

dSPACE Release

# New Features and Migration

Release 2016-B – November 2016

## How to Contact dSPACE

Mail:	dSPACE GmbH Rathenaustraße 26 33102 Paderborn Germany
Tel.:	+49 5251 1638-0
Fax:	+49 5251 16198-0
E-mail:	info@dspace.de
Web:	<a href="http://www.dspace.com">http://www.dspace.com</a>

## How to Contact dSPACE Support

To contact dSPACE if you have problems and questions, fill out the support request form provided on the website at <http://www.dspace.com/go/supportrequest>.

The request form helps the support team handle your difficulties quickly and efficiently.

In urgent cases contact dSPACE via phone: +49 5251 1638-941 (General Technical Support)

## Software Updates and Patches

dSPACE strongly recommends that you download and install the most recent patches for your current dSPACE installation. Visit <http://www.dspace.com/go/support> for software updates and patches.

## Important Notice

This document contains proprietary information that is protected by copyright. All rights are reserved. The document may be printed for personal or internal use provided all the proprietary markings are retained on all printed copies. In all other cases, the document must not be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form, in whole or in part, without the prior written consent of dSPACE GmbH.

© 2000 - 2016 by:  
dSPACE GmbH  
Rathenaustraße 26  
33102 Paderborn  
Germany

This publication and the contents hereof are subject to change without notice.

CalDesk, ConfigurationDesk, ControlDesk, MicroAutoBox, MicroLabBox, SCALEXIO, SYNECT, SystemDesk, TargetLink and VEOS are registered trademarks of dSPACE GmbH in the United States or other countries, or both. Other brand names or product names are trademarks or registered trademarks of their respective companies or organizations.

# Contents

<b>About This Document</b>	11
<i>Conventions Used in the Documentation</i> .....	11
<i>Accessing Online Help and PDF Files</i> .....	13
<b>Overview of dSPACE Release 2016-B</b>	15
<i>General Enhancements and Changes</i> .....	15
<i>Product Version Overview</i> .....	22
<i>New Product Key Features</i> .....	26
<b>Aspects of Migrating from Previous Releases</b>	35
<i>Migrating to dSPACE Release 2016-B</i> .....	35
<b>Changes to TRC File Generation</b>	37
<i>Basics on the TRC File Changes</i> .....	37
<i>Migrating Changes in Software That Generates TRC Files</i> .....	43
<i>Migrating Changes in Software That Uses TRC Files</i> .....	44
<b>AutomationDesk</b>	47
<i>New Features of AutomationDesk 5.3</i> .....	47
<i>Migrating to AutomationDesk 5.3</i> .....	50
<b>Automotive Simulation Models (ASM)</b>	55
All ASM Blocksets.....	57
<i>Migrating all ASM Blocksets</i> .....	57
ASM Base InCylinder Blockset.....	58
<i>New Features of ASM Base InCylinder Blockset 2.3</i> .....	58
<i>Migrating to ASM Base InCylinder Blockset 2.3</i> .....	58
ASM Brake Hydraulics Blockset.....	60
<i>Migrating to ASM Brake Hydraulics Blockset 1.6.1</i> .....	60
ASM Diesel Engine Blockset.....	63
<i>New Features of ASM Diesel Engine Blockset 2.4</i> .....	63

<i>Changes in the ASM Diesel Engine Demo Model</i> .....	64
<i>Migrating to ASM Diesel Engine Blockset 2.4</i> .....	65
ASM Diesel Exhaust Blockset.....	69
<i>Migrating to ASM Diesel Exhaust Blockset 2.1.3</i> .....	69
ASM Diesel InCylinder Blockset.....	71
<i>Changes in the ASM Diesel InCylinder Demo Model</i> .....	71
<i>Migrating to ASM Diesel InCylinder Blockset 2.3</i> .....	71
ASM Drivetrain Basic Blockset.....	73
<i>New Features of ASM Drivetrain Basic Blockset 4.3</i> .....	73
<i>Migrating to ASM Drivetrain Basic Blockset 4.3</i> .....	74
ASM Electric Components Blockset.....	77
<i>New Features of ASM Electric Components Blockset 3.3</i> .....	77
<i>Changes in the ASM Electric Components Demo Model</i> .....	77
<i>Migrating to ASM Electric Components Blockset 3.3</i> .....	78
ASM Environment Blockset.....	80
<i>New Features of ASM Environment Blockset 4.5</i> .....	80
<i>Migrating to ASM Environment Blockset 4.5</i> .....	80
ASM Gasoline Engine Basic Blockset.....	82
<i>New Features of ASM Gasoline Engine Basic Blockset 2.1</i> .....	82
<i>Changes in the ASM Engine Gasoline Basic Demo Model</i> .....	82
<i>Migrating to ASM Gasoline Engine Basic Blockset 2.1</i> .....	83
ASM Gasoline Engine Blockset.....	85
<i>New Features of ASM Gasoline Engine Blockset 3.4</i> .....	85
<i>Changes in the ASM Engine Gasoline Demo Model</i> .....	86
<i>Migrating to ASM Gasoline Engine Blockset 3.4</i> .....	87
ASM Gasoline InCylinder Blockset.....	91
<i>Changes in the ASM Gasoline InCylinder Demo Model</i> .....	91
<i>Migrating to ASM Gasoline InCylinder Blockset 2.3</i> .....	91
ASM Optimizer.....	93
<i>New Features of ASM Optimizer 1.8</i> .....	93
ASM Pneumatics Blockset.....	94
<i>Migrating to ASM Pneumatics Blockset 2.0.4</i> .....	94
ASM Traffic Blockset.....	95
<i>New Features of ASM Traffic Blockset 3.5</i> .....	95
<i>Changes in the ASM Traffic Demo Model</i> .....	95
<i>Migrating to ASM Traffic Blockset 3.5</i> .....	95
ASM Trailer Blockset.....	97
<i>New Features of ASM Trailer Blockset 2.6</i> .....	97

<i>Changes in the ASM Trailer Demo Model</i> .....	97
<i>Migrating to ASM Trailer Blockset 2.6</i> .....	98
ASM Truck Blockset.....	101
<i>New Features of ASM Truck Blockset 3.0</i> .....	101
<i>Changes in the ASM Truck Demo Model</i> .....	101
<i>Migrating to ASM Truck Blockset 3.0</i> .....	102
ASM Turbocharger Blockset.....	104
<i>New Features of ASM Turbocharger Blockset 3.1.3</i> .....	104
<i>Migrating to ASM Turbocharger Blockset 3.1.3</i> .....	104
ASM Utils.....	106
<i>Migrating ASM Utils</i> .....	106
ASM Vehicle Dynamics Blockset.....	107
<i>New Features of ASM Vehicle Dynamics Blockset 3.4</i> .....	107
<i>Changes in the ASM Vehicle Dynamics Demo Model</i> .....	108
<i>Migrating to ASM Vehicle Dynamics Blockset 3.4</i> .....	108
<b>Bus Manager (Stand-Alone)</b> .....	113
<i>Features of the Bus Manager (Stand-Alone) 5.6</i> .....	113
<b>ConfigurationDesk</b> .....	115
ConfigurationDesk – Implementation.....	116
<i>New Features of ConfigurationDesk 5.6 (Implementation Version)</i> .....	116
<i>Migrating to ConfigurationDesk 5.6</i> .....	119
<b>ControlDesk</b> .....	121
New Features of ControlDesk (ControlDesk 6.0).....	122
<i>New General Features (ControlDesk 6.0)</i> .....	123
<i>New Project and Experiment Features (ControlDesk 6.0)</i> .....	123
<i>New Features of Platform Management and Platforms/Devices (ControlDesk 6.0)</i> .....	124
<i>New Instrument Features (ControlDesk 6.0)</i> .....	126
<i>New Measurement and Recording Features (ControlDesk 6.0)</i> .....	130
<i>New Message Handling Features (ControlDesk 6.0)</i> .....	131
<i>New Bus Navigator Features (ControlDesk 6.0)</i> .....	131
<i>New ECU Diagnostics Features (ControlDesk 6.0)</i> .....	132
<i>New Electrical Error Simulation Features (ControlDesk 6.0)</i> ..	133
<i>New Signal Editor Features (ControlDesk 6.0)</i> .....	134

Migrating to ControlDesk (ControlDesk 6.0).....	136
<i>Discontinuations in ControlDesk</i> .....	136
<i>Migrating to ControlDesk (ControlDesk 6.0)</i> .....	138
<b>DCI Configuration Tool</b>	145
<i>New Features of the DCI Configuration Tool 3.7</i> .....	145
<b>dSPACE CAN API Package</b>	147
<i>New Features of dSPACE CAN API Package 3.0</i> .....	147
<b>dSPACE ECU Flash Programming Tool</b>	149
<i>New Features of the dSPACE ECU Flash Programming</i>	
<i>Tool 2.3.1</i> .....	149
<b>dSPACE FlexRay Configuration Package</b>	151
<i>New Features of dSPACE FlexRay Configuration Package</i>	
<i>3.8</i> .....	151
<b>dSPACE Python Extensions</b>	153
<i>New Features of dSPACE Python Extensions 2.2</i> .....	153
<i>Migrating to dSPACE Python Extensions 2.2</i> .....	153
<b>dSPACE XIL API .NET</b>	155
<i>New Features of dSPACE XIL API .NET 2016-B</i> .....	155
<i>Migrating to dSPACE XIL API .NET 2016-B</i> .....	157
<b>ECU Interface Manager</b>	159
<i>Migrating to ECU Interface Manager 2.0</i> .....	159
<b>Firmware Manager</b>	161
<i>New Features of Firmware Manager 2.2</i> .....	161
<b>Model Compare</b>	163
<i>New Features of Model Compare 2.7</i> .....	163
<i>Migration to Model Compare 2.7</i> .....	164

<b>ModelDesk</b>	165
<i>New Features of ModelDesk 4.4</i> .....	165
<i>Migration to ModelDesk 4.4</i> .....	166
<b>Model Interface Package for Simulink</b>	167
<i>New Features of the Model Interface Package for Simulink 3.3</i> .....	167
<i>Migration Aspects of the Model Interface Package for Simulink</i> .....	168
<b>MotionDesk</b>	171
<i>New Features of MotionDesk 3.9</i> .....	171
<i>Migrating to MotionDesk 3.9</i> .....	172
<b>Real-Time Testing</b>	173
<i>New Features of Real-Time Testing 3.1</i> .....	173
<i>Migrating to Real-Time Testing 3.1</i> .....	174
<b>RTI/RTI-MP and RTLib</b>	175
<i>New Features of RTI/RTI-MP and RTLib</i> .....	175
<i>Migration Aspects of RTI/RTI-MP and RTLib</i> .....	177
<b>RTI Bypass Blockset</b>	179
<i>New Features of the RTI Bypass Blockset 3.7</i> .....	179
<i>Migrating to RTI Bypass Blockset 3.7</i> .....	180
<b>RTI CAN MultiMessage Blockset</b>	181
<i>New Features of the RTI CAN MultiMessage Blockset 4.4</i> ....	181
<i>Migrating to RTI CAN MultiMessage Blockset 4.4</i> .....	181
<b>RTI FPGA Programming Blockset</b>	183
<i>New Features of the RTI FPGA Programming Blockset 3.2</i> ....	183
<i>Migrating to RTI FPGA Programming Blockset 3.2</i> .....	184
<b>RTI LIN MultiMessage Blockset</b>	187
<i>New Features of the RTI LIN MultiMessage Blockset 2.7</i> .....	187
<i>Migrating to RTI LIN MultiMessage Blockset 2.7</i> .....	187

<b>RTI Watchdog Blockset</b>	189
<i>Features of RTI Watchdog Blockset 2.0</i> .....	189
<b>SCALEXIO Firmware</b>	191
<i>New Features of the SCALEXIO Firmware 3.5</i> .....	191
<b>SystemDesk</b>	193
New Features of SystemDesk 4.7.....	194
<i>New General Features</i> .....	194
<i>Configuring ECUs</i> .....	194
<i>Creating Simulation Systems for Virtual Validation</i> .....	198
Migrating to SystemDesk 4.7.....	200
<i>Migrating to SystemDesk 4.7</i> .....	200
<b>TargetLink</b>	201
New Features of TargetLink 4.2 and TargetLink Data Dictionary 4.2.....	202
Modeling in Simulink or Stateflow.....	202
<i>Newly Supported Simulink Blocks</i> .....	203
<i>Improved Data Store Blocks and Bus Support</i> .....	203
<i>Improvements to Stateflow Code Generation</i> .....	204
<i>Improvements to Model Referencing</i> .....	205
Code Generation Core Functionality.....	205
<i>MISRA C Compliance</i> .....	206
<i>Improved Code Stability</i> .....	206
<i>Improved Code Efficiency</i> .....	207
<i>Improved Enumeration Support</i> .....	211
<i>More Flexibility When Using TargetLink Custom Code         Blocks</i> .....	211
AUTOSAR.....	212
<i>Supported AUTOSAR Releases</i> .....	212
<i>Asynchronous Client-Server Communication</i> .....	213
<i>Data Transformer in Client-Server Communication</i> .....	213
<i>Non-Scalar Interrunnable Variables</i> .....	214
<i>Improved Roundtrip with SystemDesk and other         AUTOSAR Tools</i> .....	214
<i>Miscellaneous AUTOSAR Features</i> .....	214



Target Simulation (PIL).....	215
<i>High-Speed Baud Rates for PIL Simulations</i> .....	215
<i>Changes in the Target Simulation Modules</i> .....	216
Data Dictionary and Data Management.....	217
<i>Improvements to the Data Dictionary</i> .....	217
<i>Predefined Code Generator Option Sets in Data Dictionaries</i> .....	218
Code Generator Options.....	218
<i>New Code Generator Options</i> .....	218
Tool Chain Integration.....	219
<i>Modeling with RTI Bypass Blocks and Generating Code for On-Target Bypassing</i> .....	219
<i>Build Binary Files for Functional Mock-up Units</i> .....	220
Other.....	221
<i>General Enhancements and Changes</i> .....	221
<i>TargetLink Demos</i> .....	223
API Functions and Hook Scripts.....	225
<i>New API Functions</i> .....	225
<i>New Hook Scripts</i> .....	226
Migrating to TargetLink 4.2 and TargetLink Data Dictionary 4.2 ...	227
MATLAB-Related Changes.....	228
Upgrade of Models, Libraries, and Data Dictionaries.....	228
<i>Migrating to TargetLink 4.2</i> .....	228
<i>How to Upgrade a Data Dictionary with Included DD Files</i> ... ..	230
<i>How to Manually Upgrade Libraries and Models Via the API</i> .....	232
Code Generator Options.....	234
<i>Migration Aspects Regarding Code Generator Options</i> .....	234
API Functions and Hook Scripts.....	236
<i>Changes in TargetLink and TargetLink Data Dictionary API Functions</i> .....	236
AUTOSAR-Related Migration Aspects.....	238
<i>Migrating Interrunnable Variable Specifications</i> .....	238
<i>Migrating Transformer Error Code Specifications</i> .....	242
Code Changes.....	242
<i>Code Changes</i> .....	242
Other.....	254
<i>Various Migration Aspects</i> .....	254

Obsolete.....	255
<i>Discontinued TargetLink Features</i> .....	255
<i>Obsolete Limitations</i> .....	256
<i>Obsolete API Functions</i> .....	257
Changes in Future TargetLink Versions.....	258
<i>Features to Be Discontinued</i> .....	258
<i>API Functions to Be Discontinued</i> .....	259
<i>Deprecated Code Generator Options</i> .....	259
<b>VEOS</b> .....	261
<i>New Features of VEOS 3.7</i> .....	261
<i>Compatibility of VEOS 3.7</i> .....	264
<i>Migrating to VEOS 3.7</i> .....	266
<i>Discontinuations in VEOS</i> .....	269
<b>Compatibility Information</b> .....	271
<i>Supported MATLAB Releases</i> .....	271
<i>Operating System</i> .....	273
<i>Notes on 64-bit Compatibility of dSPACE Products</i> .....	275
<i>Run-Time Compatibility of dSPACE Software</i> .....	276
<i>Limitations for Using Windows Features</i> .....	277
<b>Index</b> .....	279

# About This Document

---

**Contents** This document informs you about the new features of all the dSPACE software products in Release 2016-B. It also gives you an overview of software products with no or minor changes. There are instructions on migrating from earlier dSPACE releases, especially from earlier product versions, if required.

---




**Where to go from here** Information in this section

<i>Conventions Used in the Documentation</i>	11
<i>Accessing Online Help and PDF Files</i>	13

## Conventions Used in the Documentation

---

**Admonitions** The following admonitions may be used in this document.

Admonition	Description
	Indicates a hazardous situation that, if not avoided, will result in death or serious injury.
	Indicates a hazardous situation that, if not avoided, could result in death or serious injury.
	Indicates a hazardous situation that, if not avoided, could result in minor or moderate injury.


Admonition	Description
<b>NOTICE</b>	Indicates a hazard that may cause property damage if you do not avoid it by following the instructions given.
<b>Note</b>	Indicates important information that should be kept in mind, for example, to avoid malfunctions.
<b>Tip</b>	Indicates tips containing useful information to make your work easier.


### Naming conventions


The following abbreviations and formats are used in this document:

**%name%** Names enclosed in percent signs refer to environment variables for file and path names.

< > Angle brackets contain wildcard characters or placeholders for variable file and path names, etc.

 Precedes the document title in a link that refers to another document.

 Indicates that a link refers to another document, which is available in dSPACE HelpDesk.

 Indicates that a link refers to a glossary entry.

### Special folders

Some software products use the following special folders:

**Common Program Data folder** A standard folder for application-specific configuration data that is used by all users.

```
%PROGRAMDATA%\dSPACE\\

```

or

```
%PROGRAMDATA%\dSPACE\\

```

**Documents folder** A standard folder for user-specific documents.

```
%USERPROFILE%\My Documents\dSPACE\\

```

**Local Program Data folder** A standard folder for application-specific configuration data that is used by the current, non-roaming user.

```
%USERPROFILE%\AppData\Local\dSPACE\\

```

## Accessing Online Help and PDF Files

---

**Objective** After you install your dSPACE software, the documentation for the installed products is available as online help and Adobe® PDF files.

---

**Online help** You can access the online help, dSPACE HelpDesk, as follows:

**Windows Start menu** Select Start – (All) Programs – <ProductName> – dSPACE HelpDesk (<ProductName>) to open dSPACE HelpDesk with the start page of the selected product displayed. You can also navigate and search in the user documentation of any other installed software product and its supported hardware.

**Context-sensitive** Press the **F1** key or click the Help button in the dSPACE software to get help on the currently active context.


### Note

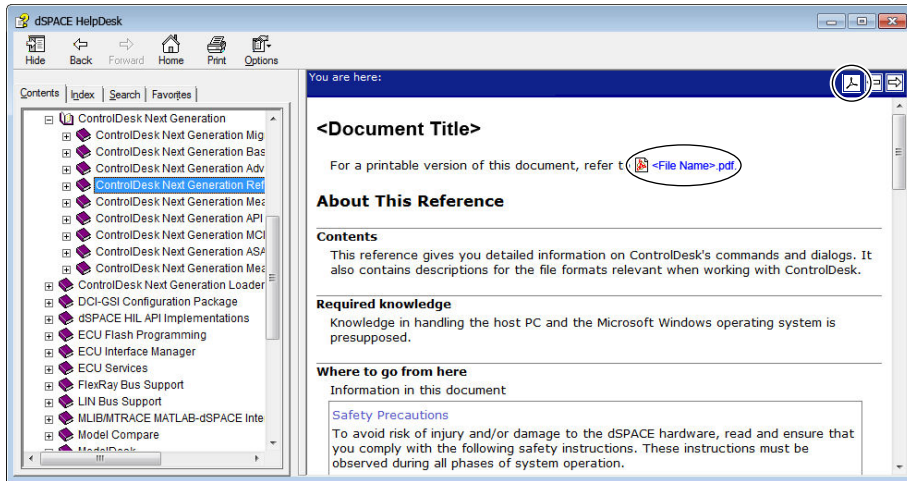
In some software products, context-sensitive help is not available.

**Help menu in the dSPACE software** On the menu bar, select Help – Contents or Help – Search (not available in all software products) to open dSPACE HelpDesk. It opens at the start page of the currently active product. You can also navigate and search in the user documentation of any other installed software product and its supported hardware.

### PDF files

You can access the PDF files as follows:

**dSPACE HelpDesk** Click the PDF link at the beginning of a document or  on a topic pane's header:



---

# Overview of dSPACE Release 2016-B

---

**Objective** Gives you an overview of the new key features in Release 2016-B and information about unchanged products.

---

**Where to go from here** Information in this section

<i>General Enhancements and Changes</i>	15
<i>Product Version Overview</i>	22
<i>New Product Key Features</i>	26

## General Enhancements and Changes

---

**Objective** The following new features and changes concern several dSPACE products.

---

**64-bit Python distribution** As of dSPACE Release 2016-B, all products are available only as 64-bit variants. As a consequence, dSPACE Release 2016-B and later support only the 64-bit variant of the Python distribution.

**Components of the dSPACE Python distribution** The component versions of the 32-bit variant and the 64-bit variant of Python 2.7 differ and some new components are available.

Python Component	32-Bit Version	64-Bit Version
Python core	2.7.10	2.7.10
PyWin32	219.10	220.10
Numpy	1.7.1	1.11.0
Matplotlib	1.2.1	1.5.1
WxPython	2.9.4.0	3.0.2.0
Py2exe	0.6.9	0.6.9
Comtypes	0.6.2	1.1.2
PIL	1.1.7	–
Python for .NET	2.0p3	2.1.0
Cycler	–	0.10.0
Pillow	–	3.2.0
Pip	–	8.1.1
Pyparsing	–	2.1.1
Python_dateutil	–	2.5.3
Pytz	–	2016.4
Six	–	1.10.0

**Using 32-bit Python and 64-bit Python in parallel** Both Python variants can be used in parallel on your computer, with the following restrictions:

- The shortcuts in the Windows Start menu and the file associations for PY, PYC and PYW files can only be set to one Python version. This is usually the latest installed Python version.
- Environment variables are used by both Python versions. Their values (e.g., for PYTHONHOME), must be set to the Python installation you want to work with. For an overview of the environment variables set by Python, refer to <http://docs.python.org/2/using/cmdline.html>.
- To switch from the 64-bit variant to the 32-bit variant of Python you have to execute the **Repair** command of the following Python components:
  - Python Core
    - Click **Repair** in the context menu of the `python-2.7.10.msi` file.



You can also start the repair for the Python 2.7.10 entry using Control Panel\Programs\Programs and Features.

- PyWin32

Execute `pywin32-219.10.win32-py2.7.exe`.

You find both files on the dSPACE DVD from dSPACE

Release 2016-A and earlier in `Disc1\Products\Common\Python2.7`.

- To switch from the 32-bit variant to the 64-bit variant of Python you have to execute the **Repair** command of the following Python components:

- Python Core

Click **Repair** in the context menu of the `python-2.7.10.amd64.msi` file.

You can also start the repair for the Python 2.7.10 entry using Control Panel\Programs\Programs and Features.

- PyWin32

Execute `pywin32-220.10.win-amd64-py2.7.exe`.

You find both files on the dSPACE DVD from dSPACE

Release 2016-B in `Disc1\Products\Common\Python2.7_x64`.

For more information, refer to *Notes on 64-bit Compatibility of dSPACE Products* on page 275.

## Contents of DVDs

The dSPACE software is provided on two disks. The disks contain the following dSPACE software packages and main products:

- Disk 1:
  - AutomationDesk 5.3
  - ControlDesk 6.0
  - TargetLink 4.2
  - Model Compare 2.7

### Note

#### Product use prohibited in United States

You are not licensed to use Model Compare in the United States. You are not allowed to use or permit others to use this product in the United States or in any way that violates the laws of the United States.

- SystemDesk 4.7

- VEOS 3.7
  - Various other dSPACE software tools
  - Disk 2:
    - RCP and HIL software
- RCP and HIL software* is a generic term for a software package containing several dSPACE software products, such as RTI, ConfigurationDesk, MotionDesk, and ModelDesk.

### Tip

Disk 2 does not contain any other dSPACE software products.

### New hardware dongles for dongle licenses

As of dSPACE Release 2014-B, the hardware dongle for dongle licenses is now a CmDongle instead of a WibuKey dongle. Both are products of WIBU-SYSTEMS and are shown below.



With dSPACE Release 2014-B, the new CmDongles are shipped with new dSPACE systems for the first time.

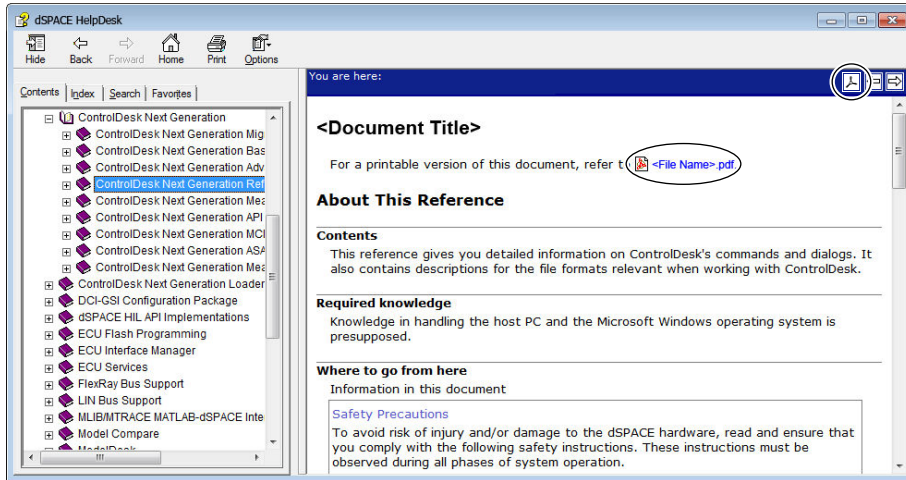
Keep the following compatibility information in mind:

- In general, you can use dSPACE Release 2016-B with an already delivered WibuKey dongle. As of dSPACE Release 2014-B, the drivers for both dongle versions are installed on your host PC. The driver software automatically detects which dongle is used. No further user action is necessary.
- If you want to use dSPACE Release 2014-A and earlier with the new CmDongle, you have to install dSPACE Installation Manager 3.8 (or later) on your host PC. This version contains the driver for the new dongle. You can download the latest version of dSPACE Installation Manager from <http://www.dspace.com/go/imupdate>.
- dSPACE Release 6.3 and earlier versions have not been tested for the new CmDongle. If necessary, contact dSPACE Support.

### Improved user documentation

The documents in dSPACE HelpDesk are also available in PDF format. As of dSPACE Release 2016-A you have faster access to these PDF files. If there is a PDF file for the currently opened topic, click [PDF](#) in the topic pane's header to open it.

For topics that are not newly published, you have still to navigate to the front page of the online book. Here, you will find the hyperlink to the related PDF file.



### Restrictions when working with dSPACE HelpDesk

dSPACE HelpDesk is installed in release-specific folders.

According to the dSPACE software products, that are now all implemented as 64-bit applications, the user documentation from dSPACE Release 2016-B is installed in C:\Program Files. Up to dSPACE Release 2016-A the user documentation was installed in C:\Program Files(x86).

Note the following restrictions:

Links to documents might not work and might return the following error message: *Selection is not associated with any topics*. The possible reasons are:

- The documents for the product are not installed, because the product is not included in your license key.
- The documents for the product are installed in another dSPACE HelpDesk. For example, if a product in the current dSPACE Release has not been changed, its user documentation is installed in the dSPACE HelpDesk version that the product setup was created for.

Each setup on dSPACE Release 2016-B installs its user documentation in dSPACE HelpDesk 2016-B.

If you are not sure where to find the user documentation for your product, use the product-specific dSPACE HelpDesk shortcut in the Windows Start menu to open the online help.

### Printed user documentation

With dSPACE Release 2016-B, the printed user documentation is not delivered automatically. You can decide which of the available printed documents you want to have. To order printed documentation, refer to <http://www.dspace.com/go/requestreleasematerial>.

#### Note

If you do not order printed documentation, use dSPACE HelpDesk or PDF files to obtain information about new features, enhancements, and the safety precautions regarding your products.

### Software support discontinuation

#### Discontinued test automation software for platform access

The following test automation software for platform access is discontinued with dSPACE Release 2016-B:

- Test Automation Python Modules no longer provides the `rtplib2` Python module.  
The `matlablib2` and `rs232lib2` Python modules are still available.
- The dSPACE Python implementation and the dSPACE .NET implementation of the HIL API MAPort are no longer available.

You can migrate your test automation projects to ASAM XIL API as the HIL API successor. The migration from HIL API .NET to XIL API .NET requires only a few modifications in your application. Refer to *Migrating HIL API Applications to XIL API Applications* ([📖 dSPACE XIL API Implementation Guide](#)). For the migration of HIL API Python and `rtplib2` to XIL API .NET, you can still use Python scripts. For more information on migration, refer to the Test Automation Tools Support Center: <http://www.dspace.com/go/pscta>.

#### Discontinuation of platform management automation

**API version 1.0 as of dSPACE Release 2016-B** Platform management automation API version 1.0 is no longer supported. You have to migrate to the API versions 2.0 or 3.0, which are now available in the dSPACE XIL API .NET setup.

For more information, refer to *Basics on the Platform Management API* ([📖 dSPACE Platform Management API Reference](#)).

**DTS V7 support in AutomationDesk** The Remote Diagnostics (COM) library is no longer supporting DTS V7. For help on migration, refer to *General Aspects on Migrating Discontinued Libraries in AutomationDesk* ([📖 AutomationDesk Guide](#)).

For further discontinuations in AutomationDesk, refer to *Migrating to AutomationDesk 5.3* on page 50.

**Discontinued methods for handling messages** As of dSPACE Release 2016-B, all dSPACE products use improved methods for handling messages such as errors and warnings.

As a consequence, messages are no longer written to the dSPACE.log file. This means that they are no longer available as ASCII text.

To collect diagnostics information and send it to dSPACE Support, you have to use the dSPACE Installation Manager.

---

**Planned discontinuation of dSPACE software**

**Discontinuation of processor-in-the-loop (PIL) simulation with VEOS and SystemDesk** As of dSPACE Release 2017-A, VEOS and SystemDesk will no longer support processor-in-the-loop (PIL) simulation. This includes the generation of V-ECUs for PIL simulation and the simulation of these V-ECUs on evaluation boards.

**Discontinuation of MPC5554 Support in ConfigurationDesk - Configuration Version** As of dSPACE Release 2017-A, ConfigurationDesk - Configuration Version will no longer support the MPC5554 microcontroller.

The MPC5554 is the core component for a RapidPro system that is used as a stand-alone prototyping ECU.

---

**Discontinuation of dSPACE hardware**

**DS1005 PPC Board** This product will be discontinued as of December 2016. New releases of dSPACE software will continue to support the DS1005 until at least the end of 2019.

However, for new projects we recommend that you use the successor, the dSPACE DS1007 PPC Processor Board.

**DS1103 PPC Controller Board** This product will be discontinued as of December 2016. New releases of dSPACE software will continue to support the DS1103 until at least the end of 2018.

However, for new projects we recommend that you use the successor, dSPACE MicroLabBox.

**MicroAutoBox II 1401/1511/1512 and MicroAutoBox II 1401/1512/1513** MicroAutoBox variants with DS1512 I/O Board will be discontinued as of December 2016. New releases of dSPACE software will continue to support these MicroAutoBox variants until at least the end of 2019. However, for new projects we recommend that you use the successors MicroAutoBox II 1401/1511/1514 or MicroAutoBox II 1401/1513/1514.

**MicroAutoBox** MicroAutoBox (board revision DS1401-19 and earlier) was discontinued in December 2013. The software support has been discontinued with dSPACE Release 2016-A. However, the

I/O boards that were used with MicroAutoBox are still supported with MicroAutoBox II.

New releases of dSPACE software will continue to support the following MicroAutoBox II variants until at least the end of 2018:

- MicroAutoBox II 1401/1501
- MicroAutoBox II 1401/1504
- MicroAutoBox II 1401/1505/1507

For new projects we recommend that you use MicroAutoBox II variants with the I/O board successors DS1511, DS1513, and DS1514. MicroAutoBox II 1401/1507 is still available.

## Product Version Overview

### Objective

The following table is an extract from product version histories showing the product versions of the current release and of the three preceding releases. If a product has new features, there is a link to the brief description in this document.

Product	dSPACE Release			
	2015-A	2015-B	2016-A	2016-B
AutomationDesk	5.0	5.1	5.2	5.3 Refer to <i>AutomationDesk</i> on page 47.
Automotive Simulation Models	8.0	8.1	8.2	8.3 Refer to <i>Automotive Simulation Models (ASM)</i> on page 55.
Bus Manager (stand-alone)	–	–	5.5	5.6 Refer to <i>Bus Manager (Stand-Alone)</i> on page 113.
ConfigurationDesk	5.3	5.4	5.5	5.6 Refer to <i>ConfigurationDesk</i> on page 115.
Container Manager	4.3	4.4	4.4	4.5

Product	dSPACE Release			
	2015-A	2015-B	2016-A	2016-B
ControlDesk	5.4	5.5	5.6	6.0 Refer to <i>ControlDesk</i> on page 121.
DCI Configuration Tool	3.4	3.5	3.6	3.7 Refer to <i>DCI Configuration Tool</i> on page 145.
dSPACE CAN API Package	2.7.1	2.7.4	2.7.5	3.0 Refer to <i>dSPACE CAN API Package</i> on page 147.
dSPACE ECU Flash Programming Tool	2.2.6	2.2.6	2.3	2.3.1 Refer to <i>dSPACE ECU Flash Programming Tool</i> on page 149.
dSPACE FlexRay Configuration Package	3.5	3.6	3.7	3.8 Refer to <i>dSPACE FlexRay Configuration Package</i> on page 151.
dSPACE HIL API .NET	1.8	2.0	2.1	-
dSPACE Python Extensions	1.8	2.0	2.1	2.2 Refer to <i>dSPACE Python Extensions</i> on page 153.
dSPACE XIL API .NET	2015-A	2015-B	2016-A	2016-B Refer to <i>dSPACE XIL API .NET</i> on page 155.
ECU Interface Manager	1.6	1.7	1.8	2.0 Refer to <i>ECU Interface Manager</i> on page 159.
Firmware Manager	1.3	2.0	2.1	2.2 Refer to <i>Firmware Manager</i> on page 161.
Model Compare	2.5	2.6	2.6	2.7 Refer to <i>Model Compare</i> on page 163.
ModelDesk	4.1	4.2	4.3	4.4 Refer to <i>ModelDesk</i> on page 165.

Product	dSPACE Release			
	2015-A	2015-B	2016-A	2016-B
Model Interface Package for Simulink	3.0	3.1	3.2	3.3 Refer to <i>Model Interface Package for Simulink</i> on page 167.
MotionDesk	3.6	3.7	3.8	3.9 Refer to <i>MotionDesk</i> on page 171.
MotionDesk Blockset	2.3.2	2.4	2.4.1	2.5 Refer to <i>MotionDesk</i> on page 171.
Real-Time Testing	2.5	2.6	3.0	3.1 Refer to <i>Real-Time Testing</i> on page 173.
RTI <sup>1)</sup>	7.4	7.5	7.6	7.7 Refer to <i>RTI/RTI-MP and RTLib</i> on page 175.
RTI-MP <sup>2)</sup>	7.4	7.5	7.6	7.7 Refer to <i>RTI/RTI-MP and RTLib</i> on page 175.
RTI Bypass Blockset	3.4	3.5	3.6	3.7 Refer to <i>RTI Bypass Blockset</i> on page 179.
RTI CAN Blockset	3.4	3.4.1	3.4.2	3.4.3
RTI CAN MultiMessage Blockset	4.1	4.2	4.3	4.4 Refer to <i>RTI CAN MultiMessage Blockset</i> on page 181.
RTI Electric Motor Control Blockset	1.1	1.2	1.3	1.3.1
RTI Ethernet Blockset	1.2	1.2	1.2	1.2.1
RTI Ethernet (UDP) Blockset	1.4	1.4	1.4	1.4.1
RTI FPGA Programming Blockset	2.9	3.0	3.1	3.2 Refer to <i>RTI FPGA Programming Blockset</i> on page 183.
RTI LIN MultiMessage Blockset	2.5	2.5.1	2.6	2.7 Refer to <i>RTI LIN MultiMessage Blockset</i> on page 187.
RTI RapidPro Control Unit Blockset	2.2.1	2.2.1	2.2.1	2.2.2



Product	dSPACE Release			
	2015-A	2015-B	2016-A	2016-B
RTI USB Flight Recorder Blockset	1.2	1.2	1.2	1.2.1
RTI Watchdog Blockset	1.0	1.0	1.0	2.0 Refer to <i>RTI Watchdog Blockset</i> on page 189.
SCALEXIO firmware	3.2	3.3	3.4	3.5 Refer to <i>SCALEXIO Firmware</i> on page 191.
SYNECT server	1.4.1	1.4.1	1.4.1	1.4.1
SystemDesk	4.4	4.5	4.6	4.7 Refer to <i>SystemDesk</i> on page 193.
TargetLink/TargetLink Data Dictionary	4.0	4.1	4.1	4.2 Refer to <i>TargetLink</i> on page 201.
Variable Editor	2.1	2.2	2.3	2.3 <sup>3)</sup>
VEOS	3.4	3.5	3.6	3.7 Refer to <i>VEOS</i> on page 261.

<sup>1)</sup> Including the standard I/O blocksets.

<sup>2)</sup> Including the RTI Gigalink Blockset.

<sup>3)</sup> The Variable Editor is no longer part of the dSPACE Release DVD. It is available at <https://www.dspace.com/go/requestreleasedownload>.

If you have not updated regularly, refer to the *New Features and Migration* documents for the dSPACE Releases listed above for information about the new features and necessary migration steps.

## New Product Key Features

**Objective** This is an overview of each product's new key features. For more information, refer to the product-specific sections.

**Information in this topic**

*AutomationDesk* on page 26  
*Bus Manager (stand-alone)* on page 27  
*ConfigurationDesk (Implementation Version)* on page 27  
*ControlDesk* on page 27  
*DCI Configuration Tool* on page 29  
*dSPACE CAN API Package* on page 29  
*dSPACE ECU Flash Programming Tool* on page 29  
*dSPACE XIL API* on page 30  
*Firmware Manager* on page 30  
*Model Compare* on page 30  
*ModelDesk* on page 30  
*MotionDesk* on page 30  
*Python Extensions* on page 31  
*Real-Time Testing* on page 31  
*RTI, RTI-MP and RTLib* on page 31  
*RTI Bypass Blockset* on page 31  
*RTI CAN MultiMessage Blockset* on page 31  
*RTI FPGA Programming Blockset* on page 31  
*RTI LIN MultiMessage Blockset* on page 32  
*RTI Watchdog Blockset* on page 32  
*SCALEXIO firmware* on page 32  
*SystemDesk* on page 32  
*TargetLink* on page 32  
*VEOS* on page 33

**AutomationDesk**

The new key features of AutomationDesk are:

- New automation blocks in the XIL API Convenience library to trigger EESPort errors by software.
- New automation block in the Evaluation library to create a Signal data object from an ASAM MDF 4.1 file.
- Enhancements to the COM API, such as getting the path of the generated result file in XML format.
- Enhancements to the user documentation.

For more information on the new features, refer to *New Features of AutomationDesk 5.3* on page 47.

**Bus Manager (stand-alone)**

The new key features of the Bus Manager (stand-alone) are:

- Support of SAE J2602 standard
- Support of FIBEX 4.1.1 files

For more information, refer to *Features of the Bus Manager (Stand-Alone) 5.6* on page 113.

**ConfigurationDesk  
(Implementation Version)**

The new key features of ConfigurationDesk are:

- Adding bus simulation container files (BSC files) to your ConfigurationDesk application
- Support of ECU interfacing
- Extended function blocks: SENT In, SENT Out, CAN
- Bus Manager:
  - Support of the SAE J2602 standard for LIN communication
  - Support of FIBEX files based on FIBEX 4.1.1 as communication matrices

For more information , refer to *ConfigurationDesk – Implementation* on page 116.

**ControlDesk**

The new key features of ControlDesk 6.0 are:

- General enhancement:
  - 64-bit application
  - New packaging of ControlDesk
- Project/experiment management enhancements:
  - Sorting folders and folder groups

For more information on the new features, refer to *New General Features (ControlDesk 6.0)* on page 123.

For more information on the new features, refer to *New Project and Experiment Features (ControlDesk 6.0)* on page 123.

- Platform/device enhancements:
  - Support of the new DCI-CAN/LIN1
  - SCALEXIO platform: Display of Ethernet and PCI/PCIe cards
  - SCALEXIO platform: Display of connected but unregistered systems
  - Display of additional important information when adding a platform/device

For more information on the new features, refer to *New Features of Platform Management and Platforms/Devices (ControlDesk 6.0)* on page 124.

- Instrument enhancements:
  - Enhanced Table Editor
  - Enhancements for the Instrument Selector
  - New STRUCT instrument caption macro

For more information on the new features, refer to *New Instrument Features (ControlDesk 6.0)* on page 126.

- Measurement and recording enhancements:
  - Improved handling of large ASAM MDF 4.1 files
    - Improved loading of signals
    - Support of reduction data
  - Removing variables from measurement data files

For more information on the new features, refer to *New Measurement and Recording Features (ControlDesk 6.0)* on page 130.

- Message handling enhancements:
  - New Message Viewer
  - Viewing the dSPACE Log

For more information on the new features, refer to *New Message Handling Features (ControlDesk 6.0)* on page 131.

- Bus Navigator enhancements:
  - Bus Instrument (TX Type for FlexRay): Support of the minimum delay time
  - Loading/unloading bus signals

For more information on the new features, refer to *New Bus Navigator Features (ControlDesk 6.0)* on page 131.

- ECU Diagnostics enhancements:
  - Support of DoIP (Diagnostics over Internet Protocol)
  - Info if default COMPARAM values are used

For more information on the new features, refer to *New ECU Diagnostics Features (ControlDesk 6.0)* on page 132.

- Electrical error simulation (failure simulation) enhancements:

- Support of software triggers
- Automatic potential mapping
- Automatic signal mapping

For more information on the new features, refer to *New Electrical Error Simulation Features (ControlDesk 6.0)* on page 133.

- Signal Editor enhancements:

- Support of the eLINEAR interpolation method
- Automatic refresh of Data File segment display during download

For more information on the new features, refer to *New Signal Editor Features (ControlDesk 6.0)* on page 134.

---

#### **DCI Configuration Tool**

The new key features of the DCI Configuration Tool are:

- Improved A2L file adaptation
- Manual ECU memory access

For more information on the new features, refer to *New Features of the DCI Configuration Tool 3.7* on page 145.

---

#### **dSPACE CAN API Package**

The new key features of the dSPACE CAN API Package are:

- Support of CAN FD (provided by the new dSPACE CAN API 2.0 only)
- Support of the DCI-CAN/LIN1 interface (provided by the dSPACE CAN API 1.0 and the new dSPACE CAN API 2.0)

For more information on the new features, refer to *New Features of dSPACE CAN API Package 3.0* on page 147.

---

#### **dSPACE ECU Flash Programming Tool**

The new key features of the dSPACE ECU Flash Programming Tool are:

- Support of the DCI-CAN/LIN1 interface
- Loading XCP on CAN flash projects from versions earlier than 2.3 without any modification

For more information on the new feature, refer to *New Features of the dSPACE ECU Flash Programming Tool 2.3.1* on page 149.

---

### dSPACE XIL API

The new key features of dSPACE XIL API are:

- Initial support of the Framework implementation
- Enhanced functionalities to the EESPort for triggering an error by software

For more information on the new features, refer to *New Features of dSPACE XIL API .NET 2016-B* on page 155.

---

### Firmware Manager

The new key feature of the Firmware Manager is:

- Direct access to the New Features and Migration document and to the PDF files via the Help ribbon in the Backstage view

For more information on the new feature, refer to *New Features of Firmware Manager 2.2* on page 161.

---

### Model Compare

The new key features of Model Compare are:

- Full, qualified support of Windows 10
- Three-way analysis with a common ancestor model
- Improved interaction with version control systems

For more information on the new features, refer to *New Features of Model Compare 2.7* on page 163.

---

### ModelDesk

The new key features of ModelDesk are:

- Start, stop, and reset of driving maneuver and driving cycles
- New Time Plotter and XY Plotter
- New Table Editor
- Roads can be exported in the OpenDRIVE format

For more information on the new features, refer to *New Features of ModelDesk 4.4* on page 165.

---

### MotionDesk

The new key features of MotionDesk are:

- Simulating real-time lens distortion
- Specifying the accuracy of road generation
- Configuration of vivid texturing

For more information on the new features, refer to *New Features of MotionDesk 3.9* on page 171.

---

---

**Python Extensions**

There are no new features for the products contained in Python Extensions, but there are the following discontinuations:

- dSPACE HIL API Python Implementation
- rtplib2 of Test Automation Python Modules

For more information on migrating, refer to *Migrating to dSPACE Python Extensions 2.2* on page 153.

---

**Real-Time Testing**

The new key features of Real-Time Testing are:

- The real-time Python interpreter is based on Python 2.7.11
- The `rttlib.hostcalls` module supports all the platforms supported by Real-Time Testing
- The `rttlib.datastream` module supports new modes for replay

For more information on the new features, refer to *New Features of Real-Time Testing 3.1* on page 173.

---

**RTI, RTI-MP and RTLib**

The new key features of RTI, RTI-MP and RTLib are:

- Support of MATLAB R2016b
- API for coding the task configuration in an M file

For more information on the new feature, refer to *New Features of RTI/RTI-MP and RTLib* on page 175.

---

**RTI Bypass Blockset**

The new key feature of the RTI Bypass Blockset is:

- TargetLink support for on-target bypassing

For more information on the new features, refer to *New Features of the RTI Bypass Blockset 3.7* on page 179.

---

**RTI CAN MultiMessage Blockset**

The new key feature of the RTI CAN MultiMessage Blockset is:

- Support of FIBEX 4.1.1

For more information on the new features, refer to *New Features of the RTI CAN MultiMessage Blockset 4.4* on page 181.

---

**RTI FPGA Programming Blockset**

The new key features of the RTI FPGA Programming Blockset are:

- Extended Xilinx® software support
- Enhancements to the FPGA frameworks for the DS2655 FPGA Base Board and I/O modules.
- Enhancements to the FPGA framework for MicroLabBox.

For more information on the new features, refer to *New Features of the RTI FPGA Programming Blockset 3.2* on page 183.

---

---

<b>RTI LIN MultiMessage Blockset</b>	<p>The new key feature of the RTI LIN MultiMessage Blockset is:</p> <ul style="list-style-type: none"><li>■ Support of FIBEX 4.1.1</li></ul> <p>For more information on the new features, refer to <i>New Features of the RTI LIN MultiMessage Blockset 2.7</i> on page 187.</p>
<b>RTI Watchdog Blockset</b>	<p>The RTI Watchdog Blockset has been enhanced by the Challenge-Response Monitoring blockset.</p> <p>For more information on the new feature, refer to <i>Features of RTI Watchdog Blockset 2.0</i> on page 189.</p>
<b>SCALEXIO firmware</b>	<p>The SCALEXIO firmware supports the new SCALEXIO_RTLIB_MC8 and SCALEXIO_FIU_500 licenses.</p> <p>For more information on the new features, refer to <i>New Features of the SCALEXIO Firmware 3.5</i> on page 191.</p>
<b>SystemDesk</b>	<p>The new key feature of SystemDesk 4.7 is:</p> <ul style="list-style-type: none"><li>■ SystemDesk now provides an <i>ECU configuration framework</i> that you can use to configure ECUs with basic software of any vendor.</li></ul> <p>For more information on the new features, refer to <i>New General Features</i> on page 194.</p>
<b>TargetLink</b>	<p>The new key features of TargetLink are:</p> <ul style="list-style-type: none"><li>■ Operating system<ul style="list-style-type: none"><li>■ Full, qualified support of Windows 10</li></ul></li><li>■ AUTOSAR<ul style="list-style-type: none"><li>■ Revision 4.2.2 support</li><li>■ Asynchronous client-server communication</li><li>■ Data Transformer in client-server communication</li><li>■ Non-Scalar interrunable variables</li><li>■ Improved roundtrip with SystemDesk</li><li>■ Improved ARXML import and export</li></ul></li><li>■ Stateflow<ul style="list-style-type: none"><li>■ Support for Superstep semantics</li><li>■ Support for (graphical) functions with multiple output data</li><li>■ Improved support for monitoring state activities (Stateflow active state data)</li></ul></li></ul>

---



- Block-specific improvements
  - Support for element selection and for bus signals in Data Store Memory blocks
  - Support of the Vector Concatenate block
  - Support of Enable blocks on model root level (controlling the execution of referenced models)
- Data Dictionary
  - Support for the comparison of Data Dictionaries with the DD Comparison pane launched via Windows Command Prompt
  - Support of custom command definition and calls
- On-target prototyping with dSPACE's RTI Bypass Blocks including full code generation for on-target bypassing
- Improved support of Simulink enumerations (including Stateflow) in the generated code
- Instance-specific replacement of user-defined strings in custom code files
- Improved MISRA C compliance for the generated production code and the Fixed-Point Library

For more information on all the new features, refer to *New Features of TargetLink 4.2 and TargetLink Data Dictionary 4.2* on page 202.

For more information on the TargetLink migration aspects (TargetLink, TargetLink AUTOSAR module, TargetLink Data Dictionary), refer to *Migrating to TargetLink 4.2 and TargetLink Data Dictionary 4.2* on page 227.

---

## VEOS

The new key features of VEOS are:

- Hierarchical display of VPU ports
- Connecting VPU ports by dragging structural elements
- Enabling/disabling logging of FMU-internal messages
- Getting version information on VEOS Player via automation

For more information on the new features, refer to *New Features of VEOS 3.7* on page 261.



---

# Aspects of Migrating from Previous Releases

---

## Objective

After you install products of the current dSPACE Release, some additional steps might be necessary. The migration steps required when you come from the last dSPACE Release are described in the product-specific migration topics in this document. If you come from an older dSPACE Release, refer to the related *New Features and Migration* document.

## Migrating to dSPACE Release 2016-B

---

### Objective

After you install Release 2016-B, some additional steps might be necessary.

---

### Migrating from dSPACE Release 2016-A

**Product-specific migration steps** Product-specific migration steps are usually performed automatically by the products. For exceptions, refer to the product-specific migration descriptions.

---

### Migrating from dSPACE Release 2015-B or earlier

To migrate from dSPACE Release 2015-B or earlier to Release 2016-B, you also have to perform the migration steps of the intervening dSPACE Releases. All of the required migration steps can be performed with Release 2016-B installed.

For more information on the required migration steps, refer to the *New Features and Migration* documents of the intervening dSPACE Releases.

### Previous release documents

The PDF files of previous releases are called `NewFeaturesAndMigrationxx.pdf`, where `xx` stands for the release number.

You can find the *New Features and Migration* files for previous releases at the following locations:

- In the installation folder of the current dSPACE HelpDesk. Refer to `C:\Program Files\Common Files\dSPACE\HelpDesk 2016-B\Print\PreviousReleases`.
- On the dSPACE DVDs. Refer to `\Doc\Print\PreviousReleases`.
- At [www.dspace.com/go/migration](http://www.dspace.com/go/migration) for download. Here you can also find *New Features and Migration* documents for very early releases.

---

# Changes to TRC File Generation

---

## Where to go from here

Information in this section

<i>Basics on the TRC File Changes</i>	37
Basics on the changes of the TRC file generation.	
<i>Migrating Changes in Software That Generates TRC Files</i>	43
Information on required manual migration.	
<i>Migrating Changes in Software That Uses TRC Files</i>	44
Information on required manual migration.	

---

## Basics on the TRC File Changes

---

### Objective

The enhanced code generation leads to improvements for the simulation behavior of the executable application. To profit from these improvements in dSPACE software, the TRC file generation was enhanced.

---

### Enhancements in the generated TRC file

With the MATLAB/Simulink R2014a release, the enhanced code generation by Simulink® Coder™ was introduced to optimize the simulation behavior. It provides a simpler behavior for tuning all parameters and support for referenced models. Additional Simulink Coder functions introduced with MATLAB R2015b now allow dSPACE to fully support these new features via the enhanced TRC file generation.

The main advantages of the enhanced TRC file generation are:

- Same view of model parameters in MATLAB workspace and TRC file.

All tunable model parameters defined by MATLAB workspace variables are available in the top-level Tunable Parameters group in the TRC file. This lets you access global parameters very quickly and independently of the model hierarchy. Modifying the model hierarchy later on will not affect the variable path already specified for layout connections or test scripts.

- Working with MATLAB structures

If a MATLAB structure is tunable according to the Simulink Coder rules, the structure levels and structure fields are generated into the code.

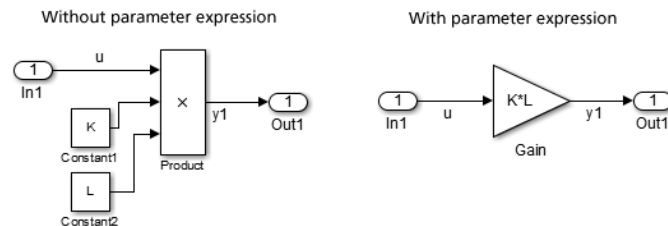
This means:

- Structured parameters are available in the TRC file.
  - Non-virtual Simulink buses are represented more efficiently in the TRC file.
  - Bus arrays are available in the TRC file.
- Higher performance

For non-virtual Simulink buses, the performance of code generation and compiling will be highly increased.

- More compact models by using tunable parameter expressions

Complex workarounds for modeling parameter expressions can be simplified, for example, as shown in the model below. The MATLAB workspace variables K and L are automatically generated as tunable parameters.



- Handling of global parameters for Default parameter behavior = Tunable or Inlined (formerly Inline Parameters option off and on)

The mapping between the configured tunable workspace variables and Simulink.Parameter objects, and variables in the generated code does not depend on the Default parameter behavior option (formerly Inline Parameters option).

- Improved model referencing support

Simulink referenced models were limited to having the Inline Parameters option set to On (MATLAB R2015b and later: Default parameter behavior set to Inlined). Now, the dSPACE tool chain also supports the Default parameter behavior option set to Tunable for referenced models when MATLAB R2015b or later is used.

- Support of Simulink mask parameters

Simulink mask parameters are now available in the TRC file and can be accessed by dSPACE software, such as ControlDesk.

- Same behavior of Simulink simulation and simulations running on dSPACE platforms

As a result of the above mentioned enhancements for consistent parameter tuning, the behavior of a Simulink simulation and a simulation on dSPACE platforms will be the same.

---

### Support of Simulink Coder enhancements

For the support of the coder enhancements in the generated TRC file, MathWorks and dSPACE together developed additional build functionality, which was released with MATLAB R2015b. The resulting additions of the TRC file syntax required complex modifications in all the TRC file-generating and TRC file-consuming dSPACE products.

The full support of these enhancements is available as of dSPACE Release 2015-B used with MATLAB R2015b. If you use dSPACE Release 2015-B or later with an earlier MATLAB version, the code generation mainly remains the same as with earlier dSPACE releases.

No migration is required if you change the dSPACE Release but keep the MATLAB Release.

For an overview of the different behavior when using the current dSPACE Release, refer to the following table:

dSPACE Release 2016-B used with MATLAB ...			
R2015a	R2015b	R2016a	R2016b
<p>Sharing the same parameter variable across multiple blocks is supported for block parameters that are defined with an <i>unstructured</i> MATLAB workspace variable and <i>without</i> expressions. All other block parameter definitions have the same behavior as with the previous MATLAB releases R2013b or R2014a.</p> <p>Internal adaptations to the coder changes are done automatically.</p> <p>Using the Inline Parameters option set to <code>off</code> for referenced models is not supported.</p>	<p>Full support of the above mentioned Simulink Coder features.</p>		
	<p>The standard Simulink Coder behavior is used.</p>		
			<p>Using the Inline Parameters option set to <code>off</code> for referenced models is supported.<sup>1)</sup></p>

<sup>1)</sup> As of MATLAB R2015b this setting is similar to the Default parameter behavior option set to `Tunable`.

### Details on the TRC file changes introduced with MATLAB R2015b

The following changes were introduced with dSPACE Release 2015-B and MATLAB R2015b.

**Model Root group** The entries in the Model Root group have changed as follows:

- To improve performance and usability, entries for virtual Simulink buses and muxed signals (e.g., `Out1{SubArray1}`) are no longer generated into the variable description.

This also applies to the labels of these signals.

This is an incompatible change that requires manual migration. Refer to *Migrating Changes in Software That Generates TRC Files* on page 43.

- Entries for non-virtual Simulink buses are now generated as one structured variable in the variable description, e.g., `Out1{MyField}` has changed to `Out1.MyField`.

This also applies to the labels of non-virtual Simulink buses.

This is an incompatible change that requires manual migration. Refer to *Migrating Changes in Software That Generates TRC Files* on page 43.

- Simulink mask parameters are now generated into the variable description at the entries of the related masked subsystems.



- Input signals of signal sink blocks are now generated into the variable description also when you use ConfigurationDesk or VEOS for the build process.
- The Include states and Include derivatives options are now also available for ConfigurationDesk and VEOS.

**Tunable Parameters group** The entries in the Tunable Parameters group have changed as follows:

- MATLAB workspace variables and Simulink.Parameter objects, which are used as block parameters in the model, are now generated as global variables in the Tunable Parameters group. Internal optimizations during code generation might be the reason that a variable will not be generated into the variable description.

If a block's parameter definition contains an expression, the local block parameter is no longer available. This is an incompatible change that requires manual migration. Refer to *Migrating Changes in Software That Generates TRC Files* on page 43.

- Structured workspace variables and Simulink.Parameter objects that are used as block parameters in the model are now generated as global structured parameters in the Tunable Parameters group.

The structure has to fulfill the Simulink Coder conditions for a tunable structured parameter.

- Previously, each referenced model of a model referencing hierarchy had its own Tunable Parameters group. These groups are no longer generated.

All global parameters referenced in the top-level model or in the referenced models are generated into the Tunable Parameters group of the top-level model.

This is an incompatible change that requires manual migration. Refer to *Migrating Changes in Software That Generates TRC Files* on page 43.

**Handling n-D look-up tables** As of dSPACE Release 2015-B Look-Up Table blocks with a dimension higher than 2, such as a 4x3x2 matrix, are no longer automatically divided into two-dimensional slices.

This is an incompatible change that requires manual migration. Refer to *Migrating Changes in Software That Generates TRC Files* on page 43.

**Data Stores group** To improve performance and data consistency with other blocks, the Data Stores group is no longer generated into the variable description.

This is an incompatible change that requires manual migration. Refer to *Migrating Changes in Software That Generates TRC Files* on page 43.

**Structured variables** Structured variables, such as non-virtual buses or tunable structured parameters, are generated into the code and represented in the variable description as a `struct` element. The hierarchy of fields and members in a structured element is described in dot notation, for example, `myStruct.mySubstruct.myValue[0]`.

**References** A variable description now contains block parameters as references. The source of a reference can be a global parameter: e.g., a MATLAB workspace variable available in the Tunable Parameters group, or a mask parameter. For structured parameters, the reference can specify a field of a structure.

### Note

For the support of structures and references, the following keywords have been added to the TRC file syntax:

- `array-incr`
- `offs`
- `struct`
- `endstruct`
- `refvar`
- `refgroup`
- `refelem`
- `DEPRECATED`

If you have used one of these keywords as a variable name, it is detected during the generation of the TRC file and not added to the file. There might be definitions in user code that you must check. Otherwise, there might be an error in the software that uses the TRC file.

### Up-to-date information

For more information on TRC file generation and the latest migration instructions, refer to the dSPACE website:  
<http://www.dspace.com/go/trc>.

## Migrating Changes in Software That Generates TRC Files

<b>Objective</b>	Despite the complex changes in the code generation, only a few manual migrations are required. Most of the changes based on the enhancements are automatically migrated by the dSPACE products.
<b>Using MATLAB R2015a SP1 with dSPACE Release 2016-B</b>	<p>There is no manual migration required. The variable description contains further global parameters in the Tunable Parameters group for unstructured workspace variables. These global parameters are shared with the corresponding block parameters if the block parameter is not defined with an expression. Writing a new value to one of the global parameters changes the related block parameters too.</p> <p>Using the Inline Parameters option set to <code>off</code> for referenced models is not supported.</p>
<b>Using MATLAB R2015b with dSPACE Release 2016-B</b>	<p>If you use MATLAB R2015b, the new Simulink Coder features are fully supported. The incompatible changes require migration steps that are described below in general. Detailed instructions are not given, because they depend on various conditions such as the complexity of your model, the software you are using, and the internal structure of your test scripts, for example. There are therefore only some basic examples to show a general way to migrate.</p> <p>For more information, refer to <a href="http://www.dspace.com/go/trc">http://www.dspace.com/go/trc</a>.</p>
<b>Using MATLAB R2016a with dSPACE Release 2016-B</b>	Same notes as with MATLAB R2015b.
<b>Using MATLAB R2016b with dSPACE Release 2016-B</b>	Same notes as with MATLAB R2015b.
<b>Migration steps required in TRC file generating software</b>	<p>dSPACE products that generate TRC files such as RTI, ConfigurationDesk and VEOS, support the new Simulink Coder enhancements as is. There are only the following changes that might require a manual migration to provide information in the variable description.</p> <p><b>Update assertion mode (only RTI)</b> The <code>rtiAssertionMode</code> variable is no longer generated into the variable description. The Assertion mode setting on the RTI simulation options page is still available for configuring the mode before you start the build process.</p>

**Update access to Data Stores group** The Data Stores group is no longer generated into the variable description. Instead of using Data Store Memory blocks, you have to use Data Store Read blocks for read access or the combination of Constant blocks with Data Store Write blocks for write access. Instead of the entries in the Data Stores group, you then find entries of the Data Store Read blocks or the Constant blocks in the Model Root group.

For this migration step, it is not required to use dSPACE Release 2015-B or later. You can also do it with earlier dSPACE releases.

## Migrating Changes in Software That Uses TRC Files

---

### Objective

Products that use TRC files, such as ControlDesk, use the generated variable description to connect elements in the software with variables in the simulation application. Most of the variable path modifications caused by the TRC file changes can be automatically migrated by the dSPACE products, but for some changes you have to do manual migration in your software product.

---

### Migration steps required in TRC file consuming software

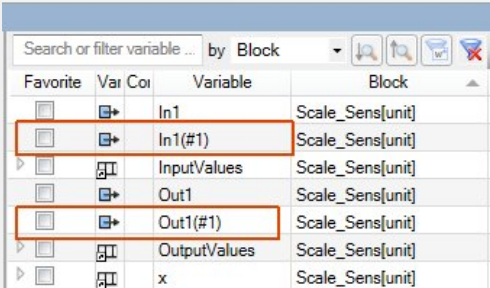
If you have already used ControlDesk, AutomationDesk, or test scripts of any kind that are accessing variables via their variable paths and you rebuild the simulation application with MATLAB R2015b or later, you have to check whether the variable paths have been discontinued or changed in the variable description.

If you are using ControlDesk, there is support for finding inconsistent connections: for example, via specially marked instruments or the `Check Mapping` command in the Signal Editor. In AutomationDesk, variable access is realized via variable aliases. Therefore, modifications in the variable description cannot be automatically recognized. However, if you are using a variable pool in your project, it is sufficient to update this.

To graphically support the new TRC file features, such as structures and references, ControlDesk and AutomationDesk provide the new Variable Browser.

If it is not possible to automatically migrate certain changes to the software, you have to perform the following manual migration.

Issue	Migration Step
Update variable paths of parameters with expressions	Update any connections in ControlDesk or variables defined in test scripts that contain expressions with MATLAB workspace variables, mask parameters, or Simulink Parameter objects. Usually, it is not sufficient to only change the variable path to the generated global parameter to get the required variable access for controlling. You also have to consider any element of the expression or the resulting variable of the block.
Update variable paths of virtual Simulink buses	Update any connections in ControlDesk and test scripts accessing signals within a virtual Simulink bus to directly accessed signal source blocks. As an alternative, you can add a Bus Selector block to your model and then connect the block's output variables.
Update variable paths of nonvirtual Simulink buses	Update connections in ControlDesk and test scripts that access signals within a nonvirtual Simulink bus to the corresponding field of the structured variable. The formerly generated measurement arrays in the variable description are now represented by struct elements.
<div style="background-color: #f0f0f0; padding: 10px;"> <p><b>Note</b></p> <p>The syntax of structured elements has changed from <code>Out1{myField.mySubField}</code> to <code>Out1.myField.mySubField</code>. This might conflict with variable names containing dots.</p> </div>	
Update variable paths of Tunable Parameters groups for referenced models	Update any connections in ControlDesk or variables defined in test scripts that refer to tunable parameters of referenced models. The variable path of such a variable must be changed to the top-level Tunable Parameters group.
Update access to Data Stores group	Update any connections in ControlDesk and test scripts that refer to the variables of the discontinued Data Stores group to the variables of the inserted Data Store Read or Write blocks in your model.
Update connections to look-up tables	<ul style="list-style-type: none"> <li>■ ControlDesk does not recognize all the look-up tables in a TRC file. As a result, these look-up tables are not available as maps or curves, for example, in ControlDesk's Variable Browser. The recognition of look-up tables does not work in the following cases: <ul style="list-style-type: none"> <li>■ The table data of the look-up table is contained in a structured parameter.</li> <li>■ The table data of the look-up table references mask parameters.</li> <li>■ The look-up table references table data, for example, in the Tunable Parameters group. In this case, ControlDesk recognizes only the first look-up table that references this table data. All the other look-up tables are not recognized.</li> <li>■ The look-up table has three or more dimensions.</li> </ul> </li> </ul> <p>To update the connection to such a look-up table, connect the individual variables of the look-up table in these cases.</p> <ul style="list-style-type: none"> <li>■ ControlDesk no longer provides a map or curve for the <code>tableData</code> parameter of a look-up table if the <code>tableData</code> parameter was parameterized with numeric values.</li> </ul>

Issue	Migration Step																																
	<p>To update the connection to such a parameter, connect the <code>LookUpTableData</code> variable of the look-up table (instead of the <code>tableData</code> parameter).</p> <p><b>Note</b></p> <ul style="list-style-type: none"> <li>ControlDesk creates a <i>duplicate variable</i> for a variable that is referenced by the table data or an axis of a look-up table. Compared to the name of the original variable, the name of the duplicate variable is extended by (#1).</li> </ul> <p>Do not connect such a duplicate variable to a ControlDesk instrument. The following illustration shows an example of duplicate variables:</p>  <p>The screenshot shows a table with the following data:</p> <table border="1"> <thead> <tr> <th>Favorite</th> <th>Var Cor</th> <th>Variable</th> <th>Block</th> </tr> </thead> <tbody> <tr> <td><input type="checkbox"/></td> <td></td> <td>In1</td> <td>Scale_Sens[unit]</td> </tr> <tr> <td><input type="checkbox"/></td> <td></td> <td>In1(#1)</td> <td>Scale_Sens[unit]</td> </tr> <tr> <td><input type="checkbox"/></td> <td></td> <td>InputValues</td> <td>Scale_Sens[unit]</td> </tr> <tr> <td><input type="checkbox"/></td> <td></td> <td>Out1</td> <td>Scale_Sens[unit]</td> </tr> <tr> <td><input type="checkbox"/></td> <td></td> <td>Out1(#1)</td> <td>Scale_Sens[unit]</td> </tr> <tr> <td><input type="checkbox"/></td> <td></td> <td>OutputValues</td> <td>Scale_Sens[unit]</td> </tr> <tr> <td><input type="checkbox"/></td> <td></td> <td>x</td> <td>Scale_Sens[unit]</td> </tr> </tbody> </table> <ul style="list-style-type: none"> <li>If a value block in the Tunable Parameters group is referenced by the table data or an axis of a look-up table, ControlDesk uses an incorrect variable type for this value block. Instead of the <i>value block</i> variable type, it uses one of the following variable types:             <ul style="list-style-type: none"> <li>Map</li> <li>Curve</li> <li>Common axis</li> </ul> </li> </ul>	Favorite	Var Cor	Variable	Block	<input type="checkbox"/>		In1	Scale_Sens[unit]	<input type="checkbox"/>		In1(#1)	Scale_Sens[unit]	<input type="checkbox"/>		InputValues	Scale_Sens[unit]	<input type="checkbox"/>		Out1	Scale_Sens[unit]	<input type="checkbox"/>		Out1(#1)	Scale_Sens[unit]	<input type="checkbox"/>		OutputValues	Scale_Sens[unit]	<input type="checkbox"/>		x	Scale_Sens[unit]
Favorite	Var Cor	Variable	Block																														
<input type="checkbox"/>		In1	Scale_Sens[unit]																														
<input type="checkbox"/>		In1(#1)	Scale_Sens[unit]																														
<input type="checkbox"/>		InputValues	Scale_Sens[unit]																														
<input type="checkbox"/>		Out1	Scale_Sens[unit]																														
<input type="checkbox"/>		Out1(#1)	Scale_Sens[unit]																														
<input type="checkbox"/>		OutputValues	Scale_Sens[unit]																														
<input type="checkbox"/>		x	Scale_Sens[unit]																														

**Note**

- ControlDesk automatically migrates variable connections after you rebuild a simulation application with MATLAB R2015b or later and reload the application's variable description. However, if you then reload the variable description of the simulation application built with a MATLAB Release earlier than R2015b, the migrated variable connections are lost, and you have to update these connections manually.
- For information on limitations in connection with the changed TRC file generation, refer to *Limitations for SDF Files* ([ControlDesk Variable Management](#)).

For more information on TRC file changes, refer to *Basics on the TRC File Changes* on page 37.

# AutomationDesk

---

## Where to go from here

Information in this section

<i>New Features of AutomationDesk 5.3</i>	47
<i>Migrating to AutomationDesk 5.3</i>	50

## New Features of AutomationDesk 5.3

---

### Information in this topic

<i>General enhancements</i> on page 47 <ul style="list-style-type: none"><li><i>Improved user documentation</i> on page 47</li></ul>
<i>Enhancements to the libraries</i> on page 48 <ul style="list-style-type: none"><li><i>Evaluation library</i> on page 48</li><li><i>XIL API Convenience library</i> on page 48</li></ul>
<i>Enhancements to the COM API</i> on page 48
<i>Enhancements to project handling</i> on page 49
<i>Discontinued libraries and blocks</i> on page 49
<i>Discontinued general functionality</i> on page 49 <ul style="list-style-type: none"><li><i>Platform Management library</i> on page 49</li><li><i>Remote Calibration (COM) library</i> on page 50</li><li><i>Third-party Python modules</i> on page 50</li><li><i>XIL API library and XIL API Convenience library</i> on page 50</li></ul>

---

### General enhancements

**Improved user documentation** The *AutomationDesk Tutorial* includes a new lesson on how to work with the Signal-Based Testing library.

## Enhancements to the libraries

The following libraries have been enhanced:

**XIL API Convenience library** The XIL API Convenience library provides the following new automation blocks in the library's Software-Triggered Errors folder:

- **PrepareMultiPinErrorWithCondition**

This block is used to create an error set that affects multiple pins and that is triggered by a specified condition.

- **PrepareMultiPinErrorWithDuration**

This block is used to create an error set that affects multiple pins and that is triggered after a specified duration.

- **PrepareSinglePinErrorWithCondition**

This block is used to create an error set that affects a single pin and that is triggered by a specified condition.

- **PrepareSinglePinErrorWithDuration**

This block is used to create an error set that affects a single pin and that is triggered after a specified duration.

- **WaitForTrigger**

This block is used to wait until the next error set is triggered for the specified EES port.

The automation blocks `PrepareSinglePinError`, `PrepareMultiPinError` and `Trigger`, have been moved to the `Manually Triggered Errors` library folder.

For more information, refer to *XIL API Convenience* ([AutomationDesk Library Reference](#)).

**Evaluation library** The Evaluation library provides a new automation block in the `Converters` library folder:

- **CreateSignalFromMDFFile**

This block is used to get an evaluation signal from a specified channel that is contained in an ASAM MDF 4.1 (MF4) file.

For more information, refer to *Evaluation* ([AutomationDesk Library Reference](#)).

## Enhancements to the COM API

The AutomationDesk COM API provides the following enhancements:

- The `Path` property has been added to `Result1` objects to hold the path of the result's `ReportPool.xml` file.

For more information, refer to [AutomationDesk API Reference](#).



### Enhancements to project handling

Automation Desk's project handling provides the following enhancements:

- In the Found Items Viewer, you can now replace matching expressions that are found in Tuple, List and Dictionary data objects.
- In the dialog of the Find command, the name of the following settings are changed:
  - Found in projects, folders, sequences, and blocks is replaced by Found elements except data objects.
  - Source code of Exec, ExecFile, and Eval blocks is replaced by Python source code.
- The Find Inconsistencies command does not consider objects that are referenced via `self` anymore. Always refer data objects via `_AD_`.
- In the `_INFO_` namespace, the message string of a currently handled exception is available. For more information, refer to *Getting Element Information* ([📖 AutomationDesk Guide](#)).

### Discontinued libraries and blocks

With AutomationDesk 5.3, the following libraries are discontinued:

- Platform Access

When you execute Platform Access library blocks, error messages are displayed because the `rtplib2` Python module is no longer available as of dSPACE Release 2016-B.

- Test Framework

With AutomationDesk 5.3, the following blocks are discontinued:

- All automation blocks for failure simulation in the ControlDesk Access library.

When you execute one of these failure simulation blocks, error messages are displayed because the corresponding functionality in ControlDesk is no longer available as of dSPACE Release 2016-B.

- `InitCaptureResultIDFReader` and `InitCaptureResultIDFWriter` automation blocks in the XIL API library.

For more information on how to migrate, refer to *Migrating to AutomationDesk 5.3* on page 50.

### Discontinued general functionality

**Platform Management library** Platform management automation API version 1.0 is no longer supported. For more information, refer to *GetPlatformManagement* ([📖 AutomationDesk Library Reference](#)).

**Remote Calibration (COM) library** Some MCD 3MC server, such as INCA or CANape, use marshalling for the FetchResults and FetchMeasurement automation blocks to increase the performance of the data transfer when using the internal data storage. Marshalling requires that the client and the server use the same bit architecture. With AutomationDesk 5.3, marshalling is only supported if the marshaller is also executed in a 64-bit process.

To use a 32-bit MCD 3MC server with AutomationDesk, you have to configure `eST_FILE` as storage type, which does not require marshalling.

**XIL API library and XIL API Convenience library** The XIL API library and the XIL API Convenience library no longer support server based on the HIL API 1.0.2 standard.

**Third-party Python modules** 32-bit third-party Python modules are no longer supported by the 64-bit Python interpreter in AutomationDesk 5.3.

## Migrating to AutomationDesk 5.3

---

### General migration aspects

If you open an AutomationDesk project with a later AutomationDesk version, the software automatically detects whether a migration is necessary. Click OK in the message dialog to start the migration. If you also want to continue working with the old project, you should not overwrite it with the migrated project, because the versions are not downward compatible. Save the migrated project to another path or name.

**Note**

Before you open an older project with the new AutomationDesk version, make sure that the following preconditions are fulfilled:

- You must create backups of the project and of the linked custom libraries.
- AutomationDesk must be running properly. There must not be any error messages displayed in the Log Viewer.
- The built-in libraries, required custom libraries and other packages must be loaded correctly.
- To import an older project to a new AutomationDesk version, the exported project or custom library must be available in the ZIP format. The automatic migration does not support the XML format.

If you are using a version control system, there are some preconditions for successful migration, refer to *How to Migrate Projects Under Version Control* ([📖 AutomationDesk Guide](#)).

For more information, refer to *Migrating AutomationDesk* ([📖 AutomationDesk Guide](#)).

**Libraries**

**ControlDesk NG Access library** The ControlDesk NG Access library is renamed **ControlDesk Access**. AutomationDesk projects are automatically adapted to the modified name.

For more information, refer to *ControlDesk Access* ([📖 AutomationDesk Library Reference](#)).

**XIL API Convenience library** The automation blocks for the electrical error simulation have been moved to the **Manually-Triggered Errors** folder. This has no effect on existing projects.

**Remote Diagnostics (COM) library** The use of *DTS V7* as an interface for a diagnostic system is discontinued. If you want to use ControlDesk as the diagnostic system, you can specify **ControlDeskNG.D3System202** as the **Interface** parameter of the **System** data object to realize access to ControlDesk's ECU Diagnostics device.

As an alternative, you can remote-control the ECU Diagnostics device via the ControlDesk Access library in AutomationDesk. For more information, refer to *ControlDesk Access* ([📖 AutomationDesk Library Reference](#)).

For more information on ControlDesk's ECU Diagnostics device, refer to [📖 ControlDesk ECU Diagnostics](#).

If you have problems when migrating to ControlDesk, contact dSPACE Support.

### Discontinued libraries and blocks

If you open a project containing discontinued elements in AutomationDesk 5.3, the discontinued data objects are replaced by Discontinued data object data objects and the discontinued blocks are replaced by Discontinued block blocks during the automatic project update. This lets you load your projects, and search for blocks and data objects to be migrated. If you execute a project containing elements of a discontinued library via AutomationDesk or API script, it will stop with an exception.

The following blocks and data objects are affected.

**Test Framework** The entire library is discontinued. You must migrate your projects to use its successor, the Test Builder library.

For this migration, you can use a tool that is provided by dSPACE Support. For further information, refer to <http://www.dspace.com/go/TestBuilderMigration>.

**Platform Access** The entire library is discontinued. You must migrate your projects to use the XIL API Convenience library.

Data objects of types that are specific for the Platform Access library are automatically converted to data objects that are still continued.

The automation blocks of the Platform Access library cannot be replaced one to one with XIL API Convenience library blocks. For migration, you must consider the task that each block is involved in. For more information, refer to *Migrating Blocks of the Platform Access Library* ([AutomationDesk Guide](#))

**ControlDesk Access** The failure simulation features of the ControlDesk Access library are discontinued. If your project or Custom Library contains elements of the library's Convenience - FailureSimulation and Basic Functions - Application - FailureSimulation folders, you must replace those elements.

With AutomationDesk 5.0, the electrical error simulation port (EESPort) of the ASAM XIL API standard is supported by the XIL API library and the XIL API Convenience library. The EESPort blocks of the XIL API Convenience library provide a future-proof access to failure simulation hardware. For further information on the XIL API electrical error simulation, refer to *Simulating Electrical Errors via the XIL API Convenience Library* ([AutomationDesk Guide](#)).

For help on migration, refer to <http://www.dspace.com/go/pscta>.

**Evaluation library** The converting of results that are captured via blocks of the Platform Access library is discontinued.

The affected block is:

- **GetSignalFromCaptureResult**

The CaptureResult data object is replaced by a Dictionary data object. To get a signal from a migrated capture result, you can use the following script in an Exec block:

```
_AD_.Signal = evaluationlibrary.Signal(
    _AD_.CaptureResult[u'xAxis'],
    _AD_.CaptureResult[_AD_.SignalName])
```

**XIL API library** The blocks that initialize access to IDF files are discontinued.

Affected blocks are:

- **InitCaptureResultIDFReader**
- **InitCaptureResultIDFWriter**

You must migrate them to access MDF files. For more information, see below.

For more information on migration aspects, refer to *Migrating AutomationDesk* ([📖 AutomationDesk Guide](#)).

---

### **Migrating the discontinued XIL API library blocks that initialize access to IDF files**

When working with the XIL API, the preferred format for files to store captured results is the ASAM Measured Data Format (MDF). AutomationDesk uses the 4.1 format that is represented by the file extension MF4. Thus, the blocks for initializing readers and writers to access IDF files have been discontinued.

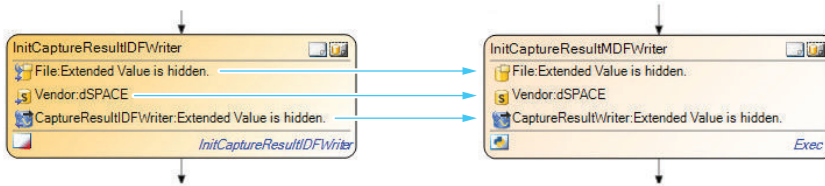
**Migrating InitCaptureResultIDFWriter blocks** Replace each InitCaptureResultIDFWriter block with an Exec block that provides a File, a String and a CaptureResultReader data object as parameters.

Rename the String data object `Vendor`, and set its value to `dSPACE` or reference it to a global String data object with the vendor information.

As the Exec block's Python source, enter:

```
_AD_.CaptureResultWriter = xilapi.InitCaptureResultWriter(
    _AD_.Vendor,                # vendor = "dSPACE"
    "CaptureResultMDFWriter",   # CaptureResultWriterType
    _AD_.File.GetAbsolutePath() ) # the instance-specific
                                # capture result MDF file
                                # with the file extension MF4
```

Parameterize the Exec block in the same way as the InitCaptureResultIDFWriter block:



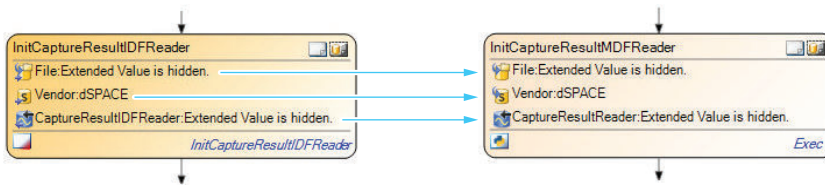
**Migrating InitCaptureResultIDFReader blocks** Replace each InitCaptureResultIDFReader block with an Exec block that provides a File data object, a String data object, and a CaptureResultReader data object as parameters.

Rename the String data object `Vendor`, and set its value to `dSPACE` or reference it to a global String data object with the vendor information.

As the Exec block's Python source, enter:

```
_AD_.CaptureResultReader = xilapi.InitCaptureResultReader(
    _AD_.Vendor,                # vendor = "dSPACE"
    "CaptureResultMDFReader",  # CaptureResultReaderType
    _AD_.File.GetAbsolutePath() # the instance-specific
                                # capture result MDF file
                                # with the file extension MF4
```

Parameterize the Exec block in the same way as the InitCaptureResultIDFReader block:



# Automotive Simulation Models (ASM)

---

## Where to go from here

Information in this section

<i>All ASM Blocksets</i>	57
<i>ASM Base InCylinder Blockset</i>	58
<i>ASM Brake Hydraulics Blockset</i>	60
<i>ASM Diesel Engine Blockset</i>	63
<i>ASM Diesel Exhaust Blockset</i>	69
<i>ASM Diesel InCylinder Blockset</i>	71
<i>ASM Drivetrain Basic Blockset</i>	73
<i>ASM Electric Components Blockset</i>	77
<i>ASM Environment Blockset</i>	80
<i>ASM Gasoline Engine Basic Blockset</i>	82
<i>ASM Gasoline Engine Blockset</i>	85
<i>ASM Gasoline InCylinder Blockset</i>	91
<i>ASM Optimizer</i>	93
<i>ASM Pneumatics Blockset</i>	94
<i>ASM Traffic Blockset</i>	95
<i>ASM Trailer Blockset</i>	97
<i>ASM Truck Blockset</i>	101
<i>ASM Turbocharger Blockset</i>	104
<i>ASM Utils</i>	106
<i>ASM Vehicle Dynamics Blockset</i>	107

Information in other sections

*Migrating ASM Models* ( *ASM User Guide*)

Provides general information on the migration of ASM models.



# All ASM Blocksets

## Migrating all ASM Blocksets

---

**Look-up table migration**

The discontinued Simulink blocks `Lookup` and `Lookup2D` within the ASM library blocks have been updated to the new standard Simulink `Lookup Table (n-D)` block.

For a list of the updated blocks of each ASM blockset, refer to the migration information of the respective ASM blockset.

The migration of the look-up table settings is performed along with the recommended migration steps in MATLAB. However, as described in the release notes of MATLAB R2011a, special handling in case of repeated breakpoints is required. Repeated breakpoints were not prohibited in the former look-up table blocks. The ASM migration only handles issues with repeated breakpoints for parameters set in the provided ASM demo models. These blocks consequently get a special migration treatment. Custom parameterizations with repeated breakpoints are not treated by this.

To also ensure the expected simulation behavior for these cases, you have to manually migrate them. More information along with a Python script, which helps to re-establish the ControlDesk connections, can be found here: <http://www.dspace.com/go/ASM-LUT-CD-migration-RLS2016-B>

---

**Discontinuation of  
ASM\_UTILS license**

The `ASM_UTILS` license is discontinued. The license check now works with any other ASM license. For a list of the updated blocks of each ASM blockset, refer to the migration information of the respective ASM blockset.

# ASM Base InCylinder Blockset

## Where to go from here

Information in this section

<i>New Features of ASM Base InCylinder Blockset 2.3</i>	58
<i>Migrating to ASM Base InCylinder Blockset 2.3</i>	58

## New Features of ASM Base InCylinder Blockset 2.3

### ENGINE\_TORQUE\_SET\_INTERVENTION

This new block facilitates the engine torque intervention during the gearshift process.

## Migrating to ASM Base InCylinder Blockset 2.3

### Look-up table migration

The discontinued Simulink blocks `Lookup` and `Lookup2D` within the ASM library blocks have been updated to the new standard Simulink `Lookup Table (n-D)` block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- DIRECTINJECTOR\_TIMING
- DIRECTINJECTOR
- EGR\_COOLER, EGR\_COOLER\_1\_0
- EGR\_RATE\_CONTROL
- EGR\_VALVE
- ENGINE\_TORQUE\_SET
- EXHAUST\_MANIFOLD
- EXHAUST\_VALVE
- FRICTION\_TORQUE
- HIGH\_PRESSURE\_PUMP
- INJECTION\_QUANTITY

- INTAKE\_MANIFOLD
- INTAKE\_MANIFOLD\_BOOST
- INTAKE\_VALVE
- INTERCOOLER, INTERCOOLER\_1\_0
- PRESSURE\_CONTROL\_VALVE
- RAIL
- RAIL\_CONTROL
- THROTTLE\_CONTROL
- THROTTLE\_MECHANICAL
- THROTTLE\_VALVE, THROTTLE\_VALVE\_2\_0
- VVT\_SETPOINT
- WALL\_HEAT

---

**License check of ASM Utils blocks**

The ASM\_UTILS license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- ANALYTIC\_THERMODYNAMIC\_ANALYSIS
- APU\_EVENT
- EGR\_COOLER, EGR\_COOLER\_1\_0
- ENGINE\_SETUP
- EXHAUST\_MANIFOLD
- INTAKE\_MANIFOLD
- INTERCOOLER, INTERCOOLER\_1\_0
- MASS\_FRACTION\_BURNED
- RAIL
- SINGLEZONE\_CYLINDER, SINGLEZONE\_CYLINDER\_3\_0

---

**SOFTECU\_SETUP**

The unit of the Const\_CylinderOffset inport has been corrected from [] to [deg].

The underscores in the names of other inports are removed.

---

**IDLE\_SPEED\_CONTROL**

During migration, the Trq\_Engine\_Ind\_Max[Nm] output is terminated.

# ASM Brake Hydraulics Blockset

## Migrating to ASM Brake Hydraulics Blockset 1.6.1

### Look-up table migration

The discontinued Simulink blocks `Lookup` and `Lookup2D` within the ASM library blocks have been updated to the new standard Simulink `Lookup Table (n-D)` block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- BRAKE\_CYLINDER\_FL
- BRAKE\_CYLINDER\_FR
- BRAKE\_CYLINDER\_RL
- BRAKE\_CYLINDER\_RR
- CONTINUOUS\_CTRL\_CHANGE\_OVER\_VALVE\_1
- CONTINUOUS\_CTRL\_CHANGE\_OVER\_VALVE\_2
- CONTINUOUS\_CTRL\_INLET\_VALVE\_FL
- CONTINUOUS\_CTRL\_INLET\_VALVE\_FR
- CONTINUOUS\_CTRL\_INLET\_VALVE\_RL
- CONTINUOUS\_CTRL\_INLET\_VALVE\_RR
- CONTINUOUS\_CTRL\_OUTLET\_VALVE\_FL
- CONTINUOUS\_CTRL\_OUTLET\_VALVE\_FR
- CONTINUOUS\_CTRL\_OUTLET\_VALVE\_RL
- CONTINUOUS\_CTRL\_OUTLET\_VALVE\_RR
- CONTINUOUS\_CTRL\_PRE\_CHARGE\_VALVE\_1
- CONTINUOUS\_CTRL\_PRE\_CHARGE\_VALVE\_2
- CONTINUOUS\_VALVE\_CONTROL
- CHANGE\_OVER\_VALVE\_1\_TABLE
- CHANGE\_OVER\_VALVE\_2\_TABLE
- INLET\_VALVE\_FL\_TABLE
- INLET\_VALVE\_FR\_TABLE

- INLET\_VALVE\_RL\_TABLE
- INLET\_VALVE\_RR\_TABLE
- MASTER\_BRAKE\_CYLINDER
- OUTLET\_VALVE\_FL\_TABLE
- OUTLET\_VALVE\_FR\_TABLE
- OUTLET\_VALVE\_RL\_TABLE
- OUTLET\_VALVE\_RR\_TABLE
- PRE\_CHARGE\_VALVE\_1\_TABLE
- PRE\_CHARGE\_VALVE\_2\_TABLE
- PUMP\_1
- PUMP\_2

**License check of ASM Utils blocks**

The ASM\_UTILS license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- BRAKE\_CYLINDER\_FL
- BRAKE\_CYLINDER\_FR
- BRAKE\_CYLINDER\_RL
- BRAKE\_CYLINDER\_RR
- CHANGE\_OVER\_VALVE\_1
- CHANGE\_OVER\_VALVE\_2
- CONNECTION\_CHAMBER\_1
- CONNECTION\_CHAMBER\_2
- DAMPER\_CHAMBER\_1
- DAMPER\_CHAMBER\_2
- INLET\_VALVE\_FL
- INLET\_VALVE\_FR
- INLET\_VALVE\_RL
- INLET\_VALVE\_RR
- NON\_RETURN\_VALVE\_FL
- NON\_RETURN\_VALVE\_FR
- NON\_RETURN\_VALVE\_RES\_1
- NON\_RETURN\_VALVE\_RES\_2

- NON\_RETURN\_VALVE\_RL
- NON\_RETURN\_VALVE\_RR
- OUTLET\_VALVE\_FL
- OUTLET\_VALVE\_FR
- OUTLET\_VALVE\_RL
- OUTLET\_VALVE\_RR
- PRE\_CHARGE\_VALVE\_1
- PRE\_CHARGE\_VALVE\_2
- RESERVOIR\_1
- RESERVOIR\_2

---

**BRAKE\_HYDRAULICS\_  
SWITCHES\_VALVE\_  
CONTROL\_1 and \_2**

The valve control mode is renamed from digital to linear and from continuous to nonlinear. The functionality of the control mode is not changed.

# ASM Diesel Engine Blockset

## Where to go from here

Information in this section

<i>New Features of ASM Diesel Engine Blockset 2.4</i>	63
<i>Changes in the ASM Diesel Engine Demo Model</i>	64
<i>Migrating to ASM Diesel Engine Blockset 2.4</i>	65

## New Features of ASM Diesel Engine Blockset 2.4

### ENGINE\_SETUP

To improve the turnaround time of the simulation model, the `Const_Inj_Matrix` parameter has been deleted. In addition, the corresponding output of the parameter is used for the new signal `active_Inj_Direct[]`.

The signal `Sw_Turbo_Stage[:1SingleStage|2TwoStage]` was renamed to `Sw_Turbo_Stage[1SingleStage|2TwoStage]`. Several outputs were also renamed.

### DIRECTINJECTOR

The block is the advancement of the INJECTOR block.

To improve the turnaround time of the simulation model, the `Const_Inj_Matrix` parameter (routed by the GoToFrom connection of the ENGINE\_SETUP block) has been deleted. The block requires the ENGINE\_SETUP block of the current release (Version 5.0).

The DIRECTINJECTOR block includes the `Sw_InjMode` parameter to switch the injection mode from mean to pulse. Until now, this was located in the SWITCHES\_INJ\_MODE block.

### UNIT\_INJECTOR

To improve the turnaround time of the simulation model, the `Const_Inj_Matrix` parameter (routed by the GoToFrom connection from the ENGINE\_SETUP block) has been deleted. The block requires the ENGINE\_SETUP of the current release (Version 5.0).

### SOFTECU\_SETUP

The unit of the `Const_CylinderOffset` signal has been corrected from [] to [deg]. The underscores in the names of other inports have been removed.

---

<b>RAIL_CONTROL_CRANKBASED</b>	Bubble sort has been removed.
<b>HPP_CRANKBASED</b>	<p>StepSize_Correction is introduced to stabilize the rail pressure. The stabilization is required while using the pulsed high-pressure pump at high engine speeds. The DeliveryLength_Correction is extended by the UpdateState_Correction, which is necessary for the mean-value approach.</p> <p>The Sw_FMU_Energized[0Off1On] signal is delayed by an iteration step during the calculation of the volume flow in the pulsed mode.</p>
<b>ENGINE_TORQUE_SET_INTERVENTION</b>	The ENGINE_TORQUE_SET_INTERVENTION block has been introduced to facilitate the engine torque intervention during the gearshift process.

---

## Changes in the ASM Diesel Engine Demo Model

---

<b>Rail control</b>	Depending on the high-pressure pump used in the model, there is a current- or a crank-angle-based rail pressure controller.
<b>MDL_DISP of engine</b>	In the demo model, just a few basic signals of the exhaust system are prepared for the visualization on the scope. Further signals can be found in the ASM Diesel Exhaust Library. The MDL_DISP of the Engine model contains a link to the location of the signals in the ASM Diesel Exhaust library.
<b>Injectors</b>	The injector (Unit or Direct) that is not used is disabled. This modification improves the turn-around time of the model.
<b>Switches of the turbo</b>	Instead of routing the entire signal bus of the ENGINE_SETUP block to the AirPath model via the GoToFrom connection, only the switches of the turbocharger model are routed. This modification improves the turn-around time of the model significantly.
<b>Torque intervention during gearshift</b>	The engine and transmission soft ECUs are interconnected, so that an engine torque intervention can be realized during the gearshift process. The amount of the intervened torque can be parameterized in the TORQUE_INTERVENTION_CONTROL block from the SoftECU_Transmission. Refer to <i>New Features of ASM Drivetrain Basic Blockset 4.3</i> on page 73.

---



---

<b>Environment signal routing</b>	The routing of the environment signals has been revised and restructured. The signal routing from the MDLUserInterface to the other demo parts is now clearer and easier to track. Moreover, the signals provided to the soft ECU models have been revised and some of them have been replaced with sensor signals from ASMSignalBus.
<b>Maneuver control</b>	The new maneuver control has been integrated in the new demo. With the new implementation, it is now possible to start, stop and reset the maneuver during the simulation on dSPACE platforms using only ModelDesk. Refer to <i>New Features of ASM Drivetrain Basic Blockset 4.3</i> on page 73.

---

## Migrating to ASM Diesel Engine Blockset 2.4

---

<b>Look-up table migration</b>	<p>The discontinued Simulink blocks Lookup and Lookup2D within the ASM library blocks have been updated to the new standard Simulink Lookup Table (n-D) block. Refer to <i>Migrating all ASM Blocksets</i> on page 57.</p> <p>The look-up tables have been updated in the following blocks within this library:</p> <ul style="list-style-type: none"><li>■ AIRFILTER</li><li>■ CATALYST</li><li>■ COMBUSTION_TORQUE_CI</li><li>■ COOLER</li><li>■ CRANKCASE</li><li>■ CYLINDER_INLET</li><li>■ DIESEL_OXIDATION_CATALYST_1_0, DIESEL_OXIDATION_CATALYST</li><li>■ DIESEL_PARTICULATE_FILTER, DIESEL_PARTICULATE_FILTER_1_0, DIESEL_PARTICULATE_FILTER</li><li>■ DIRECTINJECTOR_TIMING</li><li>■ DPF_REGENERATION</li><li>■ EGR_RATE_CONTROL</li><li>■ EGR_VALVE, EGR_VALVE_1_0</li><li>■ EGRCOOLER</li><li>■ ENGINE_SETUP</li></ul>
--------------------------------	--

- ENGINE\_TORQUE\_SET
- EXHAUST\_MANIFOLD
- EXHAUSTTHROTTLE
- FRICTION\_TORQUE
- HIGH\_PRESSURE\_PUMP
- HPP\_CRANKBASED
- HPP\_CRANKBASED\_1\_0
- INJECTION\_QUANTITY
- INJECTOR, INJECTOR\_1\_0, INJECTOR\_2\_0
- INTAKE\_MANIFOLD
- INTERCOOLER
- LP\_EGR\_VALVE
- LP\_EGRCOOLER
- LP\_INTAKE\_MANIFOLD
- LP\_INTAKE\_MANIFOLD\_4\_0
- PRESSURE\_CONTROL\_VALVE
- PUMP\_TORQUE
- RAIL
- RAIL\_CONTROL
- RAIL\_CONTROL\_CRANKBASED
- SMOKE\_LIMITATION
- SOFT\_ECU\_DIESEL\_1\_0, SOFT\_ECU\_DIESEL\_11\_0,  
SOFT\_ECU\_DIESEL\_3\_0
- THROTTLE\_MECHANICAL
- THROTTLE\_VALVE
- UNIT\_INJECTOR, UNIT\_INJECTOR\_1\_0, UNIT\_INJECTOR\_7\_0

### License check of ASM Utils blocks

The ASM\_UTILS license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- EGR\_VALVE, EGR\_VALVE\_1\_0
- EXHAUST\_MANIFOLD
- EXHAUSTTHROTTLE
- LP\_EGR\_VALVE

- LP\_INTAKE\_MANIFOLD, LP\_INTAKE\_MANIFOLD\_4\_0
- THROTTLE\_MECHANICAL

<b>INJECTOR_9_0</b>	<p>The block is the former version of the INJECTOR block, which is discontinued. During migration, the library link of the INJECTOR block is changed to the former version.</p> <p>For the new features of the block, use the DIRECT_INJECTOR block together with the ENGINE_SETUP block.</p>
<b>IDLE_SPEED_CONTROL</b>	<p>During migration, the Trq_Engine_Ind_Max[Nm] output is terminated.</p>
<b>EGR_VALVE</b>	<p>The Ctrl2Pos_EGR parameter has been renamed to Map_Ctrl2Pos.</p> <p>The Map_A_red parameter has been renamed to Map_A_Red.</p>
<b>LP_EGR_VALVE</b>	<p>The Ctrl2Pos_EGR parameter has been renamed to Map_Ctrl2Pos.</p> <p>The Map_A_red parameter has been renamed to Map_A_Red.</p>
<b>SWITCHES_INJ_MODE_1_0</b>	<p>The block is the former version of the SWITCHES_INJ_MODE block, which is discontinued. During migration, the library link of the SWITCHES_INJ_MODE block is changed to SWITCHES_INJ_MODE_1_0. The switch has been moved to the DIRECTINJECTOR block.</p>
<b>ENGINE_SETUP_4_0</b>	<p>The block is the former version of the ENGINE_SETUP block. During migration, the library link of the ENGINE_SETUP block is changed to ENGINE_SETUP_4_0. For the new features of the block, use the ENGINE_SETUP block together with a current version of the injector blocks (DIRECTINJECTOR, UNIT_INJECTOR) as these work as a pair with ENGINE_SETUP.</p>
<b>UNIT_INJECTOR_7_0</b>	<p>The block is the former version of the UNIT_INJECTOR block. During migration, the library link of the UNIT_INJECTOR block is changed to UNIT_INJECTOR_7_0. For the new features of the block, use the UNIT_INJECTOR block together with the ENGINE_SETUP block.</p>
<b>COMMON_ENGINE_PARAMETERS</b>	<p>In dSPACE Release 2015-B and Release 2016-A, the internal calculations of cpFuel and cvFuel in the COMMON_ENGINE_PARAMETERS block were connected to the wrong outputs. This is now corrected under the mask of the block.</p> <p>However, to keep simulation results reproducible, the connections to the outputs cp_Fuel[J][kgK] and cv_Fuel[J][kgK] are interchanged during migration so that cp_Fuel[J][kgK] is connected to the cv_Fuel</p>

Goto tag, for example. You can manually undo this so that cp\_Fuel[J][kgK]] is connected to cp\_Fuel.

**Note**

This is necessary only if the CNG system of the ASM Gasoline Engine Blockset is used.

# ASM Diesel Exhaust Blockset

## Migrating to ASM Diesel Exhaust Blockset 2.1.3

---

### Look-up table migration

The discontinued Simulink blocks Lookup and Lookup2D within the ASM library blocks have been updated to the new standard Simulink Lookup Table (n-D) block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- ADBLUE\_PRESSURE\_REGULATION\_VALVE
- ADBLUE\_PUMP
- AIR\_NON\_RETURN\_VALVE
- AIR\_REGULATION\_VALVE
- ATOMIZER
- DIESEL\_OXIDATION\_CATALYST,  
DIESEL\_OXIDATION\_CATALYST\_10\_0,  
DIESEL\_OXIDATION\_CATALYST\_4\_0
- DIESEL\_PARTICULATE\_FILTER, DIESEL\_PARTICULATE\_FILTER\_3\_0
- INJECTION\_VALVE
- MIXING\_CHAMBER
- MUFFLER, MUFFLER\_2\_0
- PUMP\_HOSE
- RAW\_EXHAUST\_COMPOSITION
- SCR\_CATALYST, SCR\_CATALYST\_5\_0
- THROTTLE
- UREA\_DECOMPOSITION
- VENT\_VALVE

---

### License check of ASM Utils blocks

The ASM\_UTILS license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- SCR\_CATALYST\_5\_0

# ASM Diesel InCylinder Blockset

## Where to go from here

Information in this section

<i>Changes in the ASM Diesel InCylinder Demo Model</i>	71
<i>Migrating to ASM Diesel InCylinder Blockset 2.3</i>	71

## Changes in the ASM Diesel InCylinder Demo Model

### Torque intervention during gearshift

The engine and transmission soft ECUs are interconnected, so that an engine torque intervention can be realized during the gearshift process. The amount of the intervened torque can be parameterized in the `TORQUE_INTERVENTION_CONTROL` block from the `SoftECU_Transmission`. Refer to *New Features of ASM Drivetrain Basic Blockset 4.3* on page 73.

### Environment signal routing

The routing of the environment signals has been revised and restructured. The signal routing from the `MDLUserInterface` to the other demo parts is now clearer and easier to track. Moreover, the signals provided to the soft ECU models have been revised and some of them have been replaced with sensor signals from `ASMSignalBus`.

### Maneuver control

The new maneuver control has been integrated in the new demo. With the new implementation, it is now possible to start, stop and reset the maneuver during the simulation on dSPACE platforms using only ModelDesk. Refer to *New Features of ASM Drivetrain Basic Blockset 4.3* on page 73.

## Migrating to ASM Diesel InCylinder Blockset 2.3

### Look-up table migration

The discontinued Simulink blocks `Lookup` and `Lookup2D` within the ASM library blocks have been updated to the new standard Simulink `Lookup Table (n-D)` block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- DPF\_REGENERATION
- HEAT\_RELEASE\_ARRHENIUS, HEAT\_RELEASE\_ARRHENIUS\_4\_0
- HEAT\_RELEASE\_CHMELA, HEAT\_RELEASE\_CHMELA\_4\_0
- IGNITION\_DELAY
- SMOKE\_LIMITATION
- SOFT\_ECU\_INCYLINDER\_DIESEL\_4\_0

---

### License check of ASM Utils blocks

The ASM\_UTILS license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- IGNITION\_DELAY



# ASM Drivetrain Basic Blockset

## Where to go from here

Information in this section

<i>New Features of ASM Drivetrain Basic Blockset 4.3</i>	73
<i>Migrating to ASM Drivetrain Basic Blockset 4.3</i>	74

## New Features of ASM Drivetrain Basic Blockset 4.3

### **SOFT\_ECU\_TRANSMISSION\_BASIC**

The `SOFT_ECU_TRANSMISSION_BASIC` block has been modularized. The old functionalities and controller strategies have been subdivided into several modular blocks.

You can now modify and extend the controller structure according to your needs. The new implementation offers also the possibility to integrate a third-party controller strategy.

The following blocks have been added to the library:

- `CLUTCH_ENGAGEMENT_CONTROL`
- `LOCKUP_CLUTCH_CONTROL`
- `SHIFT_LOCK_CONTROL`
- `SHIFT_STRATEGY`
- `SOFTECU_TRANSMISSION_SETUP`
- `TORQUE_INTERVENTION_CONTROL`
- `TIP_SHIFT_CONTROL`

Because of its completely new structure, the previous `SOFT_ECU_TRANSMISSION_BASIC` block is not migrated to the new implementation. Instead, the previous block is saved in `FormerVersions`, and the link is changed to this block during migration.

### **MANEUVER\_CONTROL**

The new block `MANEUVER_CONTROL` is introduced which describes a central maneuver control for the engine demos. This block controls the start, pause and stop of maneuver time, as well as the maneuver status.

Until now, the maneuver control functionality was divided between the CYCLES and other non-library blocks. Using linked library blocks, offers a unified implementation for all engine demos. This implementation can be extended and benefit from new features in the future.

Moreover, with the new block, it is now possible to start, stop and reset the maneuver during the simulation on dSPACE platforms using only ModelDesk.

---

### LONGITUDINAL\_ CONTROLLER\_HYBRID

The block has several new features:

- It is now possible to limit the actuation rate of both pedals.
- The minimum time needed to switch between different pedals can be set to a nonzero value.
- The driver can tolerate certain speed differences.
- A new controller parameter for the preview reference speed has been added as a preparation for improved controller strategy.

## Migrating to ASM Drivetrain Basic Blockset 4.3

---

### Look-up table migration

The discontinued Simulink blocks Lookup and Lookup2D within the ASM library blocks have been updated to the new standard Simulink Lookup Table (n-D) block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- CRANKSHAFT
- CYCLES\_7\_0
- GEARBOX\_AT, GEARBOX\_AT\_1\_0, GEARBOX\_AT\_8\_0
- GEARBOX\_MT, GEARBOX\_MT\_1\_0, GEARBOX\_MT\_8\_0
- GEAR\_SHIFTER\_2\_0, GEAR\_SHIFTER\_6\_0
- LONGITUDINAL\_CONTROL
- SOFT\_ECU\_TRANSMISSION\_1\_0
- SOFT\_ECU\_TRANSMISSION\_BASIC\_7\_0
- TORQUE\_CONTROLLER\_3\_0
- TORQUE\_CONVERTER, TORQUE\_CONVERTER\_4\_0

<b>License check of ASM Utils blocks</b>	<p>The ASM_UTILS license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.</p> <p>The Utils blocks in the following blocks within this library have been updated:</p> <ul style="list-style-type: none"> <li>■ GEAR_SHIFTER_6_0</li> <li>■ SOFT_ECU_TRANSMISSION_BASIC_7_0</li> </ul>
<b>DRIVING_RESISTANCES</b>	<p>The friction between the tire and road can be now described with the <code>Const_Fric_Coeff</code> parameter. You can use the parameter to limit the driving and braking forces.</p>
<b>ENGINE</b>	<p>The size of the engine torque parameter has been expanded from [30x30] to [31x30]. The old parameter is extrapolated and initialized in <code>asmmigratepost</code>.</p>
<b>TORQUE_CONTROLLER</b>	<p>The size of the inverse engine torque parameter has been expanded from [40x40] to [41x40]. The old parameter is extrapolated and initialized in <code>asmmigratepost</code>.</p>
<b>LONGITUDINAL_CONTROLLER</b>	<p>The size of the inverse engine torque parameter has been expanded from [40x40] to [41x40]. The old parameter is extrapolated and initialized in <code>asmmigratepost</code>.</p> <p>The continuous integrators inside the block have been replaced with discrete ones.</p>
<b>CRANK_SHAFT</b>	<p>The continuous integrator inside the block has been replaced with a discrete one.</p>
<b>SHAFT_RIGID</b>	<p>A new parameter has been added to describe the rotation damping.</p> <p>A new inport has been added for an external initialization of the speed integrator.</p>
<b>GEAR_SHIFTER</b>	<p>The block has a new parameter (<code>Sw_ClutchPedal_EngineOff</code>) to actuate the clutch pedal while the engine is not running.</p> <p>The behavior in hard braking scenarios where the clutch should not be disengaged if the current gear is neutral has been improved.</p>
<b>CYCLES</b>	<p>The block has been moved to the FormerVersions and discontinued. During the migration, the link is changed to <code>FormerVerions/CYCLES_7_0</code>.</p>

---

### SOFT\_ECU\_TRANSMISSION\_9\_0

The block cannot be automatically migrated. Therefore, during the migration the link to the block is changed to former implementation, which is located in `FormerVersions/SOFT_ECU_TRANSMISSION_9_0`.

To benefit from the new implementation, the delivered demos can serve as templates for the use of the new blocks.

---

### LONGITUDINAL\_CONTROLLER\_HYBRID

During migration, new parameters are added:

- `Const_Rate_Max_AccPedal` and `Const_Rate_Max_BrakePedal`: Pedals maximum rate of change
- `Cont_t_min_Pedals`: Minimum time needed to switch between different pedals
- `Map_v_Dead_Driver` and `Sw_v_Dead_Driver`: Tolerated speed difference
- `Const_Preview_vCtrl_Pedals`: Controller parameter for the preview reference speed

These parameters are initialized in `asmigratepost` so that the previous behavior is kept unchanged.

# ASM Electric Components Blockset

## Where to go from here

Information in this section

<i>New Features of ASM Electric Components Blockset 3.3</i>	77
<i>Changes in the ASM Electric Components Demo Model</i>	77
<i>Migrating to ASM Electric Components Blockset 3.3</i>	78

## New Features of ASM Electric Components Blockset 3.3

### BATTERY\_MULTICELL

The thermal battery model is extended to simulate the temperature for each separate cell and the thermal interaction with neighbor cells. Also, a cooling plate can be simulated for the six battery pack surfaces.

The model includes example parameterizations for different battery types and technologies, such as lead-acid, NiMH and lithium-ion (NMC, LFP, LTO) batteries.

### THREE\_PHASE\_RECTIFIER

To convert an AC source to a DC source. It simulates DC current and line currents at conduction and commutation states based on the three phase voltage, the DC voltage and the source parameters.

### BATTERY

The model includes example parameterizations for different battery types and technologies like lead-acid, NiMH and lithium-ion (NMC, LFP, LTO) batteries.

## Changes in the ASM Electric Components Demo Model

### Electric-drive closed-loop squirrel-cage asynchronous machine dq

The power supply now consists of the THREE\_PHASE\_POWERSUPPLY, THREE\_PHASE\_RECTIFIER and DC\_LINK blocks.

## Migrating to ASM Electric Components Blockset 3.3

### Look-up table migration

The discontinued Simulink blocks `Lookup` and `Lookup2D` within the ASM library blocks have been updated to the new standard Simulink `Lookup Table (n-D)` block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- ALTERNATOR, ALTERNATOR\_2\_0
- BATTERY, BATTERY\_4\_0
- BATTERY\_CELL
- BATTERY\_MULTICELL, BATTERY\_MULTICELL\_2\_0
- BATTERY\_THERMAL
- BRUSHLESS\_DC\_MACHINE\_ALPHA\_BETA
- COMPRESSOR
- CURRENT\_SINK
- DC\_MACHINE, DC\_MACHINE\_1\_0
- DRIVE\_MANAGEMENT
- FAN
- PERMANENT\_MAGNET\_SYNCHRONOUS\_MACHINE\_D\_Q
- POTENTIOMETER
- PTC\_GRID\_DEFROSTER
- PTC\_HEATER
- SOFT\_ECU\_HYBRID\_MANAGER\_1\_0
- STARTER, STARTER\_2\_0
- TRQ\_REQUEST\_COORDINATION

### License check of ASM Utils blocks

The `ASM_UTILS` license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- BATTERY\_CELL
- BATTERY\_MULTICELL, BATTERY\_MULTICELL\_2\_0
- PMSM\_D\_Q\_NONLINEAR

---

<b>BATTERY_THERMAL</b>	The block has two new outputs: Capacity_Thermal[J/(kgK)] and P_MainAndLossReaction[W]
------------------------	---

---

<b>THREE_PHASE_POWERSUPPLY</b>	<p>The block has new inports: Init_Angle[Rad] and Reset[0 1]</p> <p>The block has new outputs: Const_R_AC[Ohm] and Const_L_AC[H]</p> <p>The following port names were changed: v_Mains[V] to V_phase_to_phase_rms[V] and v_Mains[a;b;c][V] to v [a;b;c][V].</p>
--------------------------------	---

---

<b>DC_LINK</b>	<p>The block has two new inports: I_DCLink[A] and Sw_Source[1Current 2voltage]</p> <p>The block has a new output: Const_R_DC[Ohm]</p> <p>The output V_Out[V] was renamed to V_DCLink[V].</p>
----------------	--

# ASM Environment Blockset

## Where to go from here

Information in this section

<i>New Features of ASM Environment Blockset 4.5</i>	80
<i>Migrating to ASM Environment Blockset 4.5</i>	80

## New Features of ASM Environment Blockset 4.5

### LONGITUDINAL\_ CONTROLLER\_HYBRID

The block has several new features:

- It is now possible to limit the actuation rate of both pedals.
- The minimum time needed to switch between different pedals can be set to a nonzero value.
- The driver can tolerate certain speed differences.
- A new controller parameter for the preview reference speed has been added as a preparation for improved controller strategy.

## Migrating to ASM Environment Blockset 4.5

### Look-up table migration

The discontinued Simulink blocks `Lookup` and `Lookup2D` within the ASM library blocks have been updated to the new standard Simulink `Lookup Table (n-D)` block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- `BRAKE_HYDRAULICS_VARIANT`
- `BRAKE_PNEUMATICS_VARIANT`
- `CONTROLLER`
- `GEAR_SHIFTER_9_0`

### License check of ASM Utils blocks

The `ASM_UTILS` license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.



The Utils blocks in the following blocks within this library have been updated:

- BASIC\_ROADS
- GEAR\_SHIFTER\_9\_0
- LATERAL\_CONTROL1
- MANEUVER\_SCHEDULER
- ROAD

---

**GEAR\_SHIFTER**

The block has a new parameter (*Sw\_ClutchPedal\_EngineOff*) to actuate the clutch pedal while the engine is not running.

The behavior in hard braking scenarios where the clutch should not be disengaged if the current gear is neutral has been improved.

---

**LONGITUDINAL\_  
CONTROLLER\_HYBRID**

During migration, new parameters are added:

- *Const\_Rate\_Max\_AccPedal* and *Const\_Rate\_Max\_BrakePedal*: Pedals maximum rate of change
- *Cont\_t\_min\_Pedals*: Minimum time needed to switch between different pedals
- *Map\_v\_Dead\_Driver* and *Sw\_v\_Dead\_Driver*: Tolerated speed difference
- *Const\_Preview\_vCtrl\_Pedals*: Controller parameter for the preview reference speed

These parameters are initialized in *asmigratepost* so that the previous behavior is kept unchanged.

# ASM Gasoline Engine Basic Blockset

## Where to go from here

Information in this section

<i>New Features of ASM Gasoline Engine Basic Blockset 2.1</i>	82
<i>Changes in the ASM Engine Gasoline Basic Demo Model</i>	82
<i>Migrating to ASM Gasoline Engine Basic Blockset 2.1</i>	83

## New Features of ASM Gasoline Engine Basic Blockset 2.1

### ENGINE\_SETUP

To improve the turnaround time of the simulation model, the `Const_Inj_Matrix` parameter has been deleted. In addition, the corresponding output of the parameter is used for the new signal `active_Inj_Direct[]`.

The signal `Sw_Turbo_Stage[1SingleStage|2TwoStage]` was renamed to `Sw_Turbo_Stage[1SingleStage|2TwoStage]`. Several outports were also renamed.

### ENGINE\_TORQUE\_SET\_INTERVENTION

The `ENGINE_TORQUE_SET_INTERVENTION` block has been introduced to facilitate the engine torque intervention during the gearshift process.

### SOFTECU\_SETUP

The unit of the `Const_CylinderOffset` signal has been corrected from [] to [deg]. The underscores in the names of other inports have been removed.

### PORTINJECTOR

The block has a new signal: `q_mean_inj[mm3]cyc` in the `ASMSignalBus`.

## Changes in the ASM Engine Gasoline Basic Demo Model

### Torque intervention during gearshift

The engine and transmission soft ECUs are interconnected, so that an engine torque intervention can be realized during the gearshift process. The amount of the intervened torque can be parameterized

in the TORQUE\_INTERVENTION\_CONTROL block from the SoftECU\_Transmission. Refer to *New Features of ASM Drivetrain Basic Blockset 4.3* on page 73.

---

#### Environment signal routing

The routing of the environment signals has been revised and restructured. The signal routing from the MDLUserInterface to the other demo parts is now clearer and easier to track. Moreover, the signals provided to the soft ECU models have been revised and some of them have been replaced with sensor signals from ASMSignalBus.

---

#### Maneuver control

The new maneuver control has been integrated in the new demo. With the new implementation, it is now possible to start, stop and reset the maneuver during the simulation on dSPACE platforms using only ModelDesk. Refer to *New Features of ASM Drivetrain Basic Blockset 4.3* on page 73.

## Migrating to ASM Gasoline Engine Basic Blockset 2.1

---

#### Look-up table migration

The discontinued Simulink blocks Lookup and Lookup2D within the ASM library blocks have been updated to the new standard Simulink Lookup Table (n-D) block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- COMBUSTION\_TORQUE\_SI
- COOLER
- CYLINDER\_INLET
- ENGINE\_TORQUE\_SET
- FRICTION\_TORQUE, FRICTION\_TORQUE\_3\_0
- IGNITION\_SET
- INJECTOR\_SI\_4\_0
- LAMBDA\_CONTROL
- MAPS\_TC
- PORTINJECTOR
- PORTINJECTOR\_TIMING
- REL\_AIRMASS\_MAPBASED
- SOFT\_ECU\_GASOLINE

- SOFT\_ECU\_GASOLINEBASIC\_7\_0
- THROTTLE\_CONTROL
- THROTTLE\_MECHANICAL
- THROTTLE\_VALVE
- TURBO\_CONTROL
- WALL\_FILM

---

### License check of ASM Utils blocks

The ASM\_UTILS license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- ENGINE\_SETUP
- INTAKE\_MANIFOLD
- MAPS\_TC
- THROTTLE\_MECHANICAL

---

### IDLE\_SPEED\_CONTROL

During migration, the Trq\_Engine\_Ind\_Max[Nm] output is terminated.

---

### ENGINE\_SETUP\_4\_0

The block is the former version of the ENGINE\_SETUP block. During migration, the library link of the ENGINE\_SETUP block is changed to ENGINE\_SETUP\_4\_0. For the new features of the block, use the ENGINE\_SETUP block together with a current version of the injector blocks (DIRECTINJECTOR, UNIT\_INJECTOR) as these work as a pair with ENGINE\_SETUP.

---

### COMMON\_ENGINE\_PARAMETERS

In dSPACE Release 2015-B and Release 2016-A, the internal calculations of cpFuel and cvFuel in the COMMON\_ENGINE\_PARAMETERS block were connected to the wrong outputs. This is now corrected under the mask of the block.

However, to keep simulation results reproducible, the connections to the outputs cp\_Fuel[J][kgK] and cv\_Fuel[J][kgK] are interchanged during migration so that cp\_Fuel[J][kgK] is connected to the cv\_Fuel Goto tag, for example. You can manually undo this so that cp\_Fuel[J][kgK] is connected to cp\_Fuel.

#### Note

This is necessary only if the CNG system of the ASM Gasoline Engine Blockset is used.

# ASM Gasoline Engine Blockset

## Where to go from here

Information in this section

<i>New Features of ASM Gasoline Engine Blockset 3.4</i>	85
<i>Changes in the ASM Engine Gasoline Demo Model</i>	86
<i>Migrating to ASM Gasoline Engine Blockset 3.4</i>	87

## New Features of ASM Gasoline Engine Blockset 3.4

### ENGINE\_SETUP

To improve the turnaround time of the simulation model, the `Const_Inj_Matrix` parameter has been deleted. In addition, the corresponding output of the parameter is used for the new signal `active_Inj_Direct[]`.

The signal `Sw_Turbo_Stage[:1SingleStage|2TwoStage]` was renamed to `Sw_Turbo_Stage[1SingleStage|2TwoStage]`. Several outputs were also renamed.

### DIRECTINJECTOR

To improve the turnaround time of the simulation model, the `Const_Inj_Matrix` parameter (routed by the `GoToFrom` connection of the `ENGINE_SETUP` block) has been deleted. The block requires the `ENGINE_SETUP` block of the current release (Version 5.0).

The `DIRECTINJECTOR` block includes the `Sw_InjMode` parameter to switch the injection mode from mean to pulse. Until now, this was located in the `SWITCHES_INJ_MODE` block.

### SOFTECU\_SETUP

The unit of the `Const_CylinderOffset` signal has been corrected from [] to [deg]. The underscores in the names of other inports have been removed.

### RAIL\_CONTROL\_ CRANKBASED

Bubble sort has been removed.

### INJECTOR\_MODE

The `Sw_Mode_Strat` parameter has been added to the block to conveniently deactivate stratified mode.

<b>CATALYST</b>	A new catalyst model has been implemented, which considers oxygen storage and heat balance.
<b>LAMBDA_SENSOR</b>	A new lambda sensor has been developed to consider sensor heating.
<b>HPP_CRANKBASED</b>	<p>StepSize_Correction is introduced to stabilize the rail pressure. The stabilization is required while using the pulsed high-pressure pump at high engine speeds. The DeliveryLength_Correction is extended by the UpdateState_Correction, which is necessary for the mean-value approach.</p> <p>The Sw_FMU_Energized[0Off1On] signal is delayed by an iteration step during the calculation of the volume flow in the pulsed mode.</p>
<b>ENGINE_TORQUE_SET_INTERVENTION</b>	The ENGINE_TORQUE_SET_INTERVENTION block has been introduced to facilitate the engine torque intervention during the gearshift process.

## Changes in the ASM Engine Gasoline Demo Model

<b>Rail control</b>	Depending on the high-pressure pump used in the model, there is a current- or a crank-angle-based rail pressure controller.
<b>Exhaust system</b>	The ExhaustSystem model has been restructured because of the new CATALYST and LAMBDA_SENSOR blocks.
<b>Fuel system</b>	Now, the port injector and the direct injector can both be active at the same time. The amount of fuel injected into the cylinders is the sum of both injectors.
<b>Switches of the turbo</b>	Instead of routing the entire signal bus of the ENGINE_SETUP block to the AirPath model via the GoToFrom connection, only the switches of the turbocharger model are routed. This modification improves the turn-around time of the model significantly.
<b>Torque intervention during gearshift</b>	The engine and transmission soft ECUs are interconnected, so that an engine torque intervention can be realized during the gearshift process. The amount of the intervened torque can be parameterized in the TORQUE_INTERVENTION_CONTROL block from the SoftECU_Transmission. Refer to <i>New Features of ASM Drivetrain Basic Blockset 4.3</i> on page 73.

---

<b>Environment signal routing</b>	The routing of the environment signals has been revised and restructured. The signal routing from the MDLUserInterface to the other demo parts is now clearer and easier to track. Moreover, the signals provided to the soft ECU models have been revised and some of them have been replaced with sensor signals from ASMSignalBus.
<b>Maneuver control</b>	The new maneuver control has been integrated in the new demo. With the new implementation, it is now possible to start, stop and reset the maneuver during the simulation on dSPACE platforms using only ModelDesk. Refer to <i>New Features of ASM Drivetrain Basic Blockset 4.3</i> on page 73.

---

## Migrating to ASM Gasoline Engine Blockset 3.4

---

<b>Look-up table migration</b>	<p>The discontinued Simulink blocks Lookup and Lookup2D within the ASM library blocks have been updated to the new standard Simulink Lookup Table (n-D) block. Refer to <i>Migrating all ASM Blocksets</i> on page 57.</p> <p>The look-up tables have been updated in the following blocks within this library:</p> <ul style="list-style-type: none"><li>■ AIRFILTER</li><li>■ CNG_SHUTOFF_VALVE</li><li>■ COOLER</li><li>■ CNG_PRESSURE_REGULATOR</li><li>■ COMBUSTION_TORQUE_CI</li><li>■ COMBUSTION_TORQUE_SI</li><li>■ CRANKCASE</li><li>■ CYLINDER_INLET</li><li>■ DIRECTINJECTOR</li><li>■ DIRECTINJECTOR_TIMING</li><li>■ EGRCOOLER</li><li>■ EGR_RATE_CONTROL</li><li>■ EGR_VALVE</li><li>■ ENGINE_TORQUE_SET</li><li>■ EXHAUST_MANIFOLD</li></ul>
--------------------------------	---

- FRICTION\_TORQUE, FRICTION\_TORQUE\_3\_0
- HIGH\_PRESSURE\_PUMP
- HPP\_CRANKBASED, HPP\_CRANKBASED\_1\_0
- IGNITION\_SET
- INJECTION\_QUANTITY
- INJECTOR\_4\_0
- INJECTOR\_MODE
- INTERCOOLER
- LAMBDA\_CONTROL
- PORTINJECTOR
- PORTINJECTOR\_TIMING
- PRESSURE\_CONTROL\_VALVE
- PUMP\_TORQUE
- RAIL
- RAIL\_CONTROL
- RAIL\_CONTROL\_CRANKBASED
- REL\_AIRMASS\_MAPBASED
- SOFT\_ECU\_GASOLINE\_12\_0
- THROTTLE\_MECHANICAL
- THROTTLE\_VALVE
- WALL\_FILM

---

### License check of ASM Utils blocks

The ASM\_UTILS license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- EGR\_VALVE
- ENGINE\_SETUP
- EXHAUST\_MANIFOLD
- HPP\_CRANKBASED
- INTAKE\_MANIFOLD
- RAIL\_CONTROL\_CRANKBASED
- THROTTLE\_MECHANICAL



<b>EGR_VALVE</b>	The Ctrl2Pos_EGR parameter has been renamed to Map_Ctrl2Pos. The Map_A_red parameter has been renamed to Map_A_Red.
<b>SWITCHES_INJ_MODE_1_0</b>	The block is the former version of the SWITCHES_INJ_MODE block, which is discontinued. During migration, the library link of the SWITCHES_INJ_MODE block is changed to SWITCHES_INJ_MODE_1_0. The switch has been moved to the DIRECTINJECTOR block.
<b>CATALYST_4_0</b>	The block is the former version of the CATALYST block. During migration, the library link of the block is changed to CATALYST_4_0. Use the CATALYST block for the new features.
<b>SELECTFUELINJECTION_1_0</b>	The block is the former version of the SELECTFUELINJECTION block, which is discontinued. During migration, the library link of the block is changed to SELECTFUELINJECTION_1_0. In the current release, the port injector and direct injector can both be active at the same time. The amount of fuel injected into the cylinders is the sum of both injectors.
<b>IDLE_SPEED_CONTROL</b>	During migration, the Trq_Engine_Ind_Max[Nm] output is terminated.
<b>ENGINE_SETUP_4_0</b>	The block is the former version of the ENGINE_SETUP block. During migration, the library link of the ENGINE_SETUP block is changed to ENGINE_SETUP_4_0. For the new features of the block, use the ENGINE_SETUP block together with a current version of the injector blocks (DIRECTINJECTOR, UNIT_INJECTOR) as these work as a pair with ENGINE_SETUP.
<b>DIRECT_INJECTOR_8_0</b>	The block is the former version of the DIRECT_INJECTOR block. During migration, the library link of the block is changed to DIRECT_INJECTOR_8_0. Use DIRECT_INJECTOR for the newer features the block. The block works as a pair with the ENGINE_SETUP block.
<b>COMMON_ENGINE_PARAMETERS</b>	In dSPACE Release 2015-B and Release 2016-A, the internal calculations of cpFuel and cvFuel in the COMMON_ENGINE_PARAMETERS block were connected to the wrong outputs. This is now corrected under the mask of the block. However, to keep simulation results reproducible, the connections to the outputs cp_Fuel[J][kgK] and cv_Fuel[J][kgK] are interchanged during migration so that cp_Fuel[J][kgK] is connected to the cv_Fuel Goto tag, for example. You can manually undo this so that cp_Fuel[J][kgK] is connected to cp_Fuel.

### Note

This is necessary only if the CNG system of the ASM Gasoline Engine Blockset is used.

---

### Related topics

Basics

- *Migrating ASM Models* ( *ASM User Guide*)

# ASM Gasoline InCylinder Blockset

## Where to go from here

Information in this section

<i>Changes in the ASM Gasoline InCylinder Demo Model</i>	91
<i>Migrating to ASM Gasoline InCylinder Blockset 2.3</i>	91

## Changes in the ASM Gasoline InCylinder Demo Model

### Torque intervention during gearshift

The engine and transmission soft ECUs are interconnected, so that an engine torque intervention can be realized during the gearshift process. The amount of the intervened torque can be parameterized in the `TORQUE_INTERVENTION_CONTROL` block from the `SoftECU_Transmission`. Refer to *New Features of ASM Drivetrain Basic Blockset 4.3* on page 73.

### Environment signal routing

The routing of the environment signals has been revised and restructured. The signal routing from the `MDLUserInterface` to the other demo parts is now clearer and easier to track. Moreover, the signals provided to the soft ECU models have been revised and some of them have been replaced with sensor signals from `ASMSignalBus`.

### Maneuver control

The new maneuver control has been integrated in the new demo. With the new implementation, it is now possible to start, stop and reset the maneuver during the simulation on dSPACE platforms using only ModelDesk. Refer to *New Features of ASM Drivetrain Basic Blockset 4.3* on page 73.

## Migrating to ASM Gasoline InCylinder Blockset 2.3

### Look-up table migration

The discontinued Simulink blocks `Lookup` and `Lookup2D` within the ASM library blocks have been updated to the new standard Simulink `Lookup Table (n-D)` block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- CATALYST
- HEAT\_RELEASE\_VIBE
- IGNITION\_SET
- INJECTOR\_MODE
- LAMBDA\_CONTROL
- PORTINJECTOR
- SOFT\_ECU\_INCYLINDER\_GASOLINE\_6\_0

---

### **INJECTOR\_MODE**

The block has a new Add Parameter Sw\_Mode\_Strat parameter to deactivate the stratified mode conveniently.

# ASM Optimizer

## New Features of ASM Optimizer 1.8

---

### **ModelDesk measurement data files**

The ModelDesk measurement data files (\*.md) can now be used as source for the InCylinder measurements.

This is also included in the demo projects of the ASM Diesel InCylinder and ASM Gasoline InCylinder blocksets.

# ASM Pneumatics Blockset

## Migrating to ASM Pneumatics Blockset 2.0.4

---

### Look-up table migration

The discontinued Simulink blocks `Lookup` and `Lookup2D` within the ASM library blocks have been updated to the new standard Simulink `Lookup Table (n-D)` block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- `FOOT_BRAKE_MODULE`

---

### License check of ASM Utils blocks

The `ASM_UTILS` license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- `AIR_SUSPENSION_FORCES_FRONT`
- `AIR_SUSPENSION_FORCES_REAR`
- `AIR_SUSPENSION_FORCES_REAR_2ND`
- `AIR_SUSPENSION_FORCES_REAR_3RD`
- `AIR_SUSPENSION_FORCES_REAR_4TH`
- `TRL_AIR_SUSPENSION_FORCES_FRONT`
- `TRL_AIR_SUSPENSION_FORCES_REAR`
- `TRL_AIR_SUSPENSION_FORCES_REAR_2ND`

# ASM Traffic Blockset

## Where to go from here

Information in this section

<i>New Features of ASM Traffic Blockset 3.5</i>	95
<i>Changes in the ASM Traffic Demo Model</i>	95
<i>Migrating to ASM Traffic Blockset 3.5</i>	95

## New Features of ASM Traffic Blockset 3.5

### LINE\_SENSOR

The LINE\_SENSOR block now also detects line shapes and continuous shapes on junctions.

## Changes in the ASM Traffic Demo Model

### Torque intervention during gearshift

The engine and transmission soft ECUs are interconnected, so that an engine torque intervention can be realized during the gearshift process. The amount of the intervened torque can be parameterized in the TORQUE\_INTERVENTION\_CONTROL block from the SoftECU\_Transmission. Refer to *New Features of ASM Vehicle Dynamics Blockset 3.4* on page 107.

## Migrating to ASM Traffic Blockset 3.5

### Look-up table migration

The discontinued Simulink blocks Lookup and Lookup2D within the ASM library blocks have been updated to the new standard Simulink Lookup Table (n-D) block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- SENSOR\_SELECTOR

---

### License check of ASM Utils blocks

The ASM\_UTILS license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- CUSTOM\_SENSOR\_OUTPUT
- CUSTOM\_SENSOR\_SCOPEZONE\_CALCULATION
- FELLOW\_PARAMETERS
- LINE\_SENSOR
- OBJECT\_POSITION
- OBJECT\_PROPERTIES\_BOX
- OBJECT\_SENSOR\_2D\_CALCULATION
- OBJECT\_SENSOR\_3D\_CALCULATION
- RADARSENSOR\_3D
- TRAFFIC\_OBJECTS
- TRAFFIC\_SCHEDULER
- TRAFFIC\_SIGN\_SENSOR\_CALCULATION

---

### SOFT\_ECU\_ACC

The block has two new outputs: Enable\_Trq\_Brake\_ACC[0] and Enable\_Trq\_Engine\_ACC[0]

---

### OUTPUT\_INTERFACE\_ACC

The block has two new inputs to disable torque request values: Enable\_Trq\_Brake\_ACC[0] and Enable\_Trq\_Engine\_ACC[0].

---

### USER\_INTERFACE\_ACC

The function to set the cruise control set velocity is corrected.



# ASM Trailer Blockset

## Where to go from here

Information in this section

<i>New Features of ASM Trailer Blockset 2.6</i>	97
<i>Changes in the ASM Trailer Demo Model</i>	97
<i>Migrating to ASM Trailer Blockset 2.6</i>	98

## New Features of ASM Trailer Blockset 2.6

### TIRE\_MODEL\_MAGIC\_FORMULA\_TRAILER\_\*\*\*

A new signal for the circumference of the unloaded tire radius has been added to the signal bus.

### SUSKIN\_RIGID\_TRUCK\_TRAILER\_\*\*\*

There is a new suspension kinematics for e.g. a truck rigid axle with a recirculation ball steering system.

### CABIN\_VELOCITIES

This new block calculates the velocity at the cabin center of gravity.

### ARTICULATED\_BODY\_CABIN

This new block calculates the articulated mass matrix and articulated generalized force of the cabin body.

### ABA\_ACC\_CABIN

This new block calculates the cabin body acceleration.

### PIVOT\_POS\_CABIN

This new block calculates the pivot point position of the cabin body.

### PIVOT\_VEL\_CABIN

This new block calculates the pivot point velocity of the cabin body.

### PIVOT\_FORCE\_TORQUE\_CABIN

This new block calculates the spring damper elements for the degrees of freedom of the cabin body.

## Changes in the ASM Trailer Demo Model

### Torque intervention during gearshift

The engine and transmission soft ECUs are interconnected, so that an engine torque intervention can be realized during the gearshift

process. The amount of the intervened torque can be parameterized in the TORQUE\_INTERVENTION\_CONTROL block from the SoftECU\_Transmission. Refer to *New Features of ASM Vehicle Dynamics Blockset 3.4* on page 107.

## Migrating to ASM Trailer Blockset 2.6

### Look-up table migration

The discontinued Simulink blocks Lookup and Lookup2D within the ASM library blocks have been updated to the new standard Simulink Lookup Table (n-D) block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- AERODYNAMICS\_TRAILER
- HITCH\_TORQUE\_BALL\_JOINT
- HITCH\_TORQUE\_BALL\_JOINT\_DOLLY
- MASTER\_BRAKE\_CYL\_TRAILER
- OMEGA\_WHEEL
- OVERRUN\_BRAKE\_TRAILER
- SUSCOMP\_OPP\_TRAILER\_FRONT
- SUSCOMP\_OPP\_TRAILER\_REAR
- SUSCOMP\_OPP\_TRAILER\_REAR\_2ND
- SUSCOMP\_OPP\_TRAILER\_REAR\_3RD
- SUSCOMP\_RIGID\_SYM\_TRAILER\_FRONT
- SUSCOMP\_RIGID\_SYM\_TRAILER\_REAR\_2ND
- SUSCOMP\_RIGID\_SYM\_TRAILER\_REAR
- SUSCOMP\_RIGID\_SYM\_TRAILER\_REAR\_3RD
- SUSCOMP\_TRAILER\_FRONT
- SUSCOMP\_TRAILER\_REAR
- SUSCOMP\_TRAILER\_REAR\_2ND
- SUSCOMP\_TRAILER\_REAR\_3RD
- TIRE\_MODEL\_MAGIC\_FORMULA\_TRAILER\_\*\*\*
- TIRE\_MODEL\_TMEASY\_TRAILER\_\*\*\*

**License check of ASM Utils blocks**

The ASM\_UTILS license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- DRUM\_BRAKE\_TRAILER\_FRONT
- DRUM\_BRAKE\_TRAILER\_REAR
- DRUM\_BRAKE\_TRAILER\_REAR\_2ND
- DRUM\_BRAKE\_TRAILER\_REAR\_3RD
- SUSFORCES\_ACTIVE\_TRAILER\_FRONT
- SUSFORCES\_ACTIVE\_TRAILER\_REAR
- SUSFORCES\_ACTIVE\_TRAILER\_REAR\_2ND
- SUSFORCES\_ACTIVE\_TRAILER\_REAR\_3RD
- SUSFORCES\_TRAILER\_FRONT\_ANC\_SPRING
- SUSFORCES\_TRAILER\_FRONT
- SUSFORCES\_TRAILER\_REAR
- SUSFORCES\_TRAILER\_REAR\_ANC\_SPRING
- SUSFORCES\_TRAILER\_REAR\_2ND
- SUSFORCES\_TRAILER\_REAR\_2ND\_ANC\_SPRING
- SUSFORCES\_TRAILER\_REAR\_3RD
- SUSFORCES\_TRAILER\_REAR\_3RD\_ANC\_SPRING
- SUSKIN\_RIGID\_SYM\_TRAILER\_FRONT
- SUSKIN\_TRAILER\_FRONT\_ASYM\_3DOF
- SUSKIN\_TRAILER\_FRONT\_ASYM\_ANC\_SPRING\_3DOF
- SUSKIN\_TRAILER\_FRONT\_SYM\_3DOF
- SUSKIN\_TRAILER\_FRONT\_SYM\_ANC\_SPRING\_3DOF
- SUSKIN\_RIGID\_SYM\_TRAILER\_REAR
- SUSKIN\_RIGID\_SYM\_TRAILER\_REAR\_2ND
- SUSKIN\_TRAILER\_REAR\_ASYM\_3DOF
- SUSKIN\_TRAILER\_REAR\_ASYM\_ANC\_SPRING\_3DOF
- SUSKIN\_TRAILER\_REAR\_SYM\_3DOF
- SUSKIN\_TRAILER\_REAR\_SYM\_ANC\_SPRING\_3DOF
- SUSKIN\_TRAILER\_REAR\_2ND\_ASYM\_3DOF
- SUSKIN\_TRAILER\_REAR\_2ND\_ASYM\_ANC\_SPRING\_3DOF

- SUSKIN\_TRAILER\_REAR\_2ND\_SYM\_3DOF
- SUSKIN\_TRAILER\_REAR\_2ND\_SYM\_ANC\_SPRING\_3DOF
- SUSKIN\_RIGID\_SYM\_TRAILER\_REAR\_3RD
- SUSKIN\_TRAILER\_REAR\_3RD\_ASYM\_3DOF
- SUSKIN\_TRAILER\_REAR\_3RD\_ASYM\_ANC\_SPRING\_3DOF
- SUSKIN\_TRAILER\_REAR\_3RD\_SYM\_3DOF
- SUSKIN\_TRAILER\_REAR\_3RD\_SYM\_ANC\_SPRING\_3DOF

---

**TIRE\_MODEL\_TMEASY\_  
TRAILER\*\*\***

Division by zero is now avoided if the friction coefficient is zero.

# ASM Truck Blockset

---

## Where to go from here

Information in this section

<i>New Features of ASM Truck Blockset 3.0</i>	101
<i>Changes in the ASM Truck Demo Model</i>	101
<i>Migrating to ASM Truck Blockset 3.0</i>	102

## New Features of ASM Truck Blockset 3.0

---

### TIRE\_MODEL\_MAGIC\_FORMULA\_\*\*\*

A new signal for the circumference of the unloaded tire radius has been added to the signal bus.

---

### SUSKIN\_RIGID\_TRUCK\_\*\*\*

New suspension kinematics for e.g. truck rigid axle with recirculation ball steering system.

---

### CABIN\_MASS

New block cabin mass to simulate additional cabin body with 2 degrees of freedom (vertical displacement and rotation about y-axis).

## Changes in the ASM Truck Demo Model

---

### Torque intervention during gearshift

The engine and transmission soft ECUs are interconnected, so that an engine torque intervention can be realized during the gearshift process. The amount of the intervened torque can be parameterized in the TORQUE\_INTERVENTION\_CONTROL block from the SoftECU\_Transmission. Refer to *New Features of ASM Vehicle Dynamics Blockset 3.4* on page 107.

## Migrating to ASM Truck Blockset 3.0

### Look-up table migration

The discontinued Simulink blocks Lookup and Lookup2D within the ASM library blocks have been updated to the new standard Simulink Lookup Table (n-D) block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- OMEGA\_WHEEL
- SUSCOMP\_RIGID\_SYM\_REAR\_2ND
- SUSCOMP\_RIGID\_SYM\_REAR\_3RD
- SUSPENSION\_COMPLIANCE\_OPP\_REAR\_2ND
- SUSPENSION\_COMPLIANCE\_OPP\_REAR\_3RD
- SUSPENSION\_COMPLIANCE\_REAR\_2ND
- SUSPENSION\_COMPLIANCE\_REAR\_3RD
- TIRE\_MODEL\_MAGIC\_FORMULA\_\*\*\*
- TIRE\_MODEL\_TMEASY\_\*\*\*

### License check of ASM Utils blocks

The ASM\_UTILS license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- DRUM\_BRAKE\_REAR\_2ND
- DRUM\_BRAKE\_REAR\_3RD
- PIVOT\_TORQUE\_X\_VEHICLE\_2ND
- SUSFORCES\_REAR\_2ND\_ANC\_SPRING
- SUSFORCES\_REAR\_3RD\_ANC\_SPRING
- SUSKIN\_REAR\_2ND\_ASYM\_3DOF
- SUSKIN\_REAR\_2ND\_ASYM\_ANC\_SPRING\_3DOF
- SUSKIN\_REAR\_2ND\_SYM\_3DOF
- SUSKIN\_REAR\_2ND\_SYM\_ANC\_SPRING\_3DOF
- SUSKIN\_REAR\_3RD\_ASYM\_3DOF
- SUSKIN\_REAR\_3RD\_ASYM\_ANC\_SPRING\_3DOF
- SUSKIN\_REAR\_3RD\_SYM\_3DOF
- SUSKIN\_REAR\_3RD\_SYM\_ANC\_SPRING\_3DOF

- SUSKIN\_RIGID\_SYM\_REAR\_2ND
- SUSKIN\_RIGID\_SYM\_REAR\_3RD
- SUSPENSION\_FORCES\_ACTIVE\_REAR\_3RD
- SUSPENSION\_FORCES\_ACTIVE\_REAR\_2ND
- SUSPENSION\_FORCES\_REAR\_2ND
- SUSPENSION\_FORCES\_REAR\_3RD

---

**TIRE\_MODEL\_TMEASY\_\*\*\***

Division by zero is now avoided if the friction coefficient is zero.

# ASM Turbocharger Blockset

## Where to go from here

Information in this section

<i>New Features of ASM Turbocharger Blockset 3.1.3</i>	104
<i>Migrating to ASM Turbocharger Blockset 3.1.3</i>	104

## New Features of ASM Turbocharger Blockset 3.1.3

### COMPRESSOR, COMPRESSOR\_HP

The `cp_air` inport has been renamed to `cp_In_Comp` and the `kappa_Air` inport to `kappa_In_Comp`.

The new inports `Conv_mdot_Comp` and `Conv_omega_TC` have been added to the block for custom value conversion.

### TURBINE,TURBINE\_HP, TURBINE\_SAEJ922

The new inports `Conv_mdot_Turb` and `Conv_omega_TC` have been added to the block for custom value conversion.

## Migrating to ASM Turbocharger Blockset 3.1.3

### Look-up table migration

The discontinued Simulink blocks `Lookup` and `Lookup2D` within the ASM library blocks have been updated to the new standard Simulink `Lookup Table (n-D)` block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- COMPRESSOR
- COMPRESSOR\_HP
- MAPS\_TC\_2\_0, MAPS\_TC\_6\_0
- SHAFT\_TC, SHAFT\_TC\_5\_0
- SHAFT\_TC\_HP
- TURBINE, TURBINE\_11\_0
- TURBINE\_HP, TURBINE\_HP\_6\_0



- TURBINE\_SAEJ922, TURBINE\_SAEJ922\_6\_0
- TURBO\_BASIC, TURBO\_BASIC\_8\_0
- TURBO\_BASIC\_2STAGE, TURBO\_BASIC\_2STAGE\_3\_0
- TURBO\_CONTROL
- WASTEGATE\_VALVE, WASTEGATE\_VALVE\_6\_0
- WASTEGATE\_VALVE\_HP, WASTEGATE\_VALVE\_HP\_3\_0

---

**License check of ASM Utils blocks**

The ASM\_UTILS license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- MAPS\_TC\_2\_0, MAPS\_TC\_6\_0
- POSTTURBHPMAN
- SHAFT\_TC, SHAFT\_TC\_2\_0, SHAFT\_TC\_5\_0
- SHAFT\_TC\_HP
- TURBINE, TURBINE\_11\_0
- TURBINE\_HP, TURBINE\_HP\_6\_0
- TURBO\_BASIC, TURBO\_BASIC\_8\_0
- TURBO\_BASIC\_2STAGE, TURBO\_BASIC\_2STAGE\_3\_0
- TURBINE\_SAEJ922

# ASM Utils

## Migrating ASM Utils

---

### Look-up table migration

The discontinued Simulink blocks `Lookup` and `Lookup2D` within the ASM library blocks have been updated to the new standard Simulink `Lookup Table (n-D)` block. Refer to *Migrating all ASM Blocksets* on page 57.

---

### License check of ASM Utils blocks

The `ASM_UTILS` license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- `Forward Euler`
- `asm_1d_integral_recip_lut`
- `asm_1d_lookup`
- `asm_2d_lookup`
- `asm_shifttable_lookup`
- `compare_value`

If these blocks are used in the model (not within another ASM block) the migration process looks for the nearest ASM block and sets the related library to be used for the license check.

---

### LabelInterface Bus2Vector block

The option of the `Bus2Vector` block to generate separate `Vector2Datatype` and `Datatype2Vector` blocks was discontinued. The conversion is now always included in the `Bus2Vector` and `Vector2Bus` blocks.

When the `Bus2Vector` block is updated, you have to delete the related `Vector2Datatype` and `Datatype2Vector` blocks manually.

# ASM Vehicle Dynamics Blockset

## Where to go from here

Information in this section

<i>New Features of ASM Vehicle Dynamics Blockset 3.4</i>	107
<i>Changes in the ASM Vehicle Dynamics Demo Model</i>	108
<i>Migrating to ASM Vehicle Dynamics Blockset 3.4</i>	108

## New Features of ASM Vehicle Dynamics Blockset 3.4

### TIRE\_MODEL\_MAGIC\_FORMULA\_\*\*\*

A new signal for the circumference of unloaded tire radius has been added to the block in the signal bus.

### SUSKIN\_RIGID\_TRUCK\_\*\*\*

The block has a new suspension kinematics model for, e.g., a truck rigid axle with recirculation ball steering system.

### SOFT\_ECU\_TRANSMISSION

The SOFT\_ECU\_TRANSMISSION block has been modularized. The old functionalities and controller strategies have been subdivided into several modular blocks.

You can now modify and extend the controller structure according to your needs. The new implementation offers also the possibility to integrate a third-party controller strategy.

The following blocks have been added to the library:

- CLUTCH\_ENGAGEMENT\_CONTROL
- LOCKUP\_CLUTCH\_CONTROL
- SHIFT\_LOCK\_CONTROL
- SHIFT\_STRATEGY
- SOFTECU\_TRANSMISSION\_SETUP
- TORQUE\_INTERVENTION\_CONTROL
- TIP\_SHIFT\_CONTROL

Because of its completely new structure, the previous SOFT\_ECU\_TRANSMISSION block is not migrated to the new implementation. Instead, the previous block is saved in FormerVersions, and the link is changed to this block during migration.

## Changes in the ASM Vehicle Dynamics Demo Model

---

### Torque intervention during gearshift

The engine and transmission soft ECUs are interconnected, so that an engine torque intervention can be realized during the gearshift process. The amount of the intervened torque can be parameterized in the TORQUE\_INTERVENTION\_CONTROL block from the SoftECU\_Transmission. Refer to *New Features of ASM Vehicle Dynamics Blockset 3.4* on page 107.

## Migrating to ASM Vehicle Dynamics Blockset 3.4

---

### Look-up table migration

The discontinued Simulink blocks Lookup and Lookup2D within the ASM library blocks have been updated to the new standard Simulink Lookup Table (n-D) block. Refer to *Migrating all ASM Blocksets* on page 57.

The look-up tables have been updated in the following blocks within this library:

- AERODYNAMICS
- BRAKE\_BOOSTER
- CENTRAL\_DIFFERENTIAL
- CLUTCH
- CLUTCH\_4WD
- ENGINE
- ENGINE\_BASIC\_3\_0
- ESP\_TORQUE\_INTERVENTION\_SLOW\_3\_0
- FRONT\_DIFFERENTIAL
- GEARBOX\_AT
- GEARBOX\_MT

- MANIFOLD\_PRESSURE
- REAR\_DIFFERENTIAL
- SOFT\_ECU\_TRANSMISSION\_8\_0
- STEERING
- STEERING\_VARIABLE\_RATIO
- STEERING\_3DOF\_VARIABLE\_RATIO
- SUBFRAME
- SUSCOMP\_RIGID\_SYM\_FRONT
- SUSCOMP\_RIGID\_SYM\_REAR
- SUSPENSION\_COMPLIANCE\_FRONT
- SUSPENSION\_COMPLIANCE\_OPP\_FRONT
- SUSPENSION\_COMPLIANCE\_OPP\_REAR
- SUSPENSION\_COMPLIANCE\_REAR
- TIRE\_MODEL\_MAGIC\_FORMULA\_\*\*\*
- TIRE\_MODEL\_TMEASY\_\*\*\*
- TORQUE\_CONVERTER
- TORQUE\_CONVERTER\_RIGID
- TORQUE\_INTERVENTION\_SLOW\_ENGINE\_BASIC
- WHEEL\_SPEED

---

#### License check of ASM Utils blocks

The ASM\_UTILS license was discontinued. The ASM Utils blocks now check the license of the ASM blockset in which they are used.

The Utils blocks in the following blocks within this library have been updated:

- CLUTCH
- DRUM\_BRAKE\_FRONT
- DRUM\_BRAKE\_REAR
- RIGID\_AXLE
- SOFT\_ECU\_TRANSMISSION
- STEERING\_VARIABLE\_RATIO
- STEERING\_3DOF\_VARIABLE\_RATIO
- SUSKIN\_FRONT\_ASYM\_3DOF
- SUSKIN\_FRONT\_ASYM\_ANC\_SPRING
- SUSKIN\_FRONT\_ASYM\_ANC\_SPRING\_3DOF

- SUSKIN\_FRONT\_SYM\_3DOF
- SUSKIN\_FRONT\_SYM\_ANC\_SPRING
- SUSKIN\_FRONT\_SYM\_ANC\_SPRING\_3DOF
- SUSKIN\_RIGID\_SYM\_FRONT
- SUSKIN\_REAR\_ASYM\_3DOF
- SUSKIN\_REAR\_ASYM\_ANC\_SPRING
- SUSKIN\_REAR\_ASYM\_ANC\_SPRING\_3DOF
- SUSKIN\_REAR\_SYM\_3DOF
- SUSKIN\_REAR\_SYM\_ANC\_SPRING
- SUSKIN\_REAR\_SYM\_ANC\_SPRING\_3DOF
- SUSKIN\_RIGID\_SYM\_REAR
- SUSFORCES\_FRONT\_ANC\_SPRING
- SUSFORCES\_REAR\_ANC\_SPRING
- SUSPENSION\_FORCES\_ACTIVE\_FRONT
- SUSPENSION\_FORCES\_ACTIVE\_REAR
- SUSPENSION\_FORCES\_FRONT, SUSPENSION\_FORCES\_FRONT\_1\_0
- SUSPENSION\_FORCES\_REAR, SUSPENSION\_FORCES\_REAR\_1\_0
- SUSPENSION\_KINEMATICS\_FRONT\_ASYMMETRIC
- SUSPENSION\_KINEMATICS\_FRONT\_SYMMETRIC
- SUSPENSION\_KINEMATICS\_FRONT\_1\_0
- SUSPENSION\_KINEMATICS\_REAR\_ASYMMETRIC
- SUSPENSION\_KINEMATICS\_REAR\_SYMMETRIC
- SUSPENSION\_KINEMATICS\_REAR\_1\_0

---

### TIRE\_MODEL\_TMEASY\_\*\*\*

Division by zero is now avoided if the friction coefficient is zero.

---

### ENGINE

The size of the engine torque parameter has been expanded from [30x30] to [31x30]. The old parameter is extrapolated and initialized in `asmigratepost`.

The engine torque look-up table `Map_EffectiveEngineTorque` was renamed to `Map_Trq_MeanEff_Engine[Nm]`.

---

### CRANK\_SHAFT

A new inport has been added for an external initialization of the speed integrator.

<b>SHAFT_***</b>	The names of the inports and the outports are made modular. Further signals have been added to the ASMSignalBus.
<b>CRANK_SHAFT_RIGID</b>	A new inport has been added for an external initialization of the speed integrator.
<b>SHAFT_RIGID</b>	A new parameter has been added to describe the rotation damping. A new inport has been added for an external initialization of the speed integrator.
<b>CLUTCH_4WD</b>	The block now gets the main reduction ratio of the rear differential via an inport instead of the GoToFrom blocks. A new inport has been added for an external initialization of the speed integrator.
<b>REAR_DIFFERENTIAL</b>	The global GoToFrom connection of the main reduction ratio intended for the CLUTCH_4WD block has been deleted. New inports have been added for an external initialization of the speed integrators.
<b>DRIVETRAIN_VARIANT_SWITCHES</b>	The comments of the drivetrain mode for the FF4WD variants have been extended from [1FourWheel 2FrontWheel 3RearWheel] to [1AWD 2FWD 3RWD 4FF-4WD 5FF].
<b>SOFT_ECU_TRANSMISSION_10_0</b>	The block cannot be migrated automatically. Therefore, during migration, the link to the block is changed to the former version:FormerVersions/SOFT_ECU_TRANSMISSION_10_0.
<b>LOCKUP_CLUTCH</b>	The inertia parameter of the driven plate is transferred to a mask parameter. A new inport has been added for an external initialization of the speed integrator.
<b>CLUTCH</b>	A new inport has been added for an external initialization of the speed integrator.
<b>CENTRAL_DIFFERENTIAL</b>	New inports have been added for an external initialization of the speed integrators.
<b>REAR_DIFFERENTIAL</b>	New inports have been added for an external initialization of the speed integrators.






---

# Bus Manager (Stand-Alone)

## Features of the Bus Manager (Stand-Alone) 5.6

---

### SAE J2602 support

The Bus Manager (stand-alone) now supports LIN communication according to the SAE J2602 standard. You can import communication matrices that comply with SAE J2602 and simulate J2602-compliant ECUs. However, some limitations apply for J2602-compliant LIN communication. For more information, refer to *Limitations for LIN Communication* ( *Bus Manager (Stand-Alone) Implementation Guide*).

---

### FIBEX 4.1.1 support

The Bus Manager (stand-alone) now supports FIBEX files based on FIBEX 4.1.1 as communication matrices.



---

# ConfigurationDesk

---

**Objective**

ConfigurationDesk is provided in two variants, useful for different scenarios. You can use ConfigurationDesk - Implementation Version to implement real-time applications. You can use ConfigurationDesk - Configuration Version to configure dSPACE RapidPro hardware.

# ConfigurationDesk – Implementation

## Where to go from here

Information in this section

<i>New Features of ConfigurationDesk 5.6 (Implementation Version)</i>	116
<i>Migrating to ConfigurationDesk 5.6</i>	119

## New Features of ConfigurationDesk 5.6 (Implementation Version)

### Support of BSC files

ConfigurationDesk now lets you add bus simulation container files (BSC files) to your ConfigurationDesk application. BSC files contain the configured bus communication of an application process. They are generated with the Bus Manager. Refer to *Working With Bus Simulation Containers* ([📖 ConfigurationDesk Real-Time Implementation Guide](#)).

### Improved Add model dialog

The Add model dialog in ConfigurationDesk was improved. It provides the following enhancements:

- You can now select several models to be added to the ConfigurationDesk application at once.
- Simulink models that are open in MATLAB are displayed in a list. You can add them directly to your ConfigurationDesk application without browsing for their paths.

Refer to *Add Model* ([📖 ConfigurationDesk Real-Time Implementation Reference](#)).

### Enhancement of tasks

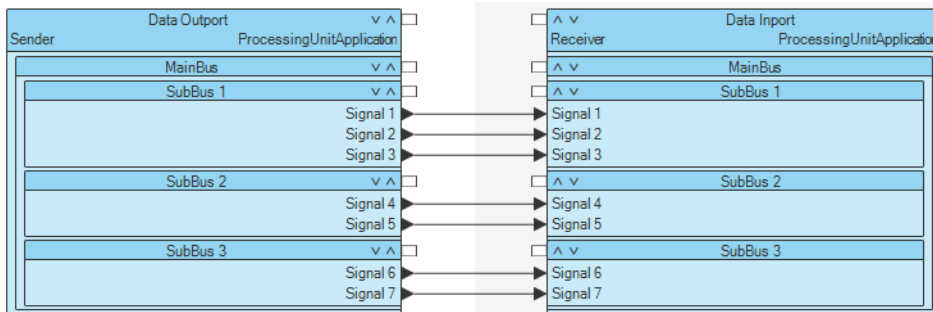
**No jitter, low latency** Tasks now provide the **Jitter and latency optimization** property that lets you run a task in polling mode. This lets you achieve smaller sample times for real-time applications. To configure a task as "No jitter, low latency", it must be the only task in the application process and triggered by a timer event. Before you configure a task as "No jitter, low latency", you should be familiar with the limitations of this feature. Refer to *How to Configure Tasks*

in ConfigurationDesk (📖 ConfigurationDesk Real-Time Implementation Guide).

**Triggering tasks** ConfigurationDesk now allows for a task to be triggered by multiple I/O events. Refer to *Basics on Modeling Asynchronous Tasks* (📖 ConfigurationDesk Real-Time Implementation Guide).

**Improved performance of model communication**

Suppose two model port blocks have the same structure, and their ports are connected one-to-one as shown in the illustration below.



In this case, ConfigurationDesk generates optimized code during the build process which leads to an improved performance of the real-time application.

**New features of the V-ECU support**

**Supported V-ECU implementation container versions** The following table shows the tool versions that export supported V-ECU implementation containers, and the related container versions:

V-ECU Implementations Created with Products of...	V-ECU Implementation Version
dSPACE Release 2016-B: ■ SystemDesk 4.7 ■ TargetLink 4.2	2.4.1
dSPACE Release 2016-A: ■ SystemDesk 4.6	2.4
dSPACE Release 2015-B: ■ SystemDesk 4.5 ■ TargetLink 4.1	2.3
dSPACE Release 2015-A: ■ SystemDesk 4.4	2.2
dSPACE Release 2014-B: ■ SystemDesk 4.3 ■ TargetLink 4.0	2.1
dSPACE Release 2013-B and earlier: ■ SystemDesk 3.2 ■ TargetLink 3.5	1.0

---

### Support of updated QNX compiler

ConfigurationDesk supports QNX compiler version 5.2.0 for 64-bit Windows. This lets you compile very large Simulink models without Out-of-Memory errors.

---

### Support of ECU interfacing

Now you can perform ECU interfacing with SCALEXIO systems. For this you must import ECU interface container (EIC) files (generated with the ECU Interface Manager) to a ConfigurationDesk application. EIC files describe an ECU application that is prepared for ECU interfacing. In ConfigurationDesk, you can integrate the prepared parts of the ECU application in the signal chain and build a real-time application for the SCALEXIO system.

**Supported ECU interfaces** To perform ECU interfacing with SCALEXIO, the target ECU must be connected to an Ethernet adapter of the SCALEXIO system via one of the following ECU interfaces:

- DCI-GSI2
- XCP on Ethernet

For details, refer to *ECU Interfacing with SCALEXIO Systems* ([📖 ConfigurationDesk Real-Time Implementation Guide](#)).

---

### New function block types

**Ethernet Setup** With the **Ethernet Setup** function block you can configure and initialize the access to an Ethernet adapter of your SCALEXIO hardware.

For more information, refer to *Ethernet Setup* ([📖 ConfigurationDesk I/O Function Implementation Guide](#))

**ECU Interface Configuration** The **ECU Interface Configuration** function block configures the data exchange between a real-time application (executed on SCALEXIO) and an ECU application (executed on an ECU) for ECU interfacing.

For more information, refer to *ECU Interface Configuration* ([📖 ConfigurationDesk I/O Function Implementation Guide](#))

---

### Enhanced function block types

**SENT In, SENT Out** The **SENT In** and **SENT Out** function blocks now provide the following enhancements:

- You can create and import user-defined protocol files to support alternative SENT protocols. Refer to *Creating User-Defined Protocol Files* ([📖 ConfigurationDesk I/O Function Implementation Guide](#)).

- New predefined protocols based on the SAE J2716 APR2016 SENT standard support more sensors, for example, temperature sensors and position/ratio sensors. For an overview of all predefined protocols, refer to *Using Application-Specific Protocols* ([📖 ConfigurationDesk I/O Function Implementation Guide](#)).

**CAN** The CAN function block now supports partial networking according to ISO 11898-6 for CAN communication. When enabled, the CAN transceiver of the assigned hardware resource can operate in the same modes as with low-power mode (stand-by, sleep, silent, normal). In contrast to low-power mode, the transceiver wakes up only if it receives a valid wake-up frame. Via properties of the CAN function block, you can specify valid wake-up frames.

For more information, refer to *CAN* ([📖 ConfigurationDesk I/O Function Implementation Guide](#)).

---

#### New features of the Bus Manager

The Bus Manager provides the following new features:

- SAE J2602 support

The Bus Manager now supports LIN communication according to the SAE J2602 standard. You can import communication matrices that comply with SAE J2602 and simulate J2602-compliant ECUs. However, some limitations apply for J2602-compliant LIN communication. For more information, refer to *Limitations for LIN Communication* ([📖 ConfigurationDesk Bus Manager Implementation Guide](#)).

- FIBEX 4.1.1 support

The Bus Manager now supports FIBEX files based on FIBEX 4.1.1 as communication matrices.

## Migrating to ConfigurationDesk 5.6

---

#### Discontinuation of platform management automation API version 1.0 as of dSPACE Release 2016-B

The platform management automation API version 1.0 was supported for the last time with ConfigurationDesk 5.5 of dSPACE Release 2016-A. Refer to *Automating Platform Management* ([📖 ConfigurationDesk Automating Tool Handling](#)).

---

#### Inconsistence Ethernet adapters

When you migrate a project to dSPACE Release 2016-B, the migration process adds the Ethernet adapter of the SCALEXIO Real-Time PC to the hardware topologies of the migrated project. The default name of the added Ethernet adapters is "[MAC address

00:00:00:00:00:00] no name assigned". This name does not match the Ethernet adapter name of the accessible platforms. Therefore, the status bar visualizes the status "No matching platform connected". This status prevents the immediate download of the real-time application after the build.

To resolve the inconsistency, specify an identical name for the Ethernet adapter of the SCALEXIO Real-Time PC in the Hardware Resource Browser and in the Platform Manager.



---

# ControlDesk

---

**Where to go from here**

Information in this section

<i>New Features of ControlDesk (ControlDesk 6.0)</i>	122
<i>Migrating to ControlDesk (ControlDesk 6.0)</i>	136

# New Features of ControlDesk (ControlDesk 6.0)

---

**Where to go from here**

## Information in this section

<i>New General Features (ControlDesk 6.0)</i>	123
<i>New Project and Experiment Features (ControlDesk 6.0)</i>	123
<i>New Features of Platform Management and Platforms/Devices (ControlDesk 6.0)</i>	124
<i>New Instrument Features (ControlDesk 6.0)</i>	126
<i>New Measurement and Recording Features (ControlDesk 6.0)</i>	130
<i>New Message Handling Features (ControlDesk 6.0)</i>	131
<i>New Bus Navigator Features (ControlDesk 6.0)</i>	131
<i>New ECU Diagnostics Features (ControlDesk 6.0)</i>	132
<i>New Electrical Error Simulation Features (ControlDesk 6.0)</i>	133
<i>New Signal Editor Features (ControlDesk 6.0)</i>	134

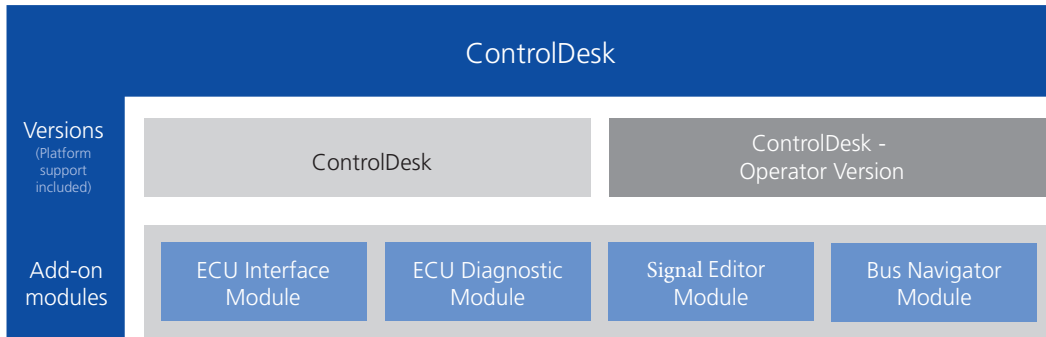
## New General Features (ControlDesk 6.0)

### 64-bit application

ControlDesk is now implemented as a 64-bit application.

### New packaging of ControlDesk

ControlDesk's packaging has been changed. The following illustration provides an overview of the available versions and modules:



Refer to *Software Versions and Modules* ([📖 ControlDesk Introduction and Overview](#)).

## New Project and Experiment Features (ControlDesk 6.0)

### Sorting folders and folder groups

ControlDesk now lets you sort folders and folder groups in the Project Manager alphabetically in ascending or descending order.

Refer to:

- *Sort - Ascending* ([📖 ControlDesk Project and Experiment Management](#))
- *Sort - Descending* ([📖 ControlDesk Project and Experiment Management](#))

## New Features of Platform Management and Platforms/Devices (ControlDesk 6.0)

### Information in this topic

*Support of the new DCI-CAN/LIN1 on page 124*

*SCALEXIO platform: Display of Ethernet and PCI/PCIe cards on page 124*

*SCALEXIO platform: Display of connected but unregistered systems on page 124*

*Display of additional important information when adding a platform/device on page 125*

### Support of the new DCI-CAN/LIN1

ControlDesk supports the new DCI-CAN/LIN1.

**CAN support** The DCI-CAN/LIN1 lets you connect your host PC to a CAN FD or CAN network.

You can use the new DCI-CAN/LIN1 with the following CAN-based devices in ControlDesk:

- CAN Bus Monitoring
- CCP
- XCP on CAN
- ECU Diagnostics

**LIN support** The DCI-CAN/LIN1 lets you connect your host PC to a LIN network.

You can use the new DCI-CAN/LIN1 with the following LIN-based device in ControlDesk:

- LIN Bus Monitoring

For details on the new DCI-CAN/LIN1, refer to the [📖 DCI-CAN/LIN1 Feature Reference](#).

### SCALEXIO platform: Display of Ethernet and PCI/PCIe cards

ControlDesk's Platform/Device Manager now displays Ethernet cards and supported PCI/PCIe cards connected to a SCALEXIO system.

### SCALEXIO platform: Display of connected but unregistered systems

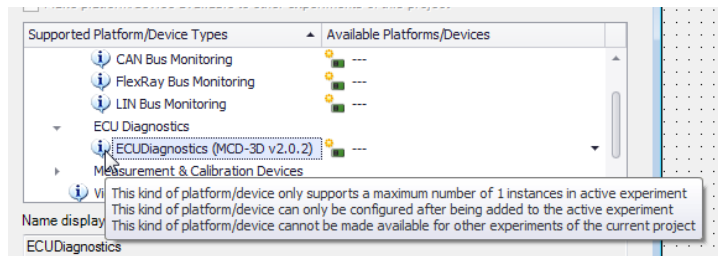
ControlDesk's Platform/Device Manager now displays connected but currently unregistered SCALEXIO systems.

Refer to *Uplink and Downlink Connections Properties* ([📖 ControlDesk Platform Management](#)).

**Display of additional important information when adding a platform/device**

If there is additional important information when adding a platform/device of a certain platform/device type (e.g., a device can only be configured after being added to the active experiment), ControlDesk displays a symbol next to the platform/device type. Move the mouse pointer over the symbol to open a tool tip with detailed information.

The following illustration shows an example:

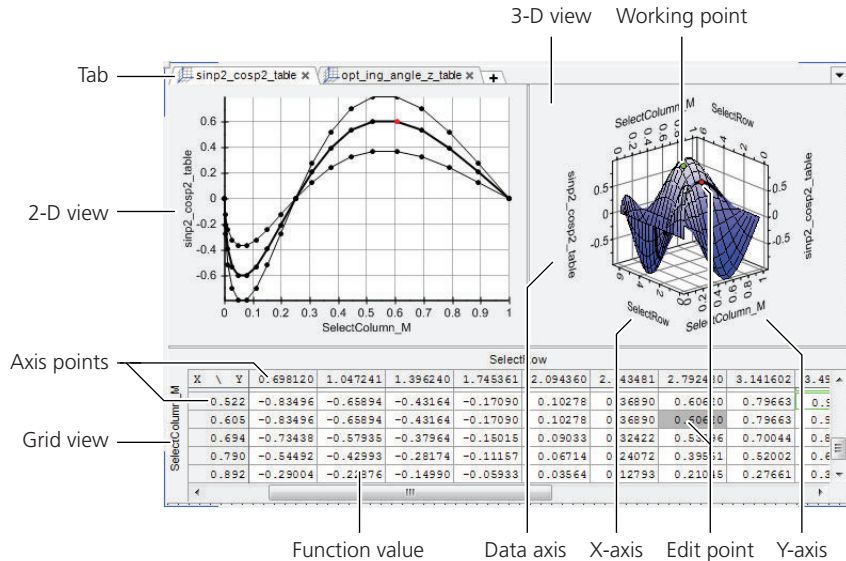


Refer to *Insert Platform / Add Platform/Device* (📖 ControlDesk Platform Management).

## New Instrument Features (ControlDesk 6.0)

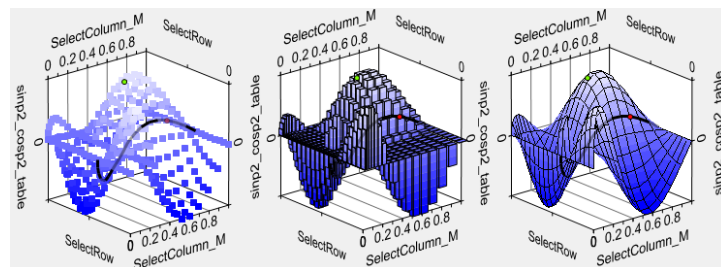
### Enhanced Table Editor

ControlDesk provides an enhanced Table Editor.



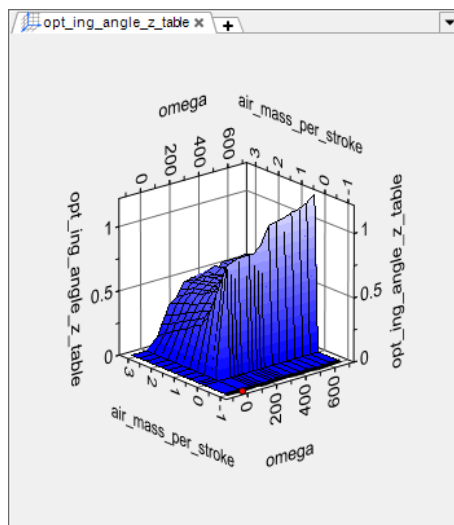
The most important enhancements are listed below.

**Selectable chart types for the 3-D view** You can select different chart types for the 3-D view: scatter plot, bar, or surface.



**Enabling/Disabling the display of views** The Table Editor now lets you enable or disable the display of each view (3-D view, 2-D view, grid view) individually by specifying the view's Visible property.

The following illustration shows a Table Editor for which the display of the 2-D and the grid view is disabled.



**Enhanced multiple selection of edit points** You can now multiselect any edit points in the Table Editor. See the following illustration as an example:

		SelectRow										
X	Y	0.000000	0.349121	0.698120	1.047241	1.396240	1.745361	2.094360	2.443481	2.792480	3.141602	3
0.000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1
0.003	0.34180	0.32861	0.29004	0.22876	0.14990	0.05933	-0.03564	-0.12793	-0.21045	-0.27661	-0.32002	2
0.012	0.64258	1.61768	0.54492	0.42993	0.28174	0.11157	-0.06714	-0.24072	-0.39551	-0.52002	-0.60000	3
0.028	0.86597	0.83228	0.73438	0.57935	0.37964	0.15015	-0.09033	-0.32422	-0.53296	-0.70044	-0.81602	4
0.049	0.98462	0.94653	0.83496	0.65894	0.43164	0.17090	-0.10278	-0.36890	-0.60620	-0.79663	-0.90000	5
0.077	0.98462	0.94653	0.83496	0.65894	0.43164	0.17090	-0.10278	-0.36890	-0.60620	-0.79663	-0.90000	6
0.111	0.86597	0.83228	0.73438	0.57935	0.37964	0.15015	-0.09033	-0.32422	-0.53296	-0.70044	-0.81602	7
0.151	0.64258	0.61768	0.54492	0.42993	0.28174	0.11157	-0.06714	-0.24072	-0.39551	-0.52002	-0.60000	8
0.198	0.34180	0.32861	0.29004	0.22876	0.14990	0.05933	-0.03564	-0.12793	-0.21045	-0.27661	-0.32002	9
0.250	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	10
0.309	-0.34180	-0.32861	-0.29004	-0.22876	-0.14990	-0.05933	0.03564	0.12793	0.21045	0.27661	0.32002	11
0.373	-0.64258	-0.61768	-0.54492	-0.42993	-0.28174	-0.11157	0.06714	0.24072	0.39551	0.52002	0.60000	12
0.444	-0.86597	-0.83228	-0.73438	-0.57935	-0.37964	-0.15015	0.09033	0.32422	0.53296	0.70044	0.81602	13

**Enhanced configuration of the 2-D and 3-D views** Configuring the 2-D and 3-D views is now more comfortable.

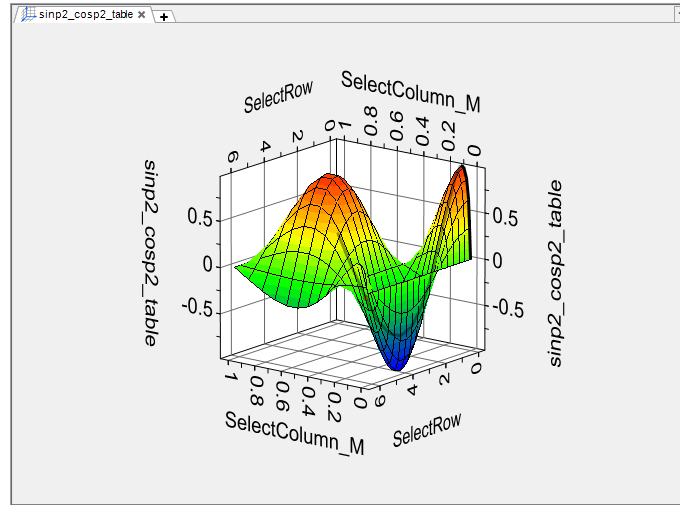
You can, for example:

- Zoom into a view with the mouse wheel
- Rotate the 3-D view with the left mouse button
- Move the 2-D view with the left mouse button
- Rescale the view with a double-click

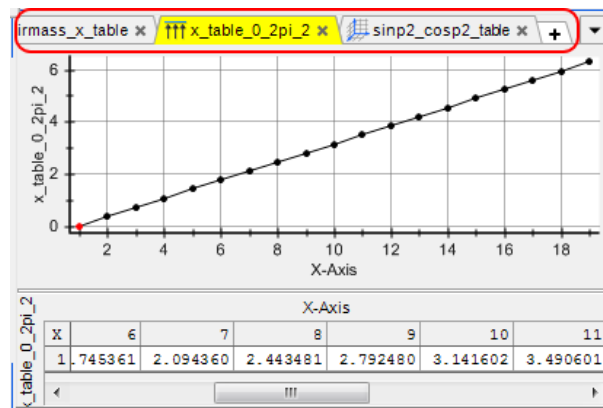
For instructions, refer to *How to Set Up a View of the Table Editor* (ControlDesk Instrument Handling).

**Enhanced color support** The Table Editor provides enhanced color support:

- You can now specify the surface color in the 3-D view. See the following illustration as an example:



- You can now specify an individual background color and a background picture for each view individually.
- You can now specify an individual background and text color for each tab of a Table Editor. See the following illustration as an example:





- You can now specify the background color of the edit point(s) in the grid view. See the following illustration as an example:

The screenshot shows a window titled 'SelectRow' with a table editor. The table has columns labeled 'X \ Y' and five numerical columns. The rows are labeled 'SelectColumn\_M' on the left. Several cells are highlighted in yellow, and some are highlighted in red. The highlighted cells are:

X \ Y	0.000000 0x1F74A	0.349121 0x1F74C	0.698120 0x1F74E	1.047241 0x1F750	1.38 0x1
0x1F6BC	0x1F0E6	0x1F0E8	0x1F0EA	0x1F0EC	0x1
0.111 0x1F6BE	0.86597 0x1F10C	0.83228 0x1F10E	0.73438 0x1F110	0.57935 0x1F112	0.3
0.151 0x1F6C0	0.64258 0x1F132	0.61768 0x1F134	> 0.46704 0x1F136	< 0.59937 0x1F138	0.2
0.198 0x1F6C2	0.34180 0x1F158	0.32861 0x1F15A	0.29004 0x1F15C	0.22876 0x1F15E	0.1
0.250 0x1F6C4	0.00000 0x1F17E	0.00000 0x1F180	0.00000 0x1F182	0.00000 0x1F184	0.0
0.309 0x1F6C6	-0.34180 0x1F1A4	-0.32861 0x1F1A6	-0.29004 0x1F1A8	-0.22876 0x1F1AA	-0.1

Refer to *Basics of Handling the Table Editor* ([ControlDesk Instrument Handling](#)).

For migration aspects refer to *Migrating from ControlDesk 5.6 to 6.0* ([ControlDesk Introduction and Overview](#)).

### Enhancements for the Instrument Selector

ControlDesk's Instrument Selector provides the following enhancements:

- You can add/remove instrument categories to/from the Instrument Selector. This allows you to structure the Instrument Selector according to your needs.
- You can easily apply changes to a custom instrument by dragging it to the original custom instrument in the Instrument Selector. ControlDesk asks you whether to overwrite the original custom instrument.
- You can add custom instruments to *any* instrument category in the Instrument Selector.
- You can exchange custom instruments by exporting the related instrument category.

Refer to *How to Configure the Instrument Selector* ([ControlDesk Instrument Handling](#)).

### New STRUCT instrument caption macro

ControlDesk now provides the `%STRUCT%` instrument caption macro. The macro lets you display the name of the variable together with the relevant parts of the struct (if the variable is part of a struct). The struct parts are separated by a dot, for example: `MyStruct.MySubstruct.MyParameter`.

Refer to *Captions/Operating Elements Properties* ([📖 ControlDesk Instrument Handling](#)).

## New Measurement and Recording Features (ControlDesk 6.0)

---

### Improved handling of large ASAM MDF 4.1 files

ControlDesk provides improvements for handling large ASAM MDF 4.1 (MF4) files:

**Improved loading of signals** ControlDesk now loads only the data that is currently needed by instruments, i.e., ControlDesk does not load complete signals stored in an MF4 file but only the signal parts to be visualized, for example, in a Time Plotter. This reduces the loading time and the required memory.

**Support of reduction data** ControlDesk supports *reduction data* of ASAM MDF 4.1 files.

Reduction data is additional content in an MF4 file that allows for visualizing the MF4 file data depending on the visualization resolution. Reduction data therefore improves the visualization and postprocessing of measurement data.

- ControlDesk's instruments such as the Time Plotter support MF4 files containing reduction data.
- ControlDesk now includes *reduction data* when measurement data is saved to the ASAM MDF 4.1 file format.

---

### Removing variables from measurement data files

ControlDesk now lets you remove one or more variables from a measurement data file opened in the Measurement Data Pool.

Refer to *Remove Variables* ([📖 ControlDesk Measurement and Recording](#)).

---

### Measuring continuously on newly added dSPACE platforms (new default setting)

As of ControlDesk 6.0, performing *continuous (untriggered) measurements* is the default for dSPACE platforms newly added to an experiment. Up to and including ControlDesk 5.6, performing *triggered measurements* was the default for newly added dSPACE platforms.

You can change the default setting for newly added dSPACE platforms on the Measurement Configuration Page.

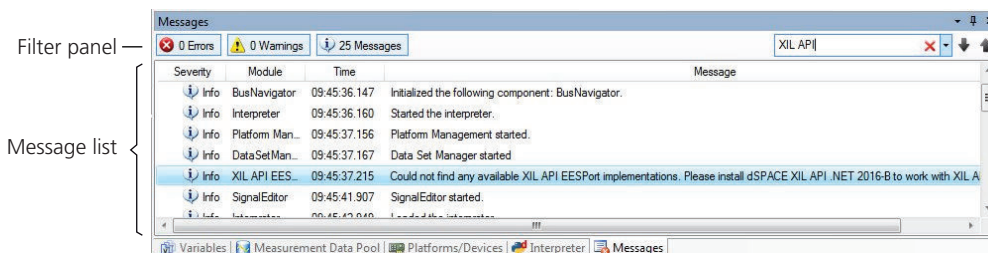
Refer to *Measurement Configuration Page* ([📖 ControlDesk Measurement and Recording](#)).

## New Message Handling Features (ControlDesk 6.0)

### New Message Viewer

ControlDesk supports the new Message Viewer. The Message Viewer provides a history of all the info, advice, error and warning messages, and all the questions that occur when you work with the product. This helps you check the system state.

The Message Viewer looks like this:



Refer to *Messages* ([ControlDesk Message Handling](#)).

### Viewing the dSPACE Log

ControlDesk lets you view the dSPACE Log.

The dSPACE Log is a collection of errors, warnings, information, questions, and advice issued by all dSPACE products and connected systems over more than one session.

To forward the dSPACE Log to dSPACE Support, use the dSPACE Installation Manager.

Refer to *dSPACE Log* ([ControlDesk Message Handling](#)).

## New Bus Navigator Features (ControlDesk 6.0)

### Bus Instrument (TX Type for FlexRay): Support of the minimum delay time

The instrument lets you display the *minimum delay time*. The minimum delay time of a PDU, which is defined in the AUTOSAR system description file, specifies the minimum delay time between successive transmissions of the new PDU data.

For reference information on the Bus Instrument (TX Type for FlexRay), refer to *Bus Instrument (TX Type for FlexRay)* ([ControlDesk Bus Navigator](#)).

---

**Loading/unloading bus signals**

ControlDesk's Bus Navigator now lets you load and unload bus signals. This is useful for bus configurations containing many bus signals which may cause memory exceptions.

ControlDesk lets you specify the default start-up behavior for loading bus signals on the [Bus Navigator Page](#).

Refer to *Bus Navigator Page* ([📖 ControlDesk Bus Navigator](#)).

To load/unload the bus signals within the [Bus Navigator](#), use the following new commands:

- *Load All Bus Signals* ([📖 ControlDesk Bus Navigator](#))
- *Unload all Bus Signals* ([📖 ControlDesk Bus Navigator](#))

## New ECU Diagnostics Features (ControlDesk 6.0)

---

**Support of DoIP (Diagnostics over Internet Protocol)**

ControlDesk's ECU Diagnostics device now also supports DoIP (Diagnostics over Internet Protocol). DoIP is a transport protocol for UDS.


For the Ethernet-based UDS on DoIP protocol, ControlDesk uses available Ethernet interfaces of the host PC to access the ECU. No further interface module is required.

For details on using the UDS on DoIP protocol in connection with the ECU Diagnostics device, refer to *Conventions in Connection with ODX Databases* ([📖 ControlDesk ECU Diagnostics](#)).

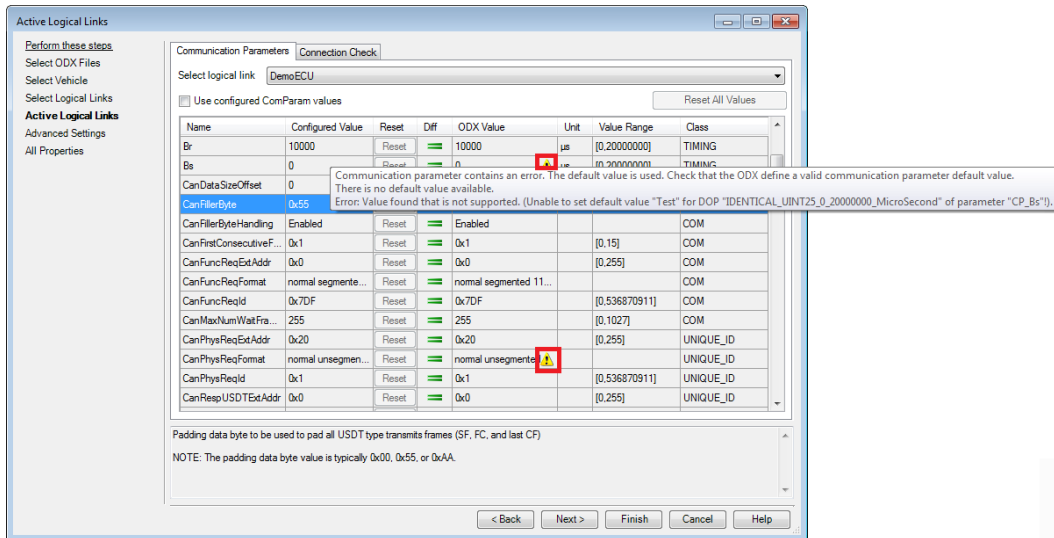
---

**Info if default COMPARAM values are used**

Up to and including version 5.6, ControlDesk implicitly used default communication parameter (`COMPARAM`) values if the `COMPARAM` values defined in the ODX database contain errors. However, ControlDesk did not display information about this behavior.

As of version 6.0, ControlDesk indicates incorrect `COMPARAM` values with a  warning symbol on the [Active Logical Links - Communication Parameters](#) page. In addition, ControlDesk provides a tooltip that informs you whether a default communication parameter is used instead of the erroneous one.

The following illustration shows an example:



Refer to *Configure Platform/Device* ([ControlDesk Platform Management](#)).

## New Electrical Error Simulation Features (ControlDesk 6.0)

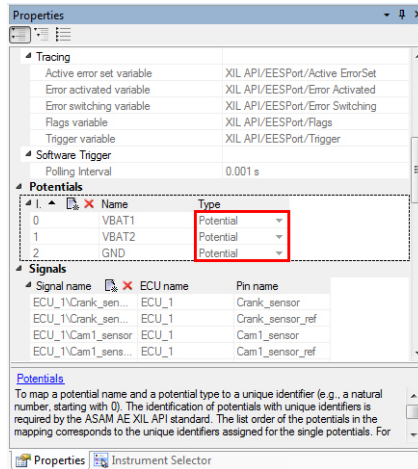
### Support of software triggers

ControlDesk now lets you activate error sets via software triggers. This lets you activate error sets in response to a defined trigger condition or duration. ControlDesk lets you configure trigger conditions according to the ASAM General Expression Syntax (GES).

Refer to *Basics on Software Triggers* ([ControlDesk Electrical Error Simulation via XIL API EESPort](#)) and *How to Configure a Software Trigger* ([ControlDesk Electrical Error Simulation via XIL API EESPort](#)).

### Automatic potential mapping

When you create an EESPort, ControlDesk automatically creates a preconfigured potential mapping based on the potential names that are provided for your selected failure simulation hardware. By default, the Type of all the potentials is set to Potential. Refer to the following illustration:



You can change the preconfigured potential mapping afterwards via the Properties controlbar.

Refer to *Basics on Potential Mapping* ([📖](#) *ControlDesk Electrical Error Simulation via XIL API EESPort*).

### Automatic signal mapping

When you create a new EESPort, you can let ControlDesk generate a default signal mapping which is based on the ECU names and pin names that are specified for your dSPACE simulator.

Refer to *Basics on Signal Mapping* ([📖](#) *ControlDesk Electrical Error Simulation via XIL API EESPort*).

### Support of drag & drop of variables

ControlDesk's Electrical Error Simulation now supports drag & drop of variables: You can drag variables from the Variable Browser to the Properties controlbar, for example, when you specify a trigger variable for the trigger condition of a software trigger.

## New Signal Editor Features (ControlDesk 6.0)

### Support of the eLINEAR interpolation method

ControlDesk's Signal Editor now supports the eLINEAR interpolation method for Data File segments and Signal Value segments.

Refer to *Interpolation Property* ([📖](#) *ControlDesk Signal Editor*).

---

**Automatic refresh of Data  
File segment display during  
download**

ControlDesk now automatically refreshes the screen display of Data File segments when you download the related signal generator.

Refer to *Data File (Segment)* ( *ControlDesk Signal Editor*).

# Migrating to ControlDesk (ControlDesk 6.0)

## Where to go from here

Information in this section

<i>Discontinuations in ControlDesk</i>	136
<i>Migrating to ControlDesk (ControlDesk 6.0)</i>	138

## Discontinuations in ControlDesk

### Information in this topic

*Discontinuations as of ControlDesk 6.0* on page 136

*ControlDesk Failure Simulation Module* on page 136

*Platform management automation API version 1.0* on page 136

*Plotter* on page 137

*Integrated Variable Editor* on page 137

*MDF file (format versions 2.0 and 3.0) export* on page 137

*Methods for handling messages* on page 137

*Migration of ControlDesk 3.x experiments* on page 138

*Migration of CalDesk projects* on page 138

*Discontinuations for ControlDesk as of dSPACE Release 2017-A* on page 138

*PAR file import* on page 138

### Discontinuations as of ControlDesk 6.0

**ControlDesk Failure Simulation Module** ControlDesk's Failure Simulation Module was delivered for the last time with ControlDesk 5.6 of dSPACE Release 2016-A.

- To prepare an electrical error simulation via the graphical user interface (GUI), use the ControlDesk XIL API EESPort GUI, which was introduced with ControlDesk 5.5 (dSPACE Release 2015-B).
- To prepare an electrical error simulation via automation, use the dSPACE XIL API .NET implementation, which supports the Electrical Error Simulation Port (EESPort).

For information on migration aspects, refer to *Migrating to ControlDesk (ControlDesk 6.0)* on page 138.

**Platform management automation API version 1.0** Platform management automation API version 1.0 was supported for the last time with ControlDesk 5.6 of dSPACE Release 2016-A.



For information on migrating to API version 2.0, which was introduced with ControlDesk 5.2 from dSPACE Release 2014-A, refer to *Migrating from ControlDesk 5.1 to 5.2* ([📖 ControlDesk Introduction and Overview](#)).

**Plotter** The Plotter was delivered for the last time with ControlDesk 5.6 of dSPACE Release 2016-A.

Use one of the following instruments instead:

- Index Plotter
- Time Plotter
- XY Plotter

For information on the differences between the different plotter types, refer to *Differences Between Time Plotter, Index Plotter, and XY Plotter* ([📖 ControlDesk Instrument Handling](#)).

For information on migration aspects, refer to *Migrating from ControlDesk 5.6 to 6.0* ([📖 ControlDesk Introduction and Overview](#)).

**Integrated Variable Editor** ControlDesk no longer provides the Variable Editor as an integrated component.

However, you can still work with the Variable Editor as a stand-alone tool. The stand-alone Variable Editor is available at <https://www.dspace.com/go/requestreleasedownload>.

**MDF file (format versions 2.0 and 3.0) export** As of ControlDesk 6.0, exporting MDF measurement data files (MDF file format versions 2.0 and 3.0) is no longer supported.

Support for *importing* MDF files (format versions 2.0 and 3.0) will continue.

To export measurement data, use one of the other file formats supported by ControlDesk. Refer to *How to Configure the Storage Settings for Recording* ([📖 ControlDesk Measurement and Recording](#)).

**Methods for handling messages** As of dSPACE Release 2016-B, all dSPACE products use improved methods for handling messages such as errors and warnings.

As a consequence:

- Messages are no longer written to the `dSPACE.log` file, i.e., they are no longer available as plain text.

To collect diagnostics information including log messages and send it to dSPACE Support, use the *dSPACE Installation Manager*.

- In ControlDesk 6.0, the dSPACE Message Monitor, which allows you to monitor log messages that are recorded by the dSPACE Message Service, has been removed.

- In ControlDesk 6.0, the `LogFilePath` property of the `Log / ILoLog <<Interface>>`, which gets the full path name of the message log file, has been removed.

For information on the improved methods for handling messages, refer to [ControlDesk Message Handling](#).

**Migration of ControlDesk 3.x experiments** ControlDesk 5.6 of dSPACE Release 2016-A was the last version that supports the migration of ControlDesk 3.x experiments for reuse in ControlDesk. This also applies to the import of ControlDesk 3.x layouts.

**Migration of CalDesk projects** ControlDesk 5.6 of dSPACE Release 2016-A was the last version that supports the migration of CalDesk projects for reuse in ControlDesk. This also applies to the import of CalDesk layouts.

---

**Discontinuations for  
ControlDesk as of  
dSPACE Release 2017-A**

**PAR file import** As of dSPACE Release 2017-A, ControlDesk will no longer support the import of PAR files created with ControlDesk 3.x.

---

## Migrating to ControlDesk (ControlDesk 6.0)

To migrate from ControlDesk 5.6 to ControlDesk 6.0 and reuse existing experiments, you might have to carry out the following migration steps.

### Note

To migrate to ControlDesk 6.0 from versions earlier than 5.6, you might also have to perform the migration steps of the intervening ControlDesk versions.

**Information in this topic**

*Failure Simulation Module: Discontinuation and migration* on page 139

*Plotter: Discontinuation and migration* on page 140

*Table Editor* on page 140

*Changed default behavior for measurements on dSPACE platforms* on page 141

*Change to the import of A2L files* on page 141

*ECU Diagnostics: DLLs called by ECU diagnostic jobs* on page 141

*Signal Editor: Changed implementation of the eBACKWARD interpolation method* on page 141

*Tool automation changes* on page 142

- Failure Simulation Module: Discontinuation and migration* on page 142
- Discontinuation of platform management automation API version 1.0* on page 142
- No support for 32-bit third-party Python extension modules* on page 142
- Change to the IXaMeasurementDataManagement interface* on page 142
- Change to the IXaECUDiagnostics202ServerSettings interface* on page 143
- Table Editor: Change to the creation of Table Editor arrays* on page 143
- Discontinuation of the dSPACE Message Monitor* on page 144
- Change to the ILoLog interface* on page 144

*Measurement Data API changes* on page 144

- MDFFormatOption2 interface no longer available* on page 144

*Migrating from prior ControlDesk versions* on page 144

**Failure Simulation Module: Discontinuation and migration**

ControlDesk's Failure Simulation Module was delivered for the last time with ControlDesk 5.6 of dSPACE Release 2016-A.


- To prepare an electrical error simulation via the graphical user interface (GUI), use the ControlDesk XIL API EESPort GUI, which was introduced with ControlDesk 5.5 (dSPACE Release 2015-B).

To use the ControlDesk XIL API EESPort GUI, the Failure Simulation Package is required, which is based on XIL API's EESPort. The implementation is based on dSPACE XIL API .NET.

Keep in mind that electrical error configurations of ControlDesk's Failure Simulation Module are not compatible with XIL API EESPort configurations.

For the migration, you can use the

`FailureSimulationExportTool` to export information from an existing ControlDesk failure simulation system (FSN) file to the following files:

- A hardware-dependent port configuration (PORTCONFIG) file  
You can use the file to create a new EESPort. For instructions, refer to *How to Create a New EESPort* ( *ControlDesk Electrical Error Simulation via XIL API EESPort*).

- One error configuration XML file for each failure pattern  
You can use the files to create and configure electrical errors, refer to *How to Create and Configure an Electrical Error* ([📖 ControlDesk Electrical Error Simulation via XIL API EESPort](#)).

The `FailureSimulationExportTool` version to use depends on the installed version of ControlDesk and dSPACE XIL API .NET as shown in the following table:

Installed ControlDesk Version	Installed dSPACE XIL API .NET Version	Required FailureSimulationExportTool Version
5.3	2.0	2014-B
5.4	2015-A	2015-A
5.5	2015-B	2015-B
5.6	2016-A	2016-A

To get the `FailureSimulationExportTool`, including a ReadMe file containing user documentation, contact dSPACE Support.

- To prepare an electrical error simulation via automation, use the dSPACE XIL API .NET implementation, which supports the Electrical Error Simulation Port (EESPort).

---

### Plotter: Discontinuation and migration

The Plotter was delivered for the last time with ControlDesk 5.6 of dSPACE Release 2016-A.

- ControlDesk 6.0 and later lets you open an experiment with Plotters. Plotter instruments including their properties and variable connections are automatically migrated to Time Plotter (refer to *Time Plotter Handling* ([📖 ControlDesk Instrument Handling](#))) instruments during experiment migration.
- The Time Plotter, which is the successor to the Plotter for displaying signals measured in a time-based raster, provides a new automation concept. As a consequence, automation scripts originally developed for automating the Plotter cannot be reused for automating the Time Plotter.

---

### Table Editor


ControlDesk 6.0 comes with an enhanced Table Editor.

- ControlDesk 6.0 and later lets you open an experiment with Table Editors created in ControlDesk 5.6 or earlier. These instruments keep their properties and variable connections during experiment migration.

- Automation scripts originally developed for automating the Table Editor in ControlDesk 5.6 or earlier can be reused in ControlDesk 6.0 and later. However, there is a migration aspect in connection with the automation of Table Editor *arrays*. Refer to *Table Editor: Change to the creation of Table Editor arrays* on page 143.

---

#### Changed default behavior for measurements on dSPACE platforms

Up to and including ControlDesk 5.6, the Measure continuously on newly added dSPACE platforms option on the Measurement Configuration Page ( *ControlDesk Measurement and Recording*) was inactive after installation, i.e., measurements on newly added dSPACE platforms were *triggered* by default.

As of ControlDesk 6.0, the Measure continuously on newly added dSPACE platforms option is active after installation, i.e., measurements on newly added dSPACE platforms are *continuous* by default.

---

#### Change to the import of A2L files

There is a change in the import of A2L files in ControlDesk 6.0:

- Up to and including version 5.6, ControlDesk did not support the COLUMN\_DIR setting for the LAYOUT flag of measurement arrays. If an A2L file to be imported contained such measurement arrays, ControlDesk assumed the ROW\_DIR setting for the LAYOUT flag of the measurement array instead.
- As of version 6.0, ControlDesk supports the COLUMN\_DIR setting for the LAYOUT flag of measurement arrays. If an A2L file to be imported contains such measurement arrays, ControlDesk uses the COLUMN\_DIR setting.

---

#### ECU Diagnostics: DLLs called by ECU diagnostic jobs

Diagnostic jobs can call DLLs, for example, Seed&Key DLLs to get security access for ECU parameter calibration.

As of version 6.0, ControlDesk supports only 64-bit DLLs called by ECU diagnostic jobs. 32-bit DLLs called by ECU diagnostic jobs are no longer supported. You therefore have to recompile these DLLs.

---

#### Signal Editor: Changed implementation of the eBACKWARD interpolation method

The implementation used for the eBACKWARD interpolation method for Data File segments and Signal Value segments has been changed:

- Up to and including ControlDesk 5.6 and Real-Time Testing 3.0 (dSPACE Release 2016-A), Real-Time Testing's `RM_SAMPLED` data streaming mode was used.

- As of ControlDesk 6.0 and Real-Time Testing 3.1 (dSPACE Release 2016-B), Real-Time Testing's `RM_BACKWARD` data streaming mode is used. The mode complies with the specification of the `eBACKWARD` interpolation method defined in the ASAM AE XIL API standard.

For details on the difference between the two interpolation methods, refer to *MatFile Class Description* ([📄 Real-Time Testing Library Reference](#)).

## Tool automation changes

### Failure Simulation Module: Discontinuation and migration

ControlDesk's Failure Simulation Module was delivered for the last time with ControlDesk 5.6 of dSPACE Release 2016-A.

To prepare an electrical error simulation via automation, use the dSPACE XIL API .NET implementation, which supports the Electrical Error Simulation Port (EESPort).

### Discontinuation of platform management automation

**API version 1.0** Platform management automation API version 1.0 was supported for the last time with ControlDesk 5.6 of dSPACE Release 2016-A.

For information on migrating to API version 2.0, which was introduced with ControlDesk 5.2 from dSPACE Release 2014-A, refer to *Migrating from ControlDesk 5.1 to 5.2* ([📄 ControlDesk Introduction and Overview](#)).

### No support for 32-bit third-party Python extension modules

As of version 6.0, ControlDesk's Internal Interpreter supports only 64-bit third-party Python extension modules. 32-bit third-party Python extension modules are no longer supported.

**Change to the IXaMeasurementDataManagement interface** As of ControlDesk 6.0, exporting MDF measurement data files (MDF file format versions 2.0 and 3.0) is no longer supported.

The following properties of the `MeasurementDataManagement / IXaMeasurementDataManagement <<Interface>>` interface have therefore been removed in ControlDesk 6.0:

- `MDFFormat`
- `VariableNameStorage`
- `MDFXAxisDataType`

Refer to `MeasurementDataManagement / IXaMeasurementDataManagement <<Interface>>`.

**Change to the IXaECUDiagnostics202ServerSettings interface**

As of ControlDesk 6.0, the diagnostic server is always started from within the ControlDesk process.

The following properties of the `ECUDiagnostics202ServerSettings / IXaECUDiagnostics202ServerSettings <<Interface>>` interface have therefore been removed in ControlDesk 6.0:

- `ExecuteDiagnosticsServerInProcessOfCDNGEnabled`  
This is always enabled and cannot be changed.
- `UseServerJVMForDatabaseOptimizationEnabled`  
This is always enabled and cannot be changed.

Refer to *ECUDiagnostics202ServerSettings / IXaECUDiagnostics202ServerSettings <<Interface>>* ([PDF](#) *ControlDesk Automation*).

**Table Editor: Change to the creation of Table Editor arrays**

ControlDesk 6.0 comes with an enhanced Table Editor. Automation scripts originally developed for automating the Table Editor in ControlDesk 5.6 or earlier can be reused in ControlDesk 6.0 and later.

However, the `Table Editor Array` instrument type is no longer available. You have to use the `Table Editor` instrument type instead. Unlike the `Table Editor Array` instrument type, the `Table Editor` instrument type already has one subinstrument by default.

The following listing shows how to create a Table Editor with two subinstruments in ControlDesk 5.6 and earlier:

**Creating a Table Editor with two Subinstruments in ControlDesk 5.6 and Earlier**

```
TableEditor = ControlDeskApplication.LayoutManagement.ActiveLayout.Instruments.Add(
    "Table Editor Array",
    TableEditorInstrumentName,
    0,
    0,
    500,
    500)

# Add sub instrument.
TableEditor.SubInstruments.Add()
(...)
# Add sub instrument.
TableEditor.SubInstruments.Add()
```

The following listing shows how to create a Table Editor with two subinstruments in ControlDesk 6.0 and later:

### Creating a Table Editor with two Subinstruments in ControlDesk 6.0 and Later

```
TableEditor = ControlDeskApplication.LayoutManagement.ActiveLayout.Instruments.Add(
    "Table Editor",
    TableEditorInstrumentName,
    0,
    0,
    500,
    500)

# Add sub instrument.
TableEditor.SubInstruments.Add()
```

**Discontinuation of the dSPACE Message Monitor** In ControlDesk 6.0, the dSPACE Message Monitor, which allows you to monitor log messages that are recorded by the dSPACE Message Service, has been removed.

**Change to the ILoLog interface** In ControlDesk 6.0, the `LogFilepath` property of the `Log / ILoLog <<Interface>>`, which gets the full path name of the message log file, has been removed.

Refer to `Log / ILoLog <<Interface>>`.

### Measurement Data API changes

**MDFFormatOption2 interface no longer available** As of ControlDesk 6.0, exporting MDF measurement data files (MDF file format versions 2.0 and 3.0) is no longer supported.

As a consequence, the `MDFFormatOption2` interface is no longer available.

Refer to [📖 ControlDesk Measurement Data API](#).

### Migrating from prior ControlDesk versions

To migrate from prior ControlDesk versions and reuse existing experiments, you might have to carry out additional migration steps. For more information on the migration steps, refer to *Migrating from Prior Versions of ControlDesk* ([📖 ControlDesk Introduction and Overview](#)).

### Related topics

Basics

- *Basics on Migrating from Prior Versions of ControlDesk* ([📖 ControlDesk Introduction and Overview](#))



# DCI Configuration Tool

## New Features of the DCI Configuration Tool 3.7

---

### Improved A2L file adaptation

The DCI Configuration Tool comes with improvements related to the adaptation of an existing A2L file for use with a DCI-GSI2.

Refer to *A2L File Page* ([📖 DCI Configuration](#)).

---

### Manual ECU memory access

The DCI Configuration Tool comes with the new **ECU Memory Access** dialog page, which lets you manually check read and write accesses to any specific ECU memory address.

Refer to *ECU Memory Access Page* ([📖 DCI Configuration](#)).

---

### Firmware versions for DCI-GSI1 and DCI-GSI2 interfaces

The following firmware versions for the DCI-GSI1 and DCI-GSI2 interfaces are delivered with the DCI Configuration Tool 3.7:

- DCI-GSI1 firmware version 1.6.7
- DCI-GSI2 firmware version 1.4.4

#### Note

The firmware version delivered with the DCI Configuration Tool is not always the latest firmware version available. If you encounter problems, contact dSPACE Support to check if a later firmware version is available.



# dSPACE CAN API Package

## New Features of dSPACE CAN API Package 3.0

---

### dSPACE CAN API 1.0 versus dSPACE CAN API 2.0

dSPACE CAN API Package 3.0 provides the following API versions:

- dSPACE CAN API 1.0
- dSPACE CAN API 2.0

dSPACE CAN API 2.0 is introduced with dSPACE Release 2016-B. It is the successor of dSPACE CAN API 1.0, includes all previous features and additionally supports CAN FD. Unlike dSPACE CAN API 1.0, dSPACE CAN API 2.0 will be developed further.

---

### Support of CAN FD

The new dSPACE CAN API 2.0 supports CAN FD.

Refer to *Basics on CAN FD* ([📖 dSPACE CAN API 2.0 C Reference](#)).

---

### Support of the DCI-CAN/LIN1 interface

The dSPACE CAN API 1.0 and the new dSPACE CAN API 2.0 support the DCI-CAN/LIN1 interface.

Refer to [📖 dSPACE CAN API 1.0 C Reference](#) and [📖 dSPACE CAN API 2.0 C Reference](#).



---

# dSPACE ECU Flash Programming Tool

## New Features of the dSPACE ECU Flash Programming Tool 2.3.1

---

### **Support of the DCI-CAN/LIN1 interface**

The dSPACE ECU Flash Programming Tool supports dSPACE's DCI-CAN/LIN1 interface for CAN and CAN FD.

Refer to *Supported ECU Interface Types* ([📖 ECU Flash Programming](#)).

---

### **Loading XCP on CAN flash projects from versions earlier than 2.3 without any modification**

As of dSPACE ECU Flash Programming Tool 2.3, CAN and CAN FD are supported as the XCP transport layer. In this context, further configuration settings were added to the Configure Flash Project wizard.

dSPACE ECU Flash Programming Tool 2.3.1 now automatically migrates XCP on CAN flash projects from versions earlier than 2.3 during the loading process.



# dSPACE FlexRay Configuration Package

## New Features of dSPACE FlexRay Configuration Package 3.8

---

### Discontinuations for future versions

**Planned discontinuation of signal-based modeling as of dSPACE Release 2017-A** Signal-based modeling with the RTI FlexRay Configuration Blockset is being supported for the last time with dSPACE Release 2016-B. As of dSPACE Release 2017-A, the RTI FlexRay Configuration Blockset supports only PDU-based modeling.

As a consequence, you should switch from the signal-based modeling concept to PDU-based modeling with the RTI FlexRay Configuration Blockset. From the Simulink configuration data, created by the FlexRay Configuration Tool, you have to generate configured RTI FlexRay blocks for PDU-based modeling. The automatically generated FlexRay model has RTI blocks for each configured PDU that comprises several signals, i.e., PDU-based modeling handles several signals with one Simulink block. You can access the single signals via the PDU blocks.

PDU-based modeling with the RTI FlexRay Configuration Blockset offers more functionality than signal-based modeling.





---

# dSPACE Python Extensions

---

## Where to go from here

Information in this section

<i>New Features of dSPACE Python Extensions 2.2</i>	153
<i>Migrating to dSPACE Python Extensions 2.2</i>	153

---

## New Features of dSPACE Python Extensions 2.2

### 64-bit software support

dSPACE Python Extensions is now implemented as 64-bit software.

This has the following effects:

- Installation path is C:\Program Files instead of C:\Program Files (x86).
- Only 64-bit client applications are supported.
- The `matlablib2` and `rs232lib2` Python modules can only be used in a 64-bit Python interpreter.

---

## Migrating to dSPACE Python Extensions 2.2

### Discontinuation of software contained in dSPACE Python Extensions

With dSPACE Release 2016-B, dSPACE Python Extensions no longer provides:

- dSPACE HIL API Python Implementation
- `rtplib2`

You can migrate your test automation projects to ASAM XIL API as the HIL API successor.

For information on migrating from HIL API Python or `rtplib2` to XIL API .NET, refer to the Test Automation Tools Support Center: <http://www.dspace.com/go/pscta>.

# dSPACE XIL API .NET

---

## Where to go from here

Information in this section

<i>New Features of dSPACE XIL API .NET 2016-B</i>	155
<i>Migrating to dSPACE XIL API .NET 2016-B</i>	157

---

## New Features of dSPACE XIL API .NET 2016-B

### 64-bit software support

dSPACE XIL API .NET is now implemented as 64-bit software.

This has the following effects:

- Installation path is `C:\Program Files` instead of `C:\Program Files (x86)`.
- The product components of dSPACE XIL API .NET 2016-B are only available as 64-bit variants.

The consequences are:

- Applications written in a .NET programming language, e.g., C#, that use the XIL API .NET MAPort or EESPort must be compiled using 64-bit as the target architecture.
- Python scripts that use the XIL API .NET MAPort or EESPort can only be used in a 64-bit Python interpreter and are not available in a 32-bit Python interpreter.
- Scripts written in M that use the XIL API .NET MAPort can only be executed with 64-bit MATLAB.

---

**Platform Management API**

The Platform Management API has moved from the Python Extensions setup to the dSPACE XIL API setup. As before, it can be installed without a license.

**Discontinuation of platform management automation**

**API version 1.0 as of dSPACE Release 2016-B** Platform management automation API version 1.0 is no longer supported. You must use the API version 2.0.

For more information, refer to *Basics on the Platform Management API* ([📖 dSPACE Platform Management API Reference](#)).

---

**XIL API Framework**

The ASAM XIL API Framework allows you to initialize several Testbench ports from different vendors and to access variables via variable objects. Using variable objects means, that the concrete model path of a model variable in the real-time application is mapped to an abstract identifier in the Framework. These mappings are stored in a specific XML-based file format that is also supported by ControlDesk and AutomationDesk.

The XIL API Framework also supports unit conversion, i.e., the unit used in the test environment can be converted to the unit used in the real-time application. For example, you can convert from km/h to miles/h.

For more information, refer to *Basics on the Framework* ([📖 dSPACE XIL API Implementation Guide](#)).

---

**Enhanced MAPort functionality**

The MAPort implementation comes with the following enhancements:

- The MAPort stimulation now supports the `eLINEAR` interpolation method for segments of `DataFileSegment` and `SignalValueSegment` type. In this method, signal values are calculated by linear interpolation between the data points given by the segment.
- The `eBACKWARD` interpolation method has been modified to fulfill the behavior specified in the ASAM XIL API standard. For more information, refer to *Migrating to dSPACE XIL API .NET 2016-B* on page 157.
- With dSPACE XIL API 2016-B, the use of the `eFORWARD` interpolation method introduces an exception. Up to dSPACE XIL API 2016-A, the `eFORWARD` interpolation method was implicitly mapped to the `eBACKWARD` method.
- The Reduction Data feature of ASAM MDF 4.1 (MF4) files is supported.

**Enhanced EESPort functionality**

**Software trigger** An error set can be now triggered by software. You can either specify a condition or a duration to activate an error set.

For more information, refer to *Basic Information on Configuring Errors* ([📖 dSPACE XIL API Implementation Guide](#)).

## Migrating to dSPACE XIL API .NET 2016-B

**Migrating applications from dSPACE HIL API .NET to dSPACE XIL API .NET**

dSPACE HIL API .NET is discontinued with dSPACE Release 2016-B. For information on the required migration steps, refer to *Migrating HIL API Applications to XIL API Applications* ([📖 dSPACE XIL API Implementation Guide](#)).

**Modified behavior for eBACKWARD interpolation**

The implementation used for the eBACKWARD interpolation method for segments of `DataFileSegment` and `SignalValueSegment` type has been changed:

- Up to and including XIL API 2016-A and Real-Time Testing 3.0, Real-Time Testing's `RM_SAMPLED` data streaming mode was used.
- As of XIL API 2016-B and Real-Time Testing 3.1, Real-Time Testing's `RM_BACKWARD` data streaming mode is used. The mode complies with the specification of the eBACKWARD interpolation method defined in the ASAM AE XIL API standard.

For details on the difference between the two interpolation methods, refer to *MatFile Class Description* ([📖 Real-Time Testing Library Reference](#)).

**Changed signal evaluation on the host PC**

With XIL API 2016-B, the evaluation of the signals (when you execute the `CreateSignalValue` method for a `SignalDescription`) on the host PC changed:

- Up to and including XIL API 2016-A, it is possible that a segment boundary is between two sample steps, so that the successive segment is able to start between sample steps.
- As of XIL API 2016-B, the sampled values of a segment always start with the time stamp of a sample step.

**New version of EESPortConfiguration API**

The version of the EESPortConfiguration API is to be changed from 2.2.0.0 to 3.0.0.0.

The version is required when you reference the related assembly in your application:

- Using dSPACE XIL API .NET 2016-A:

- dSPACE.XIL.Testbench.EESPort.Interfaces.Extended.dll  
(Version 2.2.0.0, PublicKeyToken=f9604847d8afbfb)

- Using dSPACE XIL API .NET 2016-B:

- dSPACE.XIL.Testbench.EESPort.Interfaces.Extended.dll  
(Version 3.0.0.0, PublicKeyToken=f9604847d8afbfb)

---

# ECU Interface Manager

## Migrating to ECU Interface Manager 2.0

---

### **Migrating projects last saved with a former version of ECU Interface Manager**

In the ECU Interface Manager 2.0, you can reuse projects that were last saved with a former version of the ECU Interface Manager.

When you open such a project for the first time, you are asked if you want to update it.

- When you start the update, you can continue working with the project with ECU Interface Manager 2.0.
- When you postpone the update, actions are blocked, except for exporting the application. You can update the project later.
- When you save the project, you are asked if you want to overwrite the old project file:
  - When you overwrite the old project, you can no longer use it with a former version of the ECU Interface Manager.
  - When you do not overwrite the old project, you have to specify another location and/or name for the project file. This lets you keep a version of the project that you can work with in the former version of ECU Interface Manager.

---


### **New software module description file schema**


As of ECU Interface Manager 1.6, ECU suppliers can now use a generic schema to create a software module description file.

You can also import software module description files based on the dSPACE-specific schema, which was originally introduced with ECU Interface Manager 1.0.

### Note

- The dSPACE-specific schema is supported for downward compatibility reasons only. It will be replaced by the generic schema in the next dSPACE releases.
- Multicore support and further developments are not available with the dSPACE-specific schema.

Use the *generic schema* (refer to *Using the Generic Schema to Create Software Module Description Files* ( *ECU Interface Manager Reference*)) instead.

For more information on the generic schema, refer to *Using the Generic Schema to Create Software Module Description Files* ( *ECU Interface Manager Reference*).



---

# Firmware Manager

## New Features of Firmware Manager 2.2

---

### Usability improvements

**Enhanced Help ribbon** The Help ribbon in the Backstage view now provides direct access to the New Features and Migration document and to the PDF files installed with your dSPACE software.

For more information, refer to *Basics on Ribbons* ([📖 Firmware Manager Document](#)).

**New Message Viewer** The Firmware Manager supports the new Message Viewer. The Message Viewer provides a history of all the info, advice, error and warning messages, and all the questions that occur when you work with the product. This helps you check the system state.



# Model Compare

## Note

### Product use prohibited in United States

You are not licensed to use Model Compare in the United States. You are not allowed to use or permit others to use this product in the United States or in any way that violates the laws of the United States.

## Where to go from here

Information in this section

<i>New Features of Model Compare 2.7</i>	163
<i>Migration to Model Compare 2.7</i>	164

## New Features of Model Compare 2.7

### Three-way analysis

Model Compare lets you specify a third model (e.g., *common ancestor model*) which refers to the reference model and the comparison model for a more precise comparison analysis. The analysis comprises model parts and property values.

#### Related documentation

- *How to Create Comparison Sessions* ([📖 Model Compare Guide](#))
- *tSimulinkBusObject* ([📖 TargetLink API Reference](#))

---

### Improved interaction with version control systems

Model Compare supports version control maintained by external version control systems. You can assign a string or identifier, respectively, to comparison, reference, and common ancestor models via command line (API), which allows the provisioning of version labels as required by version control systems.

#### Related documentation

- *Basics on the Interaction with Version Control Systems* ( *Model Compare Guide*)

---

### New auto-highlight mode

Blocks and subsystems that are being selected in the hierarchy in Model Compare can be automatically shown and highlighted in MATLAB/Simulink at the very same time.

#### Related documentation

- *How to Browse Differences* ( *Model Compare Guide*)

---

### Customized layouting of HTML/PDF reports

For layout and language customization purposes, Model Compare lets you specify your own stylesheet for PDF and HTML report generation.

#### Related documentation

- *How to Generate Difference Reports* ( *Model Compare Guide*)

## Migration to Model Compare 2.7

---

### No adaptation necessary

You can migrate from Model Compare 2.6 to Model Compare 2.7 without any adaptations.

---

# ModelDesk

---

## Where to go from here

Information in this section

<i>New Features of ModelDesk 4.4</i>	165
<i>Migration to ModelDesk 4.4</i>	166

---

## New Features of ModelDesk 4.4

---

### Maneuver control

You can start, stop, and reset driving maneuver and driving cycles directly from ModelDesk.

---

### Road Generator

**OpenDRIVE export** The Road Generator can export and import roads in the OpenDRIVE format 1.4H.

**Table Editor** The Road Generator has a new Table Editor for specifying height points, lateral slope points and custom height and friction maps.

---

### Plotting

**New plotter instruments** ModelDesk has new plotters: Time Plotter and XY Plotter

**Signal Selector** The updated Signal Selector provides features such as a signal search or multiselection like in other tree lists of ModelDesk.

**Plotting and downloading** Now you can plot and download at the same time.

---

### Processing

**Raw data file** Information about the raw data file is available in Matlab during processing execution.

**Messages during processing execution** During processing execution messages of different types such as info, error, or warning can occur. The messages are structured according to their types.

**Searching parameter pages** You can search for the names of parameter pages on the Parameter Configuration pane.

---

### Parameter management

**Table Editor** ModelDesk has a new Table Editor to specify multi-dimensional parameters.

## Migration to ModelDesk 4.4

---

### Tool automation for plotting

As ModelDesk has new plotters, the tool automation for plotting has been changed. To reuse scripts for plotting, you must adapt these scripts.

---

# Model Interface Package for Simulink

---

## Where to go from here

Information in this section

<i>New Features of the Model Interface Package for Simulink 3.3</i>	167
<i>Migration Aspects of the Model Interface Package for Simulink</i>	168

---

## New Features of the Model Interface Package for Simulink 3.3

---

### Simplified generation of communication interfaces for behavior models

The `dsmpb_createreversedmodelportblock()` API command has been enhanced. You can now specify the destination path of the created inverse model port blocks. Refer to [`dsmpb\_createreversedmodelportblock`](#) ([📄 Model Interface Package for Simulink Reference](#)).

---

### Improved support of generated S-functions with SIC files

Previously, you had to use the `dsrt_addnonbuildfiles()` API command to add the source code directory of a generated S-function to an SIC file. As of dSPACE Release 2016-B, the Model Interface Package for Simulink tries to detect whether your model contains generated S-functions. If an S-function is used and its name ends with “\_sf” (which is the default name postfix when generating S-functions), the

Model Interface Package for Simulink automatically adds the corresponding source code directory to the SIC file.

### Limitations when using MATLAB R2016b

Note the following limitations when you use Model Interface Package for Simulink with MATLAB R2016b:

- Code generation for recursive functions  
The code generation of recursive functions is now supported by the Simulink Coder, but it should not be used for generating real-time applications executed on real-time hardware.
- Using the Upgrade Advisor API  
The `upgradeadvisor` function lets you analyze and perform required migration steps for your model when you upgrade to a newer MATLAB version. The third-party blocksets that you use, such as RTI blocksets, should not be upgraded by this function. Do not change the default value of the `SkipBlocksets` argument from `true` to `false`.
- Standard math library changed  
The default value for the `Standard math library` setting and the `TargetLangStandard` parameter is now `C99 (ISO)`. The `C99` standard is not supported by all compilers. The setting is therefore automatically switched to `C89/90 (ANSI)` when a dSPACE platform is selected.
- New Simulink blocks  
The following new Simulink blocks are not supported:
  - Initialize Function block
  - Terminate Function block

## Migration Aspects of the Model Interface Package for Simulink

### Modified features in MATLAB R2016b

**Changed startup behavior** MATLAB has changed the execution order of startup scripts.

- Up to MATLAB R2016a, the `startup.m` script is executed first, before the dSPACE-specific initialization scripts `dsstartup.m` and `dspoststartup.m`.
- As of MATLAB R2016b, the `startup.m` script is executed last, after the dSPACE-specific initialization scripts.



This modification might result in a changed startup behavior.

**Switching to a newer MATLAB version** If you install a new MATLAB version, some settings are adopted from already installed MATLAB versions. To prevent unexpected behavior of your Simulink models when switching to a newer MATLAB version or dSPACE Release, always reset the MATLAB and Simulink preferences to their defaults before you start working.



# MotionDesk

---

**Where to go from here**

Information in this section

<i>New Features of MotionDesk 3.9</i>	171
<i>Migrating to MotionDesk 3.9</i>	172

## New Features of MotionDesk 3.9

---

**Real-time lens distortion**

MotionDesk can simulate the distortion of a real lens. If you use MotionDesk for testing a camera-based recognition algorithm, you normally place a camera including a lens in front of a MotionDesk display. MotionDesk can now simulate the distortion of lenses, so it is not necessary to use the real lens in the ECU test. MotionDesk supports several effects of distortion, such as, vignetting, barrel distortion, pincushion distortion, and chromatic aberration.

---

**Road generation**

You can specify the accuracy of the generated road geometry. It is possible to specify the accuracy based on the road curvature or based on the segment types used for the road.

---

**Selectable vivid texturing**

When you use vivid texturing, the scenery gets more detailed textures to make the scene look more realistic. You can now select which elements of the scene are improved. It is possible to visualize flowers on the ground, worn out areas, random cracks, or random patterns on road surfaces.

---

**Scene Editor**

Scene editing mode is automatically activated when you drag a 3-D object from the Library Browser to the scene. This feature can be disabled on the Visualization page of the MotionDesk Properties dialog.

The dSPACE object library now has 3-D object with compressed textures. So it is not necessary to compress the 3-D object during run time.

---

**MotionDesk blockset**

The MD\_Object block can calculate the transformation matrix using two different rotation orders (conventions) when the Cardan angle interpretation is used:

- Z-Y'-X'' convention: The object is rotated around its z-axis, y-axis, and then the actual x-axis.
- Z-X'-Y'' convention: The object is rotated around its z-axis, x-axis, and then the actual y-axis.

---

## Migrating to MotionDesk 3.9

---

**MicroAutoBox support discontinued**

dSPACE Release 2015-B was the last release supporting MicroAutoBox with its variants 1401/1501, 1401/1504, 1401/1505/1506, 1401/1505/1507, and 1401/1507.

As of dSPACE Release 2016-A, dSPACE software supports only MicroAutoBox II with its variants 1401/1501, 1401/1504, 1401/1505/1507, 1401/1507, 1401/1511, 1401/1513, 1401/1511/1514, and 1401/1513/1514.

---

**Migrating from MotionDesk 2.2.1 and earlier**

In MotionDesk 2.2.1 and earlier, MotionDesk uses 3-D objects in VRML format. To use the scenes and custom 3-D objects included in these MotionDesk versions, they must be migrated so that they can be used in MotionDesk 3.9. For more information, refer to *Migrating from MotionDesk 2.2.1 and Lower* ([📖 MotionDesk Guide](#)).

---

**Migrating from MotionDesk experiments in MDX file format**

MotionDesk 3.9 cannot read old MotionDesk experiments in the MDX file format any longer. It is therefore not possible to migrate from a MotionDesk experiment with a version earlier than 2.2.

If you want to migrate such old experiments, you can migrate using MotionDesk 3.0 up to MotionDesk 3.6.

---

# Real-Time Testing

---

## Where to go from here

Information in this section

<i>New Features of Real-Time Testing 3.1</i>	173
<i>Migrating to Real-Time Testing 3.1</i>	174

---

## New Features of Real-Time Testing 3.1

---

### New Python version

The Python interpreter running on the simulation platform is based on Python 2.7.11.

---

### Host calls

Now the `rttlib.hostcall` module supports all the platform that are supported by Real-Time Testing. This includes especially SCALEXIO systems, DS1007, MicroLabBox, and VEOS.

---

### Datastreaming

The `rttlib.datastreaming` module has new modes for replay. Depending on the mode, different values are replayed when the time stamp of the MAT file time vector does not fit the model step.

- `datastream.RM_LINEAR`: The value used for replay is the linear interpolation between the MAT file time vector values before and after the model step.
- `datastream.RM_BACKWARD`: The value used for replay is the MAT file time vector value which time stamp is directly before the model step.

## Migrating to Real-Time Testing 3.1

---

### Incompatible BCG files

The BCG files that are generated with Real-Time Testing 2.6 and lower cannot be used for Real-Time Testing 3.1. You must create the BCG file of the RTT sequence again. Refer to *Creating and Starting RTT Sequences in Python Scripts* ([📖 Real-Time Testing Guide](#)).

---

# RTI/RTI-MP and RTLib

---

## Where to go from here

Information in this section

<i>New Features of RTI/RTI-MP and RTLib</i>	175
<i>Migration Aspects of RTI/RTI-MP and RTLib</i>	177

---

## New Features of RTI/RTI-MP and RTLib

---

### MicroAutoBox

**Improved safety functions** The RTI Watchdog Blockset that is supported by MicroAutoBox now provides the Challenge-Response Monitoring sublibrary.

For more information, refer to *Features of RTI Watchdog Blockset 2.0* on page 189.

---

### Enhanced RTI license package

The license for the RTI base product now includes the licenses for:

- RTI USB Flight Recorder Blockset
- RTI DS1552 I/O Extension Blockset

---

### New features of MATLAB R2016b

The following new feature introduced with MATLAB R2016b is supported by RTI/RTI-MP:

- New settings in the Hardware Implementation dialog  
The dialog provides the `size_t` and `ptrdiff_t` settings, which should be both set to 32 bits for all RTI platforms.

**Limitations when using  
MATLAB R2016b**

Note the following limitations when you use RTI/RTI-MP with MATLAB R2016b:

**■ Using the Upgrade Advisor API**

The `upgradeadvisor` function lets you analyze and perform required migration steps for your model when you upgrade to a newer MATLAB version. The third-party blocksets that you use, such as RTI blocksets, should not be upgraded by this function. Do not change the default value of the `SkipBlocksets` argument from `true` to `false`.

**■ Using the Property Inspector**

The Property Inspector lets you edit parameters and properties of a model element. This supports only blocks whose parameters are configured by using a standard Simulink mask interface. It does not support most of the RTI blocks that provide own dialogs.

**■ Code generation for recursive functions**

The code generation of recursive functions is now supported by the Simulink Coder, but it should not be used for generating applications executed on real-time hardware.

**■ Standard math library changed**

The default value for the `Standard math library` setting and the `TargetLangStandard` parameter is now `C99 (ISO)`. The `C99` standard is not supported by all compilers. The setting is therefore automatically switched to `C89/90 (ANSI)` when a dSPACE platform is selected.

**■ New Simulink blocks**

The following new Simulink blocks are not supported:

- Initialize Function block
- Terminate Function block

**New features in RTI/RTIMP**

The following features in RTI/RTI-MP are introduced with dSPACE Release 2016-B:

**■ Programmatic task configuration**

The RTI and RTI-MP task configuration can be accessed via a programmatic API. For more information, refer to *RTI and RTI-MP Task Configuration API Reference* ([📖 RTI and RTI-MP Implementation Reference](#)).



- As of MATLAB R2016a, you can create a new model by selecting a Simulink template on the Simulink Start page. As of dSPACE Release 2016-B, you find RTI template files in the My Templates category in the Simulink Start page.

When you use MATLAB R2016b, templates can also be used for models created by `new_system`. By switching the RTI platform support, the platform-related template with its preconfigured settings is set as default.

You have to note the following limitations:

- Using RTI blocks in custom model templates is not supported.
- Using model templates from previous dSPACE Releases is not supported.
- New `rti_build2` command that replaces the `rti_build` command to reduce the risk of inconsistencies in the build process.

For further information, refer to *rti\_build2* ([RTI and RTI-MP Implementation Reference](#)) and *rtimp\_build2* ([RTI and RTI-MP Implementation Reference](#)).

## Migration Aspects of RTI/RTI-MP and RTLib

### Changes in TRC file generation

There are some modifications to the TRC file generation in RTI and RTI-MP. Refer to *Changes to TRC File Generation* on page 37.

### Modified features in MATLAB R2016b

**Changed startup behavior** MATLAB has changed the execution order of startup scripts.

- Up to MATLAB R2016a, the `startup.m` script is executed first, before the dSPACE-specific initialization scripts `dsstartup.m` and `dspoststartup.m`.
- As of MATLAB R2016b, the `startup.m` script is executed last, after the dSPACE-specific initialization scripts.

This modification might result in a changed startup behavior.

**Changed input parameter when building** As of MATLAB R2016b, the RTI commands `rti_build` and `rtimp_build` no longer support the `MakeOptions` input parameter. Build processes started with the `MakeOptions` parameter will abort with an error message.

**Switching to a newer MATLAB version** If you install a new MATLAB version, some settings are adopted from already installed

MATLAB versions. To prevent unexpected behavior of your Simulink models when switching to a newer MATLAB version or dSPACE Release, always reset the MATLAB and Simulink preferences to their defaults before you start working.

# RTI Bypass Blockset

---

## Where to go from here

Information in this section

<i>New Features of the RTI Bypass Blockset 3.7</i>	179
<i>Migrating to RTI Bypass Blockset 3.7</i>	180

---

## New Features of the RTI Bypass Blockset 3.7

---

### RTI Bypass Blockset

**On-target bypassing: TargetLink support** Besides Simulink® Coder™, the RTI Bypass Blockset now also supports the TargetLink Code Generator for model code generation when performing on-target (internal) bypassing. When you work with a Simulink model containing on-target bypass parts, you can select which code generator is to be used.

Moreover, the RTI Bypass Blockset supports TargetLink as modeling tool. This means that you can use blocks of the RTI Bypass Blockset in TargetLink models to easily integrate your TargetLink model into your existing ECU application.

Refer to [RTI Bypass Blockset Reference](#).

---

### RTI Bypass Blockset MATLAB API

**Support of enhancements to RTI Bypass Blockset** The RTI Bypass Blockset MATLAB API supports the enhancements to the RTI Bypass Blockset.

Refer to the [RTI Bypass Blockset MATLAB API Reference](#).

## Migrating to RTI Bypass Blockset 3.7

### Working with models from earlier RTI Bypass Blockset versions 3.x and 2.x

The current release contains RTI Bypass Blockset 3.7, which is compatible with earlier blockset versions 3.x and 2.x. However, there are some points to note:

#### ■ *Working with models from RTI Bypass Blockset 2.5 or earlier*

Data management was changed in comparison to the prior RTI Bypass Blockset versions. If you have a Simulink model built with RTI Bypass Blockset 2.5 or earlier, and you open it with RTI Bypass Blockset 3.7, the old data dictionary file (with file name extension `.dd`) is replaced by a new data dictionary file (`.vdb`) using the information stored in the Setup block. This happens as soon as you open and close the Setup block dialog by clicking OK, or you open the Read, Write, Upload or Download block dialog and click Fill Variable Selector on the Variables page.

If you have a model that was saved with RTI Bypass Blockset 3.7 and want to use it with RTI Bypass Blockset 2.5 or earlier, the model's data dictionary file required for blockset version 2.5 or earlier (file name extension `.dd`) is created. This happens as soon as you update the A2L files in the Setup block, or you open the Read, Write, Upload or Download block and click Fill Variable Selector on the Variables page. The data dictionary file created under RTI Bypass Blockset 3.7 (`*.vdb`) remains on the disk.

To enable the RTI Bypass Blockset to recreate the data dictionary, the database files specified in the Setup block must be accessible at the specified location and must be unchanged.

#### ■ *Working with models from RTI Bypass Blockset 2.6 up to and including RTI Bypass Blockset 3.6*

If you have a Simulink model built with RTI Bypass Blockset 2.6 up to RTI Bypass Blockset 3.6, and you open it with RTI Bypass Blockset 3.7, the old data dictionary file is replaced by a new data dictionary file. However, the new data dictionary file cannot be used in earlier RTI Bypass Blockset versions. If you want to reuse the model with RTI Bypass Blockset 2.6 up to RTI Bypass Blockset 3.6, you have to create a suitable database in the earlier RTI Bypass Blockset version by reimporting the database files (A2L files) specified in the Setup block.

---

# RTI CAN MultiMessage Blockset

---

## Where to go from here

Information in this section

<i>New Features of the RTI CAN MultiMessage Blockset 4.4</i>	181
<i>Migrating to RTI CAN MultiMessage Blockset 4.4</i>	181

## New Features of the RTI CAN MultiMessage Blockset 4.4

---

### Support of FIBEX 4.1.1

The RTI CAN MultiMessage Blockset now also supports FIBEX 4.1.1 files as database files.

Refer to *General Settings Page (RTICANMM MainBlock)* ([RTI CAN MultiMessage Blockset Reference](#)).

## Migrating to RTI CAN MultiMessage Blockset 4.4

---

### Working with models from earlier RTI CAN MultiMessage Blockset versions

To reuse a model created with an earlier RTI CAN MultiMessage Blockset version, you must update the S-functions for all the RTICANMM blocks and save the model before modifying the CAN configuration.

To create new S-functions for all the RTICANMM blocks in your model in one step, you can perform one of the following actions after opening the model:

- In the MATLAB Command Window, enter `rtimmsu_update('System', gcs)`.  
For more information on the command and its options, enter `help rtimmsu_update` in the MATLAB Command Window.
- Select the **Create S-Function for all CAN Blocks** command from the Options menu of the RTICANMM GeneralSetup block.

For more information, refer to *Limitations with RTICANMM* ([RTI CAN MultiMessage Blockset Reference](#)).

---

### Compiler messages when using code generated by an RTI CAN MultiMessage Blockset version < 4.0

If you use code that was generated by an RTI CAN MultiMessage Blockset version < 4.0, several compiler warning messages containing the phrase `<<argument of type "can_tpl_canChannel *" is incompatible with parameter of type "DsTCanCh">>` will appear during the build process of your simulation model. This is due to a modified data type. These warnings can be ignored and will disappear after you regenerate the RTICANMM code by using the current blockset version.

---

### Using existing checksum algorithms

Checksum algorithms originally developed for an application containing CAN messages cannot be reused for applications containing CAN FD messages, because CAN FD includes new message types and longer data fields. Existing checksum algorithms can still be used for applications that just contain classic CAN messages. For CAN FD applications, you must adapt the checksum algorithms.

# RTI FPGA Programming Blockset

---

## Where to go from here

Information in this section

<i>New Features of the RTI FPGA Programming Blockset 3.2</i>	183
<i>Migrating to RTI FPGA Programming Blockset 3.2</i>	184

---

## New Features of the RTI FPGA Programming Blockset 3.2

---

### Extended Xilinx® support

The RTI FPGA Programming Blockset now supports the following products and versions of the Xilinx design tools.

<b>Xilinx Design Tools Version</b>	<b>Operating System</b>	<b>MATLAB Version <sup>1)</sup></b>
Vivado 2016.2 (64-bit version)	Windows 7 Business, Ultimate, and Enterprise SP1 (64-bit version)	64-bit versions of: ■ MATLAB R2015a SP1 ■ MATLAB R2015b

<sup>1)</sup> The Processor Interface sublibrary of the RTI FPGA Programming Blockset also supports MATLAB R2016a and MATLAB R2016b.

---

### Enhanced timing analysis

In addition to the timing analysis of the FPGA model, a timing analysis of the entire FPGA application is available. The entire FPGA application includes the FPGA model and the framework for the used platform. The build process automatically starts the timing analysis of the entire FPGA application and creates a link to the results in the MATLAB command window.

---

### Enhancements to the frameworks of the DS2655 FPGA Base Board

The frameworks for the DS2655 FPGA Base Board and for the I/O modules provide the following enhancement.

#### **Configuring the electrical interface with ConfigurationDesk**

You can change the electrical characteristics of the FPGA I/O interface in ConfigurationDesk. This lets you reuse the same FPGA custom function block in several ConfigurationDesk projects and reduces the number of variants.

**Tracing FPGA signals** You can specify that all output signals of Simulink Bus Creator and Bus Selector blocks can be traced directly with your experiment software. The FPGA\_SETUP\_BL block lets you make FPGA signals traceable.

To reduce the number of traceable signals, you can exclude signals of a subsystem from tracing by using the subsystem omission tag (DsVdOmit). You enter the DsVdOmit tag in the Block Properties dialog of the subsystem.

---

### Enhancements to the framework of MicroLabBox

**Multiple clock periods** You can use up to 10 individual clock periods for modeling specific parts of your FPGA design. The clock periods are in the range 1.6e-7 s ... 1.25e-9 s (6.25 MHz ... 800 MHz).

The FPGA\_SETUP\_BL block provides the Subsystems Clocks page to configure the clock period for subsystems with an individual clock period.

**Buzzer** You can use the buzzer of MicroLabBox to generate acoustic signals.

---

### Related topics

Basics

- [Migrating to RTI FPGA Programming Blockset 3.2 on page 184](#)

## Migrating to RTI FPGA Programming Blockset 3.2

---

### Objective

There are different ways to migrate an existing model, depending on the blockset version used.

---

### Migrating from RTI FPGA Programming Blockset 1.1 and higher to 3.2

If you have implemented your FPGA application using RTI FPGA Programming Blockset Version 1.1 and higher, and want to use it with RTI FPGA Programming Blockset 3.2, the framework will automatically update itself to the current framework version.



The update handles all the subsystems in the model/subsystem. The parameters of the blocks stay the same after updating to the current framework version.

---

### ConfigurationDesk custom functions incompatible with dSPACE Release 2016-B

#### Note

Relevant for SCALEXIO systems with a DS2655 FPGA Base Board and a DS2655M1 Multi-I/O Module

An FPGA custom function block generated by using RTI FPGA Programming Blockset 2.5 from dSPACE Release 2013-A and the real-time applications (\*.rta) containing the FPGA custom function block are incompatible with dSPACE Release 2016-B. To produce a usable custom function, you have to rebuild the FPGA model by using RTI FPGA Blockset 3.2 from dSPACE Release 2016-B.

---

### Using different dSPACE hardware

Using an FPGA model on different dSPACE hardware requires some model modifications. Refer to *Migrating to Another dSPACE Hardware* ([📖 RTI FPGA Programming Blockset Guide](#)).



---

# RTI LIN MultiMessage Blockset

---

## Where to go from here

Information in this section

<i>New Features of the RTI LIN MultiMessage Blockset 2.7</i>	187
<i>Migrating to RTI LIN MultiMessage Blockset 2.7</i>	187

---

## New Features of the RTI LIN MultiMessage Blockset 2.7

---

### Support of FIBEX 4.1.1

The RTI LIN MultiMessage Blockset now also supports FIBEX 4.1.1 files as database files.

Refer to *General Settings Page (RTILINMM MainSetup)* ([📖 RTI LIN MultiMessage Blockset Reference](#)).

---

## Migrating to RTI LIN MultiMessage Blockset 2.7

---

### Working with models from earlier RTI LIN MultiMessage Blockset versions

To reuse a model created with an earlier RTI LIN MultiMessage Blockset version, you must update the S-functions for all the RTILINMM blocks and save the model before modifying the LIN configuration.

To create new S-functions for all the RTILINMM blocks in your model in one step, you can perform one of the following actions after opening the model:

- In the MATLAB Command Window, enter `rtimmsu_update('System', gcs)`.  
For more information on the command and its options, enter `help rtimmsu_update` in the MATLAB Command Window.
- Select the **Create S-Function for all LIN Blocks** command from the Options menu of the RTILINMM GeneralSetup block.

For more information, refer to *Limitations of RTI LIN MultiMessage Blockset* ([RTI LIN MultiMessage Blockset Reference](#)).

# RTI Watchdog Blockset

## Features of RTI Watchdog Blockset 2.0

---

### Challenge-response monitoring

The RTI Watchdog Blockset has been enhanced with the Challenge-Response Monitoring sublibrary supporting the challenge-response method used to monitor arbitrary software entities, such as periodic tasks or subsystems.

With the new Challenge-Response Monitoring blocks you can implement higher-level safety functions as with the current Watchdog Handling blocks, because of the following two features:

- A failure is not only detected if a response is not available within the specified timeout limit, but also if the returned response provides an invalid value.
- The challenge-response monitoring units are executed on a hardware component, that is not used by the real-time application. So the execution of the entity to be monitored is separated from the monitoring itself.

The watchdog blocks are now moved to the Watchdog Handling sublibrary.

For more information, refer to [RTI Watchdog Blockset Reference](#).



---

# SCALEXIO Firmware

## New Features of the SCALEXIO Firmware 3.5

---

### **New licenses**

The SCALEXIO firmware supports new licenses

- The new SCALEXIO\_RTLIB\_MC8 license allows the download of more than 4 application to the processing unit.
- The new SCALEXIO\_FIU\_500 license allows the simulation of failures with a real-time application implemented with up to 500 function blocks in ConfigurationDesk.

---

### **Support of updated QNX compiler**

The SCALEXIO firmware supports QNX compiler version 5.2.0 for 64-bit Windows. This lets you compile very large Simulink models without Out-of-Memory errors.





---

# SystemDesk

---

**Where to go from here**

Information in this section

<i>New Features of SystemDesk 4.7</i>	194
<i>Migrating to SystemDesk 4.7</i>	200

# New Features of SystemDesk 4.7

## Where to go from here

Information in this section

<i>New General Features</i>	194
<i>Configuring ECUs</i>	194
<i>Creating Simulation Systems for Virtual Validation</i>	198

## New General Features

### Objective

SystemDesk 4.7 has the following new general features.

### Software architecture modeling and RTE code generation for compu method 'BITFIELD TEXTTABLE'

SystemDesk now provides improved support for compu methods of the category BITFIELD TEXTTABLE that allow for the textual representation of internal values ranges.

You can specify compu methods of this category via the property dialogs and generate RTE code for ECU configurations with mapped software components that contain these compu methods.

### Improved validation

With this version, SystemDesk provides additional validation rules for the following:

- Improved validation of AUTOSAR constraints to help you model software architectures and configure ECUs.
- Improved validation of TargetLink compatibility to help you exchange software components between TargetLink and SystemDesk, and implement them.

## Configuring ECUs

### Objective

SystemDesk now provides an *ECU configuration framework* that you can use to configure ECUs with RTE and basic software (BSW) modules of any vendor.

## The ECU configuration framework

The ECU configuration framework has the following components:

- Plug-in command interfaces for BSW module configurations to perform tasks such as deriving parameters of BSW module configurations from the system description, generating BSW code, etc.

SystemDesk now supports AUTOSAR build action manifests to define the execution of the plug-in commands. Each module configuration references the related build action manifests and provides the defined plug-in commands in its context menu. Refer to *Plug-in commands* on page 196.

- A build tool that lets you execute a group of plug-in commands to auto configure and generate code for the modules of an ECU configuration.

SystemDesk now analyzes dependencies between plug-in commands via the related build action manifests that have to provide the relevant information.

- Support to develop plug-in commands for any BSW module configuration with IronPython scripts.

You can now generate IronPython scripts with the related build action manifests. Refer to *Developing BSW scripts* on page 196.

- A script-based mechanism to convert module configurations between ECU configuration parameter definitions without loss of data.

You can now generate IronPython scripts for this. Refer to *Converting BSW module configurations* on page 197.

- The Runnable Mapping editor and the Generate Mappings command to schedule the execution of runnable entities. You can now use this with any OS and RTE modules via an abstraction. Refer to *Runnable mapping abstraction* on page 197.

- The BSW Module editor to configure parameters of a BSW module configuration.

- SystemDesk's automation to access all the elements of an ECU's AUTOSAR model and BSW module configuration parameters of any vendor.

**Using IronPython** The ECU configuration framework executes BSW scripts for plug-in commands and BSW module conversion in an IronPython interpreter.

When you use Python Tools for Visual Studio and an IronPython development package, you can develop BSW scripts and use Visual Studio's debugging features.

---

**Plug-in commands**

To execute plug-in commands for BSW module configurations and perform tasks such as deriving parameters of BSW module configurations from the system description, generating BSW code, etc.

SystemDesk now supports AUTOSAR build action manifests to define the execution of the plug-in commands. Each module configuration references the related build action manifests and provides the defined plug-in commands in its context menu.

You can implement plug-in commands in different environments such as Microsoft .NET libraries, IronPython scripts, or executables. SystemDesk provides support for developing plug-in commands with IronPython.

Executing plug-in commands is most useful for the following tasks:

- Deriving BSW module configuration parameters from the system description. AUTOSAR has described upstream mapping rules for this.
- Generating software component descriptions for BSW modules. This is required to connect application software with basic software and generate RTE code.
- Generating code for BSW modules according to the BSW module configuration parameters.

Refer to *Basics on Build Action Manifests* ( *SystemDesk Guide*).

---

**Developing BSW scripts**

To configure an ECU, you have to analyze the underlying ECU extract and configure the BSW parameters accordingly. You can generate BSW code and BSW components based on the BSW parameters.

SystemDesk now provides the following support for developing IronPython scripts for these tasks:

- Generate template BSW module scripts for configuring ECUs.  
You can generate template scripts with stub functions for configuring all the containers and parameters of a BSW module configuration.
- Adding build action manifests for BSW scripts to the related BSW modules.  
You can run the BSW scripts via the context menu of related BSW modules.

- Accessing AUTOSAR elements of the ECU source system and BSW module configurations in BSW scripts.

You can get related information via an automation interface that SystemDesk passes to BSW scripts as a parameter.

Refer to *Basics on Developing BSW Module Scripts* ([📖 SystemDesk Guide](#)).

---

### Converting BSW module configurations

SystemDesk now lets you convert BSW module configurations on the basis of Python scripts. You can develop conversion scripts for any RTE or BSW module. So you can import/export BSW module configurations without loss of data.

SystemDesk provides the following support for developing BSW module conversion scripts:

- Generate conversion scripts (both directions) with stub functions for BSW module configuration parameters.
- Keep data that cannot be converted with a conversion script in SystemDesk. It can be restored with a reverse conversion.

SystemDesk also provides a simple conversion via the name matching of BSW module configuration containers and parameters.

Refer to *Basics on Converting Basic Software Module Configurations* ([📖 SystemDesk Guide](#)).

---

### Runnable mapping abstraction

Lets you use the Runnable Mapping editor and the Generate Mappings command with OS and RTE modules of any vendor.

The ECU configuration framework provides an IronPython script as a runnable mapping abstraction that lets you work with most AUTOSAR 4-compliant RTE and OS modules. However, if errors occur when you work with the Runnable Mapping editor, you might have to adapt the runnable mapping abstraction. Contact dSPACE Support in these cases to adapt the script for use with vendor-specific RTE and OS modules.

**Use case for the runnable mapping abstraction** The runnable mapping abstraction lets you perform the runnable mapping of application software components in the system configuration phase to exchange the mapping with a partner that performs the ECU configuration phase.

**Mapping runnables to OS tasks in the system configuration phase** Typically, mapping runnables to OS tasks is performed in the ECU configuration phase. However, you might want to map runnables of application software components in the system configuration phase when you map application software components

to ECUs. This can be useful if ECU configuration and system configuration are performed by different partners, e.g., in a supplier and OEM collaboration.

**Exchanging runnable mappings** You can exchange the required information between system and ECU configuration via a system or ECU extract. However, runnable mappings are not included for this, they are part of the RTE and OS configuration. To exchange this information in a supplier and OEM collaboration that use BSW configuration and software architecture tools, you have to also exchange the RTE and OS configurations.

When you use SystemDesk in the system configuration phase, the runnable mapping abstraction allows you to perform the runnable to OS task mapping with the vendor-specific RTE and OS modules. The other partner in the supplier and OEM collaboration can use the RTE and OS configurations with the runnable mapping as initial configurations or merge them with existing ones. You do not have to perform additional conversion steps to do so.

Refer to *Basics on the Runnable Mapping Abstraction* ( *SystemDesk Guide*).

## Creating Simulation Systems for Virtual Validation

### SWC template file generation

You can configure the RTE generation via the `RteGeneration/RteGenerateSwcTemplateFiles` module configuration parameter. You can use this option to generate template files for software components.

You can use the generated template files for the following:

- To test the build of simulation systems before all the software components are implemented. This helps you prepare virtual validation and set up your validation process.

For this you should copy the template file to your work folder and reference the file to the respective SWC implementation.

- To start implementing hand-written software components. The template provides the runnable signatures and related RTE API functions for accessing data.

For this, you should copy the template file to your working folder and add code for the software component's behavior.

The following illustration shows part of an example template file.

```

/*****\
*** FUNCTION:
***   Linearization
***
*** DESCRIPTION:
***
*** PARAMETERS:
***   Type           Name           Description
***   -----
***
*** RETURNS:
***   void
***
*** SETTINGS:
***
/*****\
FUNC(void, RTE_CODE) Linearization(void)
{
{
  VAR(sint16, AUTOMATIC) PosSensor = Rte_InitValue_PosSensor_PosSensor;

  /* Performs an implicit read on a sender-receiver communication data item with data semantics.
  */
  PosSensor = Rte_IRead_Linearization_PosSensor_PosSensor();
}
{
  VAR(sint16, AUTOMATIC) LinPos = 0;

  /* Performs an implicit write on an interruptible variable. */
  Rte_IrVWrite_Linearization_LinPos(LinPos);
}
}

```

### Import V-ECU implementations

You can now import V-ECU implementations to simulation systems. During the import SystemDesk creates a code-based V-ECU and imports/references the V-ECU implementation files (C/H, ARXML, A2L).

This lets you perform the following:

- Adding V-ECUs to a simulation system without having the related AUTOSAR model in the same SystemDesk project.
- Easily exchanging V-ECU implementations to build and debug them with SystemDesk.

# Migrating to SystemDesk 4.7

## Migrating to SystemDesk 4.7

---

SystemDesk 4.7 automatically migrates SystemDesk 4.5, and 4.6 SDP project files during the loading process.

### **Note**

You are recommended to install the most recent patch for SystemDesk 4.5 or 4.6. Then, save the SDP project files you want to migrate before opening them in SystemDesk 4.7.



---

# TargetLink

---

**Where to go from here**

## Information in this section

<i>New Features of TargetLink 4.2 and TargetLink Data Dictionary 4.2</i>	202
<i>Migrating to TargetLink 4.2 and TargetLink Data Dictionary 4.2</i>	227

# New Features of TargetLink 4.2 and TargetLink Data Dictionary 4.2

---

**Where to go from here**

Information in this section

<i>Modeling in Simulink or Stateflow</i>	202
<i>Code Generation Core Functionality</i>	205
<i>AUTOSAR</i>	212
<i>Target Simulation (PIL)</i>	215
<i>Data Dictionary and Data Management</i>	217
<i>Code Generator Options</i>	218
<i>Tool Chain Integration</i>	219
<i>Other</i>	221
<i>API Functions and Hook Scripts</i>	225

## Modeling in Simulink or Stateflow

---

**Where to go from here**

Information in this section

<i>Newly Supported Simulink Blocks</i>	203
<i>Improved Data Store Blocks and Bus Support</i>	203
<i>Improvements to Stateflow Code Generation</i>	204
<i>Improvements to Model Referencing</i>	205

## Newly Supported Simulink Blocks



### Supported Simulink blocks (code-relevant)

TargetLink now additionally supports the following Simulink block, which affects TargetLink code generation:

- Vector Concatenate

This block is similar to the Matrix Concatenate block. These blocks differ only in their Mode setting, Vector versus Multidimensional array.

#### Related documentation

- *Code-Relevant Simulink Blocks* ( [TargetLink Block and Object Reference](#))
- *Basics on Reusing Variables of Preceding and Subsequent Blocks* ( [TargetLink Customization and Optimization Guide](#))

### Supported Simulink blocks (not code-relevant)

TargetLink now supports the following Simulink blocks, which do not affect TargetLink code generation:

- Data Type Duplicate
- Data Type Propagation
- Floating Scope
- XY Graph
- Stop Simulation
- Timed-Based Linearization
- Trigger-Based Linearization

#### Related documentation

- *Code-Irrelevant Simulink Blocks* ( [TargetLink Block and Object Reference](#))

## Improved Data Store Blocks and Bus Support

### Element selection and bus-capable Data Store

This TargetLink version supports bus signals in Data Store Memory, Data Store Read and Data Store Write blocks. You can map entire buses to predefined structure type definitions and to predefined structure variables. This is especially helpful if the bus consists of many bus elements, and/or the type or variable is used several times in a model or across models.

**Obsolete limitations** Some limitations concerning Data Store blocks have become obsolete. For more information, refer to *Obsolete Limitations* on page 256.

- The number of input and output ports (of Data Store Read and Data Store Write blocks) can now be more than one.
- Partial reading from and writing to data stores is now also supported by TargetLink.

#### Related documentation

- *Basics on Modeling Buses via Data Store Blocks* ([📖 TargetLink Customization and Optimization Guide](#))
- *Basics on the Representation of Buses in the Production Code* ([📖 TargetLink Customization and Optimization Guide](#))

#### Bus-related API commands

You can create Simulink.Bus objects from structured DD objects. Vice versa, you can create DD Variable objects from Simulink.Bus objects.

#### Related documentation

- *Basics on Modeling Buses via Data Store Blocks* ([📖 TargetLink Customization and Optimization Guide](#))
- *tlSimulinkBusObject* ([📖 TargetLink API Reference](#))

## Improvements to Stateflow Code Generation

#### Support for Superstep semantics

You can now work with Superstep semantics in state machines. This lets you perform all possible state transitions at once.

#### Related documentation

- *Basics on Working with Superstep Semantics* ([📖 TargetLink Preparation and Simulation Guide](#))

#### Support for functions with multiple output data

TargetLink now supports Stateflow functions (e.g., graphical functions) with multiple output data.

#### Related documentation

- *Working with Stateflow* ([📖 TargetLink Preparation and Simulation Guide](#))

#### Improved support for monitoring state activities

For the monitoring of state activity via Stateflow active state data there are three activity modes: Self Activity, Child Activity and Leaf State Activity.

TargetLink now supports Self Activity and Child Activity activity modes and generates optimized production code, including C code enums or C preprocessor macros for state encoding. However, Leaf State Activity is not supported.

**Related documentation**

- *Basics on Working with Active State Data* ([📖 TargetLink Preparation and Simulation Guide](#))

## Improvements to Model Referencing

**Enable blocks on model root level**

With this TargetLink version, you can control the execution of referenced models not only via Trigger block but also via Enable block.

**Related documentation**

- *Basics on Model Referencing* ([📖 TargetLink Customization and Optimization Guide](#))

## Code Generation Core Functionality

**Where to go from here**

Information in this section

<i>MISRA C Compliance</i>	206
<i>Improved Code Stability</i>	206
<i>Improved Code Efficiency</i>	207
<i>Improved Enumeration Support</i>	211
<i>More Flexibility When Using TargetLink Custom Code Blocks</i>	211

## MISRA C Compliance

### Improvements for MISRA C

Several improvements were made to TargetLink's Fixed-Point Library and to the Code Generator itself to improve the compliance with individual MISRA C rules. The improvements include the following:

- Functions have only one return path (MISRA C:2012 - 15.5).
- Variables are always created in the minimal useful scope (Fixed-Point Library).
- Less unreachable code (MISRA C:2012 - 2.1, 2.2, 14.3).
- Improved support for the conversion rules of MISRA C:2012 by adding casts for return values or comparisons.
- TargetLink now supports the C99 Bool type for the implementation of Bool. Refer to *General Enhancements and Changes* on page 221.

#### Related documentation

- None

## Improved Code Stability

### Changed calculation sequence in atomic subsystems

The production code calculation sequence in atomic subsystems can change in contexts where the block execution order was not completely determined by the control and data flow.

**Reason** To make the production code's block scheduling more resilient against model changes (like adding an Addfile or Unit Delay block).

**Migration issue** Depending on your specific situation and use cases, obtaining production code sorting as in previous TargetLink versions might be possible by applying a temporary workaround. Contact dSPACE product support for TargetLink.

## Improved Code Efficiency

### Omitting init section for Assignment blocks

The code for Assignment blocks changed in the following context:

- The block is used to model reduced rewrites to NVRAM in AUTOSAR NvData communication.
- The block does not have more than one successor block.
- The block is not placed within an iterated subsystem.

In this case, TargetLink no longer generates the output initialization, as shown in the following code examples:

#### TargetLink ≤ 4.1

```
p_Err = Rte_IRead_Run_PR_NV_Err_Err();
...
SCtrl2_Logical_Operator = SCtrl2_Relational_Operator &&
SCtrl2_Relational_Operator1;

for (Aux_ = 0; Aux_ < 10; Aux_++)
{
    /* Assignment: pidcontroller/Efficient_Write_To_NVRAM - output
    initialization */
    SCtrl1_Efficient_Write_To_NVRAM[Aux_] = p_Err[Aux_];
}
SCtrl1_Efficient_Write_To_NVRAM[0] = SCtrl2_Logical_Operator;

p_Err = Rte_IRead_Run_PR_NV_Err_Err();
if (p_Err[0] != SCtrl1_Efficient_Write_To_NVRAM[0]) {
    p_Err_a = Rte_IWriteRef_Run_PR_NV_Err_Err();
    p_Err_a[0] = SCtrl1_Efficient_Write_To_NVRAM[0];
}
```

#### TargetLink 4.2

```
p_Err = Rte_IRead_Run_PR_NV_Err_Err();
...
SCtrl2_Logical_Operator = SCtrl2_Relational_Operator && SCtrl2_Relational_Operator
1;
SCtrl1_Efficient_Write_To_NVRAM[0] = SCtrl2_Logical_Operator;
p_Err = Rte_IRead_Run_PR_NV_Err_Err();
if (p_Err[0] != SCtrl1_Efficient_Write_To_NVRAM[0]) {
    p_Err_a = Rte_IWriteRef_Run_PR_NV_Err_Err();
    p_Err_a[0] = SCtrl1_Efficient_Write_To_NVRAM[0];
}
```

**Benefit** TargetLink can now completely optimize `SCtrl1_Efficient_Write_To_NVRAM`, which results in increased efficiency

**Omitting superfluous casts**

TargetLink now removes casts from expressions like (Type) <Boolean operation, variable, macro> if they are used in the following contexts:

- In if-, while-, do ... while-, for-, and ?: conditions
- As operand of logical operations

This also holds for comparisons with 0 or 1 that occur in the contexts mentioned above.

In addition, TargetLink omits casts in switch conditions that cast constants and do not change the value. Usually, this allows for further optimizations such as replacing the switch statement by the code of one of the case branches or the default branch.

TargetLink ≤ 4.1	TargetLink 4.2
switch((Type) <constant>)	switch(<constant cast to T>)

**Definitions of variables with local scope**

TargetLink now makes sure that variables of local scope are never defined directly at the start of loop compound statements. They are now always consistently defined at the top of the function. Keep in mind that optimization might reduce function-local variables to minimal block scope in compound statements nested within loops, depending on the option

ReduceScopeOfVariablesOnlyDownToFunctionLevel.

TargetLink ≤ 4.1	TargetLink 4.2
<pre>void Subsystem(void) {     for (Aux_b = 0; Aux_b &lt; Width; Aux_b++)     {         UInt8 TlAuxVar;          /* ... */     } }</pre>	<pre>void Subsystem(void) {     UInt8 TlAuxVar;      for (Aux_b = 0; Aux_b &lt; Width; Aux_b++)     {         /* ... */     } }</pre>

All identifiers of function-local variables and macros now have function-wide visibility, i.e., are treated as if the respective variable or macro is defined at the beginning of the function. This primarily affects the identifiers of auxiliary variables in unoptimized code.

**TargetLink ≤ 4.1**

```
for (Aux_b = 0; Aux_b < BlockWidth; Aux_b++)
{
    /* SLLutLocal: Default storage class for local variables | Width: 8 */
    UInt8 TlAuxVar; /* Index saturation for Direct Look-Up Table block */
    TlAuxVar = C_U8SAT116_SATb(Sal_In[Aux_b], 9, 0);
}
```



```

        /* Subsystem/TableAsParam_BlockVWV */
        BlockVar4[Aux_b] = Sal_TableAsParam_BlockVWV_table[TlAuxVar];
    }
...
    for (Aux_d = 0; Aux_d < BlockWidth; Aux_d++)
    {
        /* SLlutLocal: Default storage class for local variables | Width: 8 */
        UInt8 TlAuxVar; /* Index saturation for Direct Look-Up Table block */
TlAuxVar = C_U8SATI16_SATb(Sal_In[Aux_d], 9, 0);
        /* Subsystem/TableAsInput_TableAndBlockVWV */
        BlockVar3[Aux_d] = TableVar1[TlAuxVar];
    }

```

### TargetLink 4.2

```

for (Aux_b = 0; Aux_b < BlockWidth; Aux_b++)
{
    /* SLlutLocal: Default storage class for local variables | Width: 8 */
    UInt8 TlAuxVar_a; /* Index saturation for Direct Look-Up Table block */
TlAuxVar_a = C_U8SATI16_SATb(Sal_In[Aux_b], 9, 0);
    /* Subsystem/TableAsParam_BlockVWV */
    BlockVar4[Aux_b] = Sal_TableAsParam_BlockVWV_table[TlAuxVar_a];
}
...
    for (Aux_d = 0; Aux_d < BlockWidth; Aux_d++)
    {
        /* SLlutLocal: Default storage class for local variables | Width: 8 */
        UInt8 TlAuxVar_b; /* Index saturation for Direct Look-Up Table block */
TlAuxVar_b = C_U8SATI16_SATb(Sal_In[Aux_d], 9, 0);
        /* Subsystem/TableAsInput_TableAndBlockVWV */
        BlockVar3[Aux_d] = TableVar1[TlAuxVar_b];
    }

```

**Reason** To establish consistent naming of auxiliary variables across code patterns, including different auxiliary variable identifiers for different blocks.

#### Optimization of logical expressions

TargetLink can now optimize logical expressions containing constants other than zero:

TargetLink ≤ 4.1	TargetLink 4.2
<b>Logical And</b>	
<code>1 &amp;&amp; boolexpr =&gt; boolexpr</code>	<code>&lt;non-zero const&gt; &amp;&amp; boolexpr =&gt; boolexpr</code>
<b>Logical Or</b>	
<code>1    boolexpr =&gt; optimized to: 1</code>	<code>&lt;non-zero const&gt;    boolexpr =&gt; optimized to: 1</code>

#### Floating-point modulo and remainder

The code generated for the Math block can change if one of its inputs or its output has a floating-point type:

**TargetLink  $\leq$  4.1** For a Math block with modulo function whose output is Float64 and inputs are Float32 and Int16:

```
if (I16DenomIn != 0.F) {
    F64_F32I16_ = F32NumIn -
    (((Float64) I16DenomIn) * ((Float64) (Int32) (F32NumIn / ((Float64) I16DenomIn)))));
}
...
```

For a Math block with modulo function whose output is Int16 and saturated and inputs are Float32:

```
if (F32DenomInInt != 0.F) {
    ...
} else {
    I16_F32F32_ = (Int16) F32NumIn;
}
```

**TargetLink 4.2** For a Math block with modulo function whose output is Float64 and inputs are Float32 and Int16:

```
if (I16DenomIn != 0) {
    F64_F32I16_ = ((Float64) F32NumIn) -
    (((Float64) I16DenomIn) * ((Float64) (Int32) (((Float64) F32NumIn) / ((Float64) I16DenomIn))));
}
```

For a Math block with modulo function whose output is Int16 and saturated and inputs are Float32:

```
if (F32DenomIn != 0.F) {
    ...
} else {
    if (F32NumIn > 32767.F) {
        I16_F32F32_ = 32767;
    }
    else {
        if (F32NumInInt < -32768.F) {
            I16_F32F32_ = -32768;
        }
        else {
            I16_F32F32_ = (Int16) F32NumIn;
        }
    }
}
```

**Details** The code can change because TargetLink now behaves as follows:

- Explicitly cast all operands to the type the operation is supposed to be calculated in.
- Calculate all intermediate operations using Float64 if the operation result or one of its operands is Float64.
- Saturate the code in the else branch of conditional control flows.

**Reason** Improves precision and suppresses implicit casts to comply with MISRA C.

## Improved Enumeration Support

### Enumerations in the generated code

TargetLink supports the use of Simulink enumerations in the model (including Stateflow) as well as their implementation in the generated code. Simulink enumerations are implemented either as *C enums*, as *macros*, or as integers. Hence, simulation (MIL and SIL/PIL) is possible. The following code example is from the ENUM\_TYPES demo model:

```
typedef enum eBlinkMode_tag {
    eBlinkMode_NONE = 0,
    eBlinkMode_BLINKER_ON = 1,
    eBlinkMode_BLINKER_OFF = 2
} eBlinkMode;

switch (Cal_ChartMode) {
    case CHARTMODE_DO_BLINK: {
        if (!(Sal_ENABLE)) {
            Cal_BlinkState = eBlinkMode_NONE;
            Cal_ChartMode = CHARTMODE_OFF;
        }
        else {
            if (Cal_BlinkState == eBlinkMode_BLINKER_ON) {
                Cal_BlinkState = eBlinkMode_BLINKER_OFF;
            }
            else {
                Cal_BlinkState = eBlinkMode_BLINKER_ON;
            }
        }
    }
}
```

### Related documentation

- [Applying Simulink Enumeration Data Types in TargetLink](#) ([📖 TargetLink Preparation and Simulation Guide](#))
- [ENUM\\_TYPES](#) ([📖 TargetLink Demo Models](#))

## More Flexibility When Using TargetLink Custom Code Blocks

### Substitution of strings in custom code template files

TargetLink custom code blocks now support the use of placeholders for arbitrary strings in the custom code template file for each block instance. The placeholders can be replaced during code generation by arbitrary, user-provided strings, which makes custom code template files a lot more flexible.

### Related documentation

- [Basics on Custom Code and Custom Code Files](#) ([📖 TargetLink Preparation and Simulation Guide](#))

- *Code and Logging Page (Custom Code Block)* ([📖 TargetLink Block and Object Reference](#))
- *VARIABLE\_INITIALIZATION* ([📖 TargetLink Demo Models](#))

### Removing braces from code sections

Braces that encapsulate a Custom Code section can be removed by using the new `nobraces` keyword at the beginning of this code section.

**Related documentation** *Basics on Custom Code and Custom Code Files* ([📖 TargetLink Preparation and Simulation Guide](#))

## AUTOSAR

### Where to go from here

Information in this section

<i>Supported AUTOSAR Releases</i>	212
<i>Asynchronous Client-Server Communication</i>	213
<i>Data Transformer in Client-Server Communication</i>	213
<i>Non-Scalar Interrunnable Variables</i>	214
<i>Improved Roundtrip with SystemDesk and other AUTOSAR Tools</i>	214
<i>Miscellaneous AUTOSAR Features</i>	214

## Supported AUTOSAR Releases

### Supported AUTOSAR Releases

The following AUTOSAR Releases are supported:

AUTOSAR Release	Revision
4.2	4.2.2 <sup>1)</sup>
	4.2.1
4.1	4.1.3
	4.1.2
	4.1.1
4.0	4.0.3
	4.0.2



AUTOSAR Release	Revision
3.2	3.2.3
	3.2.2
	3.2.1
3.1	3.1.5
	3.1.4
	3.1.2
3.0	3.1.0
	3.0.7
	3.0.6
	3.0.4
2.1	3.0.2
	2.1.4

<sup>1)</sup> New in TargetLink 4.2 (and also supported since TargetLink 4.1 patch 1)

## Asynchronous Client-Server Communication

You can now model asynchronous client-server communication to generate calls of the `Rte_Call` and `Rte_Result` API functions in your production code.

### Related documentation

- *Modeling Client-Server Communication* ( [TargetLink AUTOSAR Modeling Guide](#))
- *Changes in TargetLink and TargetLink Data Dictionary API Functions* on page 236
- *AR\_COLLISION\_DETECTION* ( [TargetLink Demo Models](#))

## Data Transformer in Client-Server Communication

You can now model transformer error logic for client-server communication.

**Related documentation**

- *Modeling Transformer Error Logic* ([📖 TargetLink AUTOSAR Modeling Guide](#))
- *AR\_COLLISION\_DETECTION* ([📖 TargetLink Demo Models](#))

## Non-Scalar Interrunnable Variables

---

You can now model non-scalar interrunnable variables.

**Related documentation**

- *Modeling Interrunnable Communication* ([📖 TargetLink AUTOSAR Modeling Guide](#))
- *Migrating Interrunnable Variable Specifications* on page 238

## Improved Roundtrip with SystemDesk and other AUTOSAR Tools

---

The roundtrip between TargetLink and SystemDesk was improved. You can now specify package information for AUTOSAR elements that are implicitly generated during export via DD `ExportRule` objects. The internal behavior of a software component can be exported into a separate file. Moreover, units can also be exported in separate files and packages.

**Related documentation**

- *Basics on Packages* ([📖 TargetLink Interoperation and Exchange Guide](#))
- *ExportRule* ([📖 TargetLink Data Dictionary Reference](#))

## Miscellaneous AUTOSAR Features

---

**Improved import and export**

TargetLink can now import and export `CompuMethod` elements whose category is set to `BITFIELD_TEXTTABLE`.

**Related documentation**

- None

**Data Store Memory blocks outside of runnable subsystems**

You can now use Data Store Memory blocks to model NvData communication outside of *runnable subsystems* ([TargetLink Glossary](#)).

**Related documentation**

- None

**Data store blocks for simulation**

You can now place Data Store Read and Data Store Write blocks outside of runnable subsystems to access Data Store Memory blocks that model NvData or interrunnable communication. This lets you directly access the communication variable in the simulation frame for simulation purposes.

## Target Simulation (PIL)

**Where to go from here**

Information in this section

<i>High-Speed Baud Rates for PIL Simulations</i>	215
<i>Changes in the Target Simulation Modules</i>	216

## High-Speed Baud Rates for PIL Simulations

**Virtual COM ports**

TargetLink supports faster baud rates to transfer data to/from TargetLink evaluation boards (EVB) for PIL simulations. Virtual COM ports support communication baud rates that are 2-, 4-, and 8-times faster than 115,200, but the actual communication baud rate depends on the hardware. Note that the download baud rate is not affected, its maximum limit remains 115,200 (depending on the EVB even less).


**Related documentation** *Topic Settings* ([TargetLink Tool and Utility Reference](#))

## Changes in the Target Simulation Modules

### New and discontinued compiler versions

The following table shows the compiler versions that are now supported by TargetLink 4.2, refer to the New and No changes columns. Compiler versions that are no longer supported are listed in the Discontinued column.

Microcontroller Family	Compiler	New	No Changes	Discontinued
ARM CortexM3	Keil	5.2	—	5.1
C16x	TASKING	—	8.6	—
MPC57xxVLE	Diab	—	5.9	—
	GreenHill	—	2014	—
MPC560xVLE	Diab	—	5.9	—
	GreenHill	—	2012, 2014	—
RH850	GreenHill	2015	—	2014
S12X	Cosmic	—	4.8	—
	Metrowerk	—	5.1	—
SH2	Renesas	—	9.3	—
SH2A-FPU	Renesas	—	9.4	—
TriCore17xx	TASKING	6.0	3.2	5.0
TriCore2xx	TASKING	6.0	—	5.0
	GCC	—	4.6	—
V850	GreenHill	2015	—	2014
XC22xx	TASKING	—	3.0	—

For more information on the evaluation boards supported by TargetLink, refer to *Combinations of Evaluation Boards, Microcontrollers, and Compilers* ( [TargetLink Evaluation Board Hardware Reference](#)).

### Note

For further PIL support combinations that are part of a valid Software Maintenance Service (SMS) contract, refer to dSPACE's TargetLink PIL Support website at the TargetLink Product Support Center.



# Data Dictionary and Data Management

## Where to go from here

Information in this section

<i>Improvements to the Data Dictionary</i>	217
<i>Predefined Code Generator Option Sets in Data Dictionaries</i>	218

## Improvements to the Data Dictionary

### DD Comparison Pane (GUI) via Windows Command Prompt

You can now graphically compare two DD files via Windows Command prompt. You can start the Data Dictionary Manager and its DD Comparison pane stand-alone without a MATLAB environment, e.g., from inside a version control system.

#### Related documentation

- *Basics on Comparing and Merging DD Workspaces in the DD Comparison Pane* ([📖 TargetLink Data Dictionary Basic Concepts Guide](#))

### Custom command calls

You can call your own commands from within the Data Dictionary passing an arbitrary number of parameters. Custom commands can be external programs, your own M scripts or other preparatory scripts or functions you want to call.

#### Related documentation

- *How to Specify Custom Command Calls* ([📖 TargetLink Data Dictionary Basic Concepts Guide](#))
- *DD\_CUSTOM\_COMMAND* ([📖 TargetLink Demo Models](#))

### Embedded help

Embedded help was added to several objects and properties in the Data Dictionary Manager.

### Improved handling of non-integer types

Via the Plain Variable dialog, you can now specify a local scaling for non-integer types in the Data Dictionary.

Additionally, the Adjust to Typedef context-menu command now also works for non-integer types.

**Related documentation**

- Adjust to Typedef

## Predefined Code Generator Option Sets in Data Dictionaries

**Adjusting code generation via DD optionsets**

TargetLink Data Dictionaries (based on predefined system templates, e.g. dsdd\_master\_advanced.dd) now provide different option sets to easily adjust the production code generation for specific objectives, e.g., high MISRA C compliance or speed versus code-size.

**Related documentation**

- *Basics on Configuring the Code Generator for Production Code Generation* ([📖 TargetLink Customization and Optimization Guide](#))

## Code Generator Options

### New Code Generator Options

**Overview of new Code Generator options**

The following new Code Generator options are available with TargetLink 4.2.

- **ForceBooleanOperandsInBooleanOperations**

Inserts additional *unequal to zero* ( $x \neq 0$ ) comparisons into the generated code to force a Boolean context in logical operations and control flow expressions for non-Boolean types to achieve MISRA C compliance even in such modeling situations.

**Related documentation**

- *ForceBooleanOperandsInBooleanOperations* ([📖 TargetLink Block and Object Reference](#)).

For reference information on all Code Generator options, refer to *Alphabetical List of Code Generator Options* ([📖 TargetLink Block and Object Reference](#)).

### Migration aspects of Code Generator options

Migration aspects include:

- Removed Code Generator option
- Changed Code Generator options
- Recommended compatibility settings
- Basics on changed defaults

For details, refer to *Migration Aspects Regarding Code Generator Options* on page 234.

## Tool Chain Integration

### Where to go from here

Information in this section

<i>Modeling with RTI Bypass Blocks and Generating Code for On-Target Bypassing</i>	219
<i>Build Binary Files for Functional Mock-up Units</i>	220

## Modeling with RTI Bypass Blocks and Generating Code for On-Target Bypassing

In TargetLink, you can model with the RTI Bypass Blockset. The RTI Bypass Blockset can be used to configure ECU interfaces and implement new ECU functions for bypassing or for tests. You can then use TargetLink's Code Generator for code generation in order to add a new or replace an existing control function directly on the target hardware (on-target bypassing). This is done by using the free resources on the target ECU. On-target bypassing is also often called *On-Target Prototyping*, because you can develop and replace ECU functions without the use of additional (external) hardware.



## Other

### Where to go from here

Information in this section

<i>General Enhancements and Changes</i>	221
<i>TargetLink Demos</i>	223

## General Enhancements and Changes

### Improved documentation generation

**HTML support** TargetLink now supports the adding of HTML files in the generated documentation.

**Incremental documentation generation** TargetLink now supports incremental documentation generation which means that you can reuse parts of already generated documentation during documentation generation, for example, for an integration model.

**Support for Scalable Vector Graphics** TargetLink now supports SVG in the generated documentation. SVG is the default format. PNG is also possible. You can set the file format in the `ImageFormatType` property of the `tlDoc` command. However, EPS is no longer supported. Also refer to *Discontinued TargetLink Features* on page 255.

**Improved usability for embedded commands** You can now select predefined configurations of embedded commands in the Autodoc Customization block dialog instead of typing them. TargetLink inserts the selected command into its edit field using the correct syntax.

**Categorizing and filtering blocks** You can categorize and filter Autodoc Customization blocks so that you can easily control which blocks are considered or left out by TargetLink during the documentation generation.

**Hook script for further customization** You can now use a new hook script to manage Autodoc Customization blocks. For example, you can set the order, in which Autodoc Customization blocks are generated. The hook script always overwrites competing settings specified at block level.

### Related documentation

- [Basics on Documentation Generation \(TargetLink Interoperation and Exchange Guide\)](#)

#### Faster dialog opening

TargetLink now opens the Main dialog and other block dialogs twice as fast as with TargetLink 4.1.

### Related documentation

- None

#### Defining Simulink data types for ddv IC structures

TargetLink now lets you map Simulink data types from the components of structured DD Variable objects used to generate IC structures (ddv).

### Related documentation

- [How to Manually Create a Mapping Between a DD Variable Object and a Simulink Bus \(TargetLink Customization and Optimization Guide\)](#)

#### New Coded Type for C99 `_Bool` supported in `TargetConfig.xml`

You can now use the `C99_Boolean` data type, i.e., TargetLink's `Boolean` typedef maps to `_Bool`.

Introducing this coded type makes the production code more manageable with some MISRA C checkers. Currently, TargetLink provides its own `Boolean` data typedef that is MISRA-compliant. However, some MISRA checkers do not accept this data type, which causes false positive violations. The `C99_Boolean` data type is always recognized as an effective `boolean` data type during MISRA compliance checking. This might reduce the number of false positive violations of MISRA rules.

### Tip

To adjust `TargetConfig.xml` settings, you can create a new target or clone an existing one, especially for PIL simulation.

**New Generic C99 code generation target** In addition, you can use the new Generic C99 code generation target in the TargetLink Main Dialog. This uses the `C99_Boolean` data type as default.

### Note

The use of `_Bool` also depends on the C compiler. For more information, refer to your target compiler's manual.

For SIL compilation, the supported MEX/SIL compilers support the `_Bool` data type with their default compiler options.

**Related documentation**

- *Example of Controlling the Generation of the File Containing TargetLink Base Data Types* ([📖 TargetLink Customization and Optimization Guide](#))
- *How to Clone Target/Compiler Combinations to Outside the TargetLink Installation* ([📖 TargetLink Customization and Optimization Guide](#))

**Support of Windows-1252 character set**

TargetLink now also supports the Windows-1252 (CP1252, Western European) character set.

**Related documentation**

- *Basics on the Code Generation Report* ([📖 TargetLink Preparation and Simulation Guide](#))
- *How to Specify the Used Character Set* ([📖 TargetLink Interoperation and Exchange Guide](#))

**Signal injection and tunneling for struct variables**

You can now access the whole structure variable and not only its components when injecting or tunneling signals.

The associated Simulink.Signal object must be of the Bus datatype whose hierarchy tree corresponds to the hierarchy tree of the structure variable. The names of the nodes are not taken into account.

**Related documentation**

- *Basics on Injecting or Tunneling Signals During Simulation* ([📖 TargetLink Preparation and Simulation Guide](#))
- *Modeling Buses via Data Store Blocks* ([📖 TargetLink Customization and Optimization Guide](#))

## TargetLink Demos

**New demos**

The following new demos come with TargetLink

**AR\_COLLISION\_DETECTION** This new demo shows the following features:

- Modeling asynchronous client-server communication to generate calls of the `Rte_Call` and `Rte_Result` API functions in your production code.
- Modeling transformer error logic for client-server communication.

Refer to *AR\_COLLISION\_DETECTION* ([📖 TargetLink Demo Models](#)).

**AUTODOC\_GENERATION** This new demo shows the following features:

- How to control the general structure and content of the document by using a dedicated M script to call the documentation generation
- How to work with Autodoc Customization blocks to control which functions should be documented
- How to insert user-provided content (texts, images, HTML commands, ...) into the documentation generation process

Refer to *AUTODOC\_GENERATION* ([📖 TargetLink Demo Models](#)).

**DD\_CUSTOM\_COMMAND** This new demo shows the following features:

- You can call your own commands from within the Data Dictionary. You can also specify additional arguments. Custom commands can be external programs, your own M scripts or other preparatory scripts or functions you want to call. The `dd_customcommand` demo contains four different custom command calls in the Pool area of the Data Dictionary Manager.

Refer to *DD\_CUSTOM\_COMMAND* ([📖 TargetLink Demo Models](#)).

**EMBEDDING\_EXTERNAL\_CODE** This new demo shows the following features:

- The demo shows different ways of embedding external code, e.g. via a Function block or a Custom Code block.

Refer to *EMBEDDING\_EXTERNAL\_CODE* ([📖 TargetLink Demo Models](#)).

**ENUM\_TYPES** This new demo shows the following features:

- Using Enumerations in model and generated production code, for example with Stateflow active state data.

Refer to *ENUM\_TYPES* ([📖 TargetLink Demo Models](#)).

**FUNCTION\_VARIANTS** This new demo shows the following features:

- You can choose different binding times for function variants, e.g. while modeling or before generating production code.

Refer to *FUNCTION\_VARIANTS* ([📖 TargetLink Demo Models](#)).



**VARIABLE\_INITIALIZATION** This new demo shows the following features:

- Different ways to initialize variables, e.g. via a `Main Restart` function.

Refer to *VARIABLE\_INITIALIZATION* ([📄 TargetLink Demo Models](#)).

**Extended demos**

The following demos contain new feature demonstrations:

**BUS\_STRUCT** This changed demo now also shows the following features:

- How to handle buses with Data Stores.

Refer to *BUS\_STRUCT* ([📄 TargetLink Demo Models](#)).

## API Functions and Hook Scripts

**Where to go from here**

Information in this section

<i>New API Functions</i>	225
<i>New Hook Scripts</i>	226

## New API Functions

API Function	Purpose
<code>tlEnumDataType</code>	Imports a Simulink enumeration data type to the Data Dictionary.
<code>tlExecuteDDCustomCommand</code>	Executes a Data Dictionary CustomCommand or CustomCommandGroup object.
<code>tlSimulinkBusObject</code>	Processes Simulink.Bus objects in TargetLink.
<code>tlStartCallbackWithTimer</code>	Starts a callback within a timer.

**Related documentation:**

- `tlEnumDataType('CreateDDTypedef', propertyName, propertyValue, ...)` ([📄 TargetLink API Reference](#))

- *tlExecuteDDCustomCommand(ddCmdObj, propertyName, propertyValue, ...)* ([TargetLink API Reference](#))
- *tlSimulinkBusObject* ([TargetLink API Reference](#))
- *tlStartCallbackWithTimer(callback)* ([TargetLink API Reference](#))

## New Hook Scripts

TargetLink provides the following new hook scripts:

Hook Script	Purpose
<code>tl_post_add_operationresultprovidersubsystem_hook</code>	Lets you customize AUTOSAR frame model generation.
<code>tl_post_autodoc_filter_hook</code>	Lets you enter your own commands, e.g., to filter the list of the found Autodoc Customization blocks.

### Related documentation

- *tl\_post\_add\_operationresultprovidersubsystem\_hook* ([TargetLink File Reference](#))
- *tl\_post\_autodoc\_filter\_hook* ([TargetLink File Reference](#))

# Migrating to TargetLink 4.2 and TargetLink Data Dictionary 4.2

## Upgrade process

To upgrade to a new TargetLink version you have to adjust the following:

- Your data dictionaries
- Your models
- Your scripts and hook scripts

To migrate libraries/models from TargetLink versions older than 4.1, you also have to perform the migration steps of the TargetLink versions in between. Refer to [TargetLink Migration Guide](#). The general and TargetLink 4.2 release-specific migration information is identical in both documents.

You can launch a model upgrade manually by using the `t1Upgrade` API function. For detailed instructions, refer to *How to Manually Upgrade Libraries and Models Via the API* on page 232.

Carefully read all of the following information and modify your tool chain accordingly, where necessary.

## Where to go from here

Information in this section

<i>MATLAB-Related Changes</i>	228
<i>Upgrade of Models, Libraries, and Data Dictionaries</i>	228
<i>Code Generator Options</i>	234
<i>API Functions and Hook Scripts</i>	236
<i>AUTOSAR-Related Migration Aspects</i>	238
<i>Code Changes</i>	242
<i>Other</i>	254
<i>Obsolete</i>	255
<i>Changes in Future TargetLink Versions</i>	258

## MATLAB-Related Changes

### Modified features in MATLAB R2016b

**Changed startup behavior** MATLAB has changed the execution order of startup scripts.

- Up to MATLAB R2016a, the `startup.m` script is executed first, before the dSPACE-specific initialization scripts `dsstartup.m` and `dspoststartup.m`.
- As of MATLAB R2016b, the `startup.m` script is executed last, after the dSPACE-specific initialization scripts.

This modification might result in a changed startup behavior.

## Upgrade of Models, Libraries, and Data Dictionaries

### Where to go from here

Information in this section

<i>Migrating to TargetLink 4.2</i>	228
<i>How to Upgrade a Data Dictionary with Included DD Files</i>	230
<i>How to Manually Upgrade Libraries and Models Via the API</i>	232

## Migrating to TargetLink 4.2

### Using TargetLink models with earlier TargetLink versions

Models that were saved with a later TargetLink version cannot be opened by an earlier TargetLink version.

### Indirect upgrade from TargetLink 2.x

Libraries, models, and DD files from TargetLink versions prior to TargetLink 3.1 cannot be upgraded directly.

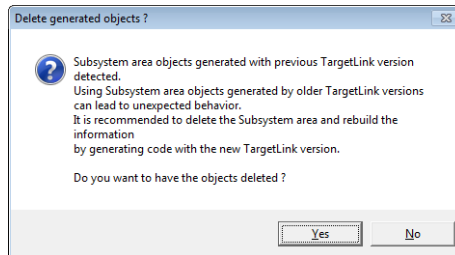
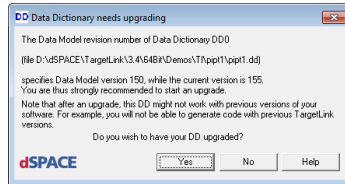
However, you can perform an indirect upgrade. First, migrate older libraries, models, and DD files to at least TargetLink 3.1. Then you can upgrade them to TargetLink 4.2.

We recommend to upgrade to TargetLink 3.5 before updating to TargetLink 4.2.

### Direct upgrade from TargetLink 3.1 or higher

TargetLink 4.2 automatically upgrades models, libraries, and Data Dictionaries created with TargetLink 3.1 or higher.

The following dialogs might appear during the upgrade process of the Data Dictionary file:



**User interaction** In the following cases, user interaction is required:

- *Legacy libraries never prepared for TargetLink* on page 229
- *Migrating from TargetLink 32-bit to TargetLink 64-bit* on page 230
- *DD files with included partial DD files* on page 230
- *How to Manually Upgrade Libraries and Models Via the API* on page 232

### Legacy libraries never prepared for TargetLink

Libraries created with TargetLink 3.x or 4.0 that were never prepared using the `tl_prepare_system(propertyName, propertyValue, ...)` ([TargetLink API Reference](#)) API function cannot be upgraded automatically by TargetLink 4.2.

#### Solution

1. Open the library in the previous TargetLink version and prepare it for the upgrade by using `tl_prepare_system(propertyName, propertyValue, ...)` ([TargetLink API Reference](#)).
2. Save the library.
3. Open the library with TargetLink 4.2.

### Related documentation

- *How to Make TargetLink User Libraries Upgrade-Capable*  
(📖 [TargetLink Orientation and Overview Guide](#))

#### Migrating from TargetLink 32-bit to TargetLink 64-bit

Custom code S-functions built with 32-bit TargetLink versions do not work with 64-bit versions of TargetLink and vice versa.

**Solution** Initiate a rebuild of all custom code S-functions using the `tlUpgrade('Model', <MyModel>, 'CheckModel', 'FixIssues')` API function. For more information, refer to *tlUpgrade(propertyName, propertyValue, ...)* (📖 [TargetLink API Reference](#)).

#### DD files with included partial DD files

To upgrade DD files with included partial DD files, refer to *How to Upgrade a Data Dictionary with Included DD Files* on page 230.

## How to Upgrade a Data Dictionary with Included DD Files

### Objective

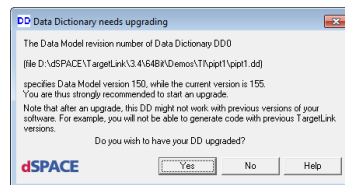
If you open a TargetLink model with an old Data Dictionary file that was not upgraded, you have to upgrade the Data Dictionary file.

### Method

#### To upgrade a Data Dictionary with included DD files

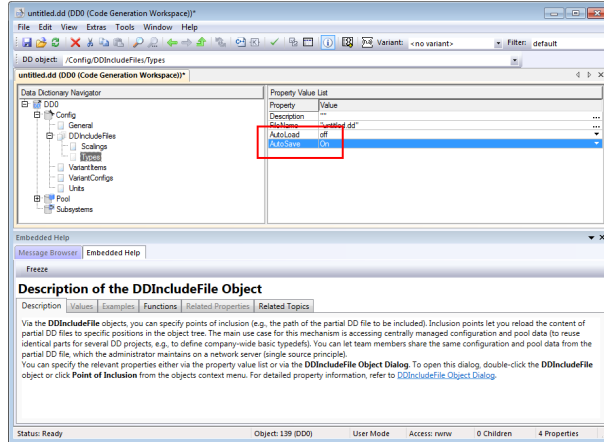
- 1 Open the model and the referenced TargetLink Data Dictionary, or type `dsdd('Open', <DDFile>)` in the MATLAB Command Window.

The Data Dictionary needs upgrading dialog automatically opens if an earlier DD version is involved.



- 2 Select No in the upgrade dialog.

- Under /Config/DDIncludeFiles, set the AutoLoad and AutoSave properties for each included DD file as shown in the following screenshot.

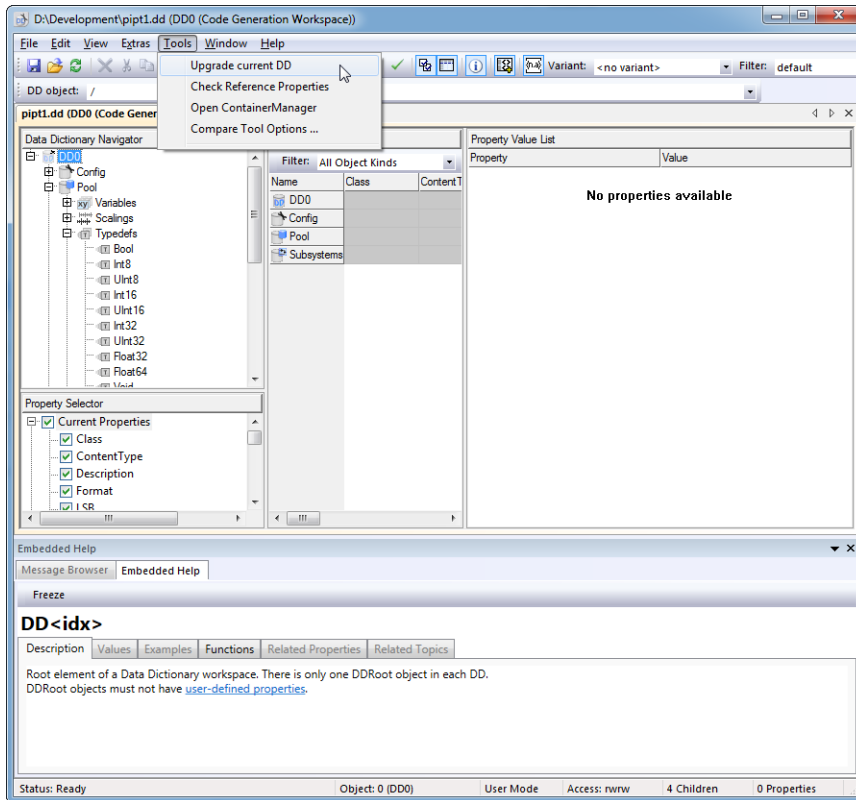


This ensures that after the Data Dictionary and the included DD files were upgraded, the included DD files that have been upgraded are saved when the Data Dictionary is saved. You can set these properties for a large number of included DD files via the Object Explorer.

**Tip**

You can also use the Point of Inclusion dialog to set the included DD file properties.

- Start the Data Dictionary upgrade (with the included DD files) via Tools – Upgrade current DD in the DD Manager, or enter `dsdd ('Upgrade')` in the MATLAB Command Window.



- 5 Save the Data Dictionary (with write permission to the relevant DD file). This completes the upgrade of the DD file and the included partial DD files.

## Result

When you open the DD file again, the upgrade dialog does not open, because the DD file and the included partial DD files are up-to-date. After the files were properly upgraded, it is recommended to restore the old settings for the included DD files.

## How to Manually Upgrade Libraries and Models Via the API

### Objective

You can manually update libraries and models via the `tlUpgrade(propertyName, propertyValue, ...)` ([TargetLink API Reference](#)) API function and save them afterwards, e.g., to prepare a



central upgrade of libraries and models in a tool chain scenario with several users.

### Note

When upgrading models and libraries, first upgrade models or libraries that do not reference any other libraries, i.e., the blocks/subsystems they contain have no links to other libraries. Start with the bottom library and then upgrade the libraries above it in ascending order.

For details on libraries that were never prepared, refer to *Legacy libraries never prepared for TargetLink* in *Migrating to TargetLink 4.2* on page 228.

---

### Method

#### To manually upgrade libraries and models via the API

- 1 In the MATLAB Command Window, type `dsdd_manage_project('Open', '<name>.dd')` to load the required and already upgraded DD project file (one way to upgrade DD project files is to use the `dsdd('Upgrade'[, <DD_Identifier>])` command, refer to [Upgrade](#) ([TargetLink Data Dictionary Reference](#))).
- 2 Type `tlUpgrade('Model', '<Model|Library>.mdl', 'CheckModel', 'FixIssues')` to upgrade single models or libraries.
- 3 Save the upgraded model or library files, e.g., `Library.mdl`.
- 4 Repeat steps 2 and 3 for all other models or libraries.

---

### Result

You upgraded your models and libraries.

# Code Generator Options

## Migration Aspects Regarding Code Generator Options

### Removed Code Generator option

The following Code Generator options were removed from TargetLink:

Removed Option
ConsiderFunctionClassesForBlockDiagramBasedSwitchOptimization
ConsiderStateflowAuxiliariesForVariableSharing
EnableVariableVectorWidths
TreatAllStateflowFunctionsAsWeakAtomic

### Changed Code Generator options

Code Generator Option	Old Default	New Default
EnableReportGeneration	off	on
ReportFailedFunctionReuseVariablePropagation	on	off
ReportMismatchingPortBlockSignalSpecifications	on	off
ReportVariablesExcludedFromPIM	on	off

### Recommended compatibility settings

Specify the following settings for new TargetLink 4.2 Code Generator options to ensure best possible downward compatibility:

- Set `ForceBooleanOperandsInBooleanOperations` to `off`.

### Basics on changed defaults

The settings of the Code Generator options are stored with the model (model-based option storage). In addition, you can store user-defined sets of Code Generator options in DD `CodegenOptionSet` objects (DD-based option storage). You can use DD `CodegenOptionSet` objects as a central source for overwriting and instead of model-based option settings since TL 4.1.

If a model-based option value equals the old default value, it is automatically changed to the new default value during the upgrade. If a DD-based option value equals the old default value, it is not changed to the new default value during the upgrade but keeps the old value.

**Option value = old default** If Code Generator options equal default values in the former TargetLink version and the new TargetLink version uses modified default values, note the following points:

- Model-based option:  
If you want to keep the old default values, you must reset them manually.
- DD-based option:  
If you want to use the new default values, you must adjust them manually.

The following table is an example describing the impact of a TargetLink upgrade (TL<sub>Old</sub> to TL<sub>New</sub>) on three arbitrary option values: 9, 11, and 13. The table illustrates two basic migration scenarios:

- Scenario #1: New default = old default  
The default value of a Code Generator option has not changed in the new TargetLink version, i.e., the default value remains 9. None of the option values is changed.
- Scenario #2: New default ≠ old default  
The default value of a Code Generator option changed with the new TargetLink version, i.e., the default value changed to 11.

Option Storage	Option Value (TL <sub>Old</sub> )	Option Value (≤ TL <sub>New</sub> )	
	Default = 9	Default = 9 (Scenario #1)	Default = 11 (Scenario #2)
Model-based	9 <sup>1)</sup>	9 <sup>1)</sup>	11 <sup>2)</sup>
	11	11	11 <sup>1)</sup>
	13	13	13
DD-based	9	9	9 <sup>3)</sup>
	11	11	11
	13	13	13

<sup>1)</sup> The option value is not stored with the model because it equals the default.  
<sup>2)</sup> Manual reset might be necessary.  
<sup>3)</sup> Manual adjustment might be necessary.

**Option value = new default** If the Code Generator options did not equal default values in the former TargetLink version (A) but do in the new TargetLink version (B), TargetLink assumes that you intentionally specified the default value in the new TargetLink version. The same applies if the default changes again in the next TargetLink version (C).

**Note**

Upgrading  $TL_A \Rightarrow TL_B \Rightarrow TL_C$  and upgrading  $TL_A \Rightarrow TL_C$  can cause different option values (see the following table).

If the default values for TargetLink versions A, B, and C read 9, 11, and 13, and an option value equaled 11 in version A, an upgrade to version C would change the option value as follows:

Upgrade Strategy	Option Value $TL_A$ Default = 9	Option Value $TL_B$ Default = 11	Option Value $TL_C$ Default = 13
A $\Rightarrow$ B $\Rightarrow$ C	11 ( $\neq$ default)	11 (= default) <sup>1)</sup>	13 (= default) <sup>1)</sup>
A $\Rightarrow$ C	11 ( $\neq$ default)	—	11 ( $\neq$ default)

<sup>1)</sup> The option value is not stored with the model, because it equals the default.

**New Code Generator options**

For more details on new Code Generator options, refer to *New Code Generator Options* on page 218.

**Related topics**

References

- *Code Generator Options* ( [TargetLink Block and Object Reference](#))


## API Functions and Hook Scripts

### Changes in TargetLink and TargetLink Data Dictionary API Functions

**tl\_set**

**Migration issue** You can now use `tl_set` to change the block properties of library items from inside mask callbacks and similar callbacks in the same API call, by setting the `AllowLibraryBlockModification` property to on:

```
tl_set(<block>, ..., 'AllowLibraryBlockModification', 'on', ...)
```

**Related documentation** `tl_set` (refer to *tl\_set(hBlock, property\_1, value\_1, ..., property\_n, value\_n)*) ( [TargetLink API Reference](#))

**tl\_generate\_swc\_model**

**Migration issue** With the introduction of asynchronous client-server communication, the `AddOperationCallTriggerPort` property of the `tl_generate_swc_model` ([📖 TargetLink API Reference](#)) API function was renamed to `AddOperationSubsystemTriggerPort`.

**Solution** Adapt your user scripts.

TargetLink automatically changes the name of the `/Pool/Autosar/Config/FrameModelGeneration/AddOperationCallTriggerPort` property to `AddOperationSubsystemTriggerPort` when upgrading the Data Dictionary file/contents to the latest revision.

**Related documentation**

- *Asynchronous Client-Server Communication* on page 213
- *Modeling Client-Server Communication* ([📖 TargetLink AUTOSAR Modeling Guide](#))
- *tl\_generate\_swc\_model* ([📖 TargetLink API Reference](#))

**tldoc**

The generation of RTF documentation is no longer supported. The `CreateRTFFile` property of the `tldoc('Convert')` command is obsolete. If you used this property in the past, you must edit your scripts and remove the property from the `tldoc('Convert')` commands. Use HTML or PDF instead.

To postprocess the documentation manually, use an HTML editor or a PDF editor of your choice. However, you should consider to customize the documentation generation which might render manual edit steps unnecessary. New features of the TargetLink documentation generation process might help you. Refer to *General Enhancements and Changes* on page 221.

**Related documentation**

- *Basics on Customizing the Generated Documentation* ([📖 TargetLink Interoperation and Exchange Guide](#))
- `tldoc('Convert', propertyName, propertyValue, ...)` ([📖 TargetLink API Reference](#))
- *Discontinued TargetLink Features* on page 255

**Data model filter rule files**

**Migration issue** Existing data model filter rule files can contain invalid elements because the TargetLink Data Dictionary's data model changed. The following files shipped with TargetLink 4.1 are affected:

- `DD_Filter_Admin.xml`
- `DD_Filter_AR_User.xml`
- `DD_Filter_NonAR_NonRTOS_User.xml`

**Solution** You can check filter rule files via the API in the MATLAB Command Window:

Checking a Single File	Checking Filter Rule Sets <sup>1)</sup>
<pre>dsdd_free; dsdd('ReadFilterRuleSet', 'file', '&lt;myFile&gt;.xml'); ds_error_register(dsdd('GetMessageList')); ds_msgdlg('update');</pre>	<pre>dsdd_free; dsdd('ReloadFilterRuleSets'); ds_error_register(dsdd('GetMessageList')); ds_msgdlg('update');</pre>

<sup>1)</sup> All the files contained in the directory defined in Data Dictionary - Filter Rules in the TargetLink Preferences Editor.

TargetLink informs you about errors in TargetLink's Message Browser. Each error contains the following information, so that you can fix it in an XML-capable editor of your choice:

- Filename
- Row number
- Column number

## AUTOSAR-Related Migration Aspects

### Where to go from here

Information in this section

<i>Migrating Interrunnable Variable Specifications</i>	238
<i>Migrating Transformer Error Code Specifications</i>	242

## Migrating Interrunnable Variable Specifications

### Migration issue

Interrunnable variables (IRVs) are now modeled via dedicated DD InterRunnableVariable objects and not via DD Variable objects that reference one of the predefined DD VariableClass objects called EXPLICIT\_IRV, DISP\_EXPLICIT\_IRV, IMPLICIT\_IRV and DISP\_IMPLICIT\_IRV (access function mechanism).

---

### Preparing your old Data Dictionary

Before you can migrate your IRV specifications to TargetLink 4.2, you have to prepare your old Data Dictionary:

- Every DD VariableGroup object that contains DD Variable objects that represent IRVs must be referenced by *exactly one* DD SoftwareComponent object via its RelatedVariables/InterrunnableVariablesRef property.
- For each IRV that you want to be initialized, make sure that an AUTOSAR constant specification is well defined by the following properties of the DD Variable object:
  - Value
  - InitConstantName
  - InitConstantFileName
  - InitConstantPackage
  - InitConstantUuid
- Save your Data Dictionary.

---

### Upgrading your Data Dictionary

You can now upgrade your Data Dictionary. TargetLink performs the following steps for each DD Variable object contained in a DD VariableGroup object that is referenced via a DD SoftwareComponent object's InterrunnableVariablesRef property, contained in the RelatedVariables subtree:

- Creates a DD InterRunnableVariable object and adds it to the software components' InterRunnableVariables subtree.
- Copies the IRV-specific property values and all the custom properties from the DD Variable object to the DD InterRunnableVariable object.
- References the old DD Variable object at the DD InterRunnableVariable object's UpgradeFromVariableRef custom property.
- If the old DD Variable object defines a constant specification to initialize the IRV, it searches the DD Variable object that matches the IRV's constant specification:
  - If it finds a suitable Variable object, TargetLink references it at the InterRunnableVariable object's InitValueRef property.
  - If it does not find a suitable Variable object, TargetLink creates a new object and references it at the InterRunnableVariable object's InitValueRef property.

- Sets the old DD Variable object's Class property as follows after all objects were processed:

Old Variable Class	New Variable Class
EXPLICIT_IRV	default
IMPLICIT_IRV	
DISP_EXPLICIT_IRV	DISP
DISP_IMPLICIT_IRV	

- Removes the old DD VariableClass objects from the Data Dictionary.

### Note

- Do not delete or unset the UpgradeFromVariableRef property or the DD Variable object it references.  
This information is required during model upgrade.
- If you want to create a fresh DD workspace that contains only the data imported from an AUTOSAR file or SWC container, make sure to merge each DD InterRunnableVariable object's UpgradeFromVariableRef property and the DD Variable object it references into your new workspace.

## Upgrading your model

After you upgraded your Data Dictionary, you can upgrade your model. The model upgrade is automatically performed the first time a model is opened in a new TargetLink Version. If the attached Data Dictionary is not upgraded, TargetLink will prompt you to upgrade the Data Dictionary.

During model upgrade, TargetLink searches on the AUTOSAR page of port blocks and the Output page of Data Store Memory blocks for references of DD Variable objects that are referenced via a DD InterRunnableVariable object's UpgradeFromVariableRef property and makes the following settings:



Port Blocks	Data Store Memory Blocks
Replaces the references to the old DD Variable objects with references to the new DD InterRunnableVariable objects.	Deactivates references to the old Variable objects from the block's Output page and makes the following settings on the block's AUTOSAR page: <ul style="list-style-type: none"> <li>■ Selects the Use AUTOSAR communication checkbox.</li> <li>■ Sets the Kind property to InterRunnable.</li> <li>■ References the new DD InterRunnableVariable object.</li> </ul>

**Purging old variables**

After you made sure that the model upgrade succeeded, you can purge the DD from the old DD Variable objects and delete the DD InterRunnableVariable objects' UpgradeFromVariableRef property.

TargetLink provides the following API for this:

```
errorCode = dsdd(
    'DeleteObsoleteInterRunnableVariables',
    <DdSwcObjectIdentifier>);
```

**Note**

Executing this API function performs the following steps without warning. This cannot be undone.

- Deletes all DD Variable objects that are referenced via DD InterRunnableVariable objects' UpgradeFromVariableRef property.
- Deletes the DD VariableGroup objects that contained the DD Variable objects if they are empty.
- Deletes the UpgradeFromVariableRef property from the DD InterRunnableVariable objects.

**Related documentation**

- *Modeling Interrunnable Communication* ([📖 TargetLink AUTOSAR Modeling Guide](#))
- *Non-Scalar Interrunnable Variables* on page 214
- *DeleteObsoleteInterRunnableVariables* ([📖 TargetLink Data Dictionary Reference](#))

## Migrating Transformer Error Code Specifications

<b>Migration issue</b>	AUTOSAR changed transformer error codes from revision 4.2.1 to 4.2.2.  These changes are not resolved during the automatic upgrade but require your attention.
<b>Solution</b>	The latest version of the dsdd_master_autosar4.dd [System] DD template contains the new transformer error codes.  You can open this template in a new workspace and merge the error codes into your DD project file using the DD comparison pane. They are located in the following DD VariableGroup:  <code>/Pool/Variables/AUTOSAR/TransformerError</code>

## Code Changes

### Code Changes

<b>Non-boolean operands in boolean contexts</b>	The code for non-boolean operands in boolean contexts changed. This affects logical operations, control flow conditions, constants, and enums in unclear modeling situations (i.e., feeding non-boolean signals):
---	---

TargetLink ≤ 4.1	TargetLink 4.2
<pre>Int16Var &amp;&amp; BoolVar if(Int16Var) { ... } 42 0</pre> <p>No support for enum constants.</p>	<pre>(Int16Var != 0) &amp;&amp; BoolVar if (Int16Var != 0) { ... } 42 != 0 0 != 0</pre> <p>For operands that are an enum variable or enum constant, whose enum type provides an enum-constant for 0, this constant is used:</p> <pre>enum BasicColors = {RED = 0, GREEN = 1, ...} BasicColors color;  color != RED GREEN != RED</pre>

**Reason** To comply with MISRA C.

**Migration issue** To revert to code as generated by TargetLink versions ≤ 4.1, deactivate the ForceBooleanOperandsInBooleanOperations Code Generator option.

**Changes to the enum code section**

The code section for enums is moved directly behind the code section for includes. This affects source and header files.

**Reason** Enums can be required for other definitions whose section occurs after the include section.

**Auxiliary variables for interruptible communication**

For improved code readability, auxiliary variables created for interruptible communication now have the same name as the DD InterRunnableVariable object:

TargetLink ≤ 4.1	TargetLink 4.2
<pre>sint32 Aux_;   Aux_ = (sint32) Sa2_Rescaler;   Rte_IrvWrite_Run_MyIrv(Aux_);</pre>	<pre>sint32 MyIrv;   MyIrv = (sint32) Sa2_Rescaler;   Rte_IrvWrite_Run_MyIrv(MyIrv);</pre>

**Pointer variables** The names of IRV variables that are pointers are prefixed by p\_.

**Math block's pow function**

When using the pow function of the Math block, calculation differences between Simulink and TargetLink's production code in corner cases are removed.

TargetLink ≤ 4.1	TargetLink 4.2
<pre>if ((Sal_u_ &lt;= 0.F) &amp;&amp; ((Float64) Sal_v) != floor((Float64) Sal_v)) { ... }</pre>	<pre>if ((Sal_u_ &lt; 0.F) &amp;&amp; ((Float64) Sal_v) != floor((Float64) Sal_v)) { ... }</pre>

**Discrete-Time Integrator block**

The code generated for the Discrete-Time Integrator block with Integration set to Forward Euler and Limits other than -inf/inf can change:

TargetLink ≤ 4.1	TargetLink 4.2
<pre>UInt16 X_Sal_DTI_FW_Sat = 10 /* 5. */ /* LSB: 2^-1 OFF: 0 MIN/MAX: 2 .. 730 */;</pre>	<pre>Int16 X_Sal_DTI_FW_Sat = 10 /* 5. */ /* LSB: 2^-1 OFF: 0 MIN/MAX: -98 .. 630 */;</pre>

**Reason** To fit the data type chosen during code generation due to a corrected range calculation.

The code generated for the External reset changed.

TargetLink ≤ 4.1	TargetLink 4.2
<pre> /* Discrete Integrator: Subsystem/DTI: Condition for either edge trigger */ if ((Sal_Reset &gt;= 0) &amp;&amp; (X_Sal_DTI_TriggerIn &lt;= 0) &amp;&amp; ((Sal_Reset != X_Sal_DTI_TriggerIn) &amp;&amp; (Sal_DTI_LastEvent != 1))) {     Sal_DTI_LastEvent = 1; } else {     if ((Sal_Reset &lt;= 0) &amp;&amp; (X_Sal_DTI_TriggerIn &gt;= 0) &amp;&amp; ((Sal_Reset != X_Sal_DTI_TriggerIn) &amp;&amp; (Sal_DTI_LastEvent != -1))) {         Sal_DTI_LastEvent = -1;     }     else {         Sal_DTI_LastEvent = 0;     } } </pre>	<pre> /* Discrete Integrator: Subsystem/DTI: Condition for either edge trigger */ if ((Sal_Reset &gt; 0) &amp;&amp; (X_Sal_DTI_TriggerIn &lt;= 0) &amp;&amp; ((Sal_Reset != X_Sal_DTI_TriggerIn) &amp;&amp; (Sal_DTI_LastEvent != 1))) {     Sal_DTI_LastEvent = 1; } else {     if ((Sal_Reset &lt;= 0) &amp;&amp; (X_Sal_DTI_TriggerIn &gt; 0) &amp;&amp; ((Sal_Reset != X_Sal_DTI_TriggerIn) &amp;&amp; (Sal_DTI_LastEvent != -1))) {         Sal_DTI_LastEvent = -1;     }     else {         Sal_DTI_LastEvent = 0;     } } </pre>

**Reason** Matching Simulink's behavior where only positive values are considered as rising edges.

**Rate Limiter**

The code generated for the RemainderState variable of the Rate Limiter block can change:

TargetLink ≤ 4.1	TargetLink 4.2
<pre> static Int8 XRem_Sal_Rate_Limiter1; static Int16 XRem_Sal_Rate_Limiter2; </pre>	<pre> static Int16 XRem_Sal_Rate_Limiter1; static Int32 XRem_Sal_Rate_Limiter2; </pre>

**Reason** To fix an incorrectly chosen data type that was too small.

An additional cast can also be inserted in the calculation of the remainder:

TargetLink ≤ 4.1	TargetLink 4.2
<pre> Aux_S16 = Aux_U16 % 100; </pre>	<pre> Aux_S16 = (Int16) (Aux_U16 % 100); </pre>

**Reason** To comply with MISRA C.

**Function reuse and pointer-to-struct for Stateflow**

The code optimization for function reuse and pointer-to-structs in Stateflow was improved.

TargetLink ≤ 4.1	TargetLink 4.2
<pre> pISV-&gt;comp = 0; pISV-&gt;comp = 1; </pre>	<pre> pISV-&gt;comp = 1; </pre>

**Order of # combined # comments**

In some cases, the order of the # combined # comments above function calls can change:

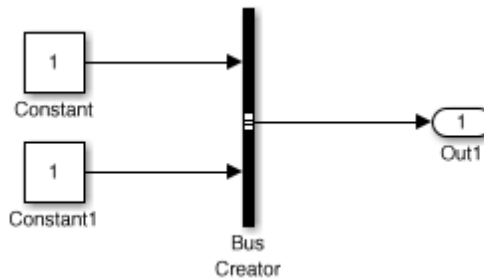
TargetLink ≤ 4.1	TargetLink 4.2
<pre> /* call of function: SimpleCall/Subsystem # combined # update(s) for inport SimpleCall/Subsystem/InPort # combined # Gain: SimpleCall/Gain # combined # Gain: SimpleCall/Gain1 # combined # TargetLink output: SimpleCall/OutPort */ foo(a, h);                     </pre>	<pre> /* call of function: SimpleCall/Subsystem # combined # update(s) for inport SimpleCall/Subsystem/InPort # combined # Gain: SimpleCall/Gain1 # combined # Gain: SimpleCall/Gain # combined # TargetLink output: SimpleCall/OutPort */ foo(a, h);                     </pre>

**IF variable in classic initialization mode**

In *classic initialization mode* (see [TargetLink Glossary](#)), the generated code changes for unenhanced Outport blocks that receive data from a non-virtual bus created by a Bus Creator block that directly precedes them. This can result in IF variables having a different name:

TargetLink ≤ 4.1	TargetLink 4.2
<pre> void Sa2_Subsystem(void) {   /* update of variable(s) associated with TL_Root/Subsystem/Constant */   IF_Sa2_a = 1;   /* update of variable(s) associated with TL_Root/Subsystem/Constant1 */   IF_Sa2_b = 1; }                     </pre>	<pre> static void Sa2_Subsystem(void) {   /* Outport: TL_Root/Subsystem/Out1 */   IF_Sa2_Out1 = 1;   IF_Sa2_Out1_a = 1; }                     </pre>

**Model used** The code example in the preceding table was generated from a model such as the following:



The signals in the non-virtual bus are called a and b.

**Unit Delay Reset Enabled**

The code generated for Unit Delay Reset Enabled blocks whose Use external initial value checkbox is selected was changed. When calculating the block's state variable, TargetLink no longer checks the enable condition for the FirstRun variable:

TargetLink ≤ 4.1	TargetLink 4.2
<pre> /* Subsystem/UDRE: reset and external initial value condition */ if ((Sa3_R != 0)    Sa3_Subsystem_FirstRun) { /* Subsystem/UDRE: state initialization */ X_Sa3_UDRE = Sa3_IV; }  /* Subsystem/UDRE: output */ Sa3_UDRE = X_Sa3_UDRE;  /* Subsystem/UDRE: enable condition */ if ((Sa3_E != 0) &amp;&amp; (!(Sa3_R != 0)) &amp;&amp; !( (Sa3_Subsystem_FirstRun != 0))) { /* Subsystem/UDRE: state update */ X_Sa3_UDRE = Sa3_u; } </pre>	<pre> /* Subsystem/UDRE: reset and external initial value condition */ if ((Sa3_R != 0)    Sa3_Subsystem_FirstRun) { /* Subsystem/UDRE: state initialization */ X_Sa3_UDRE = Sa3_IV; }  /* Subsystem/UDRE: output */ Sa3_UDRE = X_Sa3_UDRE;  /* Subsystem/UDRE: enable condition */ if ((Sa3_E != 0) &amp;&amp; !(Sa3_R != 0)) { /* Subsystem/UDRE: state update */ X_Sa3_UDRE = Sa3_u; } </pre>

**Reason** The block's behavior for the internal and external initial values was not consistent, if the following conditions applied:

- The R input received the value `false`.
- The E input received the value `true`

In this case the internal and external initial values are now identical in MIL and SIL simulation modes.

## AUTOSAR version as code comment

When generating code in AUTOSAR mode, TargetLink now places the AUTOSAR version in the code files' opening headers.

**Reason** The AUTOSAR version impacts on code generation and thus should be clearly visible.

## Auxiliary variables instead of iteration variables

Iteration variables are in some cases replaced by auxiliary variables, if optimization is enabled.

TargetLink ≤ 4.1	TargetLink 4.2
Sa3_For_Iterator_it <sup>1)</sup>	Aux_S32_a <sup>2)</sup>

1) Name configured as `$$_B_it`.

2) Name configured as `Aux_$$R`.

**Reason** Changes in the code generator optimization.

## Definitions of variables and macros

The definitions of variables and macros can be different:

- There can be variables of an `enum` type.
- Macro definitions are now sorted within functions and they receive code comments depending on the variable class.
- Parameterized macros (like macro access functions) are now grouped and sorted.

**Reason**

- Support for C enum types.
- Code readability.

**Iteration variable of while-subsystems**

The iteration variable generated for do-while-subsystems now always is incremented at the very end of the *while* loop. This also affects variables for further conditions:

TargetLink ≤ 4.1	TargetLink 4.2
<pre>while (Sa2_While_Iterator_cond &amp;&amp; (Sa2_While_Iterator_it &lt;= 4)) {     ...     Sa2_While_Iterator_cond = 1;     Sa2_While_Iterator_it++;      /* Unit delay: TL_SS/For Iterator Subsystem/Unit     Delay */     X_Sa2_Unit_Delay = (Int16) Sa2_While_Iterator_it; }</pre>	<pre>while (Sa2_While_Iterator_cond &amp;&amp; (Sa2_While_Iterator_it &lt;= 4)) {     ...     /* Unit delay: TL_SS/For Iterator Subsystem/Unit     Delay */     X_Sa2_Unit_Delay = (Int16) Sa2_While_Iterator_it;      Sa2_While_Iterator_cond = 1;     Sa2_While_Iterator_it++; }</pre>

**Reason** There was a possibility for incorrect code in the context of Unit Delay blocks.

**Code format of exponents**

The code format of exponents changed from three to two digits in case one or two digits suffice.

**Reason** To follow the change in Microsoft's CRT runtime library to improve floating-point precision and better compliance with the C standard. For more information, refer to [https://msdn.microsoft.com/en-us/library/bb531344.aspx#BK\\_CRT](https://msdn.microsoft.com/en-us/library/bb531344.aspx#BK_CRT).

**Reuse of table maps**

Code has changed for table map variables of look-up table functions with regard to function reuse:

TargetLink ≤ 4.1	TargetLink 4.2
Tables and axes referenced via pointers remained in the reused system.	Tables and axes referenced by pointers are moved to the file that contains the reuse structure.

**Reason** To facilitate component-based development.

**If Action and Switch Case Action subsystems with SateReset**

The code generated for If Action and Switch Case Action subsystems with StateReset was changed. The variable that stores the currently executed system is no longer superfluously initialized.

TargetLink ≤ 4.1	TargetLink 4.2
<pre> UInt8 Sal_If_System = 0; if (Sal_t1In1 &gt; 16384 /* 0.5 */) {     Sal_If_System = 1;     ... } else {     Sal_If_System = 2; } </pre>	<pre> UInt8 Sal_If_System; if (Sal_t1In1 &gt; 16384 /* 0.5 */) {     Sal_If_System = 1;     ... } else {     Sal_If_System = 2; } LastSystem = Sal_If_System; </pre>

**Reason** To comply with MISRA C .

**State charts with change detection**

The code generated for state charts with change detection for `input`, `Datastore`, or `Machine` data receives a new variable called `FirstRun`.

**Reason** Remove erroneous differences between Stateflow and the generated production code pattern.

**New code comments**

TargetLink now generates additional or changed code comments for the following code elements:

- Additional comments for grouping declarations and definitions of variables or functions.
- `PreBlockStatements` and `PostBlockStatements` are now always enclosed by empty lines.
- Non-empty `CodeSection-StatementTables` like `includes` or `typedefs` are now always followed by an empty line.

**Reason** To increase consistency across code patterns.

**Algebraic simplification**

TargetLink's code changed for algebraic simplification of expressions like `UInt16Var > 0`:

TargetLink ≤ 4.1	TargetLink 4.2
<code>UInt16Var</code>	<code>UInt16Var != 0</code>

**Reason** Keep boolean-context for MISRA C compliance and possibly facilitate further code optimizations.

**Logical operations with floating-point operands**

For logical operations that contain at least one operand of a floating-point type, TargetLink no longer casts all the operands to the greatest floating-point type contained within the operation:



TargetLink ≤ 4.1	TargetLink 4.2 <sup>1)</sup>
<pre> Sa1_F32F64AND = ((Float64) F32Var) &amp;&amp; F64Var; Sa1_F32I16OR = ((Float32) I16Var)    F32Var; Sa1_F32ScaledAND = F32Var &amp;&amp; (((Float32) ScaledI16Var) * 0.5F) + 2.F); // ScaledI16Var: Lsb = 0.5, Offset = 2.0 Sa1_F32IntValueAND = F32Var &amp;&amp; 1.F; Sa1_F32FlpValueOR = 1.25F    F32Var;                     </pre>	<pre> Sa1_F32F64AND = (F32Var != 0.F) &amp;&amp; (F64Var != 0.); Sa1_F32I16OR = (I16Var != 0)    (F32Var != 0.F); Sa1_F32ScaledAND = (F32Var != 0.F) &amp;&amp; ((ScaledI16Var + 4) != 0); // ScaledI16Var: Lsb = 0.5, Offset = 2.0 Sa1_F32IntValueAND = (F32Var != 0.F) &amp;&amp; (1 != 0); Sa1_F32FlpValueOR = (1.25F != 0.F)    (F32Var != 0.F);                     </pre>

<sup>1)</sup> With the ForceBooleanOperandsInBooleanOperations Code Generator option set to on.

**Reason** To comply with MISRA C.

### Scaling-invariant graphical functions in Stateflow

The code generated for scaling-invariant graphical functions in Stateflow can lack optimizations.

**Reason** This is a fail-safe to prevent false code optimization of code carrying side effects.

**Solution** Use a function class whose Optimization property is set to SIDE\_EFFECT\_FREE for the graphical function.

#### Note

Do this only if all the variables that this function operates on are function local and the function does not trigger any event.

### Simulink enum constants

The code for Simulink enum constants changed for following modeling elements. They now longer are integers but enumeration constants of an enum data type that is implicitly generated by TargetLink:

- Case conditions of Simulink's Switch-Case block
- Simulink enum constants in the Stateflow action language.
- Data port indices of TargetLink's Multiport Switch block
- Threshold of TargetLink's Switch blocks specified as follows:
  - They do not reference a DD Variable object.
  - Their Class is set to default.
- TargetLink's Constant blocks specified as follows:
  - They do not reference a DD Variable object.
  - Their Allow signal specification checkbox is cleared.
  - Their Class is set to default.

The following code is generated from a Multiport Switch block:

TargetLink ≤ 4.1	TargetLink 4.2
<pre>switch (Sal_InPort1) {   case 0: {     Sal_Merge = Sal_InPort2;     break;   }   case 2:   case 3: {     Sal_Merge = Sal_InPort3;     break;   }   default: {     Sal_Merge = Sal_InPort4;     break;   } }</pre>	<pre>switch (Sal_InPort1) {   case BasicColors_Red: {     Sal_Merge = Sal_InPort2;     break;   }   case BasicColors_Green:   case BasicColors_Blue: {     Sal_Merge = Sal_InPort3;     break;   }   default: {     Sal_Merge = Sal_InPort4;     break;   } }</pre> <p>The definition of TargetLink's enum data type looks like this:</p> <pre>typedef enum BasicColors_tag {   BasicColors_Red = 0,   BasicColors_Yellow = 1,   BasicColors_Green = 2,   BasicColors_Blue = 3,   BasicColors_Orange = 4,   BasicColors_Black = 5,   BasicColors_Pink = 6,   BasicColors_Brown = 7,   BasicColors_Magenta = 8 } BasicColors; /* Description: Enumeration type derived from Simulink type BasicColors */</pre>

**Reason** To fully support enum data types.

**Solution** To revert to code as generated by TargetLink versions ≤ 4.1, create a DD EnumTemplate object whose Filter property is set to ALWAYS and that references the DD EnumImplementation object

Additionally, you may adjust your mapping files for model preparation to EnumImplementation.

### Constraint ranges of reused variables and data variants

The code for variables has changed in the context of function reuse and data variants, if the corresponding DD Variable object is specified as follows:

- Its Min property is not specified.
- Its Max property is not specified.
- Its VariableClass object's Const property is set to on.
- Its VariableClass object's Info property is set to none, readonly, or bypassing\_readonly.

TargetLink now uses the type ranges as these variables' constraint range limits. This can lead to code that is less efficient.

**Reason** To prevent possibly incorrectly over-optimized code in the context of function reuse and data variants.

**Migration issue** If you observe less efficient code for such variables, consider specifying their Min and Max values.

### Code comments for type definitions and declarations of variables

The code comments for the following code elements changed:

- The constraints (ranges, scaling) of type definitions are now documented at the type definition:

```
typedef Uint16 ADC; /*
    Unit: m
    LSB: 2^-11 OFF: 0 MIN/MAX: -10 .. 10 */
```

- The order within code comments for declarations was changed. The information about the scaling and the description have changed places.

```
Int16 Sa1_InPort3; /*
    <BlockComment>
    Unit: Gigawatt
    LSB: 2^-8 OFF: 0 MIN/MAX: 0 .. 0.99
    Description: wash dish */
```

Additionally, a superfluous whitespace was removed from OFF and MIN/MAX.

- The ; in variable declarations and the , in struct initializers are now placed directly behind the identifier, not behind the optional comment:

```
Int16 Sa1_InPort3; /* Description: something */

struct my_tiny_struct S {
    5, /* Description: five */
    6
};
```

This does not affect comments describing the physical value of numeric constants:

```
Int32 MyVal = 37 /* 0.145 */; /* LSB: 2^-8 OFF: 0 MIN/MAX: 0 .. 0.99 */
```

- Comments for variables with the default scaling (LSB = 1, Offset = 0, no unit, relevant for all dimensions) no longer receive a scaling comment except when ranges are explicitly specified.
- Float variables with explicitly specified ranges now get MIN/MAX comments but never scaling comments.
- Macro variables (with the exception of RDI-macros) only get MIN/MAX comments for explicitly specified ranges but not for calculated ranges.

- Float variables now get Unit comments:

```
Float32 Sal_Out1; /* Unit: <unit> */
```

**Reason** To establish consistency across code patterns and improve readability.

**Moving code into conditionally executed branches**

TargetLink no longer moves variables that are defined in all branches of a conditionally executed control flow, which is contained within a function, into a conditionally executed branch of a following control flow contained in the same function, if this variable is accessed within another function.

This can lead to less efficient code.

**Reason** To prevent incorrect code calculations by conservative optimization.

**ExtendedLifeTimeOptimization**

TargetLink now performs the ExtendedLifeTimeOptimization optimization for formal out parameters of conditionally executed subsystems:

TargetLink ≤ 4.1	TargetLink 4.2
<pre>Foo(Int * out) {   Int B;   B = ... ;   ... //some Code   *out = B; }</pre>	<pre>Foo(Int * out) {   Int B;   B = ... ;   *out = B;   ... //some Code }</pre>

**Reason** Improved optimization and code efficiency.

This optimization also affects access to return variables passed by reference in the `Rte_IWriteRef` and `Rte_IrvIWriteRef` functions of the AUTOSAR RTE.

**Stateflow and casts to boolean**

Boolean casts of non-booleans in Stateflow are now forced to become effectively `boolean` in the generated production code, independent of the coded type of TargetLink's `Bool` basetype:

TargetLink ≤ 4.1	TargetLink 4.2
<pre>F32Var = (Float32) ((Bool) I16Var);</pre>	<pre>F32Var = (Float32) (I16Var != 0);</pre>

**Reason** Improve MISRA C compliance and fix different behavior in MIL and SIL simulation..

**Details** This code change cannot be influenced by the `ForceBooleanOperandsInBooleanOperations` Code Generator option.

### Stateflow state encoding

For states whose Create output for monitoring property is set to `Self activity`, Targetlink now always uses this state's Active State Data to code activity of this state.

If TargetLink uses a boolean active state data for one child of S, it uses boolean state data for every child of S.

**Reason** To increase consistency accross code patterns.

### Inheritance for blocks with multiple outputs

The code of the following blocks changed if they have multiple inputs and their Inherit properties checkbox is selected:

- Switch block
- Multiport Switch block
- MinMax block
- Merge block

This can also affect the code generated for the Bitwise Operator block under the following conditions:

- The block's Use bit mask checkbox is cleared.
- The block does not reference a DD Variable or ReplaceableDataItem object.

The output data type in TargetLink 4.2 is determined in the following way:

- The output data type is the data type of the first block input with valid signal specification provided all inputs with valid signal specification have the same minimum/maximum constraints.
- Otherwise, the output data type is the base data type of the first block input with valid signal specification if at least one input with valid signal specification has a different minimum/maximum constraints.

#### TargetLink ≤ 4.1

```
DataTypeSecondInput Sa1_SwitchScalar /* LSB: 2^0 OFF: 0 MIN/MAX: -100 .. 100 */;
```

#### TargetLink 4.2

```
DataTypeFirstInput Sa1_SwitchScalar; /* LSB: 2^0 OFF: 0 MIN/MAX: -100 .. 100 */
```

**Reason** To increase consistency accross code patterns.

**Further effect** One further effect of this change is that in some cases structure types may not be inherited through multi input blocks. This may affect the code optimizations because structure variables are optimized differently than a collection of plain variables:

TargetLink ≤ 4.1	TargetLink 4.2
<pre> <b>struct StructUDT_tag</b> Sal_SwitchStruct;  /* Switch: Subsystem/SwitchStruct */ if (Sal_CtrlIn &gt;= 0) {      /* Switch: Subsystem/SwitchStruct */     <b>Sal_SwitchStruct.a</b> = <b>Sal_InPortNoUDTWithMinMax</b>;     <b>Sal_SwitchStruct.b</b> = <b>Sal_InPortInt16</b>; } else {      /* Switch: Subsystem/SwitchStruct */     <b>Sal_SwitchStruct</b> = <b>X_Sal_Unit_Delay</b>; } </pre>	<pre> <b>Int16</b> <b>Sal_SwitchStruct</b>; /* LSB: 2^0 OFF: 0 MIN/MAX: -100 .. 100 */ <b>Int16</b> <b>Sal_SwitchStruct_a</b>;  /* Switch: Subsystem/SwitchStruct */ if (Sal_CtrlIn &gt;= 0) {      /* Switch: Subsystem/SwitchStruct */     <b>Sal_SwitchStruct</b> = <b>Sal_InPortNoUDTWithMinMax</b>;     <b>Sal_SwitchStruct_a</b> = <b>Sal_InPortInt16</b>; } else {      /* Switch: Subsystem/SwitchStruct */     <b>Sal_SwitchStruct</b> = <b>X_Sal_Unit_Delay.a</b>;     <b>Sal_SwitchStruct_a</b> = <b>X_Sal_Unit_Delay.b</b>; } </pre>

**Solution** If you want structured variables in production code as in TargetLink 4.1, do one of the following:

- Make sure that the corresponding bus elements at all the inputs of the multi-input block have the same data type (i.e., the first bus element at all the inputs have the same data type, the second bus element at all the inputs have the same data type and so on).
- Explicitly specify the structure in the Data Dictionary and reference the type or variable at each block in the model.

## Other

## Various Migration Aspects

### Requirement information

Requirement information is no longer stored as a visible block property.

Instead, TargetLink blocks now have the `HasRequirementInfos` flag. This lets you obtain a list of blocks that carry requirement information via the following API function:

```
hBlockList = tl_find('<System>', 'HasRequirementInfos', true)
```

To add or remove requirement information, use the `tlRequirementInfo` API function.

**Remapping TargetLink scaling properties during system clearance**

**Migration issue** When TargetLink specifies a scaled variable (LSB != 2^0 or Offset != 0.0) and Simulink specifies an unscaled datatype (for example, one of the built-in datatypes int8, int16, ... , single, double, boolean), TargetLink 4.2 does not set Simulink's datatype to a scaled type, but displays message E03819.

**Solution** If you want to know, which TargetLink scaling property could not be mapped to a Simulink scaling parameter, select the Verbose output checkbox in the Clear System from TargetLink dialog.

In verbose mode, TargetLink displays a message for each property that could not be mapped.

For example:

- Simulink's settings already match
- Simulink specifies property inheritance
- Simulink specifies a bus or enum datatype

**Note**

Big models can result in a huge number of messages.

## Obsolete

**Where to go from here**

Information in this section

<i>Discontinued TargetLink Features</i>	255
<i>Obsolete Limitations</i>	256
<i>Obsolete API Functions</i>	257

## Discontinued TargetLink Features

**Fully built V-ECUs as OSA**

TargetLink no longer builds ECUs as OSA files. TargetLink only exports the generated production code as V-ECU implementations (CTLGZ files) and leaves the platform-specific build process to VEOS for offline simulation and ConfigurationDesk for real-time simulation.

<b>Generation of RTF documents</b>	TargetLink can no longer generate documentation in Rich Text Format (RTF). If you need RTF files, use the HTML generation functionality in combination with third party tools such as Microsoft Word to produce RTF files.
<b>EPS graphics for documentation generation</b>	EPS files are no longer supported for documentation generation ( <code>ImageFormatType t1doc</code> ). You can now use SVGs in HTML and PDF documentation to obtain high-resolution screenshots. Refer to <i>General Enhancements and Changes</i> on page 221.
<b>A2L import</b>	TargetLink's Data Dictionary no longer provides a default import of A2L files into the Subsystem area.

## Obsolete Limitations

With TargetLink 4.2, the following limitations of previous TargetLink versions were removed:

### General limitations

#### Starting MIL simulation via sim command

For MATLAB R2015b (64-bit), starting a MIL simulation by using the Simulink `sim` command from within a function leads to an error. Use TargetLink's `tl_sim` command instead, which is generally recommended.

#### Non-ASCII characters in Windows user name

With the 64-bit version of MATLAB R2014a, it can happen that the TargetLink demos start page is not displayed correctly if your Windows user name contains non-ASCII characters, for example, `ü` in Müller. In such a case, the right-hand pane is empty.

### Block-specific limitations

#### Matrix Concatenate block

TargetLink supports this block only if the Mode block parameter `ist` is set to Multidimensional array.

#### Data Store blocks

For Data Store Memory/Data Store Read/Data Store Write blocks the following limitation applies:

- The number of input and output ports is limited to one.
- *Partial* reading from and writing to data stores is not supported.

#### Data Store Memory block

Data Store Memory blocks whose data type specifies a bus are not supported by TargetLink.



**Code generation limitations**

**No CTC code coverage measurements with the LCC compiler**

The LCC compiler cannot be used with the CTC Testwell third party tool to perform code coverage measurements. However, you can use TargetLink's own code coverage measurement instead.

**AUTOSAR limitations**

**Data type of interrunnable variables**

Interrunnable variables specified in TargetLink must be scalar.

**Stateflow**

**Graphical function with more than one data output**

It is not possible to define and call graphical functions with more than one output data.

**Obsolete API Functions**

Function	Status	Replacement
get_tloptions	Error <sup>1)</sup>	t1_global_options
t1_build_vecu	Error <sup>1)</sup>	-
t1_compile_vecu	Error <sup>1)</sup>	-
t1_vecu_compiler	Error <sup>1)</sup>	-
dsdd('AddCodegenOptions'...)	Error <sup>1)</sup>	dsdd('AddCodegenOptionSet'...)
dsdd('CreateCodegenOptionSetsTLPredefinedOptionset'...)	Error <sup>1)</sup>	dsdd('CreateTLPredefinedOptionSet'...)

<sup>1)</sup> The function was removed from TargetLink.

**Compatibility consideration** Adapt your user scripts and tool chain accordingly.

## Changes in Future TargetLink Versions

### Where to go from here

Information in this section

<i>Features to Be Discontinued</i>	258
<i>API Functions to Be Discontinued</i>	259
<i>Deprecated Code Generator Options</i>	259

## Features to Be Discontinued

### Clean code and Do not log anything

Variables selected for logging cannot be fully optimized. When generating code with the **Global logging option** `Do not log anything` or `Log according to block data`, TargetLink does not fully optimize the code to facilitate testing. That is, the code only differs with regard to the log macros. This contrasts the **Clean code** checkbox on the **Code Generation** page of the TargetLink Main Dialog block, which activates full code optimization in any case.

The special `Do not log anything` behavior will be removed in future TargetLink versions.

### User state flags in Stateflow

It is planned to discontinue support of TargetLink's own user state flags feature in Stateflow in future TargetLink versions because the Stateflow Active State data is similar and more convenient to use.

### MISRA C:2004 Compliance Documentation document

It is planned to discontinue the MISRA C:2004 Compliance Documentation document after TargetLink versions 4.2. TargetLink users will then be advised to use the MISRA C:2012 Compliance Documentation instead.

### Simulink's classic initialization mode

It is planned to discontinue support for Simulink's *classic initialization mode* ( *TargetLink Glossary*) in future TargetLink versions.

### Dynamic components

It is planned to discontinue support for specifying dynamic components for DD Variable objects in future TargetLink versions.

<b>Code generation for special OSEK versions</b>	It is planned to discontinue the code generation for special OSEK versions, such as OsCan in future TargetLink versions.
<b>Signal logging format</b>	It is planned to discontinue the support for Simulink's logging method <code>ModelDataLogs</code> (Signal logging format parameter) in future TargetLink versions.

## API Functions to Be Discontinued

**Discontinued API functions** The following API functions are deprecated and will be removed in future TargetLink versions:

Function	Deprecated Since	Replacement Function
<code>tl_adapt_dd_references</code>	TargetLink 4.0	<code>tlMoveDDObject</code>
<code>tl_extract_subsystem</code>	TargetLink 4.0	<code>tlExtractSubsystem</code>
<code>tl_find_dd_references</code>	TargetLink 4.0	<code>tlFindDDReferences</code>
<code>tl_get_blockset_mode</code>	TargetLink 4.0	<code>tlOperationMode</code>
<code>tl_sim_interface</code>	TargetLink 4.0	<code>tlSimInterface</code>
<code>tl_switch_blockset</code>	TargetLink 4.0	<code>tlOperationMode</code>
<code>tl_upgrade</code>	TargetLink 4.0	<code>tlUpgrade</code>


### Note

See the help contents on the new API functions to adjust your user scripts accordingly.

## Deprecated Code Generator Options

The following Code Generator options are deprecated and will be removed in future TargetLink versions:

- `SideEffectFreeAnalysisThreshold` ([TargetLink Block and Object Reference](#))
- `TreatAllForcedAtomicSubsystemsAsWeakAtomic` ([TargetLink Block and Object Reference](#))

- *DisableFunctionsAsAnalysisBoundaries* ( *TargetLink Block and Object Reference*)
- *CreateRestartFunctions* ( *TargetLink Block and Object Reference*)

# VEOS

---

## Where to go from here

Information in this section

<i>New Features of VEOS 3.7</i>	261
<i>Compatibility of VEOS 3.7</i>	264
<i>Migrating to VEOS 3.7</i>	266
<i>Discontinuations in VEOS</i>	269

---

## New Features of VEOS 3.7

---

### Information in this topic

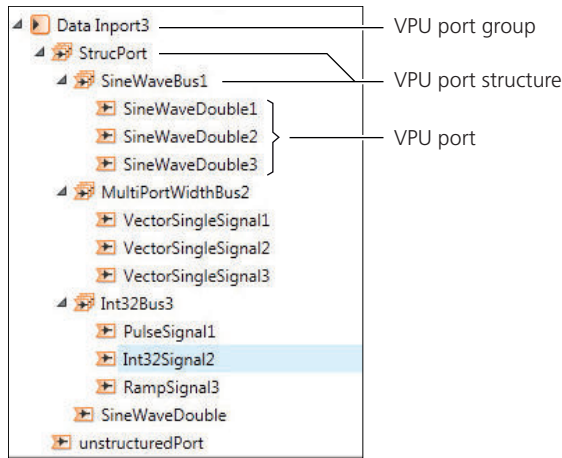
<i>Hierarchical display of VPU ports</i> on page 261
<i>Connecting VPU ports by dragging structural elements</i> on page 262
<i>Enabling/disabling logging of FMU-internal messages</i> on page 263
<i>Getting version information on VEOS Player via automation</i> on page 263
<i>Message Viewer and dSPACE Log Viewer</i> on page 263
<i>Message Viewer</i> on page 263
<i>dSPACE Log Viewer</i> on page 264

---

### Hierarchical display of VPU ports

The VEOS Player now displays the VPU ports of a VPU *hierarchically*, i.e., together with VPU port groups and VPU port structures. VPU port groups and VPU port structures result from elements of the underlying model.

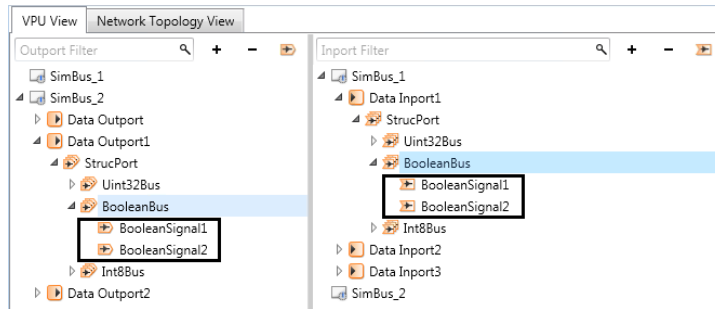
The following illustration shows the hierarchical display of VPU ports of an offline simulation application as an example.



**Connecting VPU ports by dragging structural elements**

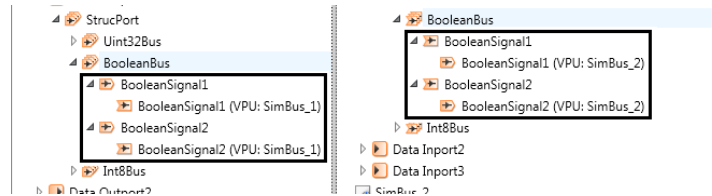
The VEOS Player lets you connect VPU ports by dragging one VPU port structure to another. The VEOS Player then automatically connects all the VPU ports of the first structure to VPU ports of the second structure if the ports have the same name.

The following illustration shows an example:



When you drag the BooleanBus VPU port structure from one tree view to the BooleanBus VPU port structure in the other tree view, the VEOS Player connects the two VPU ports.

The following illustration shows the connected VPU ports after drag & drop.



### Enabling/disabling logging of FMU-internal messages

You can enable/disable the logging of FMU-internal messages via compiler options when you import an FMU to the VEOS Player. Refer to *Import* ([VEOS Player Reference](#)).

### Getting version information on VEOS Player via automation

The `IVersionInformation` interface lets you get detailed information on the VEOS Player version including the major, minor, and maintenance version.

#### Note

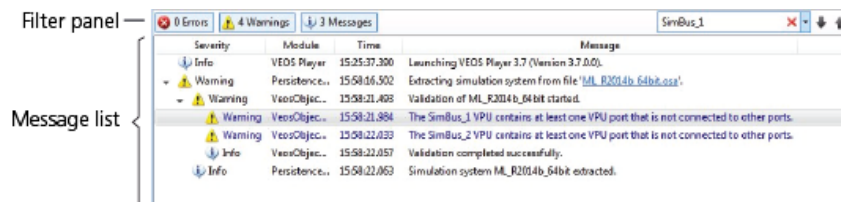
To get version information, it is recommended to no longer use the `Version` property of the `IApplication` interface.

Refer to *Automating the Start of the VEOS Player, and Getting Version Information* ([VEOS Guide](#)) and *IVersionInformation* ([VEOS Player API Reference](#)).

### Message Viewer and dSPACE Log Viewer

**Message Viewer** VEOS Player supports the new Message Viewer. The Message Viewer provides a history of all the info, advice, error and warning messages, and all the questions that occur when you work with the product. This helps you check the system state.

The Message Viewer looks like this:



Refer to *Message Viewer* ([VEOS Player Reference](#)).

**dSPACE Log Viewer** VEOS Player supports the new dSPACE Log Viewer.

The dSPACE Log is a collection of errors, warnings, information, questions, and advice issued by all dSPACE products and connected systems over more than one session.

To forward the dSPACE Log to dSPACE Support, use the dSPACE Installation Manager.

Refer to *dSPACE Log Viewer* (📖 *VEOS Player Reference*).

## Compatibility of VEOS 3.7

### Information in this topic

<i>Compatibility overview</i> on page 264
<i>Compatibility in general</i> on page 264
<i>CTLGZ compatibility</i> on page 264
<i>SIC compatibility</i> on page 265
<i>BSC compatibility</i> on page 265
<i>FMU compatibility</i> on page 265
<i>OSA compatibility</i> on page 265
<i>Real-Time Testing compatibility</i> on page 266

### Compatibility overview

**Compatibility in general** dSPACE recommends using only software products from the same dSPACE Release. This provides maximum run-time compatibility.

**CTLGZ compatibility** The following table shows the compatibility between VEOS 3.7 and CTLGZ files (V-ECU implementations):

V-ECU Implementations Created with Products of...	V-ECU Implementation Version
dSPACE Release 2016-B: ■ SystemDesk 4.7 ■ TargetLink 4.2	2.4.1
dSPACE Release 2016-A: ■ SystemDesk 4.6	2.4
dSPACE Release 2015-B: ■ SystemDesk 4.5 ■ TargetLink 4.1	2.3
dSPACE Release 2015-A: ■ SystemDesk 4.4	2.2
dSPACE Release 2014-B: ■ SystemDesk 4.3 ■ TargetLink 4.0	2.1



V-ECU Implementations Created with Products of...	V-ECU Implementation Version
dSPACE Release 2013-B and earlier: <ul style="list-style-type: none"> <li>■ SystemDesk 3.2</li> <li>■ TargetLink 3.5</li> </ul>	1.0

**SIC compatibility** VEOS 3.7 is compatible with Simulink implementation container (SIC) files created with the Model Interface Package for Simulink 3.3 of dSPACE Release 2016-B (SIC version 1.2).

**BSC compatibility** VEOS 3.7 is compatible with bus simulation container (BSC) files created with the Bus Manager of dSPACE Release 2016-B (BSC version 1.1).

**FMU compatibility** VEOS supports only the FMI for Co-Simulation interface, but not the FMI for Model Exchange interface.

For detailed and up-to-date compatibility information on FMI support in VEOS, refer to:

<http://www.dspace.com/go/FMI-Compatibility>.

**OSA compatibility** The following table shows the compatibility between VEOS 3.7 and offline simulation application (OSA) files:

**Note**

When you migrate from one VEOS version to another, it is recommended to rebuild the binary OSA files from existing model implementation container (CTLGZ, SIC, BSC, FMU) files. This is due to the limited compatibility of OSA files.

OSA Files Created with Products of ...	OSA Version
dSPACE Release 2016-B	3.7
dSPACE Release 2016-A	3.6 <sup>1)</sup>
dSPACE Release 2015-B or earlier	3.5 or earlier <sup>2)</sup>

<sup>1)</sup> OSA files created or modified with VEOS 3.6 can be loaded in VEOS 3.7, but all the VPU properties are read-only and cannot be modified. Port and network connections can be edited.

<sup>2)</sup> OSA files created or modified with VEOS 3.5 or earlier can be loaded and simulated in VEOS 3.7 if they do not contain bus communication elements. VPU properties are read-only and cannot be modified. Port and network connections can be edited.

**Note**

OSA files created or modified with VEOS 3.7 cannot be loaded in earlier VEOS versions.

**Real-Time Testing compatibility** To use RTT in connection with VEOS and ControlDesk, the Real-Time Testing (RTT) version used by the VEOS Simulator running the simulation system and the RTT version active on the host PC must be identical.

The table below shows the VEOS Simulator version and the corresponding RTT version:



VEOS Simulator	Real-Time Testing Version
... from VEOS 3.7	Real-Time Testing Version 3.1
... from VEOS 3.6	Real-Time Testing Version 3.0
... from VEOS 3.5	Real-Time Testing Version 2.6
... from VEOS 3.4	Real-Time Testing Version 2.5
... from VEOS 3.3	Real-Time Testing Version 2.4
... from VEOS 3.2	Real-Time Testing Version 2.3
... from VEOS 3.1	Real-Time Testing Version 2.2
... from VEOS 3.0	Real-Time Testing Version 2.0

ControlDesk 6.0 automatically uses the VEOS Simulator from VEOS 3.7. You can therefore use RTT in connection with VEOS and ControlDesk if RTT 3.1 is active on the host PC.

## Migrating to VEOS 3.7

### Display and connectability of array VPU ports

The display and connectability of array VPU ports has changed from VEOS 3.6 to VEOS 3.7.

Up to and Including VEOS 3.6	As of VEOS 3.7
<b>Display</b>	
Array VPU ports and their elements were displayed in the VEOS Player with the  icon.	Array VPU ports (and scalar VPU ports) are displayed in the VEOS Player with the  icon. Individual elements of an array VPU port are no longer displayed. Instead, the number of elements is displayed as the array VPU port's Data width property. To specify initial values for the individual elements of an array VPU port, enter a comma-separated list in the Initial value property edit field of the port.

Up to and Including VEOS 3.6	As of VEOS 3.7
<b>Connectability</b>	
<ul style="list-style-type: none"> <li>■ Connecting an entire array VPU port was possible.</li> <li>■ Connecting the elements of an array VPU port individually was possible.</li> </ul>	<ul style="list-style-type: none"> <li>■ Connecting an entire array VPU port is possible.</li> <li>■ Connecting the elements of an array VPU port individually is not possible.<sup>1)</sup></li> </ul>

<sup>1)</sup> In VEOS 3.7, when you open an offline simulation application last saved with VEOS 3.6 or earlier, the connections between individual elements of array VPU ports are kept.

**VEOS Player automation changes**

You have to consider the following changes to the automation interface of the VEOS Player in VEOS 3.7.

Up to and Including VEOS 3.6	As of VEOS 3.7
Accessing VPU ports was possible via the <code>VpuPorts</code> property of the <code>IVpu</code> interface.	<p>The <code>VpuPorts</code> property of the <code>IVpu</code> interface lets you access only the VPU ports that are on the root level of a VPU. Accessing VPU ports on deeper levels is not possible in such a way. Instead, you can use one of the following approaches:</p> <ul style="list-style-type: none"> <li>■ The <code>IVpuPorts</code> and the <code>IVpuPortGroups</code> interfaces provide an <code>Item</code> method that lets you return the underlying <code>IVpuPort</code> and <code>IVpuPortGroup</code> interfaces.</li> <li>■ As an alternative, the <code>IVpu</code> and the <code>IVpuPortGroup</code> interfaces provide an <code>Item</code> method that lets you return underlying generic <code>IVpuItem</code> interfaces. Via the <code>VpuItemType</code> property of the <code>IVpuItem</code> interface, you can get the item type, either a VPU port or a VPU port group. This change is due to the hierarchical display of VPU ports supported as of VEOS 3.7. Refer to <code>IVpu</code> (<a href="#">VEOS Player API Reference</a>) and <code>IVpuPortGroup</code> (<a href="#">VEOS Player API Reference</a>).</li> </ul>
The type of the <code>SendsTo</code> property of the <code>IVpuPort</code> interface was <code>System.String[]</code> .	<p>The type of the <code>SendsTo</code> property of the <code>IVpuPort</code> interface is <code>IVpuPorts</code>. This change is due to the hierarchical display of VPU ports supported as of VEOS 3.7.</p>
The type of the <code>ReceivesFrom</code> property of the <code>IVpuPort</code> interface was <code>String</code> .	<p>The type of the <code>ReceivesFrom</code> property of the <code>IVpuPort</code> interface is <code>IVpuPort</code>. This change is due to the hierarchical display of VPU ports supported as of VEOS 3.7.</p>

Up to and Including VEOS 3.6	As of VEOS 3.7
Accessing individual elements of an array VPU port was possible via the <code>DataElements</code> property of the <code>IVpuPort</code> interface.	Accessing individual elements of an array VPU port is no longer possible. As a consequence, the <code>DataElements</code> property of the <code>IVpuPort</code> interface is no longer available. Instead, the number of elements of an array VPU port is available via the <code>DataWidth</code> property of the <code>IVpuPort</code> interface.
To get version information, the <code>Version</code> property of the <code>IApplication</code> interface could be used.	<p>To get version information, the <code>IVersionInformation</code> interface can be used.</p> <div style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> <p><b>Note</b></p> <p>The <code>Version</code> property of the <code>IApplication</code> interface is still supported for compatibility reasons. However, it will be discontinued in the future.</p> </div>

### Migrating ASM models

You cannot simulate an ASM model on VEOS 3.7 (dSPACE Release 2016-B) if the model is contained in an OSA or SIC file created with a previous dSPACE Release.

On VEOS 3.7, to simulate an ASM model that was last saved with a dSPACE Release earlier than Release 2016-B on VEOS 3.7, perform the following steps:

1. Migrate the ASM model to dSPACE Release 2016-B.  
For information on migrating ASM models, refer to *Migrating ASM Models* ([📖 ASM User Guide](#)).
2. Generate a Simulink implementation container (SIC) file on the basis of the ASM model by using the *Model Interface Package for Simulink*.  
For instructions, refer to *Generating Simulink Implementation Containers* ([📖 Model Interface Package for Simulink - Modeling Guide](#)).
3. Import the SIC file to the VEOS Player of VEOS 3.7.  
For instructions, refer to *How to Import Simulink Implementations* ([📖 VEOS Guide](#)).

## Discontinuations in VEOS

---

**Discontinuations for VEOS  
as of dSPACE  
Release 2017-A**

**Discontinuation of processor-in-the-loop (PIL) simulation with VEOS and SystemDesk** As of dSPACE Release 2017-A, VEOS and SystemDesk will no longer support processor-in-the-loop (PIL) simulation. This includes the generation of V-ECUs for PIL simulation and the simulation of these V-ECUs on evaluation boards.



---

# Compatibility Information

---

## Where to go from here

Information in this section

<i>Supported MATLAB Releases</i>	271
<i>Operating System</i>	273
<i>Notes on 64-bit Compatibility of dSPACE Products</i>	275
<i>Run-Time Compatibility of dSPACE Software</i>	276
<i>Limitations for Using Windows Features</i>	277

---

## Supported MATLAB Releases

### MATLAB®

Working with various dSPACE products requires that you have installed MATLAB.

#### Tip

For system requirements of MathWorks® software, refer to [http://www.mathworks.com/support/sysreq/current\\_release](http://www.mathworks.com/support/sysreq/current_release).

MATLAB Release...	...Is Supported by dSPACE Release 2016-B					
	RCP and HIL Software	AutomationDesk 5.3 <sup>1)</sup>	TargetLink 4.2	Model Compare 2.7	dSPACE Python Extensions 2.2 <sup>2)</sup>	XIL API .NET MAPort 2016-B
R2016b (64-bit)	✓ <sup>3)</sup>	✓	✓	✓	✓	✓
R2016a (64-bit)	✓ <sup>3)</sup>	✓	✓	✓	✓	✓
R2015b (64-bit)	✓	✓	✓	✓	✓	✓
R2015a SP1 (64-bit)	✓	✓	✓	✓	✓	✓

- <sup>1)</sup> AutomationDesk's MATLAB Access library requires MATLAB.
- <sup>2)</sup> matlablib2 of dSPACE Python Extensions requires MATLAB.
- <sup>3)</sup> R2016a and R2016b are not supported by the RTI FPGA Programming Blockset – FPGA Interface.

**Note**

As of dSPACE Release 2016-A, dSPACE software supports only 64-bit MATLAB variants. 32-bit MATLAB variants are no longer supported.

For up-to-date information on additional MATLAB releases that can be used in combination with dSPACE software, refer to <http://www.dspace.com/go/MATLABCompatibility>.

**RCP and HIL software: C-compiler for building MEX files** Note that the RCP and HIL software supports only Microsoft Windows SDK 7.1 to build MEX functions.

This compiler is a free download from Microsoft. The compiler additionally requires the .NET framework 4.0, which is also available at no charge from Microsoft. For information on the supported compilers, refer to <http://www.mathworks.com/support/compilers/R2016a/index.html>.

You need to install this compiler and configure it as a MEX compiler in MATLAB if you intend to use RCP and HIL products that require a MEX compiler, such as RTI CAN MultiMessage Blockset, RTI LIN MultiMessage Blockset, or Automotive Simulation Models.



**Restricted MAT file support** The Signal Editor of ControlDesk (ControlDesk 6.0) only supports reading and writing MAT files of file format version 5.0. MAT files of this version can be created in MATLAB by using the `save` command with the option `'-v6'`.

**Limitations for ModelDesk** When you use MATLAB R2016a for Simulink simulation and the simulation runs, a download requires at least one minute.

There is no limitation when the simulation is stopped or paused.

## Operating System

### Operating system on host PC

The dSPACE products of dSPACE Release 2016-B support the following operating systems:

- Windows 7 Professional, Ultimate, and Enterprise with Service Pack 1 (64-bit versions)

Only the listed editions are supported. The Windows 7 Home and Starter editions are not supported.

- Windows 10 support of dSPACE products:

With dSPACE Release 2016-B, only TargetLink and Model Compare support Windows 10 with the following editions and servicing options:

- Windows 10 Professional, Education and Enterprise (64-bit versions)

The Windows 10 Home and Mobile editions are not supported.

- Long-Term Servicing Branch: LTSB 2016
- Current Branch for Business: CBB 1511

Support of Windows 10 for the complete dSPACE Release 2016-B will be granted by means of a service pack available in mid-2017. As of dSPACE Release 2017-A, dSPACE products will generally support Windows 10.

Some limitations apply when you use Windows together with dSPACE software. Refer to *Limitations for Using Windows Features* on page 277.

### Note

As of dSPACE Release 2016-A, dSPACE software only supports 64-bit operating systems. 32-bit operating systems are not supported any longer.

---

#### Using MicroAutobox Embedded PC as host PC:

ControlDesk can also be installed on the MicroAutoBox Embedded PC (with Intel® Core™ i7-3517UE Processor) running under Windows 7 Professional, Ultimate, and Enterprise, 64-bit version.

---

#### Allowing communication via additional firewall rules

Additional Windows firewall rules are installed during the installation of various dSPACE software products. For example, one rule allows communication with a dSPACE expansion box such as AutoBox. Another rule allows MotionDesk to receive motion data from a network channel. These example rules are created by the following commands:

- ```
netsh advfirewall firewall add rule name="dSPACE Net Service"
service=any dir=in action=allow profile=any
protocol=icmpv4:0, any description="Allow the dSPACE Net Service to connect to a dSPACE expansion box via network."
```
- ```
netsh advfirewall firewall add rule name="dSPACE MotionDesk"
program="%dSPACE_root%\MotionDesk\Bin\MotionDesk.exe"
dir=in action=allow profile=any description="Allow dSPACE MotionDesk to receive motion data via network."
```

If you are running third-party firewall software on your host PC, ensure that the TCP/IP communication of dSPACE software is not blocked.

---

#### Operating system on dSPACE License Server

If you purchased floating network licenses, you have to install and configure one of the networked PCs as the dSPACE License Server. The operating system of the dSPACE License Server must be one of the following:

- Windows Vista Business, Ultimate, or Enterprise (64-bit version) with the latest Service Pack

- Windows 7 Professional, Ultimate, or Enterprise (64-bit version) with the latest Service Pack
- Windows Server 2008 R2
- Windows Server 2012, Windows Server 2012 R2

**Note**

The dSPACE License Server does not support non-Windows operating systems.

## Notes on 64-bit Compatibility of dSPACE Products

**Notes**

As of dSPACE Release 2016-B, all products are available only as 64-bit variants. As a consequence, dSPACE Release 2016-B and later supports only the 64-bit variants of the following software:

- Windows operating systems
- MATLAB
- Python

In general, 64-bit technology lets you handle more complex models and layouts as well as larger amounts of data than 32-bit technology.

However, keep in mind the following points:

- Product extensions, e.g., ConfigurationDesk custom I/O function blocks, must be available in 64-bit.
- Python:
  - No support of 32-bit third-party extensions in the 64-bit Python installation from dSPACE.
  - No support of 64-bit dSPACE Python extensions in a parallel 32-bit Python installation.
- For some products migration tasks are necessary. For instructions, refer to the product-specific chapters in the *New Features and Migration Guide* or to the user documentation of the relevant dSPACE product.

## Run-Time Compatibility of dSPACE Software

---

### Definition

Run-time compatibility means that:

- dSPACE products can be used in parallel after software installation, even if they are installed in different folders.
- dSPACE products without interaction can run independently of each other.

---

### Compatibility of products in dSPACE Release 2016-B

dSPACE recommends using only software products from the same dSPACE Release. This provides maximum run-time compatibility.

Note that:

- Limitations regarding run-time compatibility in the dSPACE tool chain might occur if products from different dSPACE Releases are mixed.

If dSPACE products interact directly (through automation interfaces) or indirectly (through common file types like A2L), limitations might apply. For minor limitations, refer to the relevant product documentation. The major limitations are described in the following.

In rare cases, an additional patch must be installed for a product to achieve run-time compatibility. For more information on the patch and whether a patch is necessary, refer to <http://www.dspace.com/go/CompPatch>.

- RCP and HIL software products (on Release 2016-B) cannot be used in combination with RCP and HIL software products from earlier dSPACE releases.

**Major limitation for working with a SCALEXIO system** The products for working with a SCALEXIO system must be compatible. This is guaranteed only for products delivered with the same dSPACE Release. Contact dSPACE for more information if you have any questions.

**Compatibility of real-time applications loaded to a DS1005, DS1006, DS1103, DS1104 or MicroAutoBox platform** If a real-time application is loaded to one of the related platforms with a software product of dSPACE Release 2016-B, software products of dSPACE Release 2016-A do not detect that the loaded real-time application is the same as the real-time application stored on your host PC.

This also applies if you load a real-time application with a software product of dSPACE Release 2016-A and use software products of dSPACE Release 2016-B, for example, for experimenting.

**Combining dSPACE products from earlier releases**

For more information and notes on the combined use of different products from and with earlier releases, refer to [http://www.dspace.com/go/ds\\_sw\\_combi](http://www.dspace.com/go/ds_sw_combi).

## Limitations for Using Windows Features

**Objective**

Some limitations apply using dSPACE software in conjunction with features of Windows.

**Fast user switching not supported**

dSPACE software does not support the fast user switching feature of Windows.

**Closing dSPACE software before PC shutdown**

The shutdown procedure of Windows operating systems might cause some required processes to be aborted although they are still being used by dSPACE software. To avoid data loss, it is recommended to terminate the dSPACE software manually before performing a PC shutdown.

**User Account Control**

It is recommended to disable Windows' User Account Control (UAC) during the installation of dSPACE software. If you cannot disable UAC, note the following Windows behavior: If UAC is enabled, the setup programs run with the administrator account instead of the user account. Therefore, it is important that the administrator account has access to the required drives, particularly the required network drives.

**USB devices**

The first time that dSPACE USB devices using cables with optoisolation are connected to the PC, there might be a message that the device driver software was not installed successfully. The dSPACE device will nevertheless work properly later on.

**Long path names**

dSPACE software does not support the long path name syntax of the Windows API. If a pathname is directly or indirectly used which exceeds 260 characters the behavior of the dSPACE software is not defined.

### Windows' 8dot3name creation option must be enabled

#### Note

It is strongly recommended that Windows' 8dot3name creation option is enabled for all drives (drives used for installation and drives used for working) before you install third-party software (such as MATLAB®/Simulink®) and the dSPACE software.

If the option is disabled during software installation, serious errors can occur when running the dSPACE software, for example, the build process might be aborted. To repair an installation that has been installed with the disabled 8dot3name creation option, you have to reinstall the dSPACE software and the required third-party software. Using the dSPACE Maintenance Setup does not solve this problem.

For instructions on checking the setting and enabling the option, refer to <http://www.dspace.com/faq?346> or to the Microsoft Windows documentation.

**Numerics**

64-bit compatibility 275

**A**

ASM Base InCylinder Blockset  
migration 58  
new features 58

ASM blocksets  
migration 57

ASM Brake Hydraulics Blockset  
migration 60

ASM Diesel Engine Blockset  
migration 65  
new features 63

ASM Diesel Exhaust Blockset  
migration 69

ASM Diesel InCylinder Blockset  
migration 71

ASM Drivetrain Basic Blockset  
migration 74  
new features 73

ASM Electric Components Blockset  
migration 78  
new features 77

ASM Engine Gasoline Basic Blockset  
migration 83  
new features 82

ASM Engine Gasoline Blockset  
migration 87  
new features 85

ASM Environment Blockset  
migration 80  
new features 80

ASM Gasoline InCylinder Blockset  
migration 91

ASM Optimizer  
new features 93

ASM Pneumatics Blockset  
migration 94

ASM Traffic Blockset  
migration 95  
new features 95

ASM Trailer Blockset  
migration 98  
new features 97

ASM Truck Blockset  
migration 102  
new features 101

ASM Turbocharger Blockset  
migration 104  
new features 104

ASM Vehicle Dynamics Blockset  
migration 108  
new features 107

AutomationDesk  
migration 50  
new features 47

AUTOSAR  
TargetLink-related  
migration 238

**B**

Bus Manager (stand-alone)  
new features 113

**C**

Common Program Data folder 12  
CommonProgramDataFolder 12

ControlDesk  
migration 136  
new features 122

**D**

DCI Configuration Tool  
new features 145

discontinuation  
hardware 21  
software 20

Documents folder 12  
DocumentsFolder 12

DS1005 PPC Board  
discontinuation 21

DS1103 PPC Controller Board  
discontinuation 21

dSPACE ECU Flash Programming Tool  
new features 149

dSPACE FlexRay Configuration Package  
new features 151

dSPACE Python Extensions  
migration 153  
new features 153

dSPACE XIL API  
migration 157  
new features 155

DVD contents 17

**E**

ECU Interface Manager  
migration 159

**F**

Firmware Manager  
new features 161

**G**

general enhancements and changes 15

**H**

host PC software  
MATLAB 271  
operating system 273

**K**

key features 26

**L**

Limitations  
TargetLink

obsolete limitations 256  
limitations for using Windows features  
277

Local Program Data folder 12  
LocalProgramDataFolder 12

**M**

MATLAB  
requirements 271  
startup 168, 177, 228  
supported releases 271

MicroAutoBox  
new features 175

MicroAutoBox II 1401/1511/1512  
discontinuation 21

MicroAutoBox II 1401/1512/1513  
discontinuation 21

migration  
ASM Base InCylinder Blockset 58  
ASM blocksets 57  
ASM Brake Hydraulics Blockset 60  
ASM Diesel Engine Blockset 65  
ASM Diesel Exhaust Blockset 69  
ASM Diesel InCylinder Blockset 71  
ASM Drivetrain Basic Blockset 74  
ASM Electric Components Blockset 78  
ASM Engine Gasoline Basic Blockset 83  
ASM Engine Gasoline Blockset 87  
ASM Environment Blockset 80  
ASM Gasoline InCylinder Blockset 91  
ASM Pneumatics Blockset 94  
ASM Traffic Blockset 95  
ASM Trailer Blockset 98  
ASM Truck Blockset 102  
ASM Turbocharger Blockset 104  
ASM Vehicle Dynamics Blockset 108

AutomationDesk 50  
ControlDesk 136  
dSPACE Python Extensions 153  
dSPACE XIL API 157  
ECU Interface Manager 159  
Model Compare 164  
ModelDesk 166  
MotionDesk 172  
Real-Time Testing 174  
RTI 177  
RTI Bypass Blockset 180  
RTI CAN MultiMessage Blockset 181  
RTI FPGA Programming Blockset 184  
RTI LIN MultiMessage Blockset 187

Model Compare  
migration 164  
new features 163

ModelDesk  
migration 166  
new features 165

MotionDesk  
migration 172  
new features 171

**N**  
new features

- ASM Base InCylinder Blockset 58
  - ASM Diesel Engine Blockset 63
  - ASM Drivetrain Basic Blockset 73
  - ASM Electric Components Blockset 77
  - ASM Engine Gasoline Basic Blockset 82
  - ASM Engine Gasoline Blockset 85
  - ASM Environment Blockset 80
  - ASM Optimizer 93
  - ASM Traffic Blockset 95
  - ASM Trailer Blockset 97
  - ASM Truck Blockset 101
  - ASM Turbocharger Blockset 104
  - ASM Vehicle Dynamics Blockset 107
  - AutomationDesk 47
  - Bus Manager (stand-alone) 113
  - ControlDesk 122
  - DCI Configuration Tool 145
  - dSPACE ECU Flash Programming Tool 149
  - dSPACE FlexRay Configuration Package 151
  - dSPACE Python Extensions 153
  - dSPACE XIL API 155
  - Firmware Manager 161
  - MicroAutoBox 175
  - Model Compare 163
  - ModelDesk 165
  - MotionDesk 171
  - Real-Time Testing 173
  - RTI Bypass Blockset 179
  - RTI CAN MultiMessage Blockset 181
  - RTI FPGA Programming Blockset 183
  - RTI LIN MultiMessage Blockset 187
  - RTI Watchdog Blockset 189
  - RTI/RTI-MP 175
  - RTLib 175
  - SCALEXIO firmware 191
  - SystemDesk 194
  - VEOS 261
  - new MATLAB features (R2016b)
    - RTI/RTI-MP 175
- P**
- planned discontinuation
    - software 21
  - product overview 22
- R**
- RCP and HIL software
    - definition 17
  - Real-Time Testing
    - migration 174
    - new features 173
  - requirements
    - host PC software
      - MATLAB 271
    - operating system 273
  - RTI Bypass Blockset
    - migration 180
    - new features 179
  - RTI CAN MultiMessage Blockset
    - migration 181
    - new features 181
  - RTI FPGA Programming Blockset
    - migration 184
    - new features 183
  - RTI LIN MultiMessage Blockset
    - migration 187
    - new features 187
  - RTI Watchdog Blockset
    - new features 189
  - RTI/RTI-MP
    - new features 175
    - new MATLAB features (R2016b) 175
  - RTLib
    - new features 175
- S**
- SCALEXIO firmware
    - new features 191
  - supported MATLAB releases 271
  - system requirements
    - operating system 273
  - SystemDesk
    - new features 194
- T**
- TargetLink
    - API commands
      - changes 236
    - AUTOSAR features, new
      - supported releases 212
    - code changes
      - migration 242
    - code efficiency, improved 207
    - code generator options
      - backward compatibility 234
      - changed default value 234
    - custom code, improved 211
    - discontinued features 255
    - enumeration, improved 211
    - migrating to
      - new version 227
    - migration
      - AUTOSAR-related 238
      - code changes 242
      - obsolete limitations 256
      - various aspects 254
    - new API functions 225
    - new code generator options 218
    - new features 202
      - general changes 221
      - general enhancements 221
    - new version
      - migrating to 227
    - newly supported Simulink blocks 203
    - target support
      - discontinued compiler versions 216
      - discontinued evaluation boards 216
      - new compiler versions 216
      - new evaluation boards 216
      - supported targets 216
  - TargetLink Data Dictionary
    - API commands
      - changes 236
- migrating to
  - new version 227
- migration 228
  - discontinued documentation 228
  - manually upgrading libraries and models 232
  - upgrading existing data dictionaries 230
- new features 202
- new version
  - migrating to 227
- U**
- user documentation
    - improvements 18
    - printed documents 20
    - restrictions 19
- V**
- VEOS
    - new features 261
    - version history 22
- W**
- Windows
    - limitations 277