Simulink for AUTOSAR:
Best Practices
What is AUTOSAR?

AUTomotive Open System ARchitecture

- Objective: Establish an open standard for automotive E/E architecture

- Partnership
  Consisting of more than 180 companies from the global automotive industry

Agenda

Simulink for AUTOSAR - Introduction
- Workflows
- Capabilities

Simulink for AUTOSAR – User Stories
- Production Code Generation with Embedded Coder

Simulink for AUTOSAR – Best Practices
- Best Practices for using Simulink for AUTOSAR

Summary & Conclusions
Simulink & Stateflow for Behavior Modeling, Embedded Coder for Production Code

Software Architecture Definition

Application Layer

AUTOSAR Software Component 1
AUTOSAR Software Component 2
... 
AUTOSAR Software Component n

Runtime Environment (RTE)

Basic Software

BSW Configuration & RTE Generation

Behavior Modeling & Code Generation

“All toolboxes for MBD are still usable.”
Workflows

Capabilities

Simulation

Code Generation

Configuration
Agenda

Simulink for AUTOSAR - Introduction

- Workflows
- Capabilities

Simulink for AUTOSAR – User Stories

- Production Code Generation with Embedded Coder

Simulink for AUTOSAR – Best Practices

- Best Practices for using Simulink for AUTOSAR

Summary & Conclusions
Long-term Successful Collaboration with Volkswagen...

...from a “Proof of Concept” project...

...to series production across brands
More User Stories…

Market Situation and Rollout Strategies:
Volvo Cars

Model-Based Design based on AUTOSAR in an Electrical Systems Engineering Environment at Volvo Cars

Dennis Selin, Volvo Cars
Guido Sachtmann, MathWorks

Market Situation and Rollout Strategies at Our Most Important Customers: Valeo

Automatic code generation for AUTOSAR SW-Components in mass production application in Engine Management System process and benefits

Conclusion
- Key Project Data
  - More than 20 applicative SW-C developed
  - Using Auto Coding Process
  - More than 40 applicative SW-C integrated
  - Efficient solutions deployed to integrate RTE, AUTOSAR modules and historical SW modules in the same application
  - LIN Basic Software integrated as COTS
  - Use of commercial RTE and associated tools for full AUTOSAR conformity and fast response to new AUTOSAR release
  - Combined with VALEO legacy tools to increase integration efficiency

Motivation: Synergien

09. Juli 2014
Dr. David Sölder, Reinhard Jeschall
Modellbasierte Entwicklung eingebetteter Systeme für AUTOSAR mit der MathWorks-Toolkette

Validas for BMW
Agenda

Simulink for AUTOSAR - Introduction

- Workflows
- Capabilities

Simulink for AUTOSAR – User Stories

- Production Code Generation with Embedded Coder

Simulink for AUTOSAR – Best Practices

- Best Practices for using Simulink for AUTOSAR

Summary & Conclusions
#1 Decide strategy for migrating existing Simulink models to AUTOSAR

- Clean sheet start
- Start with existing Simulink models
- Maintain one model for AUTOSAR and non-AUTOSAR
#2 Use one AUTOSAR workflow

– Select top-down or bottom-up approach
– Round-trip works best with one clear owner of data

- Select tools that best support your workflow and AUTOSAR concepts
- Select simplest approach for applying AUTOSAR configuration to your Simulink model
#3 Decide data management

- Will Simulink or AUTOSAR tools manage data?
- Will projects or teams define and manage data?
- How will change management be handled?
#4 Establish modeling standards
– For Simulink and AUTOSAR

- Base it on your workflow and data management
- Use Simulink Model Advisor to enforce modeling style early in model development
#5 Simulate before you generate code
– Take advantage of **early verification** through simulation

- Make sure SWC implementation is correct early
- Simulate multiple SWC’s together in Simulink before code integration
- Use SIL and PIL to verify the generated code at the unit level before RTE generation
#6 Plan ahead for ISO 26262 – Determine how AUTOSAR process will address safety-standards

- Products supported for ISO 26262 tool qualification include:
  - Embedded Coder
  - Simulink V&V
  - Simulink Design Verifier
  - PolySpace

- Artifacts certified by TÜV SÜD
  - Requires use of V&V workflow

- ISO 26262 Advisory Service available
#7 Use Simulink to migrate legacy code to AUTOSAR

Reuse of Legacy Code
- Integration for simulation, production code generation
- Can generate AUTOSAR RTE API access points

```c
void Runnable_Runnable1(void) {
    real32_T rtb_TmpSignalConversionAtln1Out;
    real32_T rtb_UnitDelay;
    real32_T rtb_sldemo_sfun_filterV1;
    rtb_TmpSignalConversionAtln1Out = Rte_IRead_Runnable_Runnable1_Fast_in_Fast_in();
    rtb_UnitDelay = Component_DWork.UnitDelay_DSTATE;
    rtb_sldemo_sfun_filterV1 = filterV1( (real32_T)rtb_TmpSignalConversionAtln1Out,
        (real32_T)rtb_UnitDelay,
        Component_P.sldemo_sfun_filterV1_p1);
    Rte_IrvIWrite_Runnable_Runnable1_a( rtb_sldemo_sfun_filterV1);
    Component_DWork.UnitDelay_DSTATE = rtb_sldemo_sfun_filterV1;
}
```
#8 Automate, automate, automate
– Use API’s for workflow automation!

- **Manual process is difficult due to:**
  - The complexity of the standard, naming conventions
  - Iterative work cycles with AUTOSAR
  - Complex code APIs and XML file definitions

- Use documented MATLAB APIs to configure SWCs in Simulink

```matlab
%% Setup AUTOSAR Configuration programmatically
model = 'rtwdemo_autosar_counter';

% Modify AUTOSAR Properties
autosarProps = autosar.api.getAUTOSARProperties(model);
set(autosarProps, 'Input', 'IsService', true);
set(autosarProps, 'XmlOptions', 'ArxmlFilePackaging', 'SingleFile');
```
#9 Use production code generation
– Hand coding AUTOSAR is painful (Code and description)

```c
void Runnable_simple_alg_Step(void)
{
    real_T rtb_Gain;
    real_T rtb_Delay;
    real_T rtb_Delay1;
    real_T rtb_TmpSignalConversionAtFast_i;
    if (simple_alg_M->Timing.TaskCounters.TID[1] == 0) {
        Rte_Receive_Fast_in_Fast_in(&rtb_TmpSignalConversionAtFast_i);
        rtb_Delay = simple_alg_DWork.Delay_DSTATE;
        rtb_Delay1 = simple_alg_DWork.Delay1_DSTATE;
        rtb_Gain = simple_alg_DWork.Delay2_DSTATE;
    } if (simple_alg_M->Timing.TaskCounters.TID[2] == 0) {
        simple_alg_B.RateTransition = rtb_Gain;
    }
    simple_alg_DWork.Delay_DSTATE = rtb_TmpSignalConversionAtFast_i;
    simple_alg_DWork.Delay1_DSTATE = rtb_Delay;
    simple_alg_DWork.Delay2_DSTATE = rtb_Delay1;
} if (simple_alg_M->Timing.TaskCounters.TID[2] == 0) {
    Rte_IWrite_Runnable_simple_alg_Step_Out1_Out1((simple_alg_B.RateTransition + Rte_IRead_Runnable_simple_alg_Step_Slow_in_Slow_in()));
}
rate_scheduler();
}
```
#10 Actively plan for migration
– Tools and standards are changing rapidly

- Account for:
  – New versions of AUTOSAR
  – New versions of Simulink

- Consider:
  – How often to upgrade
  – What will drive upgrade

Source: AUTOSAR, 6th Open Conference 11.13.2013
Best practices for using Simulink with AUTOSAR

- Decide strategy for migrating existing Simulink models to AUTOSAR
- Use one AUTOSAR workflow
- Decide data management
- Establish modeling standard
- Simulate before code generation
- Plan ahead for ISO 26262
- Use Simulink to migrate legacy code to AUTOSAR
- Automate, automate, automate
- Use production code generation
- Actively plan for migration
Agenda

Simulink for AUTOSAR - Introduction
- Workflows
- Capabilities

Simulink for AUTOSAR – User Stories
- Production Code Generation with Embedded Coder

Simulink for AUTOSAR – Best Practices
- Best Practices for using Simulink for AUTOSAR

Summary & Conclusions
Summary

- Simulink and Embedded Coder provide extensive AUTOSAR capabilities out-of-the-box, along with API’s for workflow automation

- Leading automotive companies are successfully deploying AUTOSAR for production by leveraging MathWorks tools and industry experience

- Take advantage of best practices for deploying AUTOSAR with Production Code Generation to accelerate your projects while reducing risk and improving quality
Thank you for your attention!

Accelerating the pace of engineering and science