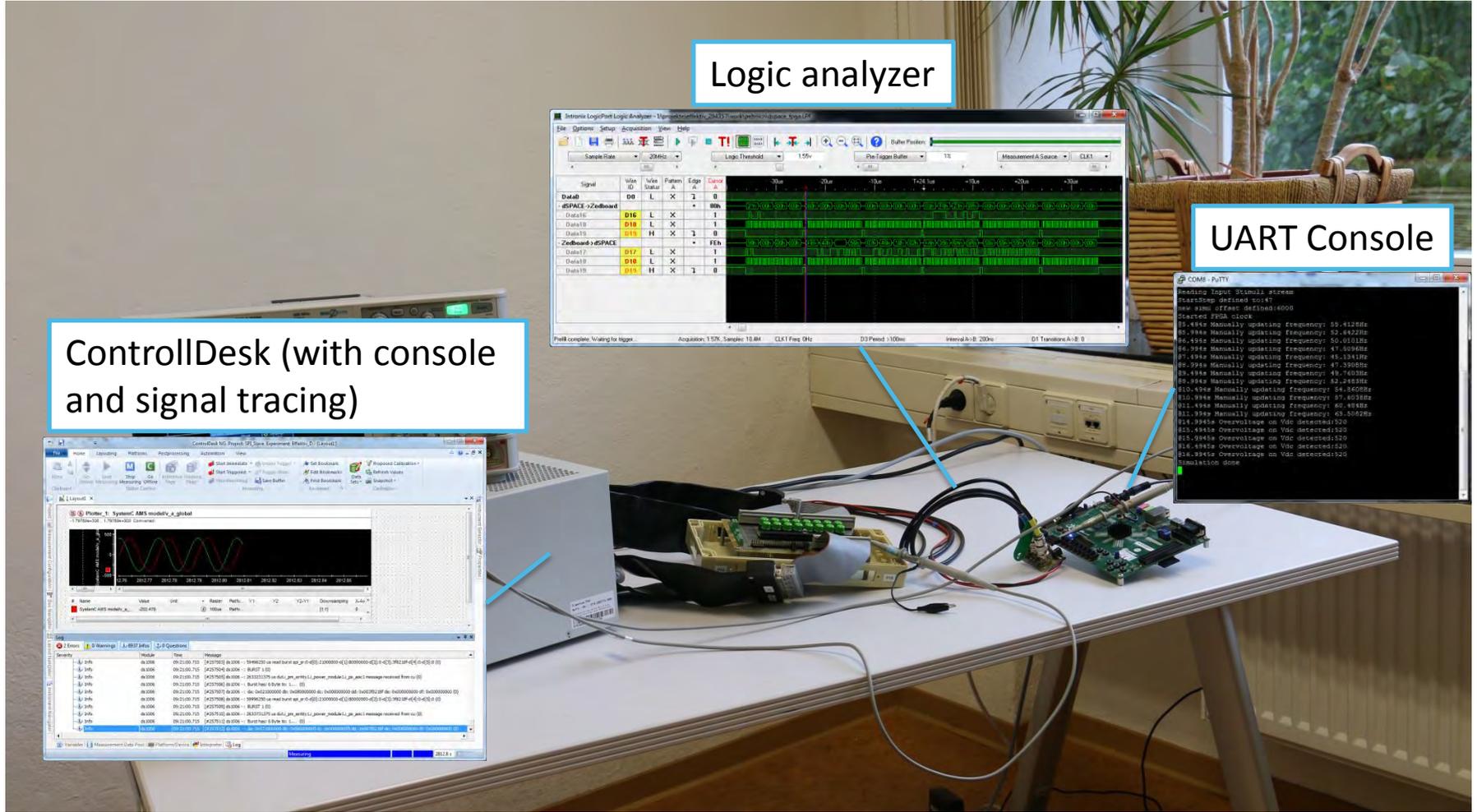
# Agenda

1. Introduction
2. Tooling
3. Failure Injection
4. **Demonstrator**

# Prototyp Demonstrator



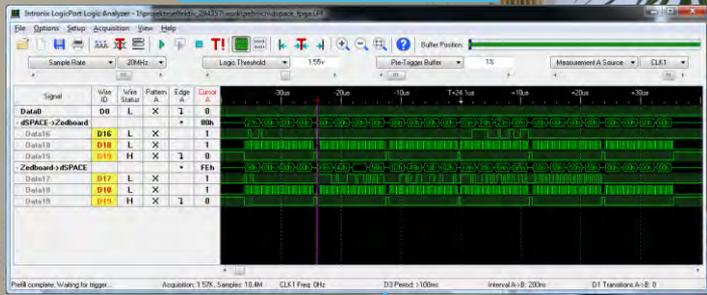
# Debug interfaces



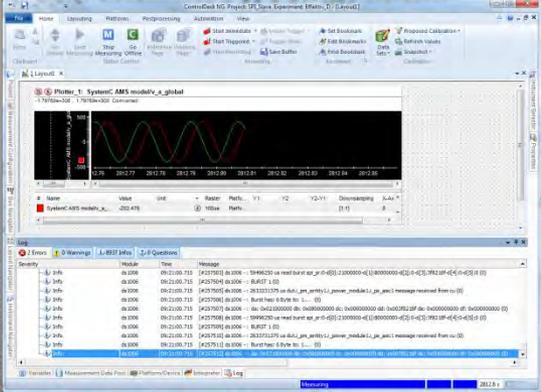
Logic analyzer

UART Console

ControllDesk (with console and signal tracing)



```
COM1 - PUTTY
Reading Input Stimuli stream
Keyboard defined port 47
New ANSI escape definition:ESC
Scanned FPGA clock
05.4944s Manually updating frequency: 50.4125MHz
05.4944s Manually updating frequency: 52.4422MHz
06.4944s Manually updating frequency: 50.4011MHz
06.4944s Manually updating frequency: 47.5036MHz
07.4944s Manually updating frequency: 45.1341MHz
08.4944s Manually updating frequency: 47.3005MHz
09.4944s Manually updating frequency: 46.7403MHz
09.4944s Manually updating frequency: 49.2463MHz
10.4944s Manually updating frequency: 54.5059MHz
11.4944s Manually updating frequency: 40.4944MHz
11.4944s Manually updating frequency: 61.6042MHz
11.4944s Overvoltage on VDD detected:520
12.4944s Overvoltage on VDD detected:520
13.4944s Overvoltage on VDD detected:520
14.4944s Overvoltage on VDD detected:520
15.4944s Overvoltage on VDD detected:520
Simulation done
```



# Acknowledgement



The presented work has been partially supported by the German Federal Ministry of Education and Research (BMBF) under the grant 01IS13022





■ Thank you for your attention!



 **Fraunhofer**  
IIS

**SYSTEM C**  
**AMS**



**Paul Ehrlich**

- Heterogeneous System Specification
- Paul.Ehrlich@eas.iis.fraunhofer.de
- +49-351-4640-756