Model-Based Design of complex embedded systems using industry standards

Tom Erkkinen, Coder and Certification Product Manager, MathWorks
Agenda

- Model-Based Design Introduction
  - Key components
  - Users
  - Standards

- Model-Based Design Technologies
  - Systems
  - Software and Hardware
  - Verification and Validation (V&V)

- Q&A
Problematic Development Process

V&V

Systems

Software

Hardware
Model-Based Design

Hardware

Software

V&V

Systems

Systems
Automotive Example of Model-Based Design

Accelerating Sensor Development with Rapid Prototyping and Model-Based Design

By Martin Hein, Hella Fahrzeugkomponenten GmbH

For designs that target a microprocessor, we use Embedded Coder™ to generate code from our Simulink model and deploy it to the TI C2000 processor on the HFK RCP unit. If all or part of the design requires an FPGA, we use HDL Coder™ to generate VHDL code from the model for deployment on the Xilinx FPGA.

Early V&V (SIL, PIL)

We use software-in-the-loop (SIL) testing to verify the implementation of our algorithms in C and processor-in-the-loop (PIL) testing to verify the algorithm on real-time hardware. In this way we ensure that the model, which we have already verified, is implemented without introducing errors.
Aerospace Example of Model-Based Design

1. Golden Reference Model (Matlab scripts/functions)
2. Simulink Model (floating-point Embedded Matlab)
3. Autocode ANSI C (PIL)
4. Software Design (C/C++)
   Hardware Design (System/HDL Coder)
5. Integration verification (PIL)
   Code validation (FIL)
6. Full system validation
   Hardware-in-the-loop testing

Motor Controls

Stabilised Mirror Motor Controller

Captured data presented to model

Doppler Radar

Application: Range Doppler Processing

Captured Data

Simulated Data

Data Captured from Xilinx FPGA based hardware

Data Captured from Simulink Simulation

Model-Based Design – Certification Examples

- **DO-178 (Level A)**
  - Honeywell Aerospace USA
  - Flight Control Systems

- **EN 50128**
  - Alstom France
  - Propulsion Control Systems

- **ISO 26262**
  - GM Global
  - Hybrid Powertrain

- **IEC 62304**
  - Weinmann Medical Germany
  - Transport ventilator

- **IEC 61508**
  - Alstom Grid UK
  - HDVC Power Systems

- **???????**
  - Academia and Schools
  - Project Based Learning
Model-Based Design is deeply rooted in ISO 26262

1.74 model-based development

development that uses models to describe the functional behavior of the elements which are to be developed

NOTE Depending on the level of abstraction used for such a model it can be used for simulation or code generation or both.

Annex B (informative)

B.1 Objectives

This Annex describes the concept of model-based development of in-vehicle software and outlines its implications on the product development at the software level.

The seamless utilisation of models facilitates a highly consistent and efficient development.
Aerospace Standard (ARP4754A)
Model-Based Design is highly recommended in ARP4754

Help uncover missing requirements

Prototypes are models of the desired system that may be hardware and/or software based, and may or may not be development versions of the system. Prototypes permit users of a system to interact with a proposed model of the system to uncover missing requirements, behaviors of the system that should be prohibited, and potential problems with user interaction.

How well the prototype represents the actual system may drive the likelihood of identifying missing requirements. The tools used to create prototypes should minimize development time.

The model should be developed in a structured manner. For example, subsets of model elements which may be used more than once may be handled and represented either as a unit or as the full contents.

Model use for requirements validation typically uses a model of the environment of a system being developed, which is interfaced to a prototype of a design solution for those requirements. An environment model that is representative of the environment of the system being developed, provides a high degree of functional coverage in exercising either a simulated or real system.

provide a high degree of function coverage
Model-Based Design has a dedicated supplement* in DO-178C

*DO-331: Model-Based Supplement to DO-178C and DO-278A
**Table MB.1-1 Model Usage Examples**

<table>
<thead>
<tr>
<th>Process that generates the life-cycle data</th>
<th>MB Example 1</th>
<th>MB Example 2</th>
<th>MB Example 3</th>
<th>MB Example 4 (See Note 1)</th>
<th>MB Example 5 (See Note 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>System Requirement and System Design Processes</strong></td>
<td>Requirements allocated to software</td>
<td>Requirements from which the Model is developed</td>
<td>Requirements from which the Model is developed</td>
<td>Requirements from which the Model is developed</td>
<td>Requirements from which the Model is developed</td>
</tr>
<tr>
<td><strong>Software Requirement and Software Design Processes</strong></td>
<td>Requirements from which the Model is developed</td>
<td>Specification Model (See Note 2)</td>
<td>Specification Model</td>
<td>Design Model</td>
<td>Design Model</td>
</tr>
</tbody>
</table>

**Notes:**
1. Note 1: Requires further clarification.
2. Note 2: Requires further clarification.
3. Note 3: Requires further clarification.
Agenda

- Model-Based Design Introduction
  - Key components
  - Users
  - Standards

- Model-Based Design Technologies
  - Systems
  - Software and Hardware
  - Verification and Validation (V&V)

- Q&A
Model-Based Design

Executable Specs, Automatic Code Generation, and Early V&V

Hardware

Software

V&V Systems

Hardware
Systems simulation
Software and hardware generation
Early verification and validation
Integral and backbone process

Or maybe you have …. AUTOSAR
AUTOSAR Target Workflow (Top-Down)

AUTOSAR Authoring Tool

- Export SWC Description
- Merge SWC Description
- Import SWC Description
- Export SWC Description/Generate SWC C code

Model Based Design
AUTOSAR Target Workflow (Bottom-Up)

AUTOSAR Authoring Tool

Import SWC Description

Export SWC Description/Generate SWC C code
Thank You!

- Questions?
Model-Based Design

System Truth
- rapid iterations/prototypes

Executable Specifications

Detailed Design

Continuous Test and Verification

Test Reuse
- detect errors earlier

Hardware & Software
- convert float- to fixed point

Code Generation

Production code
- Eliminate hand code errors

Models
Model-Based Design

Executable Specs, Automatic Code Generation, and Early V&V