ASAM | Association for standardisation of automation and measuring systems

Collaborating in California:
Open HIL Test System Architecture uses the ASAM HIL API

ASAM INTERNATIONAL CONFERENCE, Dec. 03 – 04, 2013, Dresden, Germany

Dr. Jiri Keprt, National Instruments Corporation (Speaker)
Dr. Rainer Rasche, dSPACE GmbH
Paul Liu, Tula Technology, Inc.

NATIONAL INSTRUMENTS TULA dSPACE

Preface

HIL test systems enable the implementation of effective test automation strategies. Thereby, one of the most challenging tasks is the reuse of existing test system components, as well as existing test cases inside of a project or even over project boundaries. To achieve this, open standardized interfaces for accessing the functionality of test system components are essential. This openness and standardization guaranties on one hand the reusability of existing test components and allows on the other hand the implementation of an effective modular test automation architecture with the ability to easily replace or improve test components in the future if needed.

The ASAM HIL API addresses the standardization needs in the area of HIL Test systems and provides standardized interfaces for accessing the components of a HIL Test system.

This presentation gives a report from Silicon Valley, California and shows how HIL API is at work at Tula Technology Inc. Through a combination of a unique application of digital signal processing and sophisticated powertrain controls, Tula has created the digital engine, thereby providing a cost-effective and easily integrated fuel-reduction technology. Therefore, Tula uses hardware components from dSPACE and National Instruments at a test bench accessed by test cases using HIL API.

Another part of the presentation emphasizes the cross tests performed recently in 2012 and early 2013 by major HIL System vendors und thus, shows a systematic way to get components from different suppliers into operation. It also discusses the functionality of the standard for reading, writing, capturing and stimulating variables of a HIL-Simulator or simulation model, which makes HIL API ready to use for a huge number of HIL testing applications. Since there is not much to do apart from adopting proprietary pre-conditions and configurations, when switching the vendor of a simulation platform, existing test cases can be completely reused. HIL API 1.0.2 also supports different development stages (e. g. non-real-time/real-time simulation), if functionality is provided by the simulation platform supplier.

To sum it up: HIL API is the right choice for engineers who would like to reuse their existing tests and rely on a state-of-the-art HIL standard that enables a better know-how transfer from one test bench to the other, resulting in reduced training costs for employees as well.

Contribution to ASAM International Conference,
Dec. 3.–4., 2013, Dresden

Jiri Keprt, National Instruments GmbH; Munich, Germany (Speaker)
Paul Liu, Tula Technology Inc., San Jose, California USA
Dr. Rainer Rasche, dSPACE GmbH Paderborn, Germany

NATIONAL INSTRUMENTS TULA dSPACE

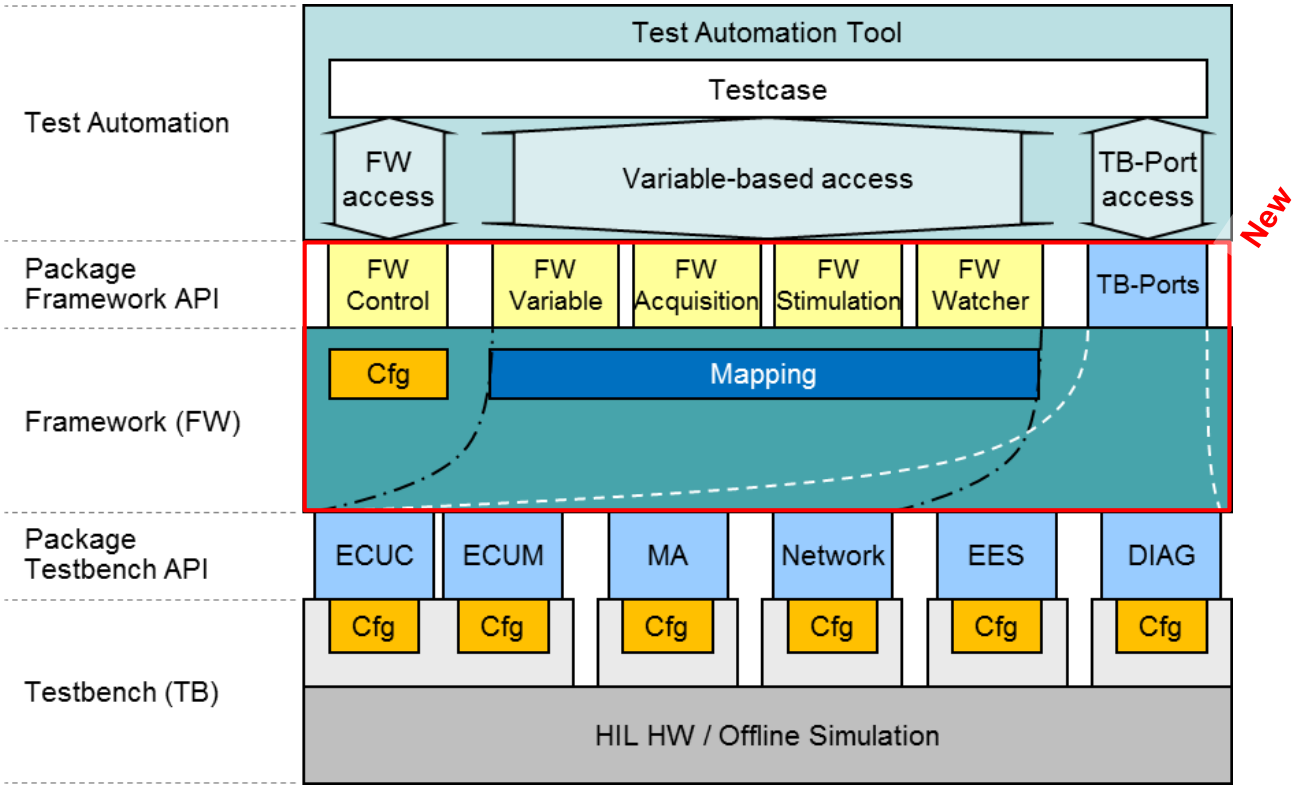
Agenda

1	**ASAM HIL API Motivation**
2	Functionality covered by the ASAM HIL API / Status of the HIL API 1.0.2
3	Tula Technology – Live application example
4	Cross tests
5	Summary

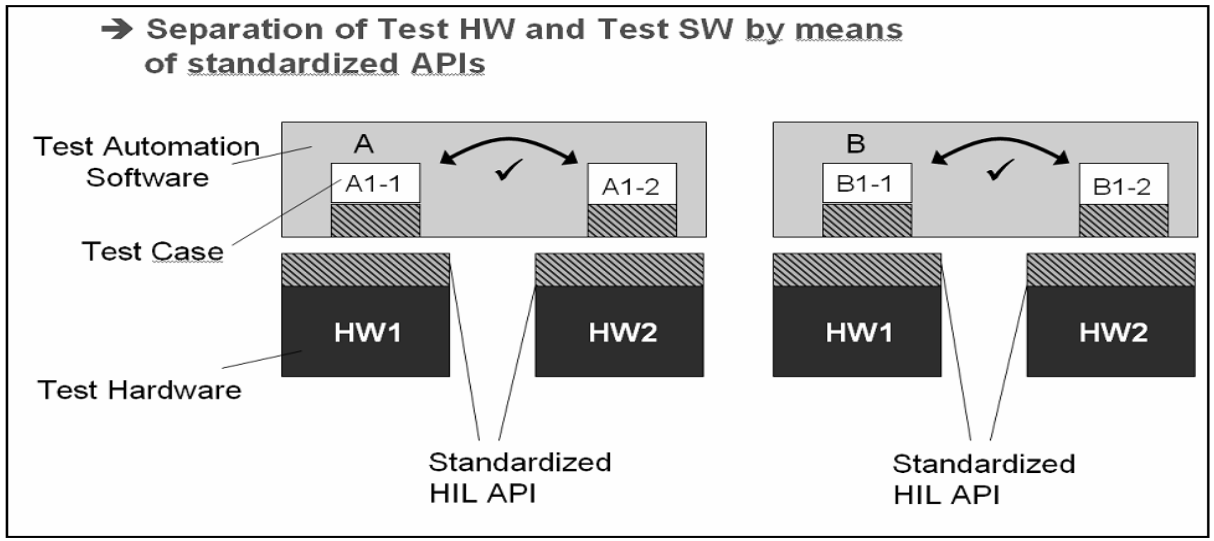
ASAM HIL API Motivation

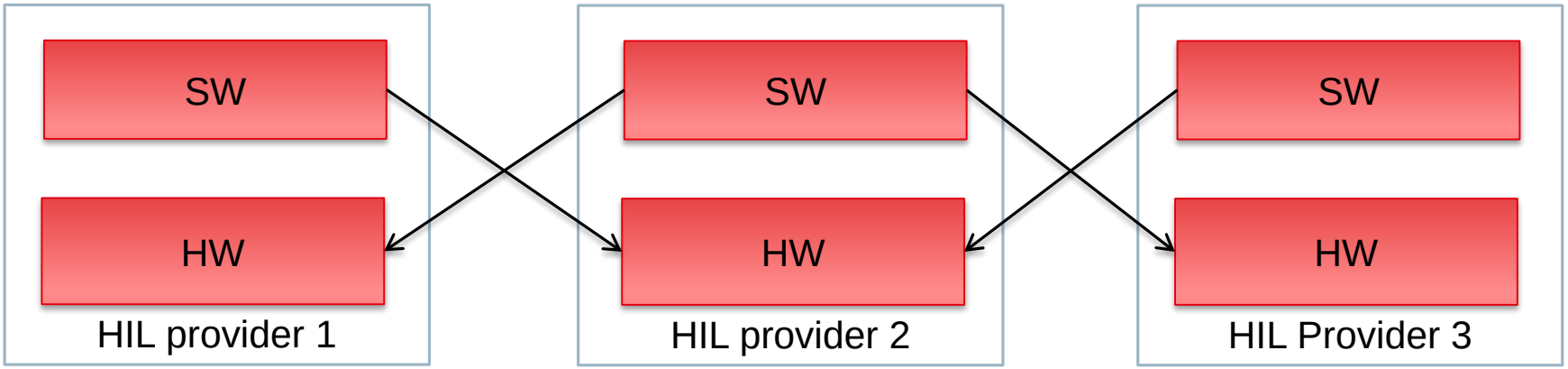
- The goal of the ASAM HIL API standardization effort is to enable the reuse of test cases and to decouple test automation software from test hardware. This leads to:
  - Re-use of automated tests on test systems from different vendors
  - Reduced training costs for employees
  - Improved know-how transfer from one test bench to another
- Suitable for all kinds of tests in HIL simulation, e.g. drivetrain, steering, electric lighting tests, etc

Framework-based Access (with XIL 2.0.0)



ASAM HIL API Motivation





Agenda

1	ASAM HIL API Motivation
2	Functionality covered by the ASAM HIL API / Status of the HIL API 1.0.2
3	Tula Technology – Live application example
4	Cross tests
5	Summary

Functionality covered by the ASAM HIL API

- The ASAM HIL API comprises access to the following HIL simulation components:
  - Reading/writing parameters in simulation models (Model access)
  - Capturing/generating signals in simulation models (Model access)
  - Capturing, reading and calibrating ECU variables (ECU access)
  - Exchanging data with an ECU via diagnostic services (ECU Diagnostics access)
  - Controlling electrical error simulation hardware (e.g. to set up short circuits) (Electrical Failure Insertion)

Functionality covered by the ASAM HIL API

- Those APIs are called *Ports*

Port	description
MAPort	The Model Access port provides access to the simulation model. It is possible to read and to write parameters and to capture and to generate signals.
DiagPort	The Diagnostic port communicates with a diagnostic system to read data via diagnostic services from an ECU or Functional Group.
EESPort	The EES port controls electrical error simulation hardware. It allows setting different types of errors.
ECUPort	The ECU ports communicate with an MC system and thus provide access to ECU internal values. The ECU M port allows to capture and to read measurement variables. The ECU C port is used for calibration.

Status of the HIL API 1.0.x

- Latest Release Version: 1.0.2 (29 Feb 2012)
- The deliverable of ASAM AE HIL Version 1.0.2 includes:
  - Programmers Guide
  - Generic UML Model
  - Templates (XML schema files)
  - Technology Reference Interfaces for Python, JAVA and C# including:
    - Sample Code Examples for Client Side
    - „Dummy“ Implementation for Server Side

Agenda

1	ASAM HIL API Motivation
2	Functionality covered by the ASAM HIL API / Status of the HIL API 1.0.2
3	Tula Technology – Live application example
4	Cross tests
5	Summary

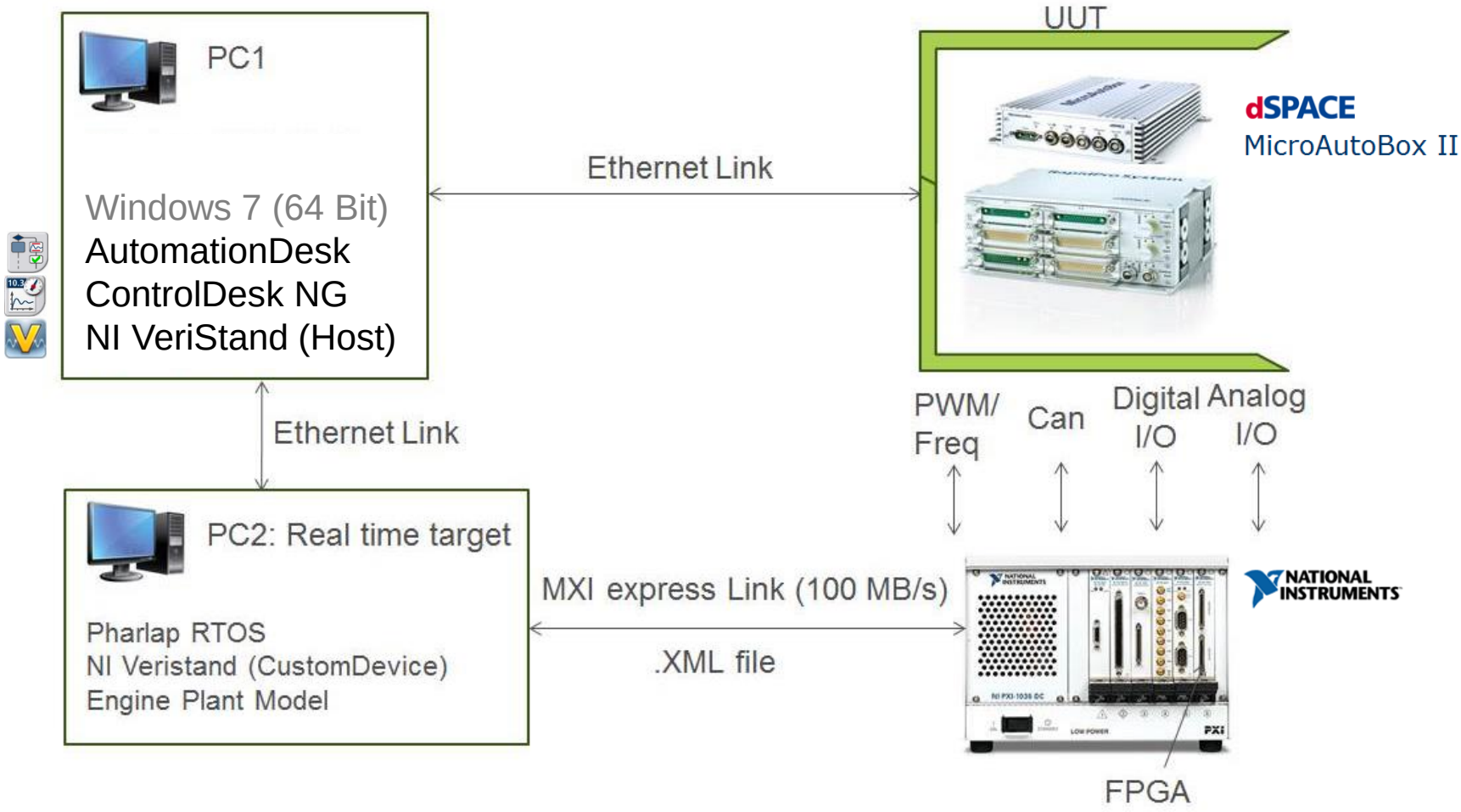
HIL Bench

Requirements

- The Tula HIL test bench shall:
  - Support full automated testing.
    - Creation of test scripts
    - Capture of test results
    - Auto-generated test reports with pass/fail evaluation
  - Provide a full test harness around the control system under test.
    - Measure outputs from UUT and simulate sensor inputs.
    - Read/Write access to all simulation model variables on MicroAutoBox II.
  - Support a real-time engine plant model.
  - Acquire and interpret signals at 1ms.
    - Fastest signals logged from MicroAutoBox and VeriStand are at 1000Hz rate

Code Verification

HIL Bench Setup



Evaluation and Reporting

Evaluation using Model Access Port (Read)

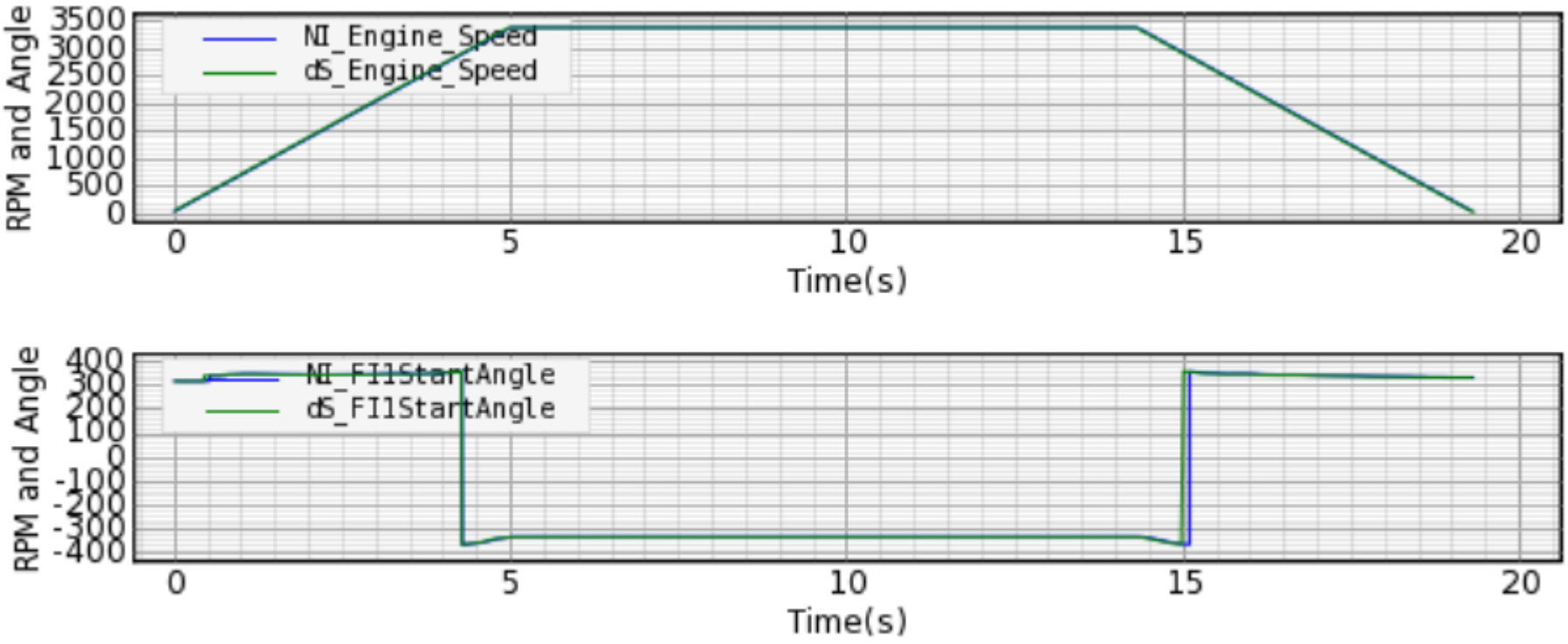
9.2.1.1.2.4 TestStep4_Evaluate : Test Builder.Step

Parameter	Value
Tolerance	3.0
ReferencePool	<tam_mainlibrary.DSDataContainer object at 0x13FFF750>

	dSPACE	NI	Difference	Result
FI1StartAngle	-329.279058434	-328.409940395	-0.869118039346	Passed
FI1EndAngle	350.834362838	351.60581977	-0.771456931943	Passed
FI2StartAngle	-329.342359675	-328.507196128	-0.835163547279	Passed
FI2EndAngle	350.833159246	351.669633244	-0.836473997963	Passed
FI3StartAngle	-328.692197107	-327.792601496	-0.899595610416	Passed
FI3EndAngle	350.833131703	351.638774485	-0.805642781521	Passed
FI4StartAngle	-329.939010087	-328.99160333	-0.947406757318	Passed
FI4EndAngle	350.828707055	351.673768798	-0.84506174309	Passed
FI5StartAngle	-329.675011708	-328.809208705	-0.865803003195	Passed
FI5EndAngle	350.829048269	351.614902246	-0.785853976987	Passed
FI6StartAngle	-329.555925737	-328.606967356	-0.948958381014	Passed
FI6EndAngle	350.830694481	351.682098941	-0.85140446022	Passed
FI7StartAngle	-329.263470327	-328.392736206	-0.870734120204	Passed
FI7EndAngle	350.829424683	351.606707174	-0.777282491656	Passed
FI8StartAngle	-329.557274271	-328.593276106	-0.963998165476	Passed
FI8EndAngle	350.834443272	351.6871219	-0.852678627226	Passed
IC1EndAngle	33.2406617822	34.1628067283	-0.922144946065	Passed
IC2EndAngle	33.2420631727	34.1613333879	-0.919270215155	Passed
IC3EndAngle	33.2397051448	34.1620883979	-0.922383253118	Passed
IC4EndAngle	33.2455775628	34.1623701882	-0.916792625383	Passed
IC5EndAngle	33.2460714371	34.1626913351	-0.916619898013	Passed
IC6EndAngle	33.240943928	34.1620320399	-0.921088111894	Passed
IC7EndAngle	33.2431546245	34.1622037911	-0.919049166562	Passed
IC8EndAngle	33.242840529	34.1619756818	-0.919135152794	Passed

Plotting of Captured Data

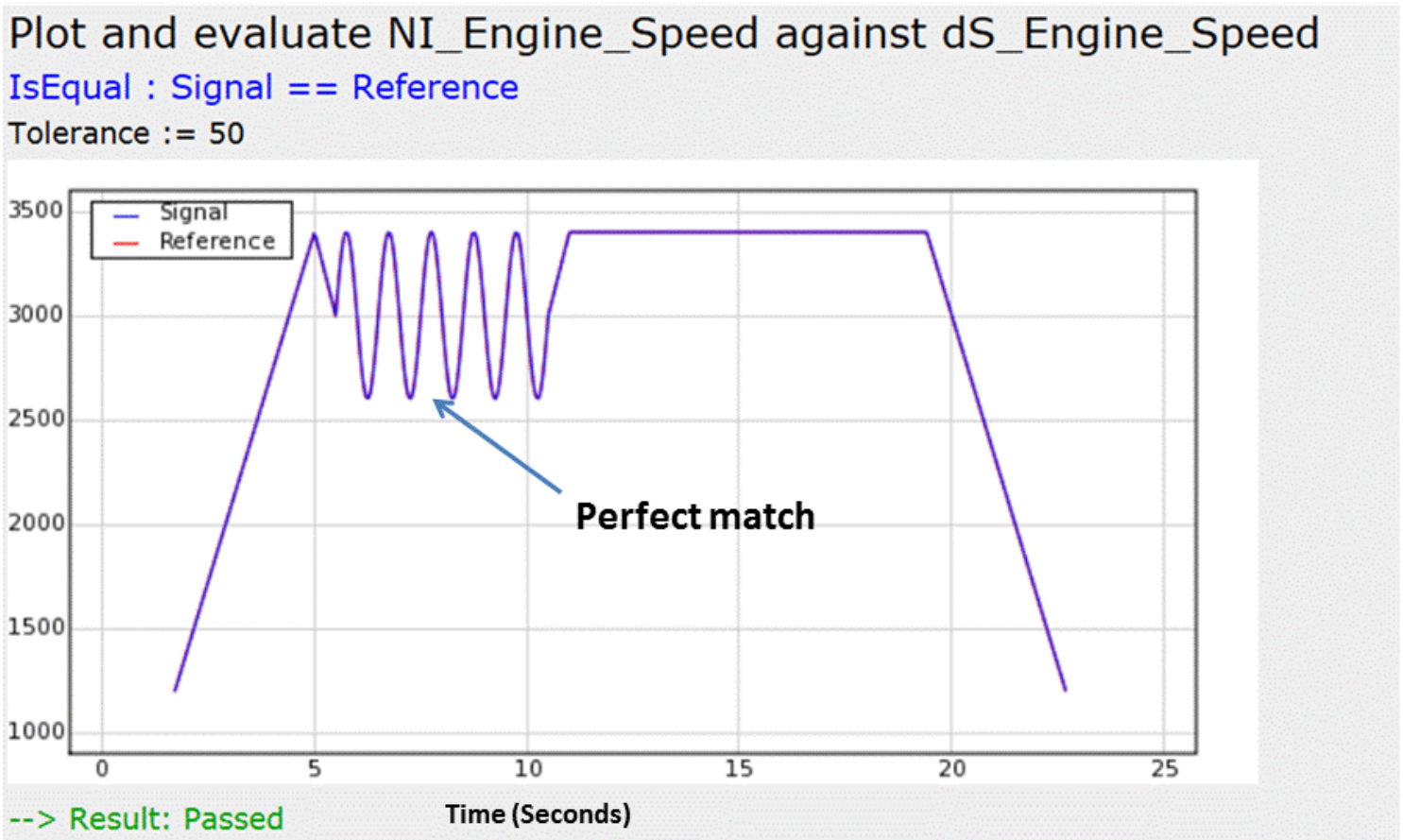
Model Access Port Capture (NI vs dSPACE data)



Plot and Evaluate Capture

Pass / Fail (NI vs dSPACE data)

Signal: Engine speed produced by NI Veristand Custom device
Reference: Engine speed measured by dSPACE Rapid Pro/ MicroAutoBox

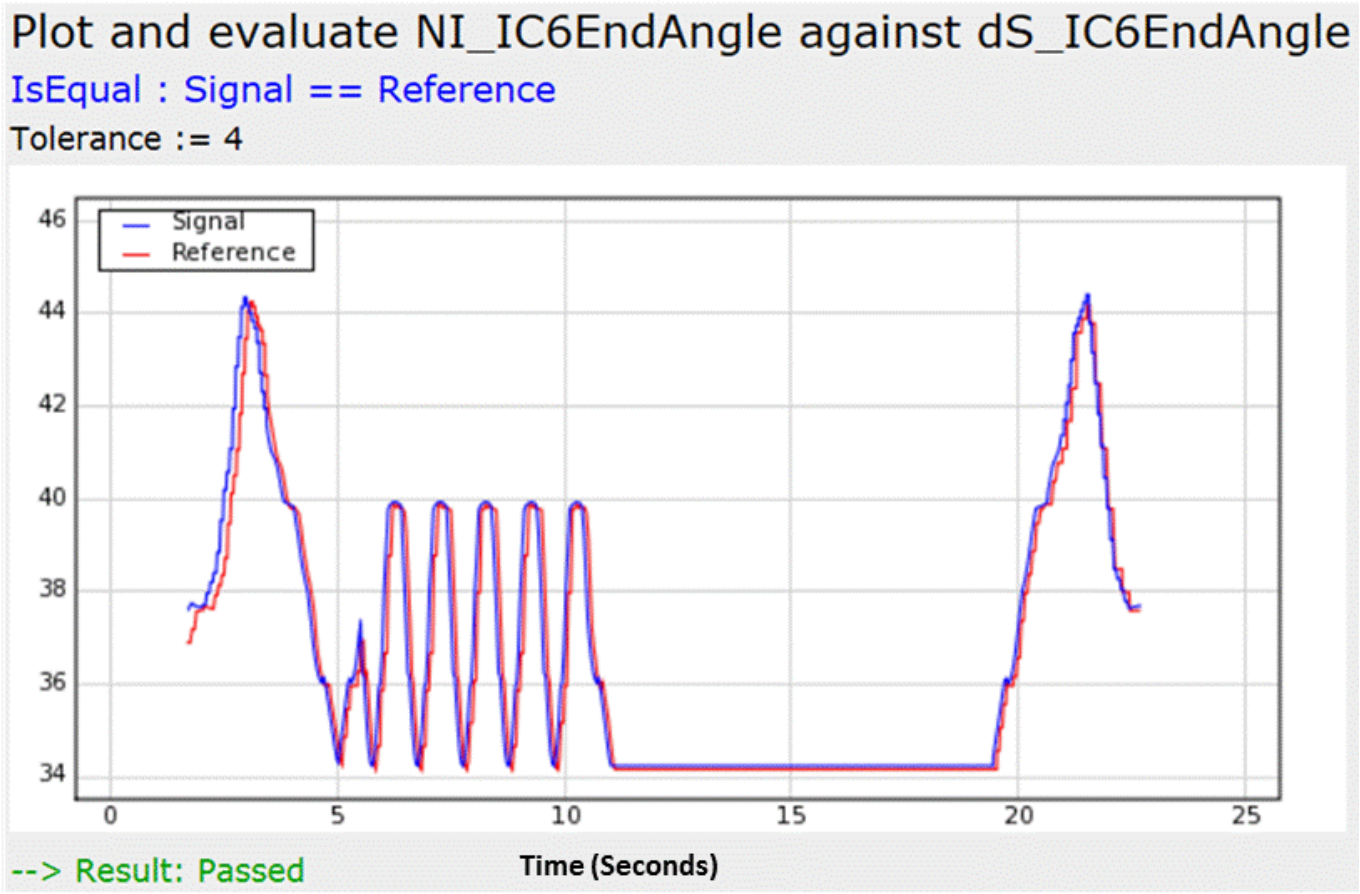


Plot and Evaluate Capture

Pass / Fail (NI vs dSPACE data)

Signal: Ignition coil angle produced by dSPACE Rapid Pro/ Microautobox

Reference: Ignition coil angle measured by NI Veristand



Agenda

1	ASAM HIL API Motivation
2	Functionality covered by the ASAM HIL API / Status of the HIL API 1.0.2
3	Tula Technology – Live application example
4	Cross tests
5	Summary

Cross Test I & II

- Location & date:
  - dSPACE, Paderborn, Germany – June 2012 (CT I)
  - Vector, Stuttgart, Germany – February 2013 (CT II)
- Companies: dSPACE, ETAS, Vector, Tracetronic (CT II only), NI
- Standard version: ASAM HIL API Version 1.0.2
- Goal: Test the interoperability between test automation tools and HIL-Systems of different vendors using the ASAM HIL API standard and check the proper functioning of the ASAM AE HIL version 1.0.2 interface
- Tested functionality: Model Access Port
  - Basic access functions: read/write model variables
  - Capturing / Logging
  - Signal Generator (Stimulus)
- 49 pre-defined test cases

Cross Test

- Terms:
  - Server = Server application (HIL-Simulator) of a vendor implementing a Port (Model Access Port) of the HIL API
  - Client = Client application (e.g. Test automation) using HIL API function calls to talk to a server (HIL Simulator)
- Two stage test procedure
- Stage 1: testing the implementation of an HIL API port using a common “lightweight” client (.NET 3.5 und .NET 4.0)
- Stage 2: testing the interoperability of real-world clients with the implementation of an HIL API port
- Vendor specific configuration through separate .NET dll
- Using reflection and XML for configuration

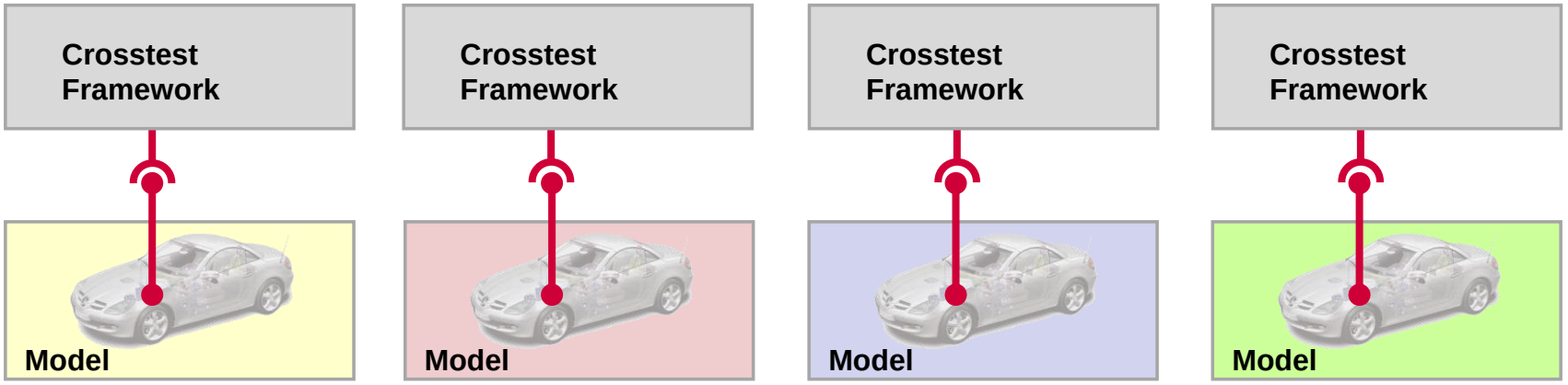
Cross Test – Typical test run

1. Test case initialization
2. Configuration-Dictionary for MAPort populated
3. MAPort instantiated
4. SignalGenerator instantiated
5. SignalGeneratorReader instantiated
6. Data read with Reader help
7. Data from HW read
8. Stimulation started

Red: Vendor specific Blue: HIL API 1.0.2 Standard Methods

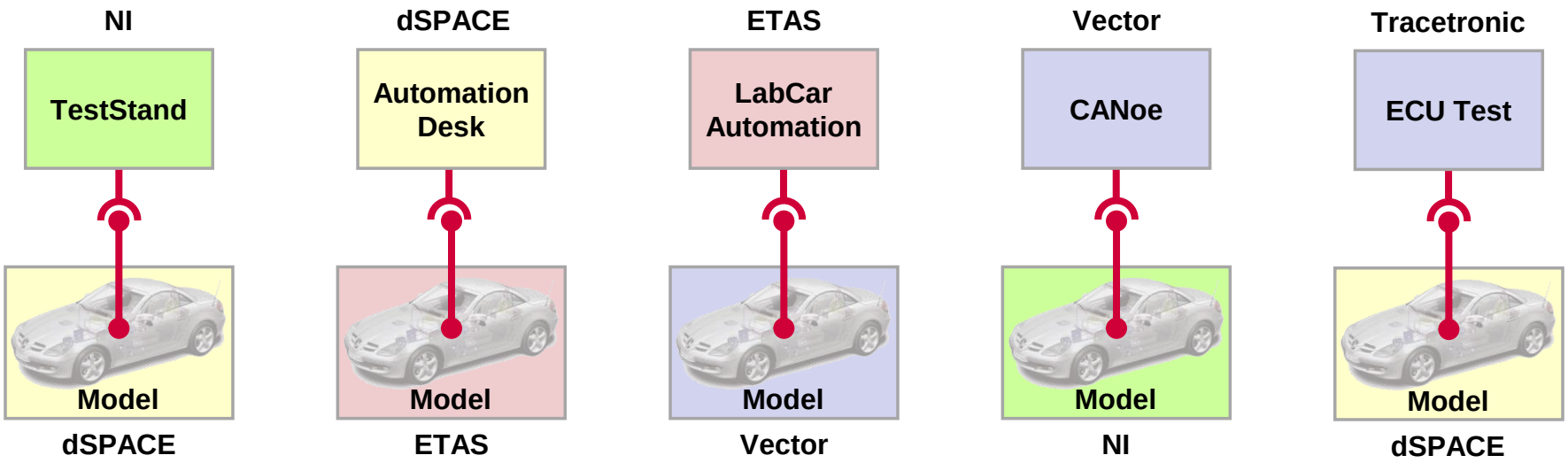
Cross Test

- Test components:
  - Matlab/Simulink model of a throttle valve controller
  - C# Test Framework (“lightweight” client)
  - 49 pre-defined test cases implemented in C#
- All test components were provided before the cross test
- Each participant were asked to implement the vendor specific part of the test Framework and check if the pre-defined test cases could be executed successfully



Cross Test – Test Framework

- Pre-defined test cases of stage 1 are implemented and executed using the “real-world” clients of each vendor
- 20 possible combinations
- It is checked if the tests can be executed and test results are compared



Cross Test - Summary

- dSPACE
  - Server tested with all Clients
  - Client (Automation Desk) tested with all other Servers
- ETAS
  - Server tested with all Clients
  - Client (LABCAR AUTOMATION) tested with all Servers
- Vector
  - Server tested with all Clients
  - Client (CANoe) tested with all Servers
- Tracetronic
  - No Server
  - Client (ECU Test) tested with all Servers
- NI
  - Server tested with all Clients
  - Client (TestStand) tested with all other Servers

Agenda

1	ASAM HIL API Motivation
2	Functionality covered by the ASAM HIL API / Status of the HIL API 1.0.2
3	Tula Technology – Live application example
4	Cross tests
5	**Summary**

Summary

- Two cross tests demonstrated that test systems and test automation software using ASAM HIL API are able to communicate immediately. The few minor functional deficits coming up due to a standard misinterpretation were identified at the first cross test in June 2012. A second cross test was conducted in early 2013 to make sure that all problems that were found was solved and removed.

- More information about version 2.0 of the ASAM HIL API will follow at next conference session

- Two HIL test system vendors: dSPACE and National Instruments have been working successfully together on supporting a live application based on the ASAM HIL API standard (V1.0.2) at Tula Technology in California.