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: http://www.dspace.com
http://www.caldesk.com

How to Contact dSPACE Support

There are different ways to contact dSPACE Support:
• Visit our Web site at http://www.dspace.com/goto?support
• Send an e-mail or phone:
  • General Technical Support:
support@dspace.de
+49 5251 1638-941
  • SystemDesk Support:
support.systemdesk@dspace.de
+49 5251 1638-996
  • CalDesk Support:
support.caldesk@dspace.de
+49 5251 1638-363
  • TargetLink Support:
support.tl@dspace.de
+49 5251 1638-700
• Use the dSPACE Support Wizard:
  • On your dSPACE DVD at \Diag\Tools\dSPACESupportWizard.exe
  • Via Start – Programs – dSPACE Tools (after installation of the dSPACE software)
  • At http://www.dspace.com/goto?supportwizard

You can always find the latest version of the dSPACE Support Wizard here.
dSPACE recommends that you use the dSPACE Support Wizard to contact dSPACE Support.

Important Notice

This document contains proprietary information that is protected by copyright. All rights are reserved. Neither the documentation nor software may 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.

© Copyright 2000 - 2009 by:
dSPACE GmbH
Rathenaustraße 26
33102 Paderborn
Germany

This publication and the contents hereof are subject to change without notice.
AutomationDesk, CalDesk, ConfigurationDesk, ControlDesk, SystemDesk and TargetLink 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

About This Document

Overview of dSPACE Release 6.5
- General Enhancements and Changes ........................................ 9
- Product Version Overview ...................................................... 10
- New Product Key Features .................................................... 11
- Migrating to dSPACE Release 6.5 ........................................... 14

AutomationDesk
- New Features of AutomationDesk 3.0 ....................................... 15
- Migrating to AutomationDesk 3.0 ............................................ 17

Automotive Simulation Models (ASM)
- All ASM Blocksets .................................................................. 20
  - New Features of MotionDesk ASM Demo Experiments and
    Blockset .................................................................................. 20
  - Migration of MotionDesk ASM Demo Experiments and
    Blockset .................................................................................. 20
- ASM Base InCylinder Blockset .................................................. 21
  - Migrating to ASM Base InCylinder Blockset 1.0.1 .................. 21
- ASM Brake Hydraulics Blockset .............................................. 22
  - New Features of ASM Brake Hydraulics Blockset 1.2 ........... 22
  - Migrating to ASM Brake Hydraulics Blockset 1.2 ................ 23
- ASM Diesel Exhaust Blockset .................................................. 24
  - ASM Diesel Exhaust Blockset 1.1 ........................................... 24
  - Migrating to ASM Diesel Exhaust Blockset 1.0.1 ................. 25
- ASM Drivetrain Basic Blockset ................................................ 27
  - New Features of ASM Drivetrain Basic Blockset 1.3 ............ 27
  - Migrating to ASM Drivetrain Basic Blockset 1.3 ................. 27
- ASM Electric Components Blockset ........................................ 29
  - New Features of ASM Electric Components Blockset 1.1 .... 29
  - Migrating to ASM Electric Components Blockset 1.1 .......... 29
- ASM Engine Diesel Blockset ................................................... 30
  - Migrating to ASM Engine Diesel Blockset 1.3.3 ................. 30
## Contents

ASM Engine Gasoline Basic Blockset ................................................................. 32
  Migrating to ASM Engine Gasoline Basic Blockset 1.3.3 ......................... 32
ASM Engine Gasoline Blockset ........................................................................... 34
  New Features of ASM Engine Gasoline Blockset 2.1.3 ............................. 34
  Migrating to ASM Engine Gasoline Blockset 2.1.3 .................................. 34
ASM Environment Blockset .................................................................................. 36
  New Features of ASM Environment Blockset 1.4.1 ................................... 36
  Migrating to ASM Environment Blockset 1.4.1 ...................................... 36
ASM Optimizer ................................................................................................. 37
  New Features of ASM Optimizer 1.2 ......................................................... 37
ASM Truck Blockset ......................................................................................... 38
  New Blockset: ASM Truck 1.0 ................................................................. 38
ASM Vehicle Dynamics Blockset ....................................................................... 39
  New Features of ASM Vehicle Dynamics Blockset 1.4 .............................. 39
  Migrating to ASM Vehicle Dynamics Blockset 1.3 ................................. 40

**ConfigurationDesk**

ConfigurationDesk Features .............................................................................. 43
  New Features of ConfigurationDesk 3.0.2 .................................................. 44
Migrating to ConfigurationDesk 3.0.1 ............................................................. 45
  Migration to ConfigurationDesk 3.0.1 ....................................................... 45
  How to Migrate a ConfigurationDesk Project with the RapidPro Projects Migrator ............................. 47

**ControlDesk**

New Features of ControlDesk 3.5 ................................................................. 51

**ModelDesk**

New Features of ModelDesk 2.2.1 ................................................................. 53

**MotionDesk**

New Features of MotionDesk 2.1.4 ................................................................. 55
  Migration to MotionDesk 2.1.4 ................................................................. 56

**RTI and RTLib**

New Features of RTI/RTI-MP and RTLib ......................................................... 57

**RTI AUTOSAR Package**

New Features of the RTI AUTOSAR Package 1.1 ......................................... 59
**Contents**

**RTI LIN MultiMessage Blockset** 61
- New Features of the RTI LIN MultiMessage Blockset 1.7.1...61
- Migrating to RTI LIN MultiMessage Blockset 1.7.1 ..........62

**RTI FPGA Programming Blockset** 63
- New Features of the RTI FPGA Programming Blockset 1.1....63
- Migrating to RTI FPGA Programming Blockset 1.1 ..........65

**SystemDesk** 67
- New Features of SystemDesk 2.1 ................................67
- Migrating to SystemDesk 2.1 ...................................69

**TargetLink** 71
- New Features of TargetLink 3.1 and dSPACE Data Dictionary 2.0 ..72
  - New Production Code Generation Features .................72
  - Code Generation from the dSPACE Data Dictionary ..........73
  - New TargetLink Bit Operation Blocks .......................77
  - Enhancements to the Target Simulation Module .............78
  - Requirements Traceability down to Generated Code and Documentation ...........................................81
  - Code Generation with Variable Vector Widths ..............81
  - Enhancements and Changes to the TargetLink Main Dialog ..82
  - Code Generator Options ......................................86
  - General Enhancements and Changes ........................89
- New AUTOSAR-Related Features ...............................92
  - Features of the TargetLink AUTOSAR Module .............92
- New Features of the dSPACE Data Dictionary ................98
  - New Key Features ............................................99
  - New and Modified DD MATLAB API Commands .............100
- Migrating to TargetLink 3.1 and dSPACE Data Dictionary 2.0 ..101
  - Model Upgrade ..............................................102
  - How to Upgrade Models and Libraries ........................103
  - Migrating to dSPACE Data Dictionary 2.0 ..................106
  - How to Upgrade a Data Dictionary Without Include Files ....109
  - How to Upgrade a Data Dictionary With Include Files ......110
  - General Migration Aspects ..................................113
  - Modified DD MATLAB API Commands ........................116
  - Obsolete and Changed Code Generator Options ............117
  - Changes in TargetLink API Functions ........................117
  - Various Migration Aspects ..................................118
  - AUTOSAR-Related Migration Aspects ........................123
  - Migrating ClientPort Blocks ................................126
About This Document

This document informs you about the new features of all the dSPACE software products in dSPACE Release 6.5. It also gives you an overview of software products with no or minor changes. There are instructions on migrating from older dSPACE releases, especially from older product versions, if required.
Overview of dSPACE Release 6.5

Objective
Gives you an overview of the new key features in dSPACE Release 6.5, and also information about unchanged products and general instructions on migrating.

Where to go from here
Information in this section

General Enhancements and Changes 9
Product Version Overview 10
New Product Key Features 11
Migrating to dSPACE Release 6.5 14

General Enhancements and Changes

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

Release update
The printed user documentation is not delivered with dSPACE Release 6.5 if you receive the release as an update for your existing dSPACE release. Use the current online help, for example, dSPACE HelpDesk, to obtain information about new features, enhancements, and the current safety precautions regarding your products.

Software support discontinued for DS4120
The DS4120 ECU Interface Board is supported by the dSPACE software only up to dSPACE Release 6.3. As of dSPACE Release 6.4, the software no longer supports the DS4120.
# Overview of dSPACE Release 6.5

## Product Version Overview

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

<table>
<thead>
<tr>
<th>Product</th>
<th>dSPACE Release</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6.2</td>
</tr>
<tr>
<td>AutomationDesk</td>
<td>2.1</td>
</tr>
<tr>
<td>Automotive Simulation Models</td>
<td>1.6</td>
</tr>
<tr>
<td>CalDesk</td>
<td>2.0</td>
</tr>
<tr>
<td>ConfigurationDesk</td>
<td>1.3.2</td>
</tr>
<tr>
<td>ControlDesk</td>
<td>3.2.2</td>
</tr>
<tr>
<td>dSPACE Data Dictionary</td>
<td>1.5.1</td>
</tr>
<tr>
<td>dSPACE FlexRay Configuration Package</td>
<td>1.10</td>
</tr>
<tr>
<td>Model Compare</td>
<td>2.0</td>
</tr>
<tr>
<td>ModelDesk</td>
<td>2.1</td>
</tr>
<tr>
<td>MotionDesk</td>
<td>2.1.2</td>
</tr>
<tr>
<td>MotionDesk Blockset</td>
<td>1.3.1</td>
</tr>
<tr>
<td>RTI</td>
<td>6.1</td>
</tr>
<tr>
<td>RTI-MP</td>
<td>5.5</td>
</tr>
<tr>
<td>RTI AUTOSAR Package</td>
<td>---</td>
</tr>
<tr>
<td>RTI Bypass Blockset</td>
<td>2.4.1</td>
</tr>
</tbody>
</table>

See [AutomationDesk](#) on page 15.

See [Automotive Simulation Models (ASM)](#) on page 19.

See [ConfigurationDesk](#) on page 43.

See [ControlDesk](#) on page 51.

See [TargetLink](#) on page 71.

See [MotionDesk](#) on page 55.

See [MotionDesk](#) on page 55.

See [RTI and RTLib](#) on page 57.

See [RTI AUTOSAR Package](#) on page 59.
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 detailed information, refer to the product-specific sections.

**AutomationDesk**
The new key features of AutomationDesk are:
- Element-wise version control
- XML import and export
- Finding disabled elements
- New HIL API library
- Enhancements to the Remote Calibration (COM) library

<table>
<thead>
<tr>
<th>Product</th>
<th>dSPACE Release</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6.2</td>
</tr>
<tr>
<td>RTI CAN Blockset</td>
<td>2.6.6</td>
</tr>
<tr>
<td>RTI CAN MultiMessage Blockset</td>
<td>2.3</td>
</tr>
<tr>
<td>RTI LIN MultiMessage Blockset</td>
<td>1.5.1</td>
</tr>
<tr>
<td>RTI RapidPro Control Unit Blockset</td>
<td>1.8</td>
</tr>
<tr>
<td>RTI FPGA Programming Blockset</td>
<td>---</td>
</tr>
<tr>
<td>SystemDesk</td>
<td>1.1</td>
</tr>
<tr>
<td>TargetLink</td>
<td>3.0</td>
</tr>
<tr>
<td></td>
<td>3.0</td>
</tr>
<tr>
<td></td>
<td>3.0</td>
</tr>
<tr>
<td>Variable Editor</td>
<td>1.1</td>
</tr>
</tbody>
</table>
Overview of dSPACE Release 6.5

- Enhancements to the Platform Access library
- Enhancements to the Report library
- Enhancements to the COM API

For details on the new features, refer to New Features of AutomationDesk 3.0 on page 15.

### Automotive Simulation Models (ASM)

The new ASM blocksets and tools of ASM are:

- ASM Truck Blockset for modeling the vehicle dynamics of a truck

For details on the new blocksets and tools, refer to Automotive Simulation Models (ASM) on page 19.

### ControlDesk

The new key features of ControlDesk are:

- The Bus Navigator now supports the handling of LIN frames configured with the RTI LIN MultiMessage Blockset.
  
  The LIN configuration is available via the Bus Navigator tree. You can:
  
  - Create layouts to view LIN RX frames and configure LIN TX frames.
  - Trigger LIN TX frames.

For details on the new features, refer to New Features of ControlDesk 3.5 on page 51.

### ModelDesk

The new key features of ModelDesk are:

- Support of the ASM Truck Blockset
- Support of the ASM Electric Components Blockset
- Several enhancements to the graphical user interface of other blocksets

For details on the new features, refer to New Features of ModelDesk 2.2.1 on page 53.

### MotionDesk

The new key features of MotionDesk are:

- Instruments usable in the scene.
- Support for antialiasing and texture filtering
- New natural look for 3-D library objects, textures and demo scenes
- New MotionDesk Blockset: easier to use and greater performance

For details on the new features, refer to New Features of MotionDesk 2.1.4 on page 55.
### RTI, RTI-MP and RTLib

The new key features of RTI, RTI-MP and RTLib are:
- RTI-MP supports model referencing
- Enhancement of the `rti_mdlversionget` function
- Changes in the Multiprocessor Setup dialog
- RTI supports the DS4004 Timing I/O functions

For details on the new features, refer to *New Features of RTI/RTI-MP and RTLib* on page 57.

### RTI AUTOSAR Package

The new key feature of RTI AUTOSAR Package is:
- Support of AUTOSAR Release 2.1 with Versions 2.1.2 and 2.1.4

For details on the new features, refer to *New Features of the RTI AUTOSAR Package 1.1* on page 59.

### RTI LIN MultiMessage Blockset

The new key feature of RTI LIN MultiMessage Blockset is:
- Support of layout generation with ControlDesk’s Bus Navigator

For details on the new features, refer to *New Features of the RTI LIN MultiMessage Blockset 1.7.1* on page 61.

### RTI FPGA Programming Blockset

The RTI FPGA Programming Blockset comes with the Processor Interface sublibrary for accessing the DS5203 FPGA board in a dSPACE modular system and the FPGA Interface sublibrary (requires a separate license) for implementing an FPGA application.

For details on the new features, refer to *New Features of the RTI FPGA Programming Blockset 1.1* on page 63.

### SystemDesk

For details on the new features, refer to *New Features of SystemDesk 2.1* on page 67.

### TargetLink

For details on the new features and migration aspects, refer to *TargetLink* on page 71.
## Migrating to dSPACE Release 6.5

<table>
<thead>
<tr>
<th><strong>Objective</strong></th>
<th>After you install dSPACE Release 6.5, some additional steps may be necessary.</th>
</tr>
</thead>
</table>

**Migrating from dSPACE Release 6.4**

There are no general migration steps to be done. The required product-specific migration steps are usually done automatically by the product. For exceptions, refer to the product-specific migration descriptions.

**Migrating from dSPACE Release 6.3 or earlier**

To migrate from dSPACE Release 6.3 or earlier to dSPACE Release 6.5, you also have to perform the migration steps of the intervening dSPACE Releases. All of the required migration steps can be done with dSPACE Release 6.5 installed.

**Example**

For example, if you want to migrate from dSPACE Release 6.1 to dSPACE Release 6.5, you have to perform the migration steps described in:

1. New Features and Migration of dSPACE Release 6.2
2. New Features and Migration of dSPACE Release 6.3
3. New Features and Migration of dSPACE Release 6.4
4. Finally, the migration steps described above.

**Previous release documents**

The New Features and Migration documents for previous releases are available via Internet and on the dSPACE DVD:

- Read them from the dSPACE DVD (see the `\Doc` folder). The PDF files are called `NewFeaturesAndMigrationxx.pdf`, where `xx` stands for the release number.

---

> Until dSPACE Release 6.2, the new features and migration steps for RCP & HIL software, CalDesk and TargetLink were described in separate documents.

# AutomationDesk

<table>
<thead>
<tr>
<th>Where to go from here</th>
<th>Information in this section</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>New Features of AutomationDesk 3.0</strong></td>
<td>15</td>
</tr>
<tr>
<td><strong>Migrating to AutomationDesk 3.0</strong></td>
<td>17</td>
</tr>
</tbody>
</table>

## New Features of AutomationDesk 3.0

**AutomationDesk 3.0**

**Element-wise version control**  
The version control interface can now be used with two different version control project types. The first type is for archiving an AutomationDesk project or a custom library in one single ZIP file, as was done in older AutomationDesk versions. The second type now allows you to separately archive elements of a project (project element itself, folders and sequences) or a custom library (custom library root folder, library folders, sequence templates and block templates). This is an extension of the multi-user support.

For further information, refer to *Using the Version Control System* (AutomationDesk Guide).
XML import and export  You can now import and export projects and custom libraries not only in ZIP format but also in XML format. Using the XML export, you can select the element in the project or the custom library where you want to start exporting. The start element and all of its subelements are written to separate XML files. With regard to AutomationDesk XML schema definitions, you can create AutomationDesk projects and custom libraries by importing the required data from custom XML files.

For further information, refer to Exporting and Importing Projects and Project Elements (AutomationDesk Guide).

Finding disabled elements  The Find dialog now allows you to search for disabled elements.

For further information, refer to Find (AutomationDesk Reference).

New HIL API library  The HIL API library is the AutomationDesk implementation of the HIL API specification published by the ASAM association as the new standard for measurement, calibration and diagnostics. The main idea is to decouple the test case implementation from the test automation software and the test hardware. With AutomationDesk 3.0, the HIL API library now has a near-complete implementation of the model access port (MAPort) as an alternative to the Platform Access library for accessing dSPACE platforms for writing and reading simulator variables.

For further information, refer to HIL API (AutomationDesk Library Reference).

Enhancements to the Remote Calibration (COM) library  The data objects of the Remote Calibration (COM) library can now be parameterized at run time. This allows you to configure calibration tasks flexibly according to the calibration tool used. The implemented Get and Set methods that you can call within Exec blocks are described in the data object’s user documentation.

For further information, refer to Remote Calibration (COM) (AutomationDesk Library Reference).

Enhancements to the Platform Access library  The Platform Access library provides the new Disconnect block to reset the connection to the real-time platform before downloading a new application.

For further information, refer to Platform Access (AutomationDesk Library Reference).
Enhancements to the Report library  The Report library provides the new InsertPageBreak block to explicitly set a page break in the generated PDF report.

For further information, refer to Report (AutomationDesk Library Reference).

Enhancements to the COM API  The AutomationDesk COM API now provides the new interfaces Project1, Folder1, Sequence1, and TestSequence1 that support the methods for the new XML import and export feature.

For faster access to the descriptions of the available objects and their properties, methods and events, the AutomationDesk API Reference has been restructured.

For further information, refer to AutomationDesk API Reference.

MTest  MTest support is discontinued since dSPACE Release 6.4.

Migrating to AutomationDesk 3.0

General migration aspects  If you open an AutomationDesk project with a newer AutomationDesk version, the software automatically detects whether migration is necessary. If you click OK in the message dialog, the migration is started. 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.

Before you open an older project with the new AutomationDesk version, ensure 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 message.
- The built-in libraries, required custom libraries and other packages must be correctly loaded.

You need not do any manual migration, except for the following point.
Migrating from AutomationDesk 1.x to AutomationDesk 2.x

The serialization of a project structure to the file system has been totally changed with AutomationDesk 2.x. Automatic migration covers only elements that are handled by the AutomationDesk project.

If you have added a file or folder to an AutomationDesk project structure in the file system manually using AutomationDesk 1.x, and you migrate from AutomationDesk 1.x to AutomationDesk 2.x or 3.x, the new AutomationDesk project does not contain that file or folder. You must copy the file or folder to the new AutomationDesk project structure in the file system to make it available to your project manually.

For example, the MainLibraryExamples.zip project contains an ExternalMaterial folder which you must copy to the migrated project manually.
Automotive Simulation Models (ASM)

Where to go from here

<table>
<thead>
<tr>
<th>Information in this section</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>All ASM Blocksets</td>
<td>20</td>
</tr>
<tr>
<td>ASM Base InCylinder Blockset</td>
<td>21</td>
</tr>
<tr>
<td>ASM Brake Hydraulics Blockset</td>
<td>22</td>
</tr>
<tr>
<td>ASM Diesel Exhaust Blockset</td>
<td>24</td>
</tr>
<tr>
<td>ASM Drivetrain Basic Blockset</td>
<td>27</td>
</tr>
<tr>
<td>ASM Electric Components Blockset</td>
<td>29</td>
</tr>
<tr>
<td>ASM Engine Diesel Blockset</td>
<td>30</td>
</tr>
<tr>
<td>ASM Engine Gasoline Basic Blockset</td>
<td>32</td>
</tr>
<tr>
<td>ASM Engine Gasoline Blockset</td>
<td>34</td>
</tr>
<tr>
<td>ASM Environment Blockset</td>
<td>36</td>
</tr>
<tr>
<td>ASM Optimizer</td>
<td>37</td>
</tr>
<tr>
<td>ASM Truck Blockset</td>
<td>38</td>
</tr>
<tr>
<td>ASM Vehicle Dynamics Blockset</td>
<td>39</td>
</tr>
</tbody>
</table>

Information in other sections

*Migrating ASM Models* (ASM User Guide)
Provides general information on the migration process of ASM models.


## All ASM Blocksets

<table>
<thead>
<tr>
<th>Where to go from here</th>
<th>Information in this section</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Features of MotionDesk ASM Demo Experiments and Blockset</td>
<td>20</td>
</tr>
<tr>
<td>Migration of MotionDesk ASM Demo Experiments and Blockset</td>
<td>20</td>
</tr>
</tbody>
</table>

### New Features of MotionDesk ASM Demo Experiments and Blockset

#### 3D Object Library

- The textures of some objects in the 3D library are updated with a new look.

### Migration of MotionDesk ASM Demo Experiments and Blockset

#### 3D Object Library

- The 3-D objects of the ASM MotionDesk demo experiments are automatically updated with the new object textures that have a more natural look, see New Features of MotionDesk 2.1.4 on page 55.

#### MotionDesk Blockset

- The MotionDesk Blockset is migrated from a previous version to version 2.0 automatically. You must only open the real-time model and rebuild the real-time application. For detailed information on the blockset, refer to MotionDesk Blockset Reference.
## ASM Base InCylinder Blockset

### Migrating to ASM Base InCylinder Blockset 1.0.1

<table>
<thead>
<tr>
<th>Block</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COOLER block</td>
<td>The COOLER block contained incorrect unit conversion in the power calculation. This has been fixed. To achieve the same model behavior after migration, scaling for inports and parameters has to be inserted.</td>
</tr>
<tr>
<td>INJECTOR_CONTINUOUS block</td>
<td>Multiple usage of the Const_Num_Cyl parameter below the mask has been removed.</td>
</tr>
<tr>
<td>WALL.HEAT block</td>
<td>The WALL.HEAT block contained incorrect calculation of C1 in the wall heat model of the Woschni approach. It is now correctly switched between gas exchange and compression/expansion. To achieve the same model behaviour after migration, the parameter for expansion/compression are set equal to the parameter for gas exchange.</td>
</tr>
</tbody>
</table>
ASM Brake Hydraulics Blockset

Where to go from here

Information in this section

New Features of ASM Brake Hydraulics Blockset 1.2 22
Migrating to ASM Brake Hydraulics Blockset 1.2 23

New Features of ASM Brake Hydraulics Blockset 1.2

Continuous valve control

Valves are sometimes continuously controlled by an ECU via a control current. The flow area of these valves depends on the valve control current and the pressure difference. The new blocks are based on a look-up table that calculates the valve control signal from the valve control current and the pressure difference.

The following new blocks are available:

- CONTINUOUS_CTRL_PRE_CHARGE_VALVE_1,
  CONTINUOUS_CTRL_PRE_CHARGE_VALVE_2
- CHANGE_OVER_VALVE_1,
  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

Table-based valves

The new blocks are based on a look-up table that calculates the valve flow from a control signal (for example, the valve control current) and the pressure difference. This model approach can be useful if the fluid flow was measured on a test bench or is available from a data sheet.

The following new blocks are available:

- PRE_CHARGE_VALVE_1_TABLE,
  PRE_CHARGE_VALVE_2_TABLE
- 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
OUTLET_VALVE_FL_TABLE, OUTLET_VALVE_FR_TABLE, OUTLET_VALVE_RL_TABLE, OUTLET_VALVE_RR_TABLE

MASTER_BRAKE_CYLINDER block
The MASTER_BRAKE_CYLINDER block supports connection to the brake booster model from the ASM Vehicle Dynamics Blockset.

Migrating to ASM Brake Hydraulics Blockset 1.2

MASTER_BRAKE_CYLINDER block
The MASTER_BRAKE_CYLINDER block has the new F_PushRod[N] inport and the new A_Piston[m2] outport. During migration the new outport is automatically connected to a terminator block and the inport is connected to a constant block of value 0.

ASMSignalBus
For nearly all the blocks of the ASM Brake Hydraulics Blockset, the signal names of the ASMSignalBus are extended with the specific component name. During migration, all the Simulink Bus Selector blocks in the model are updated with the new signal names automatically.
# ASM Diesel Exhaust Blockset

## Where to go from here

<table>
<thead>
<tr>
<th>Information in this section</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ASM Diesel Exhaust Blockset 1.1</strong></td>
</tr>
<tr>
<td><strong>Migrating to ASM Diesel Exhaust Blockset 1.0.1</strong></td>
</tr>
</tbody>
</table>

## ASM Diesel Exhaust Blockset 1.1

### UREA_DECOMPOSITION block
A urea decomposition factor map has been introduced so that the fraction of decomposed urea molecules can be adjusted via the temperature and mass flow upstream of the SCR catalyst. An additional map scales the vaporization energy of water to avoid vaporization effects at implausible temperatures.

### ADBLUE_PUMP block
A reverting valve and heaters have been implemented in order to simulate components in a non-air-assisted SCR supply system. The reverting valve changes the flow direction of the pump, and the heaters convert electrical energy into heat at the throttle and pressure side of the pump.

### INJECTION_VALVE block
A heater has been implemented in order to simulate components in a non-air-assisted SCR supply system. The block converts electrical energy into heat.

### ExhaustSystem_DOC_DPF_NonAir_SCR
A new demo model has been provided as an example of a non-air-assisted SCR supply system prepared with the ASM DieselExhaust Library.
Migrating to ASM Diesel Exhaust Blockset 1.0.1

**DIESEL_OXIDATION_CATALYST block**
The dependencies for the pressure drop have been changed from the mass flow to the volumetric flow and temperature at the inlet of the diesel oxidation catalyst. The block from the previous version has been moved to the Former versions subsystem in the ASM DieselExhaust Library. The migration procedure will change the library link to the Former versions block so that existing parameterizations are not affected.

**DIESEL_PARTICULATE_FILTER block**
The dependencies for the pressure drop have been changed from the soot mass and total mass flow to the soot mass and volumetric flow inlet of the diesel particulate filter. The block from the previous version has been moved to the Former versions subsystem in the ASM DieselExhaust Library. The migration procedure will not change the library link to the Former versions block. Existing parameterizations should not be affected negatively. The input for disabling the soot formation has been modified, such that the soot mass in the particulate filter can be either calculated, disabled, or set to a fixed, externally specified value to improve the results of test bench runs.

**MUFFLER block**
The dependencies for the pressure drop have been changed from the mass flow to the volumetric flow and temperature at the inlet of the muffler. The block from the previous version has been moved to the Former versions subsystem in the ASM DieselExhaust Library. The migration procedure will change the library link to the Former versions block so that existing parameterizations are not affected.

**SCR_CATALYST block**
As the upstream temperature of the muffler is required for the evaluation of the pressure drop, the output temperature of the SCR Catalyst has been added as a new outport.

**PUMP_HOSE block**
An external flag has been introduced to reset the states of the mass and energy balance in the pump hose. Modifications have also been made to the evaluation of the heating power in the energy balance.

**ADBLUE_TANK block**
A constant has been introduced to initialize the tank level to a predefined value.

**RAW_EXHAUST_COMPOSITION block**
An outport has been added to provide a desired soot mass for the diesel particulate filter, which can be useful during test bench simulations.
### SWITCHES_EXHAUST SYSTEM block

As the import of the diesel particulate filter for disabling the soot formation has been modified, the name of the corresponding switching block has changed, so the trace path of the real-time variable will be different. This will lead to open connections in the ControlDesk experiment, which you have to modify appropriately. The ASM demo experiment can be used as a reference.

### ExhaustSystem_DOC_DPF

The demo model has been adapted, as the scaling for the universal gas constant is not required. You should check your implementation to avoid incorrect calculations during the oxidation of the soot mass in the diesel particulate filter.
ASM Drivetrain Basic Blockset

Where to go from here

Information in this section

New Features of ASM Drivetrain Basic Blockset 1.3

27

Migrating to ASM Drivetrain Basic Blockset 1.3

27

New Features of ASM Drivetrain Basic Blockset 1.3

KEY_STATES block

This block is new. It calculates the ignition and starter request according to the key position. For further details, see Key States (ASM Drivetrain Basic Reference).

CYCLES block

Now, the CYCLES block can also handle engine cycles and is supported by ModelDesk. For details, see Driver Maneuver (CYCLES Block) (ASM Drivetrain Basic Reference).

Migrating to ASM Drivetrain Basic Blockset 1.3

GEARBOX_MT, GEARBOX_AT, STARTER, CRANK_SHAFT, DRIVING_RESISTANCES blocks

The “^” sign has been removed from several block names.

GEAR_SHIFTER block

The Sw_Gear_Set[0Off1On] inport has been introduced for activating external control of manual transmission. The inport is connected to the related output of the CYCLES block.

TEST_BENCH block

The anti-windup of the controller has been improved. The Const_Trq_Max parameter is now unique below the mask, so the real-time path to this variable has been changed.
Now, the CYCLES block can also handle engine cycles. If your model already supports ModelDesk, it will be automatically be possible to parameterize this block via ModelDesk. Default driving cycles can be copied from the ModelDesk pool of the related current demo model to the ModelDesk pool of the migrated project. To provide support for engine cycles, several new parameters and outports have been added. See the block documentation for details. The new parameters which are related to the cycles are initialized by the modified \texttt{asm_eng_drivingcycle} function. Some engine cycles are scaled with engine data. The related new parameters are set to default values and have to be adapted if such engine cycles are used. During migration, the new outports are automatically connected to the appropriate signal. If the parent system of the block has been changed, this might fail. In this case compare the connections to the current demo model and modify them manually.
ASM Electric Components Blockset

Where to go from here

Information in this section

<table>
<thead>
<tr>
<th>New Features of ASM Electric Components Blockset 1.1</th>
<th>29</th>
</tr>
</thead>
<tbody>
<tr>
<td>Migrating to ASM Electric Components Blockset 1.1</td>
<td>29</td>
</tr>
</tbody>
</table>

New Features of ASM Electric Components Blockset 1.1

BATTERY block

The effect of current loss in the thermal model and the terminal voltage model have been modified.

Migrating to ASM Electric Components Blockset 1.1

HALL_ENCODER block

The Hall_Matrix parameter has been subdivided into six Hall configuration vector parameters. During migration, an initialization M file is generated for these new vector parameters.

BLDC_CONTROLLER block

The is_index indicator matrix parameter, the is_dir indicator vector and the two Zi_HSD and Zi_LSD matrix parameters have been removed as static parameters below the mask.
ASM Engine Diesel Blockset

Migrating to ASM Engine Diesel Blockset 1.3.3

INJECTOR block
The pulsed injection mode has been updated, such that the enable flag will be generated at a crank angle corresponding to the Angle_Cyl input signal. The Const_phi_Inj_MeasUpdate variable has to be initialized in [aTDC].

COOLER block
The COOLER block contained incorrect unit conversion in the power calculation. This has been fixed. To achieve the same model behavior after migration, scaling for inports and parameters has been inserted.

Note: If you use ModelDesk to parameterize the engine model, the ModelDesk and ASMPparameterization projects must be updated manually before generating initialization files and downloading parameters to the Simulink model or real-time hardware to achieve the same model behavior with the current parameter set. The following steps have to be performed:

1. Adapt the parameters in the ASMPparameterization project
   - Open the ASMPparameterization project.
   - Navigate to the Parameters – EngineDiesel – Cooler page.
   - Open the parameter file.
   - Change the following parameters:
     - Const_T_dt_Coolant.v by a factor of 1/(2*pi)
     - Const_P_Gain_Coolant_Controller.v by a factor of (2*pi)
     - Const_Q_dt_Air.v, Const_Q_dt_Fan1.v, Const_Q_dt_Fan2.v by a factor of (2*pi)
   - This has to be done in all COOLER parameter M files in the project.

2. Adapt the cooler parameters in the ModelDesk project.
   - Open the ModelDesk project.
   - Navigate to the Engine Diesel – Cooler Diesel parameter page.
   - Change the following parameters:
     - Time constant of basic cooler by a factor of 1/(2*pi)
     - Proportional gain for temperature controller by a factor of (2*pi)
- Gain for air cooling with fan / with external fan1 / with external fan2 by a factor of (2*π)
  This has to be done in all COOLER XML files of the ModelDesk project.
- Save the ModelDesk project.

3. Generate initialization files from either ASMPParameterization or ModelDesk for all variants.

4. Remove the changes which were inserted in the post-migration file:
   - In the Simulation.current directory of the project, open the \_asmmigrate300post\IniFiles\asmmigrate300_cooler_ini.m file.
   - Remove the passage inside the marks
     ```
     %=== START COOLER PARAMETER SCALING ===%
     %=== END COOLER PARAMETER SCALING ===%
     ```
ASM Engine Gasoline Basic Blockset

Migrating to ASM Engine Gasoline Basic Blockset 1.3.3

COOLER block

The COOLER block contained incorrect unit conversion in the power calculation. This has been fixed. To achieve the same model behavior after migration, scaling for inports and parameters has been inserted.

Note: If you use ModelDesk to parameterize the engine model, the ModelDesk and ASMPParameterization projects must be updated manually before generating initialization files and downloading parameters to the Simulink model or real-time hardware to achieve the same model behavior with the current parameter set. The following steps have to be performed:

1. Adapt the parameters in the ASMPParameterization project
   - Open the ASMPParameterization project.
   - Navigate to the Parameters – EngineGasolineBasic – Cooler page.
   - Open the parameter file.
   - Change the following parameters:
     - Const_T_dt_Coolant.v by a factor of 1/(2*pi)
     - Const_P_Gain_Coolant_Controller.v by a factor of (2*pi)
     - Const_Q_dt_Air.v, Const_Q_dt_Fan1.v, Const_Q_dt_Fan2.v by a factor of (2*pi)
   
   This has to be done in all COOLER parameter M files in the project.

2. Adapt the cooler parameters in the ModelDesk project.
   - Open the ModelDesk project.
   - Navigate to the Engine Gasoline Basic – Cooler Gasoline Basic parameter page.
   - Change the following parameters:
     - Time constant of basic cooler by a factor of 1/(2*pi)
     - Proportional gain for temperature controller by a factor of (2*pi)
     - Gain for air cooling with fan / with external fan1 / with external fan2 by a factor of (2*pi)
This has to be done in all COOLER XML files of the ModelDesk project.

- Save the ModelDesk project.

3. Generate initialization files from either ASMPerameterization or ModelDesk for all variants.

4. Remove the changes which were inserted in the post-migration file:
   - In the simulation.current directory of the project, open the \_asmmigrate300post\IniFiles\asmmigrate300_cooler.ini.m file.
   - Remove the passage inside the marks
     ```
     %=== START COOLER PARAMETER SCALING ===%
     %=== END COOLER PARAMETER SCALING ===%
     ```
ASM Engine Gasoline Blockset

New Features of ASM Engine Gasoline Blockset 2.1.3

COMBUSTION_MODE_SWITCH block
The COMBUSTION_MODE_SWITCH block improves switching between stratified and homogeneous combustion.

Migrating to ASM Engine Gasoline Blockset 2.1.3

DIRECTINJECTOR block
The pulsed injection mode has been updated, such that the enable flag is generated at a crank angle corresponding to the Angle_Cyl input signal. The Const_phi_Inj_MeasUpdate variable has to be initialized in [aTDC].

COOLER block
The COOLER block contained incorrect unit conversion in the power calculation. This has been fixed. To achieve the same model behavior after migration, scaling for inports and parameters has been inserted.

Note: If you use ModelDesk to parameterize the engine model, the ModelDesk and ASMParameterization projects must be updated manually before generating initialization files and downloading parameters to the Simulink model or real-time hardware to achieve the same model behavior with the current parameter set. The following steps have to be performed:

1. Adapt the parameters in the ASMParameterization project
   - Open the ASMParameterization project.
   - Navigate to the Parameters – EngineGasoline – Cooler page.
   - Open the parameter file.
   - Change the following parameters:
     - Const_T_dt_Coolant.v by a factor of 1/(2*π)
2. Adapt the cooler parameters in the ModelDesk project.
   - Open the ModelDesk project.
   - Navigate to the Engine Gasoline – Cooler Gasoline parameter page.
   - Change the following parameters:
     - Time constant of basic cooler by a factor of 1/(2*pi)
     - Proportional gain for temperature controller by a factor of (2*pi)
     - Gain for air cooling with fan / with external fan1 / with external fan2 by a factor of (2*pi)
   - This has to be done in all COOLER XML files of the ModelDesk project.
   - Save the ModelDesk project.

3. Generate initialization files from either ASMParameterization or ModelDesk for all variants.

4. Remove the changes which were inserted in the post-migration file:
   - In the Simulation.current directory of the project, open the \_asmmigrate300post\IniFiles\asmmigrate300_cooler_ini.m file.
   - Remove the passage inside the marks
     %=== START COOLER PARAMETER SCALING ===%
     and
     %=== END COOLER PARAMETER SCALING ===%
ASM Environment Blockset

Where to go from here

Information in this section

New Features of ASM Environment Blockset 1.4.1 36
Migrating to ASM Environment Blockset 1.4.1 36

New Features of ASM Environment Blockset 1.4.1

BRAKE_HYDRAULICS_VARIANT block

The block supports connection with the brake booster model from the ASM Vehicle Dynamics Blockset.

Migrating to ASM Environment Blockset 1.4.1

BRAKE_HYDRAULICS_VARIANT block

The block contains four new inports. During migration, the new inports are connected to constant blocks with default values.
ASM Optimizer

New Features of ASM Optimizer 1.2

<table>
<thead>
<tr>
<th>Advanced postprocessing</th>
<th>The advanced, interactive postprocessing functionality enables you to disable single operating points for map generation. Additionally, end and start values for the map axis can be specified to define extrapolation values manually. When you select operating points and extrapolation values, the results are displayed simultaneously and the currently selected operating point is marked in the maps.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loading results of prior optimization runs</td>
<td>To continue canceled optimization runs, you can select whether to load the results of a prior optimization run before the current optimization run is started. You can select result loading for each task separately on the Manage Tasks page.</td>
</tr>
</tbody>
</table>
ASM Truck Blockset

New Blockset: ASM Truck 1.0

The new ASM Truck Blockset is an extension to the ASM Vehicle Dynamics Blockset for modeling trucks. It has the following features:

- 4-axle truck with 4-axle semi-trailer (arbitrary number of axles also possible)
- Multi-trailer coupling
- 8x2, 8x4 drivetrain
- Twin tires
  - Two different contact points for twin tires are calculated.
  - Twin tires can be switched to single tire online.
- Chassis with torsional elasticity frame
- Can be combined with the ASM Trailer Blockset

Related topics

References

ASM Truck Reference
ASM Vehicle Dynamics Blockset

Where to go from here

Information in this section

<table>
<thead>
<tr>
<th>New Features of ASM Vehicle Dynamics Blockset 1.4</th>
<th>39</th>
</tr>
</thead>
<tbody>
<tr>
<td>Migrating to ASM Vehicle Dynamics Blockset 1.3</td>
<td>40</td>
</tr>
</tbody>
</table>

**New Features of ASM Vehicle Dynamics Blockset 1.4**

**TEST_BENCH block**

A new block has been inserted for performing engine test bench tests. This block is equivalent to the TEST_BENCH block of the ASM DrivetrainBasic Blockset. If you set up a virtual vehicle including a detailed engine model, the TEST_BENCH block supports easy engine model testing.

**CENTRAL_DIFFERENTIAL block**

The CENTRAL_DIFFERENTIAL block now supports a parking pawl for automatic transmissions. The Ctrl_Parking_Pawl[0|1] input has been introduced for this. The parking pawl function was inserted in this block and not in the transmission block because the transmission output speed is calculated by the central differential.

**TIRE_MODEL_TMEASY block**

Three user scaling parameters have been added to the TMEasy tire model. You can scale the effect of the camber angle on the lateral force, the effect of the vertical angular velocity of the wheel on the vertical torque (bore torque), and the effect of the camber angle on the torque about the longitudinal axis of the wheel (tipping torque).

**SENSOR_MOTION block**

A fourth sensor position has been added. The parameter can be parameterized in ModelDesk.

**BRAKE_BOOSTER block**

The new model simulates a vacuum brake booster that supplements the brake pedal force with a force generated by a pneumatic booster system.

**MANIFOLD_PRESSURE block**

The new block is based on a look-up table that calculates the manifold pressure from the accelerator pedal position. It is needed to simulate the manifold pressure as an input signal for the brake booster.
Automotive Simulation Models (ASM)

Some new signals have been added to the ASMSignalBus. These signals are needed for the brake booster support of the driver in the ASM Environment Blockset.

Migrating to ASM Vehicle Dynamics Blockset 1.3

**BRAKE_DISC block**

The handling of the parking pawl has been redesigned. The old Pos_ParkingBrakePedal[\%] output has been renamed Ctrl_Parking_Pawl[0|1]. This output is active if the selector lever is in "P". During migration, a gain of 50 is added to this output to achieve the same behavior as before, where Pos_ParkingBrakePedal[\%] was set to 50 when the selector lever was in "P". This output can also be manually connected to the Ctrl_Parking_Pawl[0|1] input of the CENTRAL_DIFFERENTIAL block to add the parking pawl functionality to the model.

**CENTRAL_DIFFERENTIAL block**

The new Ctrl_Parking_Pawl[0|1] input is connected to a constant block. To introduce the new functionality, it has to be connected to the Ctrl_Parking_Pawl[0|1] output of the SOFT_ECU_TRANSMISSION block.

**STARTER block**

The STARTER block is now consistent with the STARTER block of the ASM Drivetrain Basic Blockset. The started inertia has been introduced as a new parameter, which is assigned to the new Inertia_Started[kgm^2] output if the starter is active. To avoid changes in the model behavior, the new parameter value is set to zero during migration.
### CRANK_SHAFT block

The CRANK_SHAFT block is now consistent with the CRANK_SHAFT block of the ASM Drivetrain Basic Blockset. The modulated mass torque map has been introduced as a new parameter. The effects of accelerating and decelerating the pistons can now be considered together with the crank angle and the engine speed. The modulated mass torque can be activated and deactivated by the new `SW_Trq_Mass_Mode[1On|2Off]` inport. During migration, this switch is set to off.

The new `Trq_TestBench[Nm]` inport has been added to include additional engine test bench torque on the crankshaft. During migration, this port is connected to a constant block with a value of 0.

The new `Inertia_External[kgm2]` inport has been added to consider additional inertias (for example, of the starter). During migration, this port is connected to a constant block with a value of 0.

### GEARBOX_MT, GEARBOX_AT blocks

The look-up method of all tables has been changed to interpolation/extrapolation.

### TIRE_MODEL_TMEASY block

During migration, three user scaling parameters with the default value "1" are added to the TMEasy tire model.

### VEHICLE_MASS_AND_ADDITIONAL_LOADS block

The calculation of Steiner's theorem has been improved.

### WHEEL_SPEED block

The calculation of additional damping at low speed has been improved.

### SENSOR_MOTION block

A fourth sensor position with the default value [0,0,0] has been added to the subsystem.

### Rear suspension kinematic subsystems

The rear suspension kinematic subsystems (SUSKIN_REARASYM_3DOF, SUSKIN_REAR_SYM_3DOF, SUSPENSION_KINEMATICS_REARASYMMETRIC, SUSPENSION_KINEMATICS_REAR_SYMMETRIC, and SEMI_TRAILING_ARM, RIGID_AXLE blocks) have the new `WheelPos_Offset[x;y;z][m]` inport. This inport is connected to a constant block with a value of [0,0,0]. The inport can be used to set an offset for the wheel position of suspension kinematics.
ConfigurationDesk

<table>
<thead>
<tr>
<th>Where to go from here</th>
<th>Information in this section</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConfigurationDesk Features</td>
<td>44</td>
</tr>
<tr>
<td>Migrating to ConfigurationDesk 3.0.1</td>
<td>45</td>
</tr>
</tbody>
</table>
# ConfigurationDesk Features

## New Features of ConfigurationDesk 3.0.2

**New supported RapidPro hardware**

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PS-HCFBD 1/2</strong></td>
<td>The PS-HCFBD 1/2 module (DS1767) is a high-current, full-bridge power stage module with one channel for driving bidirectional loads such as DC motors (12 V or 24 V).</td>
</tr>
<tr>
<td><strong>PS-HCHBD 2/2</strong></td>
<td>The PS-HCHBD 2/2 module (DS1768) is a high-current, half-bridge power stage module with two channels for driving loads in different modes in 12 V or 24 V power supply systems.</td>
</tr>
</tbody>
</table>

**Accessing the new RapidPro hardware**

Following software is required to work with the modules:
- ConfigurationDesk 3.0 (on dSPACE Release 6.3) with plug-in software
- ConfigurationDesk 3.0.1 (on dSPACE Release 6.4) with plug-in software
- ConfigurationDesk 3.0.2 (and later) supports the modules without additional software components.

**Related topics**

- [PS-HCFBD 1/2 Module](#)
- [PS-HCHBD 2/2 Module](#)

**References**
Migrating to ConfigurationDesk 3.0.1

Objective
Applications created with ConfigurationDesk versions prior to version 1.2. must be migrated to achieve upward compatibility with ConfigurationDesk 3.0.1.

Where to go from here
Information in this section

Migration to ConfigurationDesk 3.0.1

Objective
Applications created with ConfigurationDesk 1.1.x cannot be used with ConfigurationDesk 3.0.1 as they are incompatible with this ConfigurationDesk version.

Upward compatibility
ConfigurationDesk can convert these applications to achieve upward compatibility. Downward compatibility is not implemented.

<table>
<thead>
<tr>
<th>Application Created with ...</th>
<th>Compatible with ConfigurationDesk 1.2.0 or Earlier</th>
<th>Compatible with ConfigurationDesk 3.0.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConfigurationDesk 1.1.2 or earlier</td>
<td>Yes</td>
<td>Yes, after migration</td>
</tr>
<tr>
<td>ConfigurationDesk 1.2 or later</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
### Migrating an old project/application

When you open a project that contains an old application, or when you import an old application, ConfigurationDesk prompts you to migrate the application:

If you cancel the conversion, the application and its old subfolders are not shown in the Project Manager.

After an old project has been loaded and converted, you can configure its applications and add new ones.

### Creating application based on old hardware data

When you are creating a new application and the Project Wizard prompts you to add the hardware topology, you cannot use an old HTF file as the data source. The same applies when the Project Wizard prompts you to add the configuration settings. You cannot use an old HCF file as a data source either. It is therefore recommended to migrate only complete projects or applications.

If you only have an old HTF or an old HCF file which is part of a project or an (exported) application, contact dSPACE Support.
An old application (ConfigurationDesk 1.1.2 or earlier) comprises a hardware topology file (HTF file) and in some cases a hardware configuration file (HCF file) depending on whether configuration settings were specified. During migration, ConfigurationDesk converts an HTF file to a new HTF file. An HCF file is copied, and the copy is converted to a CDS file. The project's folder structure is not affected by migration. Thus, all the files are stored in the same subfolders as before.

For ConfigurationDesk 1.1.2 or earlier, the parameters of the user-configurable circuits of a module are stored in HCF files. Since ConfigurationDesk 1.2 these parameters are stored in HTF files. As a result, if an application that does not comprise an HCF file is converted, the parameters of the user-configurable circuits are lost.

CDL files (application) and CDP files (project) are also modified during migration.

How to Migrate a ConfigurationDesk Project with the RapidPro Projects Migrator

Objective
Projects and applications created with ConfigurationDesk 1.1.x are not compatible with ConfigurationDesk 3.0.1. You must therefore migrate them with ConfigurationDesk’s RapidPro Projects Migrator.

RapidPro Projects Migrator
To facilitate migration, ConfigurationDesk provides the RapidPro Projects Migrator. It is a stand-alone tool which migrates projects and applications created with ConfigurationDesk 1.1.x to ConfigurationDesk 3.0.1. You can find the RapidPro Projects Migrator at %DSpace_ROOT%\ConfigurationDesk\RapidProProjectsMigrator\Bin\RapidProProjectsMigrator.exe.
The RapidPro Projects Migrator provides several symbols to indicate the status of the migration. The following symbols are used:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="checkmark.png" alt="Checkmark" /></td>
<td>The project was already migrated or does not need to be migrated to use it with ConfigurationDesk 3.0.1.</td>
</tr>
<tr>
<td><img src="warning.png" alt="Warning" /></td>
<td>The project needs to be migrated to use it with ConfigurationDesk 3.0.1.</td>
</tr>
<tr>
<td><img src="checkmark.png" alt="Checkmark" /></td>
<td>The migration was successful.</td>
</tr>
<tr>
<td><img src="x.png" alt="X" /></td>
<td>The migration failed, but the affected project is not defective. It was automatically restored as all the projects to be migrated are backed up before migration. If migration is successful, the backup files are automatically deleted.</td>
</tr>
<tr>
<td><img src="lightning.png" alt="Lightning Bolt" /></td>
<td>The project is defective or is not a ConfigurationDesk project.</td>
</tr>
</tbody>
</table>

You can also find this list of symbols on the Help tab in the RapidPro Projects Migrator.

**Migrating several projects**

The RapidPro Projects Migrator allows you to migrate several projects at a time. If you do so, migration may take some time.

**Recommendation**

It is recommended to back up the projects you want to migrate. This is for safety purposes only.

**Method**

To migrate a ConfigurationDesk Project with the RapidPro Projects Migrator

1. Start the RapidPro Projects Migrator.
2. On the Projects page, search for the work folder where the project(s) you want to migrate are located.

   If you want to migrate applications only, switch to the Applications page of the RapidPro Projects Migrator.
The project located in the selected work folder is displayed in the Project name list. An automatically activated checkbox next to the project name indicates that the project will be considered for migration. The symbol next to the project name indicates that the project needs to be migrated.

3 Click Migrate Project.

**Result**

The selected project is successfully migrated, which is indicated in the Project name list by the following symbol.

The project can now be used with ConfigurationDesk 3.0.1.
ControlDesk

New Features of ControlDesk 3.5

| Bus Navigator | Support of LIN bus | The Bus Navigator now supports the handling of LIN frames configured with the RTI LIN MultiMessage Blockset. This blockset is a Simulink blockset for efficient and dynamic handling of complex LIN setups in hardware-in-the-loop (HIL) applications. With the RTI LIN MultiMessage Blockset, you can configure and control a large number of LIN frames from a single Simulink block.

The LIN configuration is available via the Bus Navigator tree. You can:
- Create layouts to view LIN RX frames and configure LIN TX frames.
- Trigger LIN TX frames. |
ModelDesk

New Features of ModelDesk 2.2.1

| New supported ASM model | The following ASM models are now supported by ModelDesk:  
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>■ ASM Truck</td>
<td>■ ASM Electric Components</td>
</tr>
</tbody>
</table>

For details on the parameters of the new supported ASM models, refer to ModelDesk ASM Parameters Reference.

<table>
<thead>
<tr>
<th>Optimized GUI</th>
<th>The graphical user interfaces (GUI) of the ASM models were optimized.</th>
</tr>
</thead>
</table>

| Road Generator | The Road Generator creates road 3-D objects in a new natural look matching the style of the MotionDesk 3-D object library. You can also generate roads with several lanes. |
# New Features of MotionDesk 2.1.4

## Instruments

You can use instruments to observe the values of simulation variables of the real-time application in the scene. This can be used, for example, to create a speedometer for a virtual car. For details, refer to *Using Instruments in the Scene* ([MotionDesk 3-D Online Visualization Guide](#)).

## Antialiasing and texture filtering

Since MotionDesk 2.1.4, the antialiasing and texture filtering of the graphics adapter is supported. The edges appear much smoother in anti-aliased images. Texture-filtering reduces the flickering of detailed textures. Both features let the resulting images may seem softer and should appear more realistic. The features are disabled after installation. For information on how to activate the features, refer to *Activating Antialiasing and Texture Filtering* ([MotionDesk 3-D Online Visualization Guide](#)).

## 3-D object library

Since MotionDesk 2.1.4, 3-D objects in the 3-D object library have a more natural look.

Some 3-D objects were revised to improve their look. For example, the shape of the trees and the textures of wheels were improved.
There is a new automotive demo showing how to use MotionDesk instruments. The old automotive demo was renamed MultiTrackDemo.

The MotionDesk Blockset has been redesigned:
- Formerly, several blocks were necessary to implement the communication from the simulation model to MotionDesk. Now this is done by one block, the MD_Communication block.
- The MD_Transform_To_Euler and MD_Transform_To_Cardan blocks have been replaced by one block, the MD_Rotation_Angles block.
- The Gigalink blocks have been removed from the MotionDesk Blockset. They have been moved to the RTI-MP Blockset.

For details on the MotionDesk Blockset, refer to MotionDesk Blockset Reference.

### Migration to MotionDesk 2.1.4

<table>
<thead>
<tr>
<th>Updating the 3-D Object Library</th>
<th>3-D objects have been changed in MotionDesk 2.1.4. Installing MotionDesk 2.1.4 replaces the relevant objects and textures in the 3-D objects library below the dSPACE_Objects folder. This can also affect existing scenes. If your scene uses 3-D objects of the Custom_Objects folder, they can have a mix of old- and new-style 3-D objects, for example, a plate in the new style and a road built in the old style. In this case, you should replace the texture files below the Custom_Objects folder by the texture files of the same name that are below the dSPACE_Objects folder. If the new style is not wanted, you can return to the previous look. You can use the 3-D Object Updater tool for this. For information on how to work with the tool, refer to How to Update the 3-D Object Gallery (MotionDesk 3-D Online Visualization Guide).</th>
</tr>
</thead>
<tbody>
<tr>
<td>MotionDesk Blockset</td>
<td>The MotionDesk Blockset is migrated from a previous version to Version 2.0 automatically. For detailed information on the blockset, refer to MotionDesk Blockset Reference.</td>
</tr>
</tbody>
</table>
RTI and RTLib

New Features of RTI/RTI-MP and RTLib

| Model referencing for RTI-MP | RTI-MP now supports Simulink Model blocks. This allows you to apply all the features of model referencing in RTI-MP models. The following modifications were also implemented:  
  - The model workspace of the main model is passed to the separated submodels.  
  - There is no longer a difference between the system target files and template makefiles of RTI and the system target files and template makefiles of RTI-MP. The files rti<xxxx>.tlc and rti<xxxx>.tmf are used for RTI and RTI-MP.  
  - The folder for the build results of an RTI-MP model is now <submodel>_rti<xxxx> instead of <submodel>_rtimp<xxxx>. |
| Getting version information | The output of the rti_mdlversionget function has been extended. It now also contains information about the MATLAB version and the MATLAB release the model was saved with.  
This function does not open the model, it is only for checking version information quickly. For details, refer to rti_mdlversionget ([RTI and RTI-MP Implementation Reference](#)). |
| Multiprocessor Setup dialog | The Global variable description file options that were placed in the Variable Description File Options page of the CPU Options dialog have been moved to the Main page of the Multiprocessor Setup dialog. |
### DS4004 support

RTI supports the timing I/O feature of the DS4004 HIL Digital I/O Board. You can now measure and generate PWM and square-wave signals via RTI blocks. For details, refer to *Timing I/O (DS4004 Features)*.
RTI AUTOSAR Package

New Features of the RTI AUTOSAR Package 1.1

RTI AUTOSAR Package 1.1 provides the following features:

<table>
<thead>
<tr>
<th>Supported AUTOSAR Releases</th>
<th>The RTI AUTOSAR Package supports:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AUTOSAR Release 3.1 with Version 3.1.0 (continued)</td>
</tr>
<tr>
<td></td>
<td>AUTOSAR Release 3.0 with Version 3.0.2 (continued)</td>
</tr>
<tr>
<td></td>
<td>AUTOSAR Release 2.1 with Versions 2.1.2 and 2.1.4 (new in RTI AUTOSAR Package 1.1)</td>
</tr>
</tbody>
</table>
RTI LIN MultiMessage Blockset

Where to go from here

Information in this section

<table>
<thead>
<tr>
<th>New Features of the RTI LIN MultiMessage Blockset 1.7.1</th>
<th>61</th>
</tr>
</thead>
<tbody>
<tr>
<td>Migrating to RTI LIN MultiMessage Blockset 1.7.1</td>
<td>62</td>
</tr>
</tbody>
</table>

New Features of the RTI LIN MultiMessage Blockset 1.7.1

Layout generation with ControlDesk’s Bus Navigator

The RTI LIN MultiMessage Blockset supports layout generation with ControlDesk’s Bus Navigator. If Bus Navigator support is enabled, the Bus Navigator lets you create layouts to view RX frames, transmit TX frames, and manipulate TX frames before transmission. Preconfiguring layout generation on the Layouter Options page therefore is not necessary when you work with ControlDesk’s Bus Navigator.

For further information, refer to Experimental Software Page (RTILINMM MainSetup) (RTI LIN MultiMessage Reference).
Migrating to RTI LIN MultiMessage Blockset 1.7.1

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 contained RTILINMM blocks before you perform modifications to the LIN configuration.

To create new S-functions for all the RTILINMM blocks in your model in one step, you can select the Create S-Function for all LIN Blocks command from the Options menu of the RTILINMM GeneralSetup block.

For further information, refer to Limitations of RTI LIN MultiMessage Blockset (RTI LIN MultiMessage Reference).
New Features of the RTI FPGA Programming Blockset 1.1

Main features of the Processor Interface sublibrary

The Processor Interface sublibrary provides the RTI blocks to implement the communication between the FPGA application and the processor application.

The main features of the processor interface of the RTI FPGA Programming Blockset are:

- Reading from and writing to the FPGA storage that can be accessed via the PHS bus.
- Accessing interrupts specified in the FPGA application.
- Managing the processor model and the FPGA model.

With the setup blocks of the processor interface, you can:

- Assign the FPGA models to the available I/O boards
- Control the build process for the processor model
- Program the FPGA application
- Configure the simulation parameters

For further information, refer to **RTI FPGA Programming Blockset - Processor Interface Reference**.
The FPGA Interface sublibrary provides RTI blocks for implementing the interface between the FPGA mounted on a dSPACE I/O board and the board’s I/O. They let you configure the access to external I/O and to a PHS-bus-based processor board, for example, a DS1005. The predefined FPGA framework provides several I/O functions, for example, ADC, DAC, Digital In and Digital Out.

**New features coming with Version 1.1** There are two new I/O functions implemented in the DS5203 with onboard I/O framework:

- **LED Out**
  To switch the LED of the dSPACE I/O board from green to yellow to display any specific activity or state of your application.

- **Status In**
  To get the state of the initialization sequence that is started after programming the FPGA.

The digital channels can now be used bidirectionally as input and output channels.

ADC and DAC functions provide a scaling parameter to switch between Volt and Bit values.

The FPGA_PHS_READ_BL block now additionally provides the Data New port to indicate changes of the data status.

Limited availability of the RTI FPGA Programming Blockset - FPGA Interface outside of Europe and Asia, please inquire.

---

<table>
<thead>
<tr>
<th>Main features of the FPGA Interface sublibrary</th>
</tr>
</thead>
<tbody>
<tr>
<td>The FPGA Interface sublibrary provides RTI blocks for implementing the interface between the FPGA mounted on a dSPACE I/O board and the board’s I/O. They let you configure the access to external I/O and to a PHS-bus-based processor board, for example, a DS1005. The predefined FPGA framework provides several I/O functions, for example, ADC, DAC, Digital In and Digital Out.</td>
</tr>
</tbody>
</table>

**New features coming with Version 1.1** There are two new I/O functions implemented in the DS5203 with onboard I/O framework:

- **LED Out**
  To switch the LED of the dSPACE I/O board from green to yellow to display any specific activity or state of your application.

- **Status In**
  To get the state of the initialization sequence that is started after programming the FPGA.

The digital channels can now be used bidirectionally as input and output channels.

ADC and DAC functions provide a scaling parameter to switch between Volt and Bit values.

The FPGA_PHS_READ_BL block now additionally provides the Data New port to indicate changes of the data status.

Limited availability of the RTI FPGA Programming Blockset - FPGA Interface outside of Europe and Asia, please inquire.

---

<table>
<thead>
<tr>
<th>New simulation feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>If you want to simulate the processor model and the FPGA model in Simulink simulation mode, you do not have to connect the blocks via signals. The data connection for the offline simulation has now been implemented internally.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Demo model</th>
</tr>
</thead>
<tbody>
<tr>
<td>The RTI FPGA Programming Blockset - FPGA Interface now provides a demo model that you can access via the blockset’s library.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hardware support</th>
</tr>
</thead>
<tbody>
<tr>
<td>You can use the RTI FPGA Programming Blockset with the DS5203 FPGA Board. This I/O board can be integrated in a modular system based on a DS1005 or DS1006 processor board.</td>
</tr>
</tbody>
</table>

For further information, refer to *Modular Systems Hardware Installation and Configuration Reference*. |
Migrating to RTI FPGA Programming Blockset 1.1

Replacing RTI blocks

Because the RTI FPGA Programming Blockset 1.0 was not fully implemented, a model that you implemented with it must be migrated manually. You must replace each block of the RTI FPGA Programming Blockset with a new one to get a model compatible with the current dSPACE RTI environment for modeling, building and executing.
SystemDesk

New Features of SystemDesk 2.1

SystemDesk 2.1 comes with several new features.

<table>
<thead>
<tr>
<th>AUTOSAR Releases and formats supported by SystemDesk 2.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>SystemDesk supports:</td>
</tr>
<tr>
<td>- AUTOSAR Release 3.1 with Versions 3.1.0 and 3.1.2 (new in SystemDesk 2.1)</td>
</tr>
<tr>
<td>- AUTOSAR Release 3.0 with Versions 3.0.0, 3.0.2, and 3.0.4 (continued)</td>
</tr>
<tr>
<td>- AUTOSAR Release 2.1 with Versions 2.1.2 and 2.1.4 (continued)</td>
</tr>
</tbody>
</table>

**Supported AUTOSAR formats**  SystemDesk supports import and export of:

- AUTOSAR software component description files
- AUTOSAR ECU parameter configuration files
- AUTOSAR system description files (new in SystemDesk 2.1)

For example, the interface descriptions for AUTOSAR software components can be created, or existing components can be imported to SystemDesk for further processing.
Refer to Basics on AUTOSAR Import/Export (Common to Import and Export) (SystemDesk Guide).

### Modeling the software architecture

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mode switch access</strong></td>
<td>You can assign mode declaration groups of provided mode switch interfaces to a runnable. Refer to Mode Switch Access Page (Runnable) (SystemDesk Reference).</td>
</tr>
<tr>
<td><strong>Mode switched ack event</strong></td>
<td>You can select the mode switched ack event to trigger runnables. The runnable will then be executed each time a mode of the referenced mode declaration group is switched. Refer to Basics on RTE Events (SystemDesk Guide).</td>
</tr>
<tr>
<td><strong>Init values for calprms values of required calprm interfaces</strong></td>
<td>You can specify an init value for a calprm value also if the calprm value is the child of a required calprm interface. This is useful, for example, when you export a software component with a required calprm interface for simulation in TargetLink: Specifying an init value for the calprm value of that software component makes it possible to simulate the component in TargetLink. Refer to General Page (Calprm Value) (SystemDesk Reference).</td>
</tr>
</tbody>
</table>

### RTE Generation Module

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Supported AUTOSAR features</strong></td>
<td>RTE code generation with SystemDesk 2.1 supports the following features of the AUTOSAR specification:</td>
</tr>
<tr>
<td>Data element invalidation</td>
<td></td>
</tr>
<tr>
<td>Transmission acknowledgement</td>
<td></td>
</tr>
<tr>
<td>Communication timeout</td>
<td></td>
</tr>
<tr>
<td><strong>Integration of software component object code</strong></td>
<td>SystemDesk lets you integrate application software component object code if the application header file is generated in compatibility mode. Refer to Integrating Object Code (SystemDesk Guide).</td>
</tr>
</tbody>
</table>

### Simulation Module

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measuring variables in a SystemDesk simulation (alternative method)</td>
<td>SystemDesk provides the new SimulationLogging option for RTE code generation. If enabled, global variables are generated for all the data elements, interrunnable variables and operation arguments of the software architecture. As a result, all these variables can be measured in a SystemDesk simulation. Using this option has the advantage that you do not have to specify measurement access for each variable individually. Refer to Basics on Configuring RTE Code Generation (SystemDesk Guide) and General Page (RTE Configuration) (SystemDesk Reference).</td>
</tr>
</tbody>
</table>
Experimenting on the offline simulation platform using CalDesk
CalDesk can communicate with the offline simulation platform via
XCP on Ethernet. This allows you to use CalDesk’s experimenting
capabilities in connection with the offline simulation of an
AUTOSAR system running in SystemDesk. Refer to Accessing a
Simulation Application from CalDesk (SystemDesk Guide).

Simulation runs can be accelerated and decelerated in relation to
the approximated real-time simulation. Refer to Preferences
(SystemDesk Reference).

SystemDesk lets you specify the step length for single-step
simulation. Refer to Preferences (SystemDesk Reference).

Offline simulation in a separate process
Offline simulation runs in a separate process from that of the
SystemDesk application. For example, if the code to be simulated
contains errors, the simulation process can crash, but since
SystemDesk runs in another process, you can continue working.

MSVC 9 is supported as a compiler for building VPUs.

**Migrating to SystemDesk 2.1**

<table>
<thead>
<tr>
<th>Reusing automation scripts created for SystemDesk 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>SystemDesk’s automation API has been changed with SystemDesk 2.1. You have to migrate automation scripts that you have created for SystemDesk versions prior to SystemDesk 2.1 to the changed API. For migration information on changed automation elements, refer to Changed Automation Elements (SystemDesk Guide).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Incompatibility between DsOffSim 1.0 and 1.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>The export files generated with DsOffSim 1.0 (part of SystemDesk 2.0) are not compatible with the export files generated with DsOffSim 1.1 (part of SystemDesk 2.1). As a consequence, if your SystemDesk project contains a Simulink® model built with DsOffSim 1.0 and integrated as a software component, you have to reintegrate it in SystemDesk when you rebuild the model with DsOffSim 1.1. Perform the following steps: 1. Rebuild the export files for the Simulink® models with DsOffSim 1.1. For instructions, refer to Preparing and Building a Simulink Model for Integration into SystemDesk (dSPACE Target for Offline Simulation).</td>
</tr>
</tbody>
</table>
2. Reintegrate the Simulink® models as atomic software components in SystemDesk 2.1. For instructions, refer to How to Integrate a Simulink Model as an Atomic Software Component (SystemDesk Guide).

If you have both DsOffSim 1.0 and 1.1 installed on your PC, you can select the DsOffSim version to be loaded when you start MATLAB.
## TargetLink

### Where to go from here

<table>
<thead>
<tr>
<th>Information in this section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Features of TargetLink 3.1 and dSPACE Data Dictionary 2.0</td>
<td>72</td>
</tr>
<tr>
<td>Migrating to TargetLink 3.1 and dSPACE Data Dictionary 2.0</td>
<td>101</td>
</tr>
</tbody>
</table>
New Features of TargetLink 3.1 and dSPACE Data Dictionary 2.0

For last minute information on TargetLink 3.1 and on potential difficulties regarding model upgrades, you are recommend to visit the TargetLink 3.1 website at http://www.dspace.com/goto?TargetLinkDocumentationUpdate.

Where to go from here

<table>
<thead>
<tr>
<th>Information in this section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Production Code Generation Features</td>
<td>72</td>
</tr>
<tr>
<td>New AUTOSAR-Related Features</td>
<td>92</td>
</tr>
<tr>
<td>New Features of the dSPACE Data Dictionary</td>
<td>98</td>
</tr>
</tbody>
</table>

New Production Code Generation Features

Where to go from here

<table>
<thead>
<tr>
<th>Information in this section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code Generation from the dSPACE Data Dictionary</td>
<td>73</td>
</tr>
<tr>
<td>New TargetLink Bit Operation Blocks</td>
<td>77</td>
</tr>
<tr>
<td>Enhancements to the Target Simulation Module</td>
<td>78</td>
</tr>
<tr>
<td>Requirements Traceability down to Generated Code and Documentation</td>
<td>81</td>
</tr>
<tr>
<td>Code Generation with Variable Vector Widths</td>
<td>81</td>
</tr>
<tr>
<td>Enhancements and Changes to the TargetLink Main Dialog</td>
<td>82</td>
</tr>
<tr>
<td>Code Generator Options</td>
<td>86</td>
</tr>
<tr>
<td>General Enhancements and Changes</td>
<td>89</td>
</tr>
</tbody>
</table>
Code Generation from the dSPACE Data Dictionary

Objective

TargetLink 3.1 offers production code generation straight from the dSPACE Data Dictionary. Beside incremental code generation and model referencing, this is another means of partitioning your models and mapping the model or model parts and the generated C code files. TargetLink allows you to generate code for code generation units (CGUs), which can be either TargetLink subsystems, subsystems configured for incremental code generation, referenced models, or CodeGenerationUnit objects in the DataDictionary. For further information, refer to Mapping Models to Production Code ([11] TargetLink Advanced Practices Guide) and Introduction to Component-Based Development and Code Generation ([11] TargetLink Advanced Practices Guide).

Use cases of code generation from the dSPACE Data Dictionary

With regard to DD-based code generation, the use cases can roughly be mapped to partitioning the model into separate components, central specification and generation of common modules, and support of A2L file generation as a whole.

Splitting models into more manageable components

TargetLink lets you split large TargetLink models into smaller ones, i.e., more manageable components such as TargetLink subsystems, referenced models, or subsystems configured for incremental code generation, and generate code for them separately.

If you want to generate code for common interface variables that are used in different subsystems, for example, it is important that the resulting modules that hold the production code do not overwrite each other unintentionally. To prevent mutual overwriting of modules by different model components, TargetLink lets you specify the module ownership of a module. This guarantees that a module is generated only together with its owner.

TargetLink lets you compile model components separately. If a model component requires variables from a module that is not yet generated and does not belong to the model component, code can be generated, but not compiled as the variable definitions are missing. Thus, for model components using objects from foreign modules, stub code versions of these modules are generated. Using these stub code versions allows you to compile and simulate the model components.
TargetLink uses production code as well as stub code for code compilation to guarantee the completeness of required code files. Thus, it lets perform tests in a component-oriented way, i.e., you can perform MIL and PIL simulations even with model components that are not the owner of certain modules, or interface variables, respectively. This allows you to test a single component, several components at a time, or to perform a complete system test.

**Central specification and generation of globally used modules**

TargetLink lets you assign variables to a specific module, for example, all calibration variables to a separate module. The generated module contains all project global data (variables) independent of their use in a specific model.

TargetLink lets you generate parameter code independently of the function algorithm. As parameters change more frequently than functional code and functional code does not have variants, it is advantageous to have the functional code separated from the variable code (variable declaration/definition).

**Generating ASAP2 files from the Data Dictionary**

TargetLink allows you to provide a measurement and calibration (MC) system with all the information required for accessing an ECU for calibration. You can, for example, generate calibration or measurement variables used in multiple TargetLink subsystems into a single module, and generate parameter code independently of the function algorithm. Using Look-Up Table objects from the DD, you can generate ASAM-MCD 2MC (A2L) files listing all the parameters and measurement variables running on an ECU as a whole. In order to generate the A2L file, all the lookup table specifications of the Look-Up Table variables must also exist in the dSPACE Data Dictionary.

**Integrating external code**

TargetLink allows the integration of external code. TargetLink modules can integrate external code by including external code files or by providing variable definitions for external code. Both can be specified in the Data Dictionary. This allows external modules to use, for example, calibratable variables that are not defined in the external modules themselves but in the calibration module that is to be generated by TargetLink.

For further information, refer to Use Cases of Model-Component-Based Development and Code Generation (TargetLink Advanced Practices Guide).
Technically, the above use cases are realized by objects in the pool area of the dSPACE Data Dictionary that are relevant for DD-based code generation.

Module objects allow you to describe your application on a file basis as a pair consisting of one *.c and one *.h file.

ModuleOwnerShip The ModuleOwnerShip object lets you define the owner of a module precisely. It specifies which code generation unit (CGU) generates the production code version of the module.

CodeGenOptionsSet Code Generator option sets allow you to configure the Code Generator for production code generation when generating code for a DD CodeGenerationUnit.

BlockSpecifications Currently, you can use block specifications for table blocks such as Look-Up Table (1D), Look-Up Table (2D), Prelookup and Interpolation Using Prelookup.
For further information, refer to Introduction to Component-Based Development and Code Generation (TargetLink Advanced Practices Guide).

New file deposition structure

In contrast to older versions, TargetLink 3.1 does not place the generated production code directly in the working directory, but in a specific file deposition structure. TargetLink creates a clear subfolder structure to provide completely separate files for the individual code generation units. The following illustration shows an example:

For details, refer to File Deposition Structure of the Files Generated by the TargetLink Code Generator (TargetLink File Reference).
New TargetLink Bit Operation Blocks

With TargetLink 3.1, native bit-operation blocks are introduced.

- **Bit Clear block**
  To set a specified bit of an integer signal to zero. For further information, refer to Bit Clear Block (TargetLink Block and Object Reference).

- **Bit Set block**
  To set a specified bit of an integer signal to one. For further information, refer to Bit Set Block (TargetLink Block and Object Reference).

- **Bitwise Operator block**
  To perform a specified bitwise operation on the integer input signal. For further information, refer to Bitwise Operator Block (TargetLink Block and Object Reference).

- **Extract Bit block**
  To output a selection of contiguous bits from an integer input signal. For further information, refer to Extract Bits Block (TargetLink Block and Object Reference).
- **Shift Arithmetic block**
  To perform a bitshift operation on an integer signal. For further information, refer to *Shift Arithmetic Block* *(TargetLink Block and Object Reference)*

## Enhancements to the Target Simulation Module

The following table shows the combinations of evaluation boards, microcontrollers, and compilers supported by TargetLink 3.1 *(TargetLink abbreviations)*. New evaluation boards, microcontrollers, and compiler versions are underscored. For details, refer to *TargetLink Target Reference*.

<table>
<thead>
<tr>
<th>Evaluation Board</th>
<th>Microcontroller Type</th>
<th>Compiler¹</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCT HCS12 T-Board (DP256)</td>
<td>Freescale MC9S12DP256</td>
<td>Cosmic 4.5, 4.6, 4.7</td>
</tr>
<tr>
<td>MCT HCS12 T-Board (DP512)</td>
<td>Freescale MC9S12DP512</td>
<td>No longer supported: Cosmic 4.4</td>
</tr>
<tr>
<td>MCT HCS12 T-Board (DP256)</td>
<td>Freescale MC9S12DP256</td>
<td>Metrowerks CodeWarrior 3.1</td>
</tr>
<tr>
<td>MCT HCS12 T-Board (DP512)</td>
<td>Freescale MC9S12DP512</td>
<td>No longer supported: Metrowerks CodeWarrior 1.2, 2.0</td>
</tr>
<tr>
<td>Freescale 56F8367 Evaluation Module</td>
<td>Freescale MC6F8367</td>
<td>Metrowerks CodeWarrior 8.1</td>
</tr>
<tr>
<td>Axiom CMD-0565</td>
<td>Freescale MPC565</td>
<td>Wind River Diab 5.3, 5.5, 5.6, 5.7</td>
</tr>
<tr>
<td>Axiom CME-0555</td>
<td>Freescale MPC555</td>
<td>Green Hills 4.2, 5.0, 5.1</td>
</tr>
<tr>
<td>Axiom CME-0555</td>
<td>Freescale MPC555</td>
<td>No longer supported: Green Hills 3.0, 3.5, 3.6, 4.0</td>
</tr>
<tr>
<td>Axiom CME-0555</td>
<td>Freescale MPC555</td>
<td>Metrowerks CodeWarrior 8.1, 8.5, 8.7</td>
</tr>
<tr>
<td>Axiom MPC5554DEMO</td>
<td>Freescale MPC5554</td>
<td>Green Hills 4.2, 5.0, 5.1</td>
</tr>
<tr>
<td>Axiom CME-0555</td>
<td>Freescale MPC555</td>
<td>No longer supported: Green Hills 4.0</td>
</tr>
<tr>
<td>Evaluation Board</td>
<td>Microcontroller Type</td>
<td>Compiler¹/²</td>
</tr>
<tr>
<td>------------------</td>
<td>----------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Axiom MPC5554DEMO</td>
<td>Freescale MPC5554</td>
<td>Metrowerks CodeWarrior 2.2, 2.3, 2.4</td>
</tr>
<tr>
<td>Axiom MPC5554DEMO</td>
<td>Freescale MPC5554</td>
<td>No longer supported: Metrowerks CodeWarrior 1.5</td>
</tr>
<tr>
<td>dSPACE DS1603</td>
<td>Freescale MPC5554</td>
<td>Wind River Diab 5.3, 5.5, 5.6, 5.7</td>
</tr>
<tr>
<td>dSPACE DS1603</td>
<td>Freescale MPC5554</td>
<td>No longer supported: Wind River Diab 5.2</td>
</tr>
<tr>
<td>Freescale MPC5561EVB</td>
<td>Freescale MPC5561</td>
<td>Green Hills 5.0, 5.1</td>
</tr>
<tr>
<td>Freescale MPC5561EVB USB</td>
<td>Freescale MPC5561</td>
<td>Wind River Diab 5.6, 5.7</td>
</tr>
<tr>
<td>Freescale MPC5561EVB</td>
<td>Freescale MPC5561</td>
<td>Metrowerks CodeWarrior 2.3, 2.4</td>
</tr>
<tr>
<td>Freescale MPC5561EVB USB</td>
<td>Freescale MPC5561</td>
<td>GNU 3.4, 4.1</td>
</tr>
<tr>
<td>Freescale MPC5561EVB USB</td>
<td>Freescale MPC5561</td>
<td>Wind River Diab 5.3, 5.5, 5.6, 5.7</td>
</tr>
<tr>
<td>Freescale MPC5561EVB USB</td>
<td>Freescale MPC5561</td>
<td>Microtec 3.2, 3.3, 3.5</td>
</tr>
<tr>
<td>Freescale MPC5561EVB</td>
<td>Freescale MPC5561</td>
<td>Wind River Diab 5.3, 5.5, 5.6, 5.7</td>
</tr>
<tr>
<td>MCT S12X T-Board</td>
<td>Freescale MC9S12XDPS12</td>
<td>Cosmic 4.6, 4.7</td>
</tr>
<tr>
<td>MCT S12X T-Board USB</td>
<td>Freescale MC9S12XDPS12</td>
<td>Metrowerks CodeWarrior 4.6, 4.7</td>
</tr>
<tr>
<td>MCT S12X T-Board</td>
<td>Freescale MC9S12XDPS12</td>
<td>No longer supported: Metrowerks CodeWarrior 4.1, 4.5</td>
</tr>
<tr>
<td>Infineon TBTC1766</td>
<td>Infineon c167</td>
<td>Altium Tasking 7.5, 8.0, 8.5, 8.6, 8.7</td>
</tr>
<tr>
<td>Infineon TBTC1766</td>
<td>Infineon TC1766</td>
<td>No longer supported: Altium Tasking 6.0</td>
</tr>
<tr>
<td>Infineon TBTC1767</td>
<td>Infineon TC1767</td>
<td>Altium Tasking 2.3, 2.5, 3.0, 3.2</td>
</tr>
<tr>
<td>Infineon TBTC1796</td>
<td>Infineon TC1796</td>
<td>No longer supported: Altium Tasking 2.2</td>
</tr>
<tr>
<td>Infineon TBTC1796</td>
<td>Infineon TC1796</td>
<td>Altium Tasking 2.3, 2.5, 3.0, 3.2</td>
</tr>
<tr>
<td>Infineon SK-EB XC2287</td>
<td>Infineon TC1796</td>
<td>No longer supported: Altium Tasking 2.2</td>
</tr>
<tr>
<td>NEC Fx3-CAN it!</td>
<td>NEC V850ES/FG3-µPD70F3377</td>
<td>Green Hills 4.2, 5.0, 5.1</td>
</tr>
<tr>
<td>NEC Fx3-CAN it!</td>
<td>NEC V850ES/FG3-µPD70F3377</td>
<td>No longer supported: Green Hills 3.5, 4.0</td>
</tr>
<tr>
<td>NEC Fx3-CAN it!</td>
<td>NEC V850ES/FG3-µPD70F3377</td>
<td>NEC 3.10, 3.20, 3.30</td>
</tr>
<tr>
<td>Renesas EVB2633F</td>
<td>Renesas H8S/2633F</td>
<td>No longer supported: Renesas 6.0, 6.2</td>
</tr>
<tr>
<td>Renesas M3A-2154</td>
<td>Renesas M32192</td>
<td>Gaio 9, 10</td>
</tr>
</tbody>
</table>
For detailed information on the evaluation boards supported by TargetLink, refer to Combinations of Evaluation Boards, Microcontrollers, and Compilers (TargetLink Target Reference).

### Discontinued boards

**No longer supported, no longer distributed**

The following boards are no longer supported by TargetLink and no longer distributed by dSPACE:

- NEC DriveIt Evaluation Board
- Renesas Evaluation Board MSA2114
- FS Forth-Systeme Start276 Development Board
- Texas Instruments TMS470R1x Evaluation Board

**No longer supported, still distributed**

The following board is no longer supported by TargetLink but still distributed by dSPACE:

- Infineon TriBoard TC1775 Evaluation Board

To use the unsupported evaluation boards with TargetLink 3.1, please contact dSPACE.

### Still supported, no longer distributed

The following boards are still supported by TargetLink but no longer distributed by dSPACE:

- Renesas EVB7055F Evaluation Board
- Renesas EVB7058 Evaluation Board
- Axiom CMD-0565 Evaluation Board
- MCT HCS12 T-Board (DP256) and Freescale M68EVB912DP256 Evaluation Boards
# Requirements Traceability down to Generated Code and Documentation

### Including requirement information in code and documentation
TargetLink supports the requirement management interface of Simulink Verification and Validation™ software by The MathWorks.

TargetLink blocks can be linked to software requirements, which are stored in requirement documents or requirement management tools in the same way as with Simulink and Stateflow objects. If a TargetLink model contains objects which are linked to requirements, TargetLink can include information on these requirements in the generated code and/or in the generated documentation.

This traceability between requirement, model and code is helpful if you want to verify whether a requirement is correctly implemented in the software, for example, when inspecting the generated code. For further information, refer to Requirement Information in the Documentation (TargetLink Production Code Generation Guide).

### Requirements linked to TargetLink subsystems
TargetLink ignores requirements linked to a TargetLink subsystem via the subsystem’s context menu. Therefore, you must not use a subsystem’s context menu to link a requirement to it.

To link a requirement to a TargetLink subsystem, open it and use the commands in its menu.

---

# Code Generation with Variable Vector Widths

TargetLink 3.1 can generate varying-width code for vectorized variables. The width of the variables can be adjusted at compilation time.

Example for code with variable vector widths:

```c
Int16 Sa5_Gain[MY_WIDTH];
...
for (Aux_S32 = 0; Aux_S32 < MY_WIDTH; Aux_S32++) {
  ...
  Sa5_Gain[Aux_S32] = ...
}
```

For further information, refer to Specifying Variable Vector Widths (TargetLink Advanced Practices Guide).
Enhancements and Changes to the TargetLink Main Dialog

Redesign of the TargetLink Main Dialog

As of TargetLink 3.1, the TargetLink Main Dialog has been redesigned.

Code Generation page

The TargetLink Main Dialog is now used to trigger code generation not just for TargetLink subsystems, but also for incremental subsystems, referenced models and Data Dictionary CodeGenerationUnit objects. For further information, refer to Component-Based Development and Code Generation (TargetLink Advanced Practices Guide).

The TargetLink Main Dialog offers new controls for the build process in MIL, SIL and PIL simulation modes. The Build SIL and Build PIL menu buttons let you generate and compile the production code for the host and the target simulation application, respectively. The simulation application is then loaded to the RAM of your development PC (SIL) or to your evaluation board (PIL). You can invoke the build process steps separately by selecting the commands on the menu button.
The Activate MIL, Activate SIL, and Activate PIL buttons are an easy way to activate a specific simulation mode.

The Activate SIL and Activate PIL buttons are active only if you built a corresponding simulation application beforehand.

The Code generation mode drop-down list lets you specify the code generation mode. The following code generation modes are available:
- Standard
- AUTOSAR
- RTOS

For details on the Code Generation Page of the TargetLink Main Dialog, refer to Code Generation Page (Main Dialog Block) (TargetLink Block and Object Reference).

With the Code generation mode drop-down list, the Enable multirate code generation checkbox on the RTOS page is obsolete and was therefore removed.

The Code generation options drop-down list was renamed Code generation target settings.

The Change ID button was removed. You can now change the ID of a code generation unit by selecting the Change subsystem ID command from the context menu of the TargetLink subsystem you selected in the Code generation units tree.
With TargetLink 3.1, the Model Referencing Control Center has been discontinued. To manage models containing model references, TargetLink now provides commands in the context menus of the referenced models listed in the Code generation units tree of the TargetLink Main Dialog.

The new Logging in referenced models drop-down list lets you specify logging options for referenced models. The following logging options are available:

- Enable in all referenced models
- Disable in all referenced models
- Enable in selected models

For details on model referencing, refer to TargetLink Advanced Practices Guide.

**Simulation page** The new Target simulation frame comprises the following controls:

- Evaluate stack consumption
- Measure execution time
- Target simulation configuration
As of TargetLink 3.1, the Disable target timeout checkbox is no longer available. However, you can still set this option via the API command `tl_set(gcbh, 'frameopt.targettimeout','on')`. With this flag set to 1 (on), the host PC will always wait for a response from the EVB target, which may cause MATLAB to hang if the EVB does not respond. You should use this option if you run the EVB in a target debugger.

**Tools page** The ASAP 1b interface drop-down list allows you to generate information specifically for ASAM-MCD 1b interfaces into the ASAM-MCD 2MC (A2L) file. You can specify several ASAM-MCD 1b interfaces. The following ASAM-MCD 1b interfaces are supported:

- ADDRESS
- DIM
- CCP
- XCP
- DCI_GME1
- DCI_GSI1
- ETK
- CANAPE_EXT
For further information, refer to Main Dialog Block (TargetLink Block and Object Reference).

Preferences dialog  The Preferences dialog of TargetLink 3.1 provides new synchronization options for updating TargetLink property values to the values of corresponding Simulink properties, and vice versa. You can selectively control the synchronization process by selecting synchronization options.

![TargetLink Preferences Dialog](TargetLink Block and Object Reference)

For details on the Preferences dialog, refer to TargetLink Preferences Dialog (TargetLink Block and Object Reference).

### Code Generator Options

| Sets of Code Generator options | TargetLink lets you configure Code Generator options to tailor the code generation process. As of TargetLink 3.1, you can also configure different sets of options, each of which influences code generation in a different way. As a result, you can analyze and test the effects of different Code Generator options on the generated code. Depending |

86  New Features and Migration  December 2009
on the type of code generation unit you want to create production code for, there are different ways of accessing these Code Generator options. You can set Code Generator options easily by clicking the All Options button on the Advanced Page of the TargetLink Main Dialog or by double-clicking a CodegenOptions object in the dSPACE Data Dictionary. The type of code generation unit also determines where the options are stored. For a TargetLink subsystem, a subsystem configured for incremental code generation, or a referenced model, the Code Generator options are stored at the model, but when you configure Code Generator options for producing C code for CodeGenerationUnit objects, they are stored in the dSPACE Data Dictionary. In contrast to the Code Generator options you specify via the TargetLink Main Dialog, an option set object applies only to those CGU objects it is assigned to. Code Generator options you configured for TargetLink subsystems, incremental subsystems, or referenced models via the TargetLink Main Dialog do not affect CodegenOptions objects stored in the dSPACE Data Dictionary and vice versa.

TargetLink provides the API commands `dsdd_import_optionset` (dSPACE Data Dictionary MATLAB API Reference) and `dsdd_export_optionset` (dSPACE Data Dictionary MATLAB API Reference) to exchange Code Generator options between TargetLink and the dSPACE Data Dictionary, and to compare a CodegenOptions object (`dsdd_compare_optionset` (dSPACE Data Dictionary MATLAB API Reference)) stored in the dSPACE Data Dictionary with Code Generator options you configured via the TargetLink Main Dialog. For further information, refer to Basics of Configuring the Code Generator for Production Code Generation (TargetLink Production Code Generation Guide).

New Options for the Code Generator

TargetLink 3.1 provides the following new Code Generator options.

<table>
<thead>
<tr>
<th>Code Generator Option</th>
<th>Subject Area in the Code Generator Options Dialog</th>
<th>TargetLink 3.1 Default</th>
<th>Most Compatible TargetLink 2.x / 3.x Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowDuplicationOfImplicitAUTOSARDataAccess</td>
<td>Code Generation \ AUTOSAR</td>
<td>ON</td>
<td>OFF</td>
</tr>
<tr>
<td>AssumeFunctionCallSemanticsForRteAPIArguments</td>
<td>Code Generation \ AUTOSAR</td>
<td>ON</td>
<td>ON</td>
</tr>
<tr>
<td>CodeGenerationMode</td>
<td>TargetLink Main Dialog</td>
<td>Standard</td>
<td>Standard</td>
</tr>
<tr>
<td>ForceCodeGenOfMisspecifiedVariableWidthsBlocks</td>
<td>Code Generation \ Pattern</td>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td>EnableVariableVectorWidths</td>
<td>Code Generation \ Pattern</td>
<td>OFF</td>
<td>OFF</td>
</tr>
<tr>
<td>HandleUnscaledStateflowExpressionsWithTlType</td>
<td>Stateflow</td>
<td>ON</td>
<td>OFF</td>
</tr>
</tbody>
</table>
New Features and Migration December 2009

New code generation modes

The Code generation mode drop-down list lets you specify the code generation mode. The following code generation modes are available:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>Lets you generate production code without any AUTOSAR or RTOS parts.</td>
</tr>
<tr>
<td>AUTOSAR</td>
<td>Lets you generate AUTOSAR-compliant code according to the supported AUTOSAR standard.</td>
</tr>
<tr>
<td>RTOS</td>
<td>Lets you generate code for multiple sample times.</td>
</tr>
</tbody>
</table>

For further information on the Code Generation Page of the TargetLink Main Dialog, refer to Code Generation Page (Main Dialog Block) ([TargetLink Block and Object Reference](#)) on page 117.

Obsolete and changed Code Generator options

For details on obsolete and changed Code Generator options refer to Obsolete and Changed Code Generator Options on page 117.
General Enhancements and Changes

Code Changes

TargetLink 3.1 provides enhanced and new features as well as bug fixes that all affect the generated code. The new features can be switched on or off by Code Generator options. The table in Code Generator Options on page 86 shows the options and their old and new values. To obtain code that is as close as possible to TargetLink 3.0, set the options’ values according to the right-hand column.

Enlargement of the Simulink menu bar

A set of frequently used TargetLink tools and commands is now accessible via the Simulink menu bar.
Integration of TargetLink Toolbox DEMOS

As of TargetLink 3.1, the TargetLink demo models are integrated in the MATLAB Help Browser. In addition, a short description of the features implemented in the models is given.

Algebraic simplifications

TargetLink 3.1 performs more algebraic optimizations than previous TargetLink versions.

This applies to:

- Bitwise operators (&, |, ^, ~, <<, >>, &=, |=)
- Logical operators (&&, ||, !, ?:)
- Arithmetical operators (+=, -=, *=, /=, %)
- Relations with a Boolean variable and a floating-point constant as operators

Changes in the code result primarily from making use of De Morgan's laws for logical NOT and the corresponding inversion of relations, such as !(a <= b) => (a > b).

Avoiding the generation of multiple log macros

If the vectorized output of a predecessor block of a Switch block is logged, and the Switch block itself has a vectorized condition, the code generated for the predecessor block is not generated into the conditionally executed control flow branch. This avoids the generation of multiple log macros. The same behavior also applies to situations where multiple potential control flow branches exist or where the conditionally executed control flow branch (including the condition) is located within a loop.
As of TargetLink 3.1, the difference between code generated with EnableBlockDiagramBasedSwitchOptimization = OFF and EnableBlockDiagramBasedSwitchOptimization = ON (default) is less significant than with previous TargetLink versions. This applies especially to:

- Code with log macros
- Assignments to variables of a variable class whose ERASABLE property is not set.
- The NoAssignmentOfBooleanExpressions option when it is set to ON.

If CleanCode = ON and NoAssignmentOfBooleanExpressions = OFF (default) are set, the code generated by TargetLink 3.1 is not different from code generated by an earlier TargetLink version.

The standard settings for HandleUnscaledStateflowExpressionsWithTtType can result in code changes originating from Stateflow charts. To be able to apply the code generation rules TargetLink uses for Simulink code, the Stateflow expression autoscale mechanism calculates a result type and a result scaling for each scaled operation specified in a Stateflow chart. A scaled operation is an operation that has at least one operand with an LSB unequal 1.0 and/or an offset unequal 0.0. In previous versions of TargetLink, all unscaled Stateflow expressions were regarded as custom code, i.e., the code specified in the chart was copied to the generated code. As of this version of TargetLink, however, unscaled operations with at least one operand that has different TargetLink and Stateflow data types are now also included in the Stateflow expression autoscale mechanism. This changed behavior usually results in an additional cast operation in the code generated for Stateflow charts and in code that is more similar to code originating from Simulink parts.

TargetLink 3.1 emits more code comments than previous TargetLink versions. This applies especially to:

- Logical and Relation blocks
- Blocks that result in cast operations
- Boundaries of atomic subsystems and inlined functions
- Variables used in custom code that are replaced by corresponding interface variables
In some cases, TargetLink might emit significantly more comments.

### Model referencing information in the headers of generated C files

With TargetLink < 3.1, the headers of generated C files which are not part of a TargetLink default module contain subsystem information:

**SUBSYS CORRESPONDING SIMULINK SUBSYSTEM**

Example:

```
Sa1 pidcontroller
```

and the Stateflow information:

**SF-NODE CORRESPONDING STATEFLOW NODE DESCRIPTION**

Example:

```
Ca1 sf_control/Chart chart description
```

In the header of generated C files, TargetLink now also emits information about model blocks and referenced models in the following form:

**SUBSYS CORRESPONDING MODEL BLOCK (REFERENCED MODEL)**

Example:

```
Sc0   fuelratecontroller/r_AirflowCalculation (r_AirflowCalculation)
```

You can prevent TargetLink emitting information about model blocks and referenced models by setting the `modelref-block-list-comment` switch (subelement of the `<TL:header-comment>` attribute) in the `cconfig.xml` file to `false`.

### Rate Limiter block

In TargetLink 3.1, the Rate Limiter block is able to process vectorized signals.

### New AUTOSAR-Related Features

### Features of the TargetLink AUTOSAR Module

**Support of AUTOSAR Release 3.1**

The TargetLink AUTOSAR Module supports the following AUTOSAR Releases:

- AUTOSAR Release 3.1 with Version 3.1.0 (new)
- AUTOSAR Release 3.0 with Version 3.0.2 (continued)
AUTOSAR Release 2.1 with Version 2.1.4 (continued)

Redesign of the TargetLink AUTOSAR Module

**Redesigned AUTOSAR Block Library** In TargetLink 3.1 the AUTOSAR Block Library has been redesigned as shown in the illustration below.
Introduction of icons for AUTOSAR objects in the Data Dictionary

AUTOSAR-related DD objects are now visualized with specific icons as shown in the illustration below.

Improved support of the TargetLink code generator software’s standard features

- You can use Simulink models with AUTOSAR software components for prototyping.
- You can use models with AUTOSAR blocks in TargetLink’s stand-alone mode.
- You can use AUTOSAR blocks in libraries.
- You can use TargetLink’s documentation generation feature for software components.
**Improved AUTOSAR import/export**

**Support of initialization constants** You can now specify and import/export constants for initialization:

- For data elements via the `InitRef` property of the data element's communication specification. This allows you to initialize scalars, arrays, and structs. For instructions on initializing data elements, refer to *How to Model Initialization of Data Elements* (TargetLink AUTOSAR Modeling Guide).

- For interrunnable variables via the `Value` property of the variable object. The new properties `InitConstantName`, `InitConstantUuid`, and `InitConstantPackage` allow you to specify import/export settings for the constant. For instructions on creating interrunnable variables, refer to *How to Create Interrunnable Variables* (TargetLink AUTOSAR Modeling Guide).

- For calprm elements via the `InitRef` property of the calprm's communication specification. This allows you to initialize scalars, arrays, and structs.

- For SWC-internal calibration parameters via the `Value` property of the variable object. The new properties `InitConstantName`, `InitConstantUuid`, and `InitConstantPackage` allow you to specify import/export settings for the constant. If you want to initialize structs, you have to specify the import/export settings at the variable object and the values at the components.

**Improved support of arrays** TargetLink now supports AUTOSAR-compliant handling of arrays in the generated software component code and exported AUTOSAR file. For instructions on creating arrays, refer to *How to Define Array Types* (TargetLink AUTOSAR Modeling Guide).

**Improved support for software components and runnables**

- Specialized software component types such as application software component type and sensor actuator software component type as defined by AUTOSAR. The software component type is specified in the new `ComponentType` property of a software component.

- You can model a software component’s initialization and termination runnables with individual names via the new `Kind` property.

- Software components have new properties for specifying the names and UUIDs of their implementations and internal behaviors.
Access points and read/write accesses of a runnable for sender-receiver communication, client-server communication, interrunnable variables, calibration and measurement, modes, and per instance memories. The access points and read/write accesses are specified as subnodes of the runnable.

**Support of admin data** You can import/export and specify additional data for each AUTOSAR element as AdminData objects in the Data Dictionary.

**Configuration of default import/export settings** You can configure the default options of AUTOSAR file import/export by editing the global configuration file, which is located at \%DSPACE_ROOT\%\dss\dd Autosar_config.xml. The options are valid for import/export via dialog and API command.

<table>
<thead>
<tr>
<th>Additionally supported RTE API functions</th>
<th>Code generation is now supported for the following RTE API functions:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rte_Invalidate, for invalidating data elements in explicit sender-receiver communication.</td>
<td></td>
</tr>
<tr>
<td>Rte_Invalidate, for invalidating data elements in implicit sender-receiver communication.</td>
<td></td>
</tr>
<tr>
<td>Rte_Feedback, for accessing acknowledgement notifications from the RTE when explicitly sending data elements.</td>
<td></td>
</tr>
<tr>
<td>Rte_IStatus, for accessing the RTE status of a data element that is to be read via Rte_IRead.</td>
<td></td>
</tr>
<tr>
<td>Rte_Pim, for accessing per instance memories.</td>
<td></td>
</tr>
</tbody>
</table>
The illustration below shows how you can model sender-receiver communication with invalidation, acknowledgement notifications, and access to the RTE status using a SenderComSpec block of the redesigned AUTOSAR Block Library. The illustration also shows how you can stimulate acknowledgement notifications and the RTE status via Data Store blocks.

For instructions on modeling sender-receiver communication with invalidation, acknowledgement notifications, and access to the RTE status, refer to "How to Model Invalidation, Acknowledgement Notifications and Access to the RTE Status (TargetLink AUTOSAR Modeling Guide)."

**Improved modeling of AUTOSAR software components**

**Improved support for client-server communication**

You can now use client-server communication involving operations with composite operation arguments (structs and arrays).
Access to RTE status and acknowledgement notifications in simulations
You can stimulate the RTE status of Rte_Send/Rte_Receive RTE API calls, etc., via Simulink Signal objects in simulations, refer to How to Simulate the RTE Status ([TargetLink AUTOSAR Modeling Guide]). You can also stimulate the acknowledgement notifications that are provided by Rte_Feedback RTE API calls via Simulink Signal objects in simulations, refer to How to Simulate Acknowledgement Notifications ([TargetLink AUTOSAR Modeling Guide]).

Modeling AUTOSAR and non-AUTOSAR controllers in one model
TargetLink now supports modeling AUTOSAR and non-AUTOSAR control algorithms in one Simulink/TargetLink model. A global option that is available via TargetLink’s Main dialog allows you to switch between generating AUTOSAR and non-AUTOSAR code, i.e., standard code. For basic information on working with models for AUTOSAR and non-AUTOSAR controllers, refer to Basics on Dual Block Data ([TargetLink AUTOSAR Modeling Guide]).

Improved migration of standard TargetLink models to AUTOSAR
AUTOSAR data is now included in Function, InPort, OutPort, Bus Inport, and Bus Outport blocks from the TargetLink Block Library. This allows improved migration of standard TargetLink models to AUTOSAR. AUTOSAR-specific blocks are required only for RTE status, acknowledgement notification, and invalidation signals.

You can migrate a standard TargetLink model to AUTOSAR using the TargetLink AUTOSAR Migration Tool, which is available for download via http://www.dspace.de/goto?tl_ar_migration.

New Features of the dSPACE Data Dictionary

Objective
dSPACE Data Dictionary 2.0 (DD) has the following new features, enhancements and changes:

Where to go from here
Information in this section

<table>
<thead>
<tr>
<th>New Key Features</th>
<th>99</th>
</tr>
</thead>
<tbody>
<tr>
<td>New and Modified DD MATLAB API Commands</td>
<td>100</td>
</tr>
</tbody>
</table>
New Key Features

Objective

Information on the new key features of dSPACE Data Dictionary 2.0 is provided below.

Message Browser

The Message Browser displays all the messages of the Data Dictionary Manager. It can be opened in the View menu.

<table>
<thead>
<tr>
<th>Severity</th>
<th>Time</th>
<th>Title</th>
<th>Object</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Info</td>
<td>10:45:17</td>
<td>Save DD file</td>
<td>DD</td>
<td>The current DD file was successfully saved.</td>
</tr>
<tr>
<td>Info</td>
<td>10:45:17</td>
<td>Save DD file</td>
<td>DD</td>
<td>The current DD file was successfully saved.</td>
</tr>
<tr>
<td>Info</td>
<td>10:45:25</td>
<td>Save DD file</td>
<td>DD</td>
<td>The current DD file was successfully saved.</td>
</tr>
</tbody>
</table>

Enhanced Object Selection dialog

The Object Selection dialog has been enhanced by a history (1) and the Find Next and Apply Filter buttons (2). The Find Next button lets you search for the next DD reference object which matches the specified string. The Apply Filter button lets you apply the string entered in the edit field filter the DD reference objects displayed in the Reference Selection window.

For further information, refer to How to Reference Data Dictionary Objects (dSPACE Data Dictionary Basic Concepts Guide).
Improved drag and drop handling in the Data Dictionary Navigator

As of TargetLink 3.1, you can see where an object will be dropped. You can also drop an object at an arbitrary place within a group, which was not the case with previous versions.

Loading data for expanded subtrees

When you click Refresh, the DD Navigator does not load all the data from the DD, but only the data for the expanded subtrees. This boosts performance with large data dictionaries (especially ones that have extensive subsystems).

New and Modified DD MATLAB API Commands

New DD MATLAB API commands: SetCurWidth / GetCurWidth

For direct access to the width property, the Data Dictionary MATLAB API provides the new `SetCurWidth` and `GetCurWidth` commands. Their behaviors are identical to the old behaviors of `GetWidth` and `SetWidth`.

References

- `GetWidth` ([dSPACE Data Dictionary MATLAB API Reference](#))
- `SetWidth` ([dSPACE Data Dictionary MATLAB API Reference](#))
- `SetCurWidth` ([dSPACE Data Dictionary MATLAB API Reference](#))
- `GetCurWidth` ([dSPACE Data Dictionary MATLAB API Reference](#))
Migrating to TargetLink 3.1 and dSPACE Data Dictionary 2.0

Upgrade process

To update existing TargetLink models and libraries to TargetLink 3.1, TargetLink provides an update process. The associated DD files must also be updated in addition to models and libraries. This is done by the DD update process.

For last minute information on TargetLink 3.1 and on potential difficulties regarding model upgrades, you are recommended to visit the TargetLink 3.1 website at http://www.dspace.com/goto?TargetLinkDocumentationUpdate.

The upgrade process in TargetLink 3.1 and dSPACE Data Dictionary 2.0 upgrades only models and Data Dictionary files created under TargetLink 2.x or 3.0 (and the associated Data Dictionary versions). Projects that were created under TargetLink 1.3 or even older versions cannot be upgraded directly to TargetLink 3.1 and dSPACE Data Dictionary 2.0. You must first perform an upgrade to a TargetLink 2.x version (including the associated Data Dictionary version) before you can upgrade to TargetLink 3.1 and dSPACE Data Dictionary 2.0.

Where to go from here

Information in this section

<table>
<thead>
<tr>
<th>Model Upgrade</th>
<th>102</th>
</tr>
</thead>
<tbody>
<tr>
<td>How to Upgrade Models and Libraries</td>
<td>103</td>
</tr>
<tr>
<td>Migrating to dSPACE Data Dictionary 2.0</td>
<td>106</td>
</tr>
<tr>
<td>How to Upgrade a Data Dictionary Without Include Files</td>
<td>109</td>
</tr>
<tr>
<td>How to Upgrade a Data Dictionary With Include Files</td>
<td>110</td>
</tr>
<tr>
<td>General Migration Aspects</td>
<td>113</td>
</tr>
<tr>
<td>Modified DD MATLAB API Commands</td>
<td>116</td>
</tr>
<tr>
<td>Obsolete and Changed Code Generator Options</td>
<td>117</td>
</tr>
<tr>
<td>Changes in TargetLink API Functions</td>
<td>117</td>
</tr>
<tr>
<td>Various Migration Aspects</td>
<td>118</td>
</tr>
</tbody>
</table>
Model Upgrade

Upgrading TargetLink models
When you upgrade existing TargetLink models to TargetLink 3.1, you also need to upgrade the associated DD files to dSPACE Data Dictionary 2.0 (this belongs to TargetLink 3.1).

Upgrading referenced models
When you upgrade an existing model with references to other models to TargetLink 3.1, you need to upgrade the referenced models before you upgrade the referencing model.

Unset dirty flag after upgrade
The first time a TargetLink model is opened with Version 3.1, an upgrade routine automatically migrates the model to TargetLink 3.1. In most cases, this results in a modification of the model which is however not indicated by the dirty flag (asterisk) being set.

Related topics
Basics
- How to Upgrade Models and Libraries on page 103
- Migrating to dSPACE Data Dictionary 2.0 on page 106
How to Upgrade Models and Libraries

Objective

If you open an older model, the upgrade routine is called automatically (by a model callback function, though first the data dictionary upgrade routine runs). However, if you open a library, it is not automatically upgraded, as the MDL file does not contain callbacks for this. You therefore have to initiate the library upgrade explicitly using the API command $\texttt{tl\_upgrade}$. To display the options for the command, enter the API command $\texttt{help tl\_upgrade}$ in the MATLAB Command Window.

The following options apply to TargetLink 3.1. It is always advisable to use the default settings.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>Name of the model or library to be upgraded.</td>
</tr>
<tr>
<td>Upgradelibs</td>
<td>Ensures that the libraries referenced by a model are automatically included in the upgrade. However, it is basically recommended to upgrade all libraries manually first, i.e., not to use this option.</td>
</tr>
<tr>
<td>Rebuildsfcn</td>
<td>Rebuilds all the custom code blocks in a model (including, for example, the Bitwise Logical Operator block). This option is automatically enabled in TargetLink 3.1 and should be used here.</td>
</tr>
<tr>
<td>useTL2xportnames</td>
<td>As of TargetLink 3.0, Simulink and TargetLink ports are merged. If you upgrade a TargetLink 2.x model to TargetLink 3.1, this option allows you to use the original name of the TargetLink port (which is removed) as a new name for the connected Simulink port. The original name of the Simulink port then no longer exists. This option is useful in cases such as user-built library blocks with mask initialization code that write data to the TargetLink port. In the future, this data has to be written to the Simulink port, as the TargetLink port and the Simulink port were merged.</td>
</tr>
<tr>
<td>Loadmodelonly</td>
<td>The model is not opened explicitly.</td>
</tr>
</tbody>
</table>

Precondition

If you have used restricted read/write permissions for individual subsystems, you first have to upgrade the read/write permission for all the subsystems to ReadWrite. This ensures that the upgrade routine is allowed to carry out the necessary modifications.
**Method**

**To upgrade models and libraries**

1. If you have used restricted read/write permissions for individual subsystems, upgrade the Read/Write permissions for all subsystems to ReadWrite. This makes sure that subsystem modifications are permitted during the upgrade process.

   ![Subsystem Block Parameters](image)

   You must do this for models before opening them with TargetLink 3.1, as otherwise the upgrade routine appears. If you have to set multiple properties, you can use the Model Explorer.

2. Type `tl_upgrade(<Options>)` in the MATLAB Command Window to upgrade all the Simulink libraries that are used, beginning with libraries that do not themselves reference any other libraries (so the blocks they contain have no links to other libraries).

   You do not have to open the file first. The recommended upgrade settings are as follows:
   - `upgradelibs = "off"`
   - `rebuildsfcn = "on"`
The libraries themselves do not need to be opened exclusively. You have to decide whether you want to use the \texttt{TL2xportname} option. This depends on your modeling style. When the upgrade/update has finished, you must save the libraries (under the same names). You can now continue with the libraries that reference only upgraded libraries.

3 After having upgraded all the referenced user libraries, continue with the TargetLink models and upgrade them.

- The data types for MIL simulation or small, individual code fragments might change after the models are updated automatically to TargetLink 3.1. In some cases this can initially cause problems with model initialization, or code that looks slightly different can be generated. You should therefore check the model initialization/model simulation and the code generation performed with the updated models. If a situation arises where a model has to be made initializable again due to changed Simulink data types after model upgrade, there are no generally valid rules for doing so. In most cases, such initialization errors can be avoided by adjusting the initial data types.

For last minute information on TargetLink 3.1 and on potential difficulties regarding model upgrades, you are recommend to visit the TargetLink 3.1 website at http://www.dspace.com/goto?TargetLinkDocumentationUpdate.

\underline{Related topics}

- Basics
  - Model Upgrade on page 102
Migrating to dSPACE Data Dictionary 2.0

Upgrading Data Dictionary files

When you upgrade existing TargetLink models to TargetLink 3.1, you also need to upgrade the associated DD files to dSPACE Data Dictionary 2.0 (this belongs to TargetLink 3.1).

- The upgrade process in TargetLink 3.1 and dSPACE Data Dictionary 2.0 upgrades only models and Data Dictionary files created under TargetLink 2.x or 3.0 (and the associated Data Dictionary versions). Projects that were created under TargetLink 1.3 or even older versions cannot be upgraded directly to TargetLink 3.1 and dSPACE Data Dictionary 2.0. You must first perform an upgrade to a TargetLink 2.x version (including the associated Data Dictionary version) before you can upgrade to TargetLink 3.1 and dSPACE Data Dictionary 2.0.

For last minute information regarding TargetLink 3.1 as well as potential difficulties regarding model upgrades, you are recommend to visit the TargetLink 3.1 website at http://www.dspace.com/goto?TargetLinkDocumentati

onUpdate.

- For information on upgrading Data Dictionary files containing AUTOSAR objects, refer to AUTOSAR-Related Migration Aspects on page 123.

Deletion of Subsystem and <Application> areas

Due to the changes in the data model, the data in the Subsystem and <Application> areas, which resulted from code generation and build processes performed before the upgrade, is not consistent with new data models. The upgrade of a DD project file therefore deletes the Subsystem and <Application> areas from the Data Dictionary. You are asked to confirm the deletion.

Delete generated objects?

- Subsystems area objects generated with previous TargetLink version detected.
- Linking Subsystems area objects generated by older TargetLink versions can lead to unexpected behavior. It is recommended to delete the Subsystems area and reload the information.
- For generating code with the new TargetLink version.

Do you want to have the objects deleted?

YES    NO
In batch mode, the subsystem node is deleted without request, i.e., only a message is issued. To create correct data in the Subsystem and <Application> areas after the upgrade, for example, data needed for generating ASAP2 files or AUTOSAR XML files, generate code for all the code generation units again.

### Method to upgrade Data Dictionary files

**dSPACE Data Dictionary 2.0** provides an upgrade process that automatically upgrades older DD files to Version 2.0.

The upgrade process can be called in three ways:

- **Automatically opening an old TargetLink model.**
  
  When you open a TargetLink model with an old (not upgraded) DD file, TargetLink first runs the dSPACE Data Dictionary's upgrade process.

- **Via the Tools menu in the Data Dictionary Manager**
  
  Manually via Tools -- Upgrade Current DD in the DD Manager.

---

New Features and Migration  December 2009  107
Data Dictionary API command

To call the upgrade process via the Data Dictionary’s API, type `dsdd('Upgrade')` in the MATLAB Command Window.

Preconditions when upgrading Data Dictionary files

To ensure the DD file is upgraded correctly, the following preconditions must be met:

- There must be write permission for the DD file and the file must not be write-protected. If Data Dictionary Include files are used, there must be write access to all of them.

- If Data Dictionary Include files are used, they must be saved after updating as well as the main DD file. This ensures that the Data Dictionary partitions into several files. To save the Include files correctly, you must first make the appropriate settings for them.

Related topics

- How to Upgrade a Data Dictionary With Include Files on page 110
- How to Upgrade a Data Dictionary Without Include Files on page 109
- Model Upgrade on page 102
How to Upgrade a Data Dictionary Without Include Files

Objective

If you open a TargetLink model with an old, nonupgraded Data Dictionary file, you have to upgrade the Data Dictionary file.

Method

To upgrade a Data Dictionary without Include files

1. Open the model and therefore the referenced dSPACE Data Dictionary, or type `dsdd('Open',<DDFile>)` in the MATLAB Command Window.

   The Data Dictionary needs upgrading dialog automatically opens if an older DD version is involved.

   ![Data Dictionary needs upgrading dialog](image)

2. Click Yes if no Include files are used in the Data Dictionary.

   If Include files are used, refer to How to Upgrade a Data Dictionary With Include Files on page 110.

   The Delete generated objects dialog automatically opens if the Data Dictionary's Subsystem area contains objects generated with a previous TargetLink version.

   ![Delete generated objects dialog](image)

3. Click Yes.

4. Save the data dictionary.

   When you have saved the Data Dictionary (assuming you have write permission to the relevant DD file), the upgrade of the DD file is completed.
How to Upgrade a Data Dictionary With Include Files

Objective
If you open a TargetLink model with an old, nonupgraded Data Dictionary file, you have to upgrade the Data Dictionary file.

Method
To upgrade a Data Dictionary with Include files

1. Open the model and therefore the referenced dSPACE Data Dictionary, or type `dadd('Open',<DDFile>)` in the MATLAB Command Window.
   The Data Dictionary needs upgrading dialog automatically opens if an older DD version is involved.

2. Select No in the upgrade dialog.

3. Under `/Config/DDIncludeFiles`, set the AutoLoad and AutoSave properties for each Include file as shown in the illustration below.
This ensures that after the Data Dictionary and the Include files have been upgraded, the upgraded Include files are saved when the Data Dictionary is saved. You can set these properties for a large number of Include files via the Object Explorer.

4 Start the DD upgrade (including the included files) via Tools - Upgrade Current DD in the DD Manager, or enter `dsdd('Upgrade')` in the MATLAB Command Window.
5 Save the Data Dictionary (assuming you have write permission to the relevant DD file). This completes the upgrade of the DD file itself as well as the included DD files.

Result

The next time you open the DD file, the upgrade dialog will not open because the DD file is up-to-date and so are the included Data Dictionary files. After the files have been properly upgraded, you might want to restore the old settings for the Data Dictionary Include files.
### General Migration Aspects

**Objective**

There are various aspects that you have to note when migrating to TargetLink 3.1.

**New file deposition structure**

TargetLink 3.1 allows you to generate code for code generation units (CGUs), which can be TargetLink subsystems, subsystems configured for incremental code generation, referenced models, or CodeGenerationUnit objects in the Data Dictionary. In contrast to older versions, TargetLink 3.1 does not place the generated production code directly in the working directory, but in a specific file deposition structure.

For the production code, TargetLink generates a `/TLProj` folder which contains one subfolder for each CGU in your model. The names of the subfolders are the names of the CGUs. For the stub code (code generated for model components using objects from foreign modules i.e., variables are declared but not defined), TargetLink generates a `/StubCode` folder in the `/TLSim` folder. Like the `/TLProj` folder, the `/StubCode` folder contains one subfolder for each CGU. For data variants, specific folders are generated with the same structure as the `/TLProj` and `/StubCode` folders.
When an application is built, the files to be built are copied to a build folder under the `/TLSim` folder. If there are still generated files in the working directory, a warning occurs. If there are files to be compiled in the working directory when an application is built, an error message is issued and the build process is stopped. Generated documentation is not stored under `/doc` but under `/TLProj/doc`. TargetLinks generates ASAM MCD 2MC files in the `/TLProj` folder.

The `ModuleOwnership` object in the Data Dictionary specifies which code generation unit (CGU) generates the production code version of a module. For further information, refer to `Introduction to Component-Based Development and Code Generation` (TargetLink Advanced Practices Guide) and to `File Deposition Structure of the Files Generated by the TargetLink Code Generator` (TargetLink File Reference).

---

**Module objects in the pool area**

As of TargetLink 3.1, module objects are created in the Pool area of the Data Dictionary. They represent modules created and/or emitted in the file system during production code generation.

The upgrade of TargetLink models and DD project files creates module objects for TargetLink internal files such as `tl_basetypes.h` or `tl_defines.h` in `/Pool/Modules/TLPredefinedModules/` and for each module name specified at variables, types, variable classes, etc. Hence TargetLink allows you to modify internal modules via the module.
objects. The module templates are applied, and all modules originating from the module templates are created. In TargetLink 3.1, module templates must be used only to modify sets of modules. Templates specifying one specific module must be replaced by a module object. Remember that all module name properties at various DD objects have been replaced by references to a specific module object. Keep in mind that if a DD module object already exists, no module template is applied to it.

During code generation, module templates are not applied to modules that are specified via module objects in the dSPACE Data Dictionary.

**ModuleOwnership**

The ModuleOwnership object lets you define the owner of a module precisely. It specifies which code generation unit (CGU) generates the production code version of the module.

If you used module templates to implement the module ownership in a model in TargetLink < 3.1, you have to create corresponding module ownership objects and use them instead. For further information, refer to *Introduction to Module Ownership* (*TargetLink Advanced Practices Guide*).

**Data types not assigned to a specific module**

In contrast to TargetLink < 3.1, all the data types that are not assigned to a specific module are generated into the user-defined data type header file (*UDT.h*).

**Prevent emitting files**

With TargetLink < 3.1, you could specify subsystems under the <Application> node of the Data Dictionary and assign a module to them to prevent them from being emitted. As of TargetLink 3.1, this specification is obsolete and will be ignored. If you used this mechanism, you have to create module objects and set the EmitFile property or specify an ownership with ModuleOwnership objects instead.

The upgrade does not provide module objects based on this specification, and deletes the Subsystem and <Application> areas from the Data Dictionary.

To prevent such subsystems from being emitted, you can explicitly create module objects in the Data Dictionary and prevent them from being emitted by setting the EmitFile property to off. This applies especially to *tl_basetypes.h*, which exists as a DD module object and which is not emitted.
Data type with buses in referenced models

If the input of a referenced model is a bus, the data type of the bus must be explicitly assigned to a module. Otherwise, an error message tells you that the data type is defined twice. With TargetLink < 3.1, this was the case for models that are connected to the same bus. If you want to distribute the data types in the previous way, you can set the compatibility option \texttt{StrictTypedefPlacement} to false.

Changed behavior of Bus Inport and Outport blocks

TargetLink no longer determines changes in the bus hierarchy automatically when the block dialog is opened. As of TargetLink 3.1 you have to click the Rescan bus hierarchy button in the Bus Inport or Bus Outport dialogs to update the bus hierarchy. The model must be initializable for this. The update of the bus hierarchy may take some time.

Modified DD MATLAB API Commands

Changes to the DD MATLAB API

There is a new feature for code generation from the dSPACE Data Dictionary providing new Module objects in the data model of the dSPACE Data Dictionary.

The upgrade of the Data Dictionary replaces all the ModuleName properties in the Data Dictionary with references to Module objects.

You have to remember this when using the DD MATLAB API, i.e., in scripts.

Changed DD MATLAB API commands: \texttt{SetWidth} / \texttt{GetWidth}

The Data Dictionary MATLAB API provides the \texttt{GetWidth} and \texttt{SetWidth} commands to read and set the \texttt{Width} property of DD variable objects. The behavior of these commands has been changed.

If a DD variable object references a typedef object, the variable object’s \texttt{Width} property is set via the typedef object’s \texttt{Width} property. In this case \texttt{GetWidth} now returns the typedef object’s \texttt{Width}, and \texttt{SetWidth} returns an error because the variable object’s \texttt{Width} can be specified only either via the local \texttt{Width} property or via the \texttt{Width} property of the referenced Typedef object.
Obsolete and Changed Code Generator Options

Obsolete options for the Code Generator

The following Code Generator options are no longer used in TargetLink 3.1:

- **EnableMultirate** The EnableMultirate option is obsolete. Use the CodeGenerationMode = RTOS setting instead. For further information, refer to CodeGenerationMode in the table in New options for the Code Generator.

- **ScopeReductionPreserveLinkerSectionEntriesUserClass** The ScopeReductionPreserveLinkerSectionEntriesUserClass option is no longer used.

- **VariantFileDefaultID** The VariantFileDefaultID option is obsolete. As of TargetLink 3.1 you can set the ModuleRef property directly in the DataVariant object specifying a data variant.

If your model contains obsolete Code Generator options that are set to values other than the default, the model upgrade does not delete these options automatically. If you generate code while the obsolete options are still stored at the model, a warning occurs.

Changed options for the Code Generator

The interpretation of the numerical values (0 to 4) of the Code Generator option Logging was changed. If you use the pre_codegen_hook function to generate clean code or to configure global logging options, this will now create wrong results. For details, refer to tl_pre_codegen_hook.m (*TargetLink File Reference*).

New Code Generator options

For further information on new Code Generator options, refer to Code Generator Options on page 86.

Changes in TargetLink API Functions

In TargetLink 3.1, the following TargetLink API command name has changed.

<table>
<thead>
<tr>
<th>TargetLink 3.0</th>
<th>TargetLink 3.1</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>get_sfobjects</td>
<td>tl_get_sfobjects</td>
<td>The command get_sfobjects no longer exists in TargetLink 3.1 and has been replaced by tl_get_sfobjects.</td>
</tr>
</tbody>
</table>
### Various Migration Aspects

<table>
<thead>
<tr>
<th>Migration Aspect</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>No automatic replacement of bit operation blocks</strong></td>
<td>The model upgrade does not replace Bit operations blocks from the TargetLink Sample Blocks (tlsamples/Bit Operations) with the new bit operation blocks introduced with TargetLink 3.1. If you want to use the new blocks, you have to replace the old ones manually.</td>
</tr>
<tr>
<td><strong>Obsolete EmitCode property</strong></td>
<td>As of TargetLink 3.1, the EmitCode = {on,off} property of FileSpecification objects (child objects to ModuleTemplate objects) no longer exists. It is replaced by the EmitFile={on,off,auto} property. EmitFile=Auto is equivalent to EmitCode=on.</td>
</tr>
<tr>
<td><strong>Fewer parentheses for unary operations</strong></td>
<td>Fewer parentheses are used for unary operations such as incrementing a loop variable.</td>
</tr>
<tr>
<td></td>
<td>In TargetLink &lt; 3.1, unary operations are generated as follows:</td>
</tr>
</tbody>
</table>
|                                                       | ```c
for (i = 0; i < 10; (i)++)
```                                                                                                                                                    |
|                                                       | In TargetLink 3.1, unary operations are generated as follows:                                                                                                                                       |
|                                                       | ```c
for (i = 0; i < 10; i++)
```                                                                                                                                                    |
| **Modified code pattern for loops for vector signals**| The code pattern representing For loops for vector signals has been changed.                                                                                                                                 |
|                                                       | In TargetLink 2.3 and 3.0, For loops were generated with a <= relation in the condition, as shown in the following example:                                                                         |
|                                                       | ```c
for (i = 0; i <= 9; i++) {...}
```                                                                                                                                                           |
|                                                       | In TargetLink 3.1, For loops are generated with a < relation, as shown in the following example:                                                                                                    |
|                                                       | ```c
for (i = 0; i < 10; i++) {...}
```                                                                                                                                                           |
| **Modified code pattern for relational operations**   | In TargetLink 3.1, the scaling in which a relational operation is performed can change. That is, if an operand of a relational operation was part of a scaled operation in a previous version of TargetLink, the other operand of the relational operation might now be part of the inverse scaled operation, as shown in the following example: |
|                                                       | Var1 << 1 > Var2                                                                                                                                  |
|                                                       | might become                                                                                                                                       |
|                                                       | Var1 > Var2 >> 1                                                                                                                                       |
No empty return value during DD upgrade

The upgrade of a DD project file generates warnings and information that are displayed in the Message Browser. The return value of the upgrade function is the message number of the latest message. This also applies to the `dadd('Open',<DDFile>)` command.

Scripts that check for an empty return value will indicate errors even if no errors occurred. After the upgrade, the messages are not displayed.

Input signal-dependent variable structure

If a block supporting signal properties inheritance has a bus signal at its input, the variables of the block are created according to the structure of the bus signal.

This can cause the following kinds of code changes in comparison with prior TargetLink versions:

- A single vector variable is replaced by multiple variables.
- A single loop is replaced by multiple single statements and/or multiple loops.
- Multirate code generation: A single message is replaced by multiple messages.
- Multirate code generation: A message is no longer protected (because the criterion ‘atomic access’ is fulfilled more often).

Merging of variables

In previous TargetLink versions, two or more variables with the same name but different variable classes were optimized in one variable provided that the MERGEABLE optimization property was set in all variable classes, even if the optimization properties were not compatible with regard to MOVABLE, ERASABLE, SCOPE_REDUCIBLE. As of TargetLink 3.1, the relevant properties must match. The recommended procedure when using MERGEABLE is to create exactly one variable in the dSPACE Data Dictionary, which is then referenced in all the appropriate places.

Subsystem-to-subsystem ID assignment without model name

In the comment in the file header of the C code file, among others, the assignment of the subsystem ID to the subsystem that code is generated for (TargetLink subsystem or subsystem configured for incremental code generation) is indicated. As of TargetLink 3.1, the name of the model is no longer output before the name of the subsystem.

Example:

- TargetLink < 3.1
  
  Sa1       MyModel/MyTLSubsystem
As of TargetLink 3.1, the default name of a module generated from a referenced model is not TL_<ModelName> but <ModelName>. This complies with the general rule that a module’s default name is the name of the system (TargetLink subsystem or subsystem configured for incremental code generation) that code is generated for.

The default names for the axes of look-up tables (1-D) and (2-D) have changed as shown below. If you do not want to use the new default names, you can also specify your own names.

The following changes apply to Look-up table (1D):

<table>
<thead>
<tr>
<th>Look-up Table (1D)</th>
<th>TargetLink 3.0</th>
<th>TargetLink 3.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>X axis Table</td>
<td>$S_8B_x_table</td>
<td>$S_8B_axis</td>
</tr>
<tr>
<td></td>
<td>$S_8B_z_table</td>
<td>$S_8B_table</td>
</tr>
</tbody>
</table>

The following changes apply to Look-up table (2D):

<table>
<thead>
<tr>
<th>Look-up Table (2D)</th>
<th>TargetLink 3.0</th>
<th>TargetLink 3.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row axis</td>
<td>$S_8B_x_table</td>
<td>$S_8B_axis_1</td>
</tr>
<tr>
<td>Column axis</td>
<td>$S_8B_y_table</td>
<td>$S_8B_axis_2</td>
</tr>
</tbody>
</table>
Changes to user-defined header files

Up to TargetLink < 3.1, the name of the user-defined header file was built from the base name of the TargetLink root file and the extension _udt. As of TargetLink 3.1, the name of the user-defined header file is by default udt_$I.h. This also necessitates a change in the Include statements.

You can use the UseRootFileNameAsUserDefinedTypesHeaderBasename Code Generator option to achieve the behavior of the previous TargetLink versions.

Additionally, you can change the name of the user-defined header file via the NameTemplate property of the /Pool/Modules/TLPredefinedModules/UserDataTypes module object.

Avoid merging Bus and Mux signals

The merging of bus and muxed signals must always be avoided, as otherwise the overflow detection for Bus Inport blocks does not work properly. If you split a signal into separate signal parts using Demux blocks and subsequently merge them in a bus signal via Bus Creator blocks, the Simulink interface provides wrong data for the overflow detection and min/max logging. This results in overflow warnings not being properly assigned to the separate bus elements.

Name macro $(FunctionName) in LookupFunction templates

With TargetLink < 3.1, the $(FunctionName) name macro could be used for the module name, function name and type name in LookUpFunction templates. With the introduction of the module objects in the dSPACE Data Dictionary, the $(FunctionName) name macro can no longer be used for the module name.

There are two alternatives:

- Set the ModuleRef property to <n.a.> and activate the Share functions between subsystems option. A separate file is then generated for each created LookUp function. The name of each file is identical to the default LookUp function name, which in turn is identical to the name that would have resulted from the $(FunctionName) name macro.

<table>
<thead>
<tr>
<th>Look-up Table (2D)</th>
<th>TargetLink 3.0</th>
<th>TargetLink 3.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table</td>
<td>$S_$_B_z_table</td>
<td>$S_$_B_table</td>
</tr>
</tbody>
</table>
If you want to extend the default LookUp table function name with a prefix or postfix, you can specify the module name directly. You need a template and a module object for each LookUp table function.

| Tool Selector block configured for model referencing | As of TargetLink 3.1, the functionality of the Model Referencing Control Center is integrated into the TargetLink Main Dialog. The Tool Selector block configured for the Model Referencing Control Center is therefore no longer needed in TargetLink models with references to other models and is automatically removed from the model by the upgrade. |
| Listing files used for building the simulation application | After code generation has finished, the MATLAB Command Window displays a link named Click here, which outputs a list of the generated files that were used for building the simulation application. |
| Changed code pattern for the Multiport-Switch block | In TargetLink 3.1, the code pattern for the Multiport-Switch block has changed. The last input of the block is not calculated in the conditionally executed (case) branch, but in the default branch. |
| No ModuleRef entry for variable object of scope ref_param | A variable object with its scope set to ref_param means call-by-reference semantics. The ref_param scope is the default in the variable classes FCN_REF_ARG, OPT_FCN_REF_ARG and FCN_REF_PARAM. In the resulting C code, such a variable becomes a formal parameter in a function declaration, as follows:

\[
\text{f(int16 } x) \{} \quad \ldots \quad \} 
\]

Consequently, the variable is bound to the module the function is generated into. The ModuleRef property of the variable object must therefore not be set. Usually, a function will not only be declared but also called in the generated code. In order to call the function, you need a variable to pass to the function, the actual parameter, as follows:

\[
\text{g()} 
\]  
\[
\text{\{} \quad \ldots \quad \} 
\]

\[
\text{int16 } Z = 1 
\]

\[
\text{f(&Z)} 
\]

\[
\text{\ldots \} 
\]
The module into which the definition of the actual parameter will be generated can be specified by setting the `ArgClass` property of the variable class of the variable object which will become the formal parameter. The property value of the `ArgClass` is a variable class itself, and the `ModuleRef` property of the variable class object specifies the module that will contain the definition of the variable that becomes the actual parameter.

**Discontinued boards**

There are various evaluation boards that are no longer supported and/or distributed. For further information, refer to *Enhancements to the Target Simulation Module* on page 78.

## AUTOSAR-Related Migration Aspects

<table>
<thead>
<tr>
<th>Migration of Data Dictionaries prior to Data Dictionary 2.0</th>
<th><strong>Automatic Data Dictionary upgrade</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>When you access Data Dictionaries prior to Data Dictionary 2.0 they are upgraded automatically.</td>
</tr>
</tbody>
</table>

AUTOSAR DD upgrade is not supported for Data Dictionaries prior to Data Dictionary 1.5. If you want to upgrade an older Data Dictionary, contact dSPACE Support.

The following steps are performed during DD upgrade:

- Synchronization of the Data Dictionary with the revised AUTOSAR DD template. The DD upgrade adapts DD objects such as type definitions and variables. The DD upgrade also moves the `/Pool/Interfaces` DD node to `/Pool/Autosar/Interfaces`.

You have to customize self-written scripts as required.

- Initial values that are specified in communication specifications have been replaced by variables. The upgrade creates variable objects that are referenced via the `InitRef` property in the communication specifications.

- `InitFunction` and `TermFunction` DD objects have been replaced by runnables. Runnables for initialization and termination are specified via their Kind property.
Migrating Data Dictionaries  If you are also working with an AUTOSAR-compliant architecture tool, you should reimport the AUTOSAR file from the architecture tool to the Data Dictionary after upgrading. The Data Dictionary 2.0 supports the following items for improved exchange of AUTOSAR files.

- Constant specifications
- Array types
- Access points and read/write accesses

Migration of Simulink/TargetLink models prior to TargetLink 3.1

Automatic model upgrade  When you access Simulink/TargetLink models prior to TargetLink 3.1, they are upgraded automatically.

AUTOSAR model upgrade is not supported for Simulink/TargetLink models prior to TargetLink 2.3. If you want to upgrade an older model, contact dSPACE Support.

The following steps are performed during model upgrade:

- Replacement of Runnable blocks by Function blocks. The AUTOSAR settings for code generation are now on the AUTOSAR page of the Function block.
- Replacement of Runnable Inport blocks by TargetLink InPort/Bus Inport blocks as required. The AUTOSAR settings for incoming signals are now on the AUTOSAR page of the InPort/Bus Inport block. If the Status port of a Runnable Inport block is enabled, a ReceiverComSpec block is added.
- Replacement of Runnable Outport blocks by TargetLink OutPort/Bus Outport blocks as required. The AUTOSAR settings for outgoing signals are now on the AUTOSAR page of the OutPort/Bus Outport block. If the Status port of a Runnable Outport block is enabled, a SenderComSpec block with enabled Status port is additionally added.
- Replacement of ClientPort blocks by atomic subsystems with a Function block that is configured for implementing an operation call.

The model upgrade does not support replacing ClientPort blocks that are connected with Bus Inport/Bus Outport blocks for modeling operations with several IN and/or OUT arguments. You have to migrate those ClientPort blocks by hand. Refer to Migrating ClientPort Blocks on page 126.
Migrating SWC SenderPort and SWC ReceiverPort blocks  If you migrate a model where TargetLink blocks other than Mux/Demux, Bus Creator/Bus Selector, or Selector are used together with SWC SenderPort/SWC ReceiverPort blocks outside runnables, code generation might fail. Perform the following migration steps:
1. Place a Simulink Inport block before/after the SWC ReceiverPort/SWC SenderPort block.
2. Transform it into a TargetLink InPort block via the TargetLink - Enhance block context menu command.
3. If the combined signal represents a struct, transform the TargetLink InPort by opening the block dialog and clicking Click here to transform the port block to a busport block.

Migrating buses for struct-based data elements  If you migrate a model with struct-based data elements, model initialization might fail due to non-matching signals. For migration, you have to name the bus signals of a bus that represents a struct-based data element according to the components of the data element.

In previous TargetLink versions, the components of a structured data element were simply mapped onto the signals within a Simulink bus based on their order of appearance in the struct and the Simulink bus. With TargetLink 3.1, component names and signal names in the Simulink bus must match.

Changed modeling practices  The following modeling practices have been changed:
- The Runnable block has been replaced by the AUTOSAR dialog page of the Function block. You have to select the Implement as AUTOSAR runnable checkbox on the dialog page to implement a subsystem as an AUTOSAR runnable. The runnable subsystem now has to be triggered in the same way as in Simulink, i.e., by triggering or enabling the runnable subsystem as required. RTE event properties have to be edited in the Data Dictionary. You have to adapt user-written scripts that have run on Runnable blocks.
- The Runnable Inport/Runnable Outport blocks have been replaced by AUTOSAR dialog pages of the TargetLink InPort/Outport or Bus Inport/Bus Outport blocks. You have to specify AUTOSAR settings for runnable subsystem interfaces via the Use AUTOSAR communication checkbox on the dialog pages. You have to adapt user-written scripts that have used Runnable Inport/Runnable Ouport blocks for reading runnable interface data.
The ClientPort block has been removed from the TargetLink AUTOSAR Block Library. You can now model operation calls using a subsystem with a Function block. Select the Implement the subsystem as operation call checkbox on the AUTOSAR dialog page for this purpose. You can also implement the same subsystem for simulation in SIL/PIL mode or even implement the same subsystem as the runnable for the operation. You have to adapt user-written scripts that have run on ClientPort blocks.

The mapping of struct type data element components to bus signal elements was performed by sequence with TargetLink versions prior to TargetLink 3.1. Now the mapping is performed by name. You have to name the bus signals according to the components of a data element.

### Discontinued support for AUTOSAR Release 2.0

With Version 2.0, the Data Dictionary no longer supports the import/export of AUTOSAR files according to AUTOSAR Release 2.0.

### Migrating ClientPort Blocks

**Objective**

To migrate TargetLink models that contain ClientPort blocks that are connected to bus signals.

The model upgrade replaces ClientPort blocks that are connected to non-bus signals without the need for further migration steps. You can generate code for those models after the model upgrade has been performed.
The illustration below shows an example from a model with a ClientPort block that had been connected to bus signals before model upgrade. After model upgrade the ClientPort block was replaced by a subsystem that implements an operation call. To fully migrate the model, you have to replace the bus signal and connect its contained signals via TargetLink InPort/OutPort blocks.

Preconditions
TargetLink must have upgraded the model.

Method
To migrate ClientPort blocks

1. In the subsystem that replaces the ClientPort block, add TargetLink InPort/OutPort blocks to match the individual signals of the incoming/outgoing bus signals.

2. Remove Demux/Bus Creator blocks that handle the bus signals in the subsystem.

3. Properly connect the signals in the subsystem and specify the AUTOSAR settings for client-server communication in the InPort/OutPort AUTOSAR dialog pages. For information on modeling client-server communication, refer to Modeling Client-Server Communication (TargetLink AUTOSAR Modeling Guide).
4. Properly connect the subsystem in your model, i.e., replace the ingoing/outcoming bus signals by individual signals using Demux/Bus Creator blocks as required.

Result

You have migrated TargetLink models with ClientPort blocks that had been connected to bus signals and you are ready to generate code.
## Compatibility Information

### Where to go from here

Information in this section

<table>
<thead>
<tr>
<th>Supported MATLAB Releases</th>
<th>129</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>130</td>
</tr>
<tr>
<td>Limitations for Windows Vista/Windows 7</td>
<td>132</td>
</tr>
<tr>
<td>Limitations for 64-bit Windows Operating Systems</td>
<td>133</td>
</tr>
</tbody>
</table>

### Supported MATLAB Releases

The table below shows which dSPACE software item supports which MATLAB release:

<table>
<thead>
<tr>
<th>MATLAB Release...</th>
<th>Is Supported by dSPACE Release 6.5 Software Item...</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RCP and HIL Software</td>
<td>TargetLink 3.1</td>
</tr>
<tr>
<td>R2009b</td>
<td>Yes &lt;sup&gt;2,3&lt;/sup&gt;</td>
<td>Yes</td>
</tr>
<tr>
<td>R2009a</td>
<td>Yes &lt;sup&gt;3&lt;/sup&gt;</td>
<td>Yes</td>
</tr>
<tr>
<td>R2008b</td>
<td>Yes &lt;sup&gt;3&lt;/sup&gt;</td>
<td>Yes</td>
</tr>
<tr>
<td>R2008a+</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>R2007b+</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>R2007a+</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>R2006b</td>
<td>Yes &lt;sup&gt;3&lt;/sup&gt;</td>
<td>No</td>
</tr>
</tbody>
</table>

New Features and Migration | December 2009 | 129
## Compatibility Information

### Operating System

The following table shows which software items in dSPACE Release 6.5 support which operating system:

<table>
<thead>
<tr>
<th>Operating System...</th>
<th>Is Supported By...</th>
<th>RCP &amp; HIL Software</th>
<th>TargetLink 3.1</th>
<th>CalDesk 3.0</th>
<th>Model Compare 2.0.2</th>
<th>SystemDesk 2.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows XP Professional (32-bit version) with Service Pack 3</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Windows Vista Business, Ultimate, and Enterprise 1) (32-bit version) with the latest Service Pack</td>
<td>Yes2)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Windows Vista Business, Ultimate, and Enterprise 1) (64-bit version) 2) with the latest Service Pack</td>
<td>Yes2)</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Windows 7 Professional, Ultimate, and Enterprise (32-bit version) 3)</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Windows 7 Professional, Ultimate, and Enterprise (64-bit version) 4)</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

1) Only the editions Windows Vista Business, Ultimate, and Enterprise are supported. The editions Windows Vista Home and Starter are not supported.
2) Not supported by the RTI FPGA Programming Blockset.
3) The 64-bit operating systems are supported by 32-bit software running in WoW64 (Windows-On-Windows64). 64-bit MATLAB versions are not supported.
4) Only the editions Windows 7 Professional, Ultimate, and Enterprise are supported. The editions Windows 7 Home and Starter are not supported.

---

### New Features and Migration December 2009


---

The table shows which software items in dSPACE Release 6.5 support which operating system:

<table>
<thead>
<tr>
<th>MATLAB Release...</th>
<th>Is Supported by dSPACE Release 6.5 Software Item...</th>
</tr>
</thead>
<tbody>
<tr>
<td>R2006a+</td>
<td>RCP and HIL Software No</td>
</tr>
</tbody>
</table>

1) DSOffSim (dSPACE target for offline simulation) installed with SystemDesk requires MATLAB.
2) This compatibility is not documented in other documents on dSPACE Release 6.5.
3) Not supported by the RTI FPGA Programming Blockset.

---

**dSPACE software supports only 32-bit versions of MATLAB. 64-bit MATLAB versions are not supported.**

---

Compatibility Information

Limitations for Windows Vista/Windows 7
Some limitations apply when you use Windows Vista or Windows 7 in combination with dSPACE software. Refer to Limitations for Windows Vista/Windows 7 on page 132.

Notes on 64-bit Windows operating systems
dSPACE software runs as a 32-bit application under 64-bit Windows operating systems in a WoW64 (Windows-on-Windows 64-bit) subsystem. WoW64 is the x86 emulator that allows 32-bit Windows-based applications to run seamlessly on 64-bit versions of Windows. This allows you to use up to 4 GB virtual memory for each 32-bit program. 32-bit versions of Windows can only address up to 3.2 GB of memory in total for all running processes including the operating system itself.

Some additional limitations apply when you use a 64-bit Windows operating system with dSPACE software. Refer to Limitations for 64-bit Windows Operating Systems on page 133.

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 XP Professional (32-bit version) with Service Pack 3
- Windows Vista Business, Ultimate, or Enterprise with the latest Service Pack
- Windows 7 Professional, Ultimate, or Enterprise
- Windows Server 2003

The dSPACE License Server does not support non-Windows operating systems.
Limitations for Windows Vista/Windows 7

**Objective**
Some limitations apply when you use Windows Vista/Windows 7 in combination with dSPACE software.

**MATLAB support**
- Under Windows Vista, the dSPACE software supports only MATLAB versions since MATLAB R2007a+.
- Under Windows 7, the dSPACE software supports only MATLAB versions since MATLAB R2009a.

**Sleep mode not supported**
The dSPACE software does not support Windows's sleep mode for power saving. When restarting the PC from the sleep mode, you must reboot it to restore communication with the dSPACE hardware.

To avoid the automatic sleep mode, disable it. Refer to *How to Disable Sleep Mode of Windows Vista and Windows 7* ([Software Installation and Management Guide](#)).

**Fast user switching not supported**
The dSPACE software does not support the fast user switching feature of Windows Vista and Windows 7.

**Closing dSPACE software before PC shutdown**
The modified shutdown procedure of Windows operating systems causes some required processes to be aborted although they are still being used by dSPACE software. To avoid data loss, the dSPACE software must be terminated manually before a PC shutdown is performed.

**USB devices under Windows 7**
The first time dSPACE USB devices using cables with opto-isolation are connected to the PC, there might be a message that the device driver software was not successfully installed. The dSPACE device will work properly later on nevertheless.

**Allowing communication via additional firewall rules**
During installation of the dSPACE software, two additional Windows firewall rules are installed. The first rule allows communication with a dSPACE expansion box, for example, AutoBox. The second rule allows MotionDesk to receive motion data from a network channel.

The rules are created by the following commands:
- `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."`
advm firewall 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."

**Limitations for 64-bit Windows Operating Systems**

**Objective**

Some additional limitations apply when you use Windows Vista/Windows 7 (64-bit versions) in combination with dSPACE software.

**Using boards with ISA interface installed in the host PC**

Using ISA boards, such as the DS813, installed directly in the host PC, with Windows Vista/Windows 7 (64-bit versions) is not possible with the standard installation routines. If necessary, contact dSPACE Support.

**Limitations of device drivers**

Third-party CAN interfaces are supported only if they are provided with 64-bit drivers from the manufacturers.

**Limitations of target compilers**

For information on the support of a given target compiler, contact the respective compiler manufacturer.

**Limitations with software**

The following table shows the known limitations of the dSPACE and third-party software under 64-bit Windows operating systems:

<table>
<thead>
<tr>
<th>Software</th>
<th>Limitations and Recommendations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naming of installation path</td>
<td>Under 64-bit Windows operating systems, the default installation folder for 32-bit applications is C:\Program Files (x86). Since parentheses in the path name can cause problems, make sure to choose installation paths for the dSPACE software, MATLAB, and the PPC compiler which contain no parentheses.</td>
</tr>
<tr>
<td>AutomationDesk 2.3</td>
<td>The winsound Python module used to create an acoustic signal in the Tutorialdemo06 project is not supported.</td>
</tr>
</tbody>
</table>
## Compatibility Information

<table>
<thead>
<tr>
<th>Software</th>
<th>Limitations and Recommendations</th>
</tr>
</thead>
<tbody>
<tr>
<td>CalDesk</td>
<td>- CalDesk 3.0 can be installed under Windows Vista/Windows 7 (64-bit versions) but has not been qualified for 64-bit Windows operating systems. &lt;br&gt; - CalDesk versions earlier than CalDesk 3.0 do not support Windows Vista/Windows 7 (64-bit versions) and cannot be installed under them.</td>
</tr>
<tr>
<td>dSPACE ECU Services</td>
<td>The setups of the dSPACE ECU Services abort due to a problem with WinZip Self-Extractor. To avoid this: &lt;br&gt; - Click the installation dialogs in less than ten seconds or &lt;br&gt; - Manually unpack the setups of the dSPACE ECU Services to an empty folder and run setup.exe.</td>
</tr>
<tr>
<td>RTI AUTOSAR Package 1.0</td>
<td>The RTI AUTOSAR Interface Generator does not start under Windows Vista/Windows 7 (64-bit versions).</td>
</tr>
<tr>
<td>MATLAB</td>
<td>If you install a 32-bit version of MATLAB under Windows Vista/Windows 7 (64-bit versions), the MATLAB installation program generates a message that a 64-bit version of MATLAB is available. Click OK and continue installing the 32-bit version of MATLAB. &lt;br&gt; dSPACE software supports only 32-bit versions of MATLAB. 64-bit versions of MATLAB are not supported.</td>
</tr>
</tbody>
</table>
# Index

## A

<table>
<thead>
<tr>
<th>Blockset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASM Base InCylinder Blockset</td>
<td>migrating 21</td>
</tr>
<tr>
<td>ASM Brake Hydraulics Blockset</td>
<td>new features 22</td>
</tr>
<tr>
<td>ASM Diesel Exhaust Blockset</td>
<td>migrating 25</td>
</tr>
<tr>
<td>ASM Drivetrain Basic Blockset</td>
<td>new features 24</td>
</tr>
<tr>
<td>ASM Electric Components Blockset</td>
<td>migrating 27</td>
</tr>
<tr>
<td>ASM Engine Diesel Blockset</td>
<td>new features 27</td>
</tr>
<tr>
<td>ASM Engine Gasoline Basic Blockset</td>
<td>migrating 32</td>
</tr>
<tr>
<td>ASM Engine Gasoline Blockset</td>
<td>new features 32</td>
</tr>
<tr>
<td>ASM Environment Blockset</td>
<td>migrating 34</td>
</tr>
<tr>
<td>ASM Optimizer</td>
<td>new features 37</td>
</tr>
<tr>
<td>ASM Truck Blockset</td>
<td>migrating 38</td>
</tr>
<tr>
<td>ASM Vehicle Dynamics Blockset</td>
<td>new features 39</td>
</tr>
<tr>
<td>AutomationDesk</td>
<td>migration 40</td>
</tr>
<tr>
<td>new features 40</td>
<td></td>
</tr>
</tbody>
</table>

## C

<table>
<thead>
<tr>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>changes</td>
<td>9</td>
</tr>
<tr>
<td>TargetLink API</td>
<td>117</td>
</tr>
<tr>
<td>ConfigurationDesk</td>
<td>45</td>
</tr>
<tr>
<td>migration to Version 3.0.1</td>
<td>45</td>
</tr>
<tr>
<td>new features</td>
<td>44</td>
</tr>
<tr>
<td>ControlDesk</td>
<td>51</td>
</tr>
<tr>
<td>new features</td>
<td>51</td>
</tr>
</tbody>
</table>

## D

<table>
<thead>
<tr>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>DD project file</td>
<td>102</td>
</tr>
<tr>
<td>upgrading</td>
<td>102</td>
</tr>
<tr>
<td>discontinued compiler versions</td>
<td>78</td>
</tr>
<tr>
<td>discontinued evaluation boards</td>
<td>78</td>
</tr>
<tr>
<td>dSPACE Data Dictionary</td>
<td>migration 106</td>
</tr>
</tbody>
</table>

## G

<table>
<thead>
<tr>
<th>Blockset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASM Fatigue Blockset</td>
<td>migrating 27</td>
</tr>
<tr>
<td>ASM Fuel Blockset</td>
<td>new features 27</td>
</tr>
<tr>
<td>ASM INTQ Blockset</td>
<td>migrating 32</td>
</tr>
<tr>
<td>ASM Key Features</td>
<td>new features 32</td>
</tr>
<tr>
<td>ASM Optimization</td>
<td>migrating 37</td>
</tr>
<tr>
<td>ASM Vehicle Mechanics Blockset</td>
<td>new features 39</td>
</tr>
<tr>
<td>AutomationDesk</td>
<td>migration 40</td>
</tr>
<tr>
<td>new features 40</td>
<td></td>
</tr>
</tbody>
</table>

## H

<table>
<thead>
<tr>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>host PC software</td>
<td>130</td>
</tr>
<tr>
<td>operating system</td>
<td>130</td>
</tr>
</tbody>
</table>

## K

<table>
<thead>
<tr>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>key features</td>
<td>11</td>
</tr>
<tr>
<td>dSPACE Data Dictionary</td>
<td>99</td>
</tr>
</tbody>
</table>

## L

<table>
<thead>
<tr>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>limitations for Windows Vista</td>
<td>132</td>
</tr>
</tbody>
</table>

## M

<table>
<thead>
<tr>
<th>Blockset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASM Base InCylinder Blockset</td>
<td>migrating 21</td>
</tr>
<tr>
<td>ASM Brake Hydraulics Blockset</td>
<td>new features 22</td>
</tr>
<tr>
<td>ASM Diesel Exhaust Blockset</td>
<td>migrating 25</td>
</tr>
<tr>
<td>ASM Drivetrain Basic Blockset</td>
<td>new features 27</td>
</tr>
<tr>
<td>ASM Electric Components Blockset</td>
<td>migrating 27</td>
</tr>
<tr>
<td>ASM Engine Diesel Blockset</td>
<td>new features 27</td>
</tr>
<tr>
<td>ASM Engine Gasoline Basic Blockset</td>
<td>migrating 32</td>
</tr>
<tr>
<td>ASM Engine Gasoline Blockset</td>
<td>new features 32</td>
</tr>
<tr>
<td>ASM Environment Blockset</td>
<td>migrating 34</td>
</tr>
<tr>
<td>ASM Optimizer</td>
<td>new features 37</td>
</tr>
<tr>
<td>ASM Truck Blockset</td>
<td>migrating 38</td>
</tr>
<tr>
<td>ASM Vehicle Dynamics Blockset</td>
<td>new features 39</td>
</tr>
<tr>
<td>AutomationDesk</td>
<td>migration 40</td>
</tr>
<tr>
<td>new features 40</td>
<td></td>
</tr>
<tr>
<td>to ConfigurationDesk 3.0.1</td>
<td>45</td>
</tr>
</tbody>
</table>

## O

<table>
<thead>
<tr>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>obsolete Code Generator options</td>
<td>86</td>
</tr>
<tr>
<td>TargetLink API changes</td>
<td>117</td>
</tr>
</tbody>
</table>

## P

<table>
<thead>
<tr>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>product overview</td>
<td>10</td>
</tr>
</tbody>
</table>

## R

<table>
<thead>
<tr>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>RapidPro Projects Migrator</td>
<td>migrating 47</td>
</tr>
<tr>
<td>migrating a ConfigurationDesk project</td>
<td>47</td>
</tr>
<tr>
<td>new features 53</td>
<td></td>
</tr>
<tr>
<td>ModelDesk</td>
<td>53</td>
</tr>
<tr>
<td>new features 53</td>
<td></td>
</tr>
<tr>
<td>motionDesk</td>
<td>56</td>
</tr>
<tr>
<td>motionDesk new features 56</td>
<td></td>
</tr>
<tr>
<td>motionDesk migrating 56</td>
<td></td>
</tr>
<tr>
<td>motionDesk new features 55</td>
<td></td>
</tr>
<tr>
<td>motionDesk migration 56</td>
<td></td>
</tr>
<tr>
<td>motionDesk new features 20</td>
<td></td>
</tr>
<tr>
<td>new features 20</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>new blockset</td>
<td>38</td>
</tr>
<tr>
<td>ASM Truck Blockset</td>
<td>38</td>
</tr>
<tr>
<td>new Code Generator options</td>
<td>86</td>
</tr>
<tr>
<td>new evaluation boards</td>
<td>78</td>
</tr>
<tr>
<td>new evaluation compiler versions</td>
<td>78</td>
</tr>
<tr>
<td>new features</td>
<td>78</td>
</tr>
<tr>
<td>ASM Brake Hydraulics Blockset</td>
<td>22</td>
</tr>
<tr>
<td>ASM Diesel Exhaust Blockset</td>
<td>24</td>
</tr>
<tr>
<td>ASM Drivetrain Basic Blockset</td>
<td>27</td>
</tr>
<tr>
<td>ASM Electric Components Blockset</td>
<td>29</td>
</tr>
<tr>
<td>ASM Engine Gasoline Blockset</td>
<td>34</td>
</tr>
<tr>
<td>ASM Environment Blockset</td>
<td>36</td>
</tr>
<tr>
<td>ASM Optimizer</td>
<td>37</td>
</tr>
<tr>
<td>ASM Vehicle Dynamics Blockset</td>
<td>39</td>
</tr>
<tr>
<td>AutomationDesk</td>
<td>15</td>
</tr>
<tr>
<td>ConfigurationDesk</td>
<td>44</td>
</tr>
<tr>
<td>ControlDesk</td>
<td>51</td>
</tr>
<tr>
<td>ModelDesk</td>
<td>53</td>
</tr>
<tr>
<td>MotionDesk</td>
<td>55</td>
</tr>
<tr>
<td>MotionDesk new features 55</td>
<td></td>
</tr>
<tr>
<td>RTI FAULTOSAR Package</td>
<td>59</td>
</tr>
<tr>
<td>RTI FPGA Programming Blockset</td>
<td>63</td>
</tr>
<tr>
<td>RTI LIN MultiMessage Blockset</td>
<td>61</td>
</tr>
</tbody>
</table>

## N

<table>
<thead>
<tr>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>new API commands</td>
<td>100</td>
</tr>
<tr>
<td>dSPACE Data Dictionary</td>
<td>migration 106</td>
</tr>
</tbody>
</table>

---

New Features and Migration December 2009 135
Index

S
supported MATLAB releases 129
supported targets 78
system requirements
  operating system 130
SystemDesk
  migrating 69

T
target support 78
TargetLink API
  changes 117
TargetLink models
  upgrading 102

U
upgrading
  DD files 102
  existing TargetLink models 102
  TargetLink libraries 103
  TargetLink models 103

V
version history 10

W
Windows Vista
  limitations 132