Real and Virtual Development with SystemDesk

Joe Fairchild

Project Manager – Software Development and Validation

dSPACE, Inc.
Goals of AUTOSAR

- Create libraries of software components
  - Reusable
  - Hardware-independent
- Be flexible
  - Easy to exchange hardware or software components
  - Easy to change configurations
- Make tools interoperable and exchangeable by
  - Standardized methodology
  - Standardized interfaces and exchange formats
- “Cooperate on standards, Compete on implementation”

**dSPACE’s AUTOSAR Involvement**

Joined the AUTOSAR partnership as a **Premium Member** in April 2004.

dSPACE is active in specifying parts of the architecture.
Logical vs. Physical Architecture

Adaptive Cruise Brake Force

Read Brake Pedal

Calculate Vehicle Brake Force

Calculate Regenerative Brake Force

Calculate Brake Force for Each Wheel

Electronic Stability Control

Standard

Premium

Hybrid
AUTOSAR Software Development Methodology

Library of software components (C code functions with XML describing interfaces)

System (interconnection of software components)

ECU
Configure OS, and basic software; generate code for RTE; compile and link

System Configuration: Mapping of software components to ECUs, data elements to messages, etc.
AUTOSAR Software Development Process

- Design System Architecture
  - SystemDesk®
  - arxml

- Develop Control Functions
  - TargetLink®
  - .c

- Configure Basic Software
  - tresos Studio
  - arxml
  - .c

- Application
  - BSW Stack
  - ECU
SystemDesk at a Glance

- Modeling tool for AUTOSAR software and system architectures
  - Support of multiple versions of AUTOSAR standard, including most current versions
  - Integration of software architecture onto an ECU
  - Virtual integration of architectures for PC-based SIL simulations

- Process support
  - Complete tool automation via Python or C# API
  - Special round-trip support with TargetLink
Work with complex AUTOSAR objects in comfortable dialogs
- Model architecture of software at a functional level

- Map functions to ECUs and Networks
Validation

Complete validation of a branch of the project

Definition of custom rules

Validation in dialogs

Visualization of incompatible interfaces
Validation – EB tresos Compatibility Check

Validate project for import into EB tresos

Validation Results (ACC_ECU)

Validation of ACC_ECU failed with 2 errors and 0 warnings.
Rule configuration: \dSPACE\v2015a_64\SystemDesk_44\bin\Rules\EB tresos Compatibility Check.xml
Context: ACC_System
197 internal and 0 custom rules applied, 1284 project elements examined.

- ACC_ECU.SleepModeSupported - According to the STRICT version of the AUTOSAR schema, this element is mandatory. Currently, this element is not available.
  - Element Path: /System/ACC_ECU
  - Rule: R103

- ACC_ECU.WakeUpOverBusSupported - According to the STRICT version of the AUTOSAR schema, this element is mandatory. Currently, this element is not available.
  - Element Path: /System/ACC_ECU
  - Rule: R103

Locations:
- Locate Element in Project Manager
- Properties...

Actions:
- Set WakeUpOverBusSupported to 'true'.
- Set WakeUpOverBusSupported to 'false'.
- Set WakeUpOverBusSupported to 'true'. (Apply to all occurrences)
- Set WakeUpOverBusSupported to 'false'. (Apply to all occurrences)
Virtual ECU Generation Process

Design System Architecture

- SystemDesk®

Develop Control Functions

- TargetLink®

Generate V-ECU

- SystemDesk®

Virtual ECU:
Production-intent application code with simulation-capable BSW stack
Re-use V-ECU across dSPACE Platforms

Virtual ECU

MicroAutoBox II
Run production-intent code on a rapid prototyping platform

VEOS
Simulate realistic ECU on desktop environment without hardware

SCALEXIO
Connect Virtual ECU to physical signals for Hardware-In-Loop testing
# Pieces Required for Simulation and Testing

<table>
<thead>
<tr>
<th>Device / System Under Test</th>
<th>Environment Models</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Device / System Under Test" /></td>
<td><img src="image2" alt="Environment Models" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Execution Platform</th>
<th>Test Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3" alt="Execution Platform" /></td>
<td><img src="image4" alt="Test Tools" /></td>
</tr>
</tbody>
</table>
dSPACE’s Philosophy: Utilize industry standards
Create scalable solutions
Enable seamless transitions between phases of development
Integrated Tool Chain for Virtual ECU Development

Instruments and Simulation Control
Visualization and Animation
Road and Maneuver Definition
Test Automation and Evaluation

Early, PC-based validation of ECU software and functions
Seamless reuse of models, layouts, tests, data during real ECU testing

Third-Party Models and Test Tools

ControlDesk®
MotionDesk
ModelDesk
ASM
AutomationDesk®

FMI
XCP
HIL-API

VEOS
VEOS
VEOS Player – Linking it all Together

SystemDesk → TargetLink

Connection to Third-Party Test Tools

XCP

HIL-API

FMI

Co-Simulation with Third-Party Models

ControlDesk  AutomationDesk  ModelDesk  MotionDesk

VEOS

dSPACE
Seamless transition from Offline Simulation to HIL

SystemDesk → TargetLink → Simulink → ASM

Connection to Third-Party Test Tools

Co-Simulation with Third-Party Models

XCP HIL-API FMI
A Standards-Based, Scalable, Seamless solution for doing real and virtual testing across the entire software development cycle
Thank you! Any questions?