What’s New In Simulink: R2015a R2015b

Fraser Macmillen

MATLAB EXPO 2015
UNITED KINGDOM
Agenda

– Interacting with models

– Handling model (design) data

– New modelling constructs & editing features

– Simulink Test
How can you easily create a human-machine interface (HMI) with your Simulink model?
Dashboard Block Library

Tune and test your simulations with graphical controls and displays

- New library of controls and displays, including knobs, switches, and gauges
- Place blocks directly within the Simulink Editor
- Provide a front-end user interface to the entire model
New Interface & Functionality for Scopes

View and debug signals with cursors and measurements

- Scope, Floating Scope, and Viewers all upgraded with new UI

- Includes simulation data analysis and debugging tools
  - Cursors
  - Measurements
  - Triggers

View and debug signals with cursors and measurements.
With this scope one can:

- Control simulation from scope
- Triggers - to stabilize the signal
- Freeze display
- Cursors

Additional capabilities with DSP or Simscape
- Transitions (e.g. slew rates)
- Bilevel measurements: overshoots & undershoots
- Pulse width / duty cycle
- Peak finder
How can you run multiple consecutive simulations faster?
Fast Restart

Run consecutive simulations faster

- Fast restart on command bar
- Fast restart from command line

```matlab
sim(mdlName, 'FastRestart', 'on');
```
%% Set parameters for batch run

machReheatArray = 0.0:0.1:0.8;
machRocketArray = 0.0:0.1:1.0;
Agenda

- Interacting with models
- Handling model (design) data
- New modelling constructs & editing features
- Simulink Test
Handling model design data

How can I tackle these challenges of using the MATLAB workspace?

- Mixed with other MATLAB data
- Lack of organization: hierarchy, componentisation
- Ease of change detection
- What was the source of the data?
- Data conflict – multiple files writing to same variable name
Data Dictionary provides:

- Clarity of base workspace
- Avoids risk of running with incorrect data, or data clash
- Organised data hierarchy
- Ease of data re-use
- Integrated change management
Data Dictionary API

Automate the creation and editing of data dictionaries with MATLAB scripts

- Migrate models to use data dictionaries
- Import & export data
- Create, delete, and reassign data
- Save or discard changes to entire dictionaries or individual entries

```matlab
web(fullfile(docroot, 'simulink/ug/store-data-in-dictionary-programmatically.html'))
```
Bloodhound data organisation:

File Type
- Simulink Model (9)
- Simulink Data Dictionary (10)
Agenda

– Interacting with models

– Handling model (design) data

– New modelling constructs & editing features

– Simulink Test
Making life easier….

Automatic solver selection

Solver: auto (Automatic solver selection)

Simulation meta data

simData.getSimulationMetadata;

Always-On Tunability

From spreadsheet block

myInputTimeHistories.xlsx
Sheet: Test1

From Spreadsheet

Create project from a model

Simulink Project > Create Project from Model
Bus “Smart Editing Cue”

Automatically create a bus from a set of signals in your model

- Adds a bus creator block with the appropriate number of inputs, size, position, and orientation.
- Automatically connects the signals.
Area Annotations

Call out and separate regions of interest in your model

- Highlight groups of blocks in your model
- Specify a name for the area annotation
- Move all blocks together by moving the area annotation
- Convert area annotation into a subsystem
Rename All

Change the name of a parameter and all its references with one click

- Rename a variable and have Simulink automatically update all references to the variable in your model
- Right-click the variable in Model Explorer and select “Rename All…” to start
- Also works for data store memory
<table>
<thead>
<tr>
<th>Name</th>
<th>DataSource</th>
<th>Value</th>
<th>DataType</th>
<th>Max</th>
<th>Min</th>
<th>Dimensions</th>
<th>StorageClass</th>
<th>Status</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>VarSpeedDiffAbsTol</td>
<td>ControllersCommon.sdd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Controllers_sample_time</td>
<td>ControllersCommon.sdd</td>
<td>0</td>
<td>double</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DriverBus</td>
<td>InterfaceDefinitions.sdd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PropulsionBus</td>
<td>InterfaceDefinitions.sdd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VehicleSensorBus</td>
<td>InterfaceDefinitions.sdd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BrakingBus</td>
<td>InterfaceDefinitions.sdd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ControllerBus</td>
<td>InterfaceDefinitions.sdd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EnvironmentBus</td>
<td>InterfaceDefinitions.sdd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GroundBus</td>
<td>InterfaceDefinitions.sdd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ThrustBus</td>
<td>InterfaceDefinitions.sdd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VehicleMotionBus</td>
<td>InterfaceDefinitions.sdd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WindBus</td>
<td>InterfaceDefinitions.sdd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WorldSensorBus</td>
<td>InterfaceDefinitions.sdd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Driver</td>
<td>InterfaceDefinitions.sdd</td>
<td>Bus: DriverBus [ ] [ ] 1</td>
<td>ExportedGlobal</td>
<td>auto</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BrakingCommands</td>
<td>InterfaceDefinitions.sdd</td>
<td>Bus: BrakingBus [ ] [ ] 1</td>
<td>ExportedGlobal</td>
<td>auto</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VehicleSensors</td>
<td>InterfaceDefinitions.sdd</td>
<td>Bus: VehicleSensorBus [ ] [ ] 1</td>
<td>ExportedGlobal</td>
<td>auto</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AliveTime</td>
<td>InterfaceDefinitions.sdd</td>
<td>double 9999999 0 1</td>
<td>ExportedGlobal</td>
<td>auto</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Controls</td>
<td>InterfaceDefinitions.sdd</td>
<td>Bus: ControllerBus [ ] [ ] 1</td>
<td>ExportedGlobal</td>
<td>auto</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Messages

Model asynchronous operations in state charts using objects that carry data and can be queued

- New message object and queue
- Message Viewer block to visualize lifetime of a message
- Signal lines in Simulink to transfer messages between charts
Testing Models: Common challenges

Throughout a project lifecycle:

- Do legacy subsystem models work in new application?
- The model worked last week… does it still?
- Another engineer modified the algorithm… still working ok?
- Does the generated code match the model?
- Does subsystem/system meet the design requirements?
Simulink Test

New product launched in R2015a

1. Test Harness
2. Test Sequence Block
3. Test Manager

Test Harness

Component under test

Main Model

Report Generated by Test Manager
Simulink Test API

Automate test creation, editing, and execution using MATLAB scripts

- Programmatically create and edit Simulink Test Harnesses
- Author and manage test cases through the API
- Automate test execution, analysis, and reporting

```
%Create a new Test file, test suite
tf = sltest.testmanager.TestFile;
ts = tf.createTestSuite('API Test');
tc = ts.createTestCase('baseline');

%Assign the system under test to the test case
tc.setProperty('Model','sldemo_automotive');
```
Qualification and Certification for Simulink Test

Qualify Simulink test for supported industry standards

- DO Qualification Kit (for DO-178) now supports Simulink Test
- IEC Certification Kit (for ISO 26262 and IEC 61508) now supports Simulink Test

Access Certification Artifacts and Qualification documents in Product documentation
MATLAB EXPO 2015

Testing, Validating, and Verifying with Model-Based Design

15:45–17:00

Simulink® and Stateflow® help individuals and teams rapidly develop complex designs. These designs must be supported by appropriate levels of testing, ideally effective test methods that identify defects as early as possible. In other words, efficient testing is just as important in speeding up design times as having capable design tools. This talk describes the range of verification and validation techniques that can be applied to models, discusses the classes of defects they can best discover, and provides practical advice on how to apply them.
% set parameters in data dictionary:
Simulink.data.assigninGlobal('BloodhoundTop', 'machReheat',
Simulink.data.assigninGlobal('BloodhoundTop', 'machRocket',

Conclusion