Le Model Based Design pour la vérification et la validation de modèles de simulations

by
Adel Allal : Sales Account Manager
Laurent Bresson : Application Engineer
Quiz

- What’s the most powerful V&V tool available in the Simulink family of products?

- The ‘Play’ button!
MATLAB & Simulink Products

La famille de produits MATLAB®

- MATLAB
- Calcul Parallèle
- Parallel Computing Toolbox
- MATLAB Distributed Computing Server
- Mathématiques, Statistiques et Optimisation
  - Symbolic Math Toolbox
  - Partial Differential Equation Toolbox
  - Statistics and Machine Learning Toolbox
  - Curve Fitting Toolbox
  - Optimization Toolbox
  - Global Optimization Toolbox
  - Neural Network Toolbox
  - Model-Based Calibration Toolbox
- Systèmes de contrôle
  - Control System Toolbox
  - System Identification Toolbox
  - Fuzzy Logic Toolbox
  - Robust Control Toolbox
  - Model Predictive Control Toolbox
  - Aerospace Toolbox
  - Robotics System Toolbox

Traitement du signal et communications
- Signal Processing Toolbox
- DSP System Toolbox
- Communications System Toolbox
- Wavelet Toolbox
- RF Toolbox
- Antenna Toolbox
- Phased Array System Toolbox
- LTE System Toolbox
- WLAN System Toolbox

Traitement d’image et Vision assistée par ordinateur
- Image Processing Toolbox
- Computer Vision System Toolbox
- Vision HDL Toolbox
- Image Acquisition Toolbox
- Mapping Toolbox

Test et mesure
- Data Acquisition Toolbox
- Instrument Control Toolbox
- Image Acquisition Toolbox
- OPC Toolbox
- Vehicle Network Toolbox

Mathématiques financières
- Financial Toolbox
- Econometrics Toolbox
- Database Toolbox
- Spreadsheet Link® EX (pour Microsoft Excel)
- Financial Instruments Toolbox
- Trading Toolbox

Biologie computationnelle
- Bioinformatics Toolbox
- Simbiology

Génération et Vérification de Code
- MATLAB Coder
- HDL Coder
- Vision HDL Toolbox
- HDL Verifier
- FPGA Design HDL Coder
- Fixed-Point Designer

Déploiement d’applications
- MATLAB Compiler
- MATLAB Compiler SDK
- Spreadsheet Link EX (pour Microsoft Excel)
- MATLAB Production Server

Accès aux bases de données et Reporting
- Database Toolbox
- MATLAB Report Generator

La famille de produits Simulink®

- Simulink
- Modélisation événementielle
  - Stateflow
  - SimEvents
- Modélisation physique
  - Simscape
  - SimMechanics
  - SimHydraulics
  - SimRF
  - SimElectronics
  - SimPowerSystems
- Systèmes de contrôle
  - Simulink Control Design
  - Simulink Design Optimization
  - Aerospace Blockset
  - Robotics System Toolbox

Traitement du signal et communications
- DSP System Toolbox
- Communications System Toolbox
- Phased Array System Toolbox
- SimRF
- Computer Vision System Toolbox

Génération de code
- SimulinkCoder
- Embedded Coder
- HDL Coder
- Vision HDL Toolbox
- Simulink PLC Coder
- Fixed-Point Designer
- DO Qualification Kit (for DO-178)
- IEC Certification Kit (for ISO 26262 and IEC 61508)

Simulation en temps réel et test
- Simulink Real-Time
- Simulink Desktop Real-Time

Vérification, validation et test
- Simulink Verification and Validation
- Simulink Design Verifier
- Simulink Test
- Simulink Code Inspector
- HDL Verifier
- Polyspace Bug Finder
- Polyspace Code Prover

Résultats graphiques de simulation et Reporting
- Simulink 3D Animation
- Gauges Blockset
- Simulink Report Generator
TRW Automotive Develops and Tests Electric Parking Brake Using Simulink and Simulink Design Verifier

Challenge
Design tests for an electric parking brake control system

Solution
Use Simulink Design Verifier to automatically generate tests that maximize model coverage and enable systematic design verification

Results
- Test development time reduced from days to hours
- 100 percent model coverage achieved
- Formal testing begun two months into the project

“Everyone knows that errors are much less expensive to fix when you find them early. With Simulink Design Verifier, we build on the advantages of Model-Based Design by performing formal testing in the first phases of development.”

Christoph Hellwig
TRW

Link to user story
Statement
Different kind of models,

- Different kind of objectives, different point of the V-cycle
  ... but you always need to verify & validate, implicitly or explicitly
Different kind of models,

- Different kind of objectives, different point of the V-cycle
  ... but you always need to verify & validate, implicitly or explicitly
Different kind of models,

- Different kind of objectives, different point of the V-cycle
  ... but you always need to verify & validate, implicitly or explicitly
Why do you need to Verify & Validate at the very beginning of your projet?
Why early and Automated Verification?

What is the Most Expensive Project Stage to Find Errors In?

Legend – effort to correct:
- Errors introduced in the coding phase

Why early and Automated Verification?
What is the Most Expensive Project Stage to Find Errors In?

Legend – effort to correct:
- Errors introduced in the coding phase
- Errors introduced in the design phase

Why early and Automated Verification?

What is the Most Expensive Project Stage to Find Errors In?

Legend – effort to correct:
- Errors introduced in the coding phase
- Errors introduced in the design phase
- Errors introduced in the requirements specifications phase

Why early and Automated Verification?
What is the Most Expensive Project Stage to Find Errors In?

Mitigate that by *early* validation through simulation and automatic code generation.

Legend – effort to correct:
- Errors introduced in the coding phase
- Errors introduced in the design phase
- Errors introduced in the requirements specifications phase

Traditional Design Process

- Lots of manual re-coding between the design stages
  - New errors introduced in every re-coding step
- Lots of design environment and languages
- Late requirements validation
  - Errors in requirements can stop projects
- Lack of capitalization
  - Each team develops their own tooling
Model-Based Design Process

- Early stage requirements validation
- Automate manual tasks
- Easy to verify implementation at system-level
- Work as a team
- Increase capitalization over models
Model Based Design
Model-Based Design - Design and test using simulation

Requirements are linked to the model.
Detail is added from specific domains to refine the model.
Intellectual property and engineering data are reused from existing designs and CAE tools, such as CAD, FEA, and SPICE models.
Testing control algorithms against requirements is done by simulating the model.

RESEARCH REQUIREMENTS

DESIGN

Environmental Models
Mechanical Electrical
Control Algorithms
Supervisory Logic

TEST & VERIFICATION

Requirements are linked to the model.
Detail is added from specific domains to refine the model.
Intellectual property and engineering data are reused from existing designs and CAE tools, such as CAD, FEA, and SPICE models.
Testing control algorithms against requirements is done by simulating the model.
Model-Based Design—Test and validate in real-time

**TEST & VERIFICATION**

**REAL-TIME TESTING**
- C, C++
- VHDL, Verilog
  - MCU
  - DSP
  - FPGA
  - ASIC

**DESIGN**
- Environmental Models
  - Mechanical
  - Electrical
- Control Algorithms
- Supervisory Logic

**RESEARCH**

**REQUIREMENTS**

Automatically generate code from the simulation model for real-time testing of the control algorithms.

Rapid Control Prototyping
Model-Based Design - Test and validate in real-time

RESEARCH

requirements

DESIGN

- Environmental Models
  - Mechanical
  - Electrical
- Control Algorithms
- Supervisory Logic

TEST & VERIFICATION

- Automatically generate code from the simulation model for real-time testing of the control algorithms.

REAL-TIME TESTING

- C, C++
- VHDL, Verilog
- MCU, DSP, FPGA, ASIC

Hardware-in-the-loop

- Automatically generate code from the simulation model for real-time system simulation of hardware for testing the real microcontroller, FPGA, or PLC.
Motivation for Model Based Design

Reasons for model-based development

- Design new functions with high complexity
- Increase product quality
- Shorter development times
- Shorter time to market

- Cost savings: 27 %
- Time savings: 36 %
VnV Tool Chain
v1.0 Features

Library of blocks

Pre-built reference applications
 Libraries include basic component and subsystem controllers
  - Powertrain Control Module (PCM)
  - Hybrid Control Module (HCM)
  - Engine Control Module (ECM)
  - Transmission Control Module (TCM)
  - Component controllers
  - State estimators

 Like plant models, the controllers are open and reconfigurable

 Provides realistic starting point for your own controller development
Simulink Real Time
Modeling Guidelines for High-Integrity Systems

- Modeling Guidelines and corresponding Model Advisor checks for:
  - ISO 26262

Library: Drivetrain

- Limited Slip Differential
- Open Differential
- Rotational Inertia
- Torsional Compliance
- Longitudinal Wheel - Disc Brake
- Longitudinal Wheel - Drum Brake
- Longitudinal Wheel - Mapped Brake
- Longitudinal Wheel - No Brake
Library: Energy Storage and Auxiliary Drive

- Starter
- Reduced Lundell Alternator
- Datasheet Battery
- Equivalent Circuit Battery
- Estimation Equivalent Circuit Battery
Library: Propulsion (Gasoline / Diesel Engines)
Library: Propulsion (Electric Motors)
Library: Transmission

- CVT Controller
- DCT Controller
- AMT Controller
- Torque Converter
- Continuously Variable Transmission
- Dual Clutch Transmission
- Automated Manual Transmission
- Ideal Fixed Gear Transmission
Library: Vehicle Dynamics and Scenario Builder

Drive Cycle Source
FTP75 (2474 seconds)

Longitudinal Driver

Vehicle Body Total Road Load

Vehicle Body 1DOF Longitudinal

Vehicle Body 3DOF Longitudinal
Powertrain Blockset Design Principles

- **Balance accuracy and speed**
  - Include detailed physics (turbo wind-up, manifold filling / emptying, driveline dynamics)
  - Exceed real-time performance requirements

- **Keep the subsystems open and reconfigurable**
  - Learn from our application of best practices
  - Parameterize subsystems with your own data
  - Modify subsystems according to your needs
  - Replace built-in subsystems entirely
    - Custom Simulink-based subsystems
    - Custom Simscape-based subsystems
    - Complementary modeling tools (e.g., connected via S-function)
## MathWorks V&V Product Portfolio

<table>
<thead>
<tr>
<th>Product</th>
<th>Capabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulink Test</td>
<td>Author, execute, and manage simulation-based tests for models and generated code</td>
</tr>
<tr>
<td>Simulink Verification &amp; Validation</td>
<td>Trace to requirements, check model standards, perform coverage analysis</td>
</tr>
<tr>
<td>Simulink Design Verifier</td>
<td>Identify design errors, automatically generate test vectors, verify designs against requirements</td>
</tr>
<tr>
<td>Report Generator</td>
<td>Design and generate reports from MATLAB applications</td>
</tr>
<tr>
<td>Polyspace Bug Finder</td>
<td>Find software bugs and check compliance to MISRA</td>
</tr>
<tr>
<td>Polyspace Code Prover</td>
<td>Prove the absence of run-time errors in software</td>
</tr>
<tr>
<td>Simulink Code Inspector</td>
<td>Automate source code reviews for safety standards</td>
</tr>
<tr>
<td>Simulink Real-Time</td>
<td>Build, run, and test real-time applications</td>
</tr>
</tbody>
</table>