dSPACE Release

New Features and Migration

Release 2020-B – November 2020

dSPACE

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

How to Contact dSPACE Support

If you encounter a problem when using dSPACE products, contact your local dSPACE representative:

- Local dSPACE companies and distributors: http://www.dspace.com/go/locations
- For countries not listed, contact dSPACE GmbH in Paderborn, Germany.
  Tel.: +49 5251 1638-941 or e-mail: support@dspace.de

You can also use the support request form: http://www.dspace.com/go/supportrequest. If you are logged on to mydSPACE, you are automatically identified and do not need to add your contact details manually.

If possible, always provide the serial number of the hardware, the relevant dSPACE License ID, or the serial number of the CmContainer in your support request.

Software Updates and Patches

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

Important Notice

Contents

About This Document

Content

This document informs you about the new features of all the dSPACE software products in Release 2020-B. It also gives you an overview of software products with no or minor changes. There are instructions on migrating from earlier dSPACE Releases, especially from earlier product versions, if required.

Printed document

A printed copy of this document is available on demand. You can order it free of charge by using the following link:
http://www.dspace.com/go/requestreleasematerial.

Symbols

dSPACE user documentation uses the following symbols:

Symbol	Description
⚠ DANGER	Indicates a hazardous situation that, if not avoided, will result in death or serious injury.
⚠ WARNING	Indicates a hazardous situation that, if not avoided, could result in death or serious injury.
⚠ CAUTION	Indicates a hazardous situation that, if not avoided, could result in minor or moderate injury.
NOTICE	Indicates a hazard that, if not avoided, could result in property damage.
Note	Indicates important information that you should take into account to avoid malfunctions.
Tip	Indicates tips that can make your work easier.
[?]	Indicates a link that refers to a definition in the glossary, which you can find at the end of the document unless stated otherwise.
🕮	Precedes the document title in a link that refers to another document.

Naming conventions

dSPACE user documentation uses the following naming conventions:

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

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

Special folders

Some software products use the following special folders:

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

`%PROGRAMDATA%\dSPACE\<InstallationGUID>\<ProductName>`

or

`%PROGRAMDATA%\dSPACE\<ProductName>\<VersionNumber>`

Documents folder A standard folder for user-specific documents.

`%USERPROFILE%\Documents\dSPACE\<ProductName>\<VersionNumber>`

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

`%USERPROFILE%\AppData\Local\dSPACE\<InstallationGUID>\<ProductName>`

Accessing dSPACE Help and PDF Files

After you install and decrypt dSPACE software, the documentation for the installed products is available in dSPACE Help and as Adobe® PDF files.

dSPACE Help (local) You can open your local installation of dSPACE Help:

- On its home page via Windows Start Menu
- On specific content using context-sensitive help via **F1**

dSPACE Help (Web) You can access the Web version of dSPACE Help at www.dspace.com.

To access the Web version, you must have a *mydSPACE* account.

PDF files You can access PDF files via the icon in dSPACE Help. The PDF opens on the first page.

Overview of dSPACE Release 2020-B

Introduction

Gives you an overview of the new key features in Release 2020-B and information about unchanged products.

Where to go from here

Information in this section

General Enhancements and Changes

Introduction

The following new features and changes concern several dSPACE products.

dSPACE certificate for dSPACE software

As of dSPACE Release 2020-B, all dSPACE executable files are digitally signed with a dSPACE certificate. To check this certificate, a valid chain of trust must be established on the PC on which the dSPACE software is installed. Part of the chain of trust are root certificates, which are available from trusted certification authorities and are installed via the Windows update mechanism.

Note

- If the dSPACE software is installed on a PC with Internet access, the root certificates are automatically installed on the PC via the Windows update mechanism. No further user actions are required.
- If your dSPACE software is to be installed on a PC without an Internet connection, the root certificates must be installed manually on your PC. dSPACE strongly recommends installing the certificates before installing the dSPACE software. It is possible that the security software installed on your PC blocks the installation of the dSPACE software if no root certificates are available on your PC.

For instructions on manual installation, refer to Installing Root Certificates Required for dSPACE Software (🕮 Installing dSPACE Software).

Providing legal notes for using third-party software

Several dSPACE software products of dSPACE Release 2020-B use or contain third-party software. The third-party components may be subject to additional terms and conditions or terms and conditions that deviate from the ones that apply to the dSPACE software. Copyright notices and license terms of third-party components must be adhered to. If the user is required to be explicitly informed of and to comply with the terms and conditions of third-party components, the relevant information on terms and conditions is stated in separate OSSAcknowledgements files for each dSPACE product.

You can find the OSSAcknowledgements files as follows:

- In the Legal folders accessible via the root directory on both dSPACE DVDs and the root directory of the downloaded installation files (mounted ISO images or unpacked ZIP files).
- After you install the dSPACE software in the <main installation path of dSPACE Setup>\<subfolder of InstallationSet>\Legal folder.

Python distribution

The libraries and components used with Python 3.6 and distributed on dSPACE DVDs have changed as shown in the following table.

Package	Release 2020-A	Release 2020-B
comtypes	1.1.7	1.1.7
Core	3.6.8	3.6.8
cycler	0.10.0	0.10.0
future	0.18.2	0.18.2
grpcio	1.25.0	1.29.0
grpcio_tools	1.25.0	1.29.0
kiwisolver	1.1.0	1.2.0
matplotlib	3.1.1	3.2.1
numpy	1.17.3	1.18.4
pillow	6.2.1	7.1.2

Package	Release 2020-A	Release 2020-B
pip	19.3.1	20.1.1
protobuf	3.10.0	3.12.2
pyget	1.4.6	1.5.5
pyparsing	2.4.2	2.4.7
pypubsub	4.0.3	4.0.3
Python-dateutil	2.8.0	2.8.1
pythonnet	2.4.1	2.4.2
pytz	2019.3	2020.1
pywin32	225.10	227.10
scipy	1.3.1	1.4.1
six	1.12.0	1.15.0
wxPython	4.0.7	4.1.0
yapsy	1.12.2	1.12.2

Using dSPACE software on virtual machines (VM)

As of dSPACE Release 2019-A, you can operate several dSPACE products on virtual machines. For more information, refer to Using dSPACE Software on Virtual Machines (VMs) on page 239.

Legacy licensing using CodeMeter licenses simplifies installation and use of dSPACE releases earlier than 2017-B

If you own a software product from dSPACE Release 2017-B or later and you want to install and use a version of this product from a dSPACE Release earlier than 2017-B, you must have legacy license files and a CodeMeter CmContainer with activated licenses. Until now, you had to contact dSPACE and provide specific information before dSPACE sent you the license files.

As of October 2019, you can use dSPACE Installation Manager 5.4 to download license files specifically prepared on the basis of the licenses you purchased. Legacy licensing using CodeMeter licenses maps former product versions to an available license so that you can install and use products from dSPACE Release 7.4 (2012-B) up to dSPACE Release 2017-A.

Refer to Legacy Licensing Using CodeMeter Licenses (🕮 Working with CodeMeter Licensing Technology).

RCP and HIL software: C/C++ compilers for building MATLAB MEX files

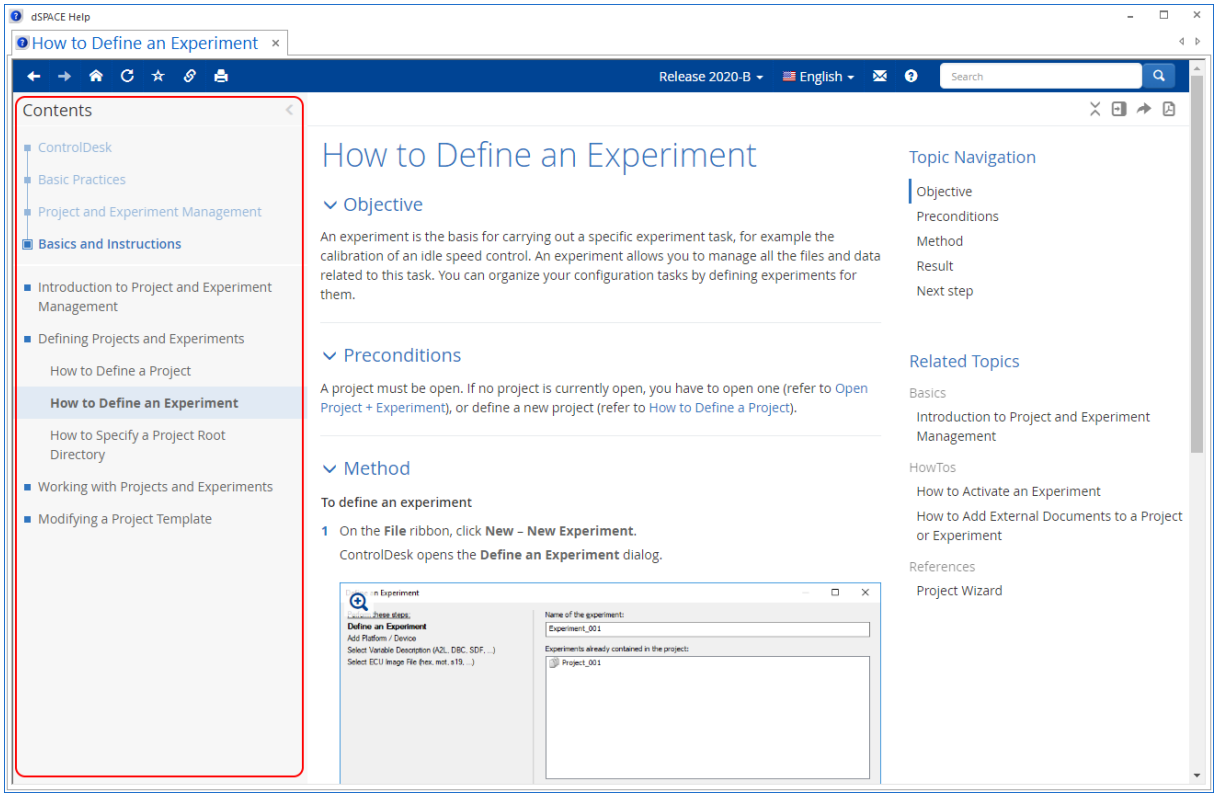
RCP and HIL software (such as RTI CAN MultiMessage Blockset, RTI LIN MultiMessage Blockset, or Automotive Simulation Models) now supports the following C/C++ compilers for building MATLAB MEX files:

- MinGW (GNU Compiler Collection (GCC 6.3.0))
- Microsoft Visual Studio 2017 Professional

New Features of dSPACE Help

Enhanced content navigation

As of Release 2020-B, dSPACE Help provides an enhanced content navigation, displaying the selected topic and the surrounding sections. The path is displayed at the top, visualized by squares that are connected with a line.



Using dSPACE Help

For a detailed description of and instructions for dSPACE Help features, click in dSPACE Help.

Discontinuations

Introduction

The following discontinuations for software and hardware are relevant to the current Release or are planned for future Releases.

For more end-of-life announcements, refer to http://www.dspace.com/go/discontinuation.

Discontinuation of software support

Discontinuation of .NET 2.0 APIs With dSPACE Release 2020-B, dSPACE discontinues the support for client programs and libraries built with .NET Runtime 2.0. This applies to any C#-based programs using the provided interfaces (APIs) for COM automation and for the Installation Manager. Applications using these interfaces have to use at least .NET Runtime 4.0.

MicroAutoBox Embedded SPU As of December 2020, software support is discontinued for the MicroAutoBox Embedded SPU.

For new projects, we recommend that you use the dSPACE AUTERA AutoBox.

DS541 DPMEM POD for MPC55xx As of Release 2020-B, software support is discontinued for the DS541 DPMEM POD for MPC55xx, which was discontinued in a previous Release.

For new projects, we recommend that you use the dSPACE DCI-GSI2.

Planned discontinuations of software support

Calibration Hub As of Release 2021-A, software support will be discontinued for the Calibration Hub, which was discontinued in a previous Release.

For new projects, we recommend that you use the DCI-CAN2 or DCI-CAN/LIN1.

Planned discontinuation of dSPACE hardware

PHS-bus hardware As of December 2021, the hardware components for PHS-bus-based systems, such as the DS1006 Processor Board, the DS1007 PPC Processor Board, and all the PHS-bus I/O boards, will be discontinued. This is also relevant to the dSPACE Simulator Mid-Size and the dSPACE Simulator EcoLine. New Releases of dSPACE software will continue to support the PHS-bus hardware components until the end of 2023.

For new projects, we recommend that you use SCALEXIO as a modular real-time system.

Product Version Overview

Product versions

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

Product	dSPACE Release			
	2019-A	2019-B	2020-A	2020-B
AutomationDesk	6.1	6.2	6.3	6.4 Refer to AutomationDesk on page 31.

Product	dSPACE Release			
	2019-A	2019-B	2020-A	2020-B
Automotive Simulation Models	9.3	9.4	9.5	9.6 Refer to Automotive Simulation Models (ASM) on page 37.
Bus Manager (stand-alone)	6.3	6.4	6.5	6.6 Refer to Bus Manager (Stand-Alone) on page 71.
ConfigurationDesk for RapidPro	6.3	6.4	6.5	6.6
ConfigurationDesk	6.3	6.4	6.5	6.6 Refer to ConfigurationDesk on page 73.
Container Manager	5.1	5.1	5.1	5.2
ControlDesk	7.0	7.1	7.2	7.3 Refer to ControlDesk on page 83.
DCI Configuration Tool	3.11	3.11	3.12	3.12
dSPACE AUTOSAR Compare	–	–	–	1.0 Refer to dSPACE AUTOSAR Compare on page 91.
dSPACE CAN API Package	4.0.2	4.0.3	4.0.4	4.0.4
dSPACE ECU Flash Programming Tool	2.6	2.6	2.7	2.7
dSPACE FlexRay Configuration Package	4.3	4.4	4.5	4.6 Refer to dSPACE FlexRay Configuration Package on page 95.
dSPACE Installation Manager	5.3	5.4	5.5	5.6 Refer to dSPACE Installation Manager on page 97
dSPACE Python Extensions	3.1	3.2	3.3	3.4
dSPACE XIL API .NET	2019-A	2019-B	2020-A	2020-B Refer to dSPACE XIL API .NET on page 99.
ECU Interface Manager	2.5	2.6	2.7	2.8 Refer to ECU Interface Manager on page 101.
Firmware Manager	2.7	3.0	3.1	3.2 Refer to Firmware Manager on page 105.
Model and Sensor Interface Blockset	–	–	–	1.0 Refer to Model and Sensor Interface Blockset on page 109.
Model Compare	2.9	3.0	3.0	3.1 Refer to Model Compare on page 111.
ModelDesk	5.1	5.2	5.3	5.4 Refer to ModelDesk on page 113.

Product	dSPACE Release			
	2019-A	2019-B	2020-A	2020-B
Model Interface Package for Simulink	4.1	4.2	4.3	4.4 Refer to Model Interface Package for Simulink on page 115.
MotionDesk	4.4	4.5	4.6	4.7 Refer to MotionDesk on page 117.
MotionDesk Blockset	2.5.5	2.6	2.6.1	2.6.2 Refer to MotionDesk on page 117.
Real-Time Testing	4.1	4.2	4.3	4.4 Refer to Real-Time Testing on page 121.
RTI[1)]	7.12	7.13	7.14	7.15 Refer to RTI/RTI-MP and RTLib on page 123.
RTI-MP[2)]	7.12	7.13	7.14	7.15 Refer to RTI/RTI-MP and RTLib on page 123.
RTI Bypass Blockset	3.12	3.13	3.14	3.15 Refer to RTI Bypass Blockset on page 127.
RTI CAN Blockset	3.4.8	3.4.9	3.4.10	3.4.11
RTI CAN MultiMessage Blockset	5.2	5.3	5.4	5.5 Refer to RTI CAN MultiMessage Blockset on page 131.
RTI Electric Motor Control Blockset	1.4.1	1.4.1	1.4.1	1.4.2
RTI Ethernet Blockset	1.2.3	1.2.3	1.2.3	1.2.4
RTI Ethernet (UDP) Blockset	1.4.3	1.4.3	1.4.3	1.4.4
RTI FPGA Programming Blockset	3.7	3.8	3.9	3.10 Refer to RTI FPGA Programming Blockset on page 133.
RTI LIN MultiMessage Blockset	3.2	3.3	3.4	3.5 Refer to RTI LIN MultiMessage Blockset on page 137.
RTI RapidPro Control Unit Blockset	2.2.3	2.2.3	2.2.3	2.2.4
RTI Synchronized Time Base Manager Blockset	1.2	1.3	1.4	1.4.1
RTI USB Flight Recorder Blockset	1.2.2	1.2.2	1.2.2	1.2.3
RTI Watchdog Blockset	2.1.1	2.1.1	2.1.1	2.1.2
Sensor Simulation	1.1	1.2	1.3	1.4 Refer to Sensor Simulation on page 143.
SCALEXIO firmware	4.4	4.5	4.6	5.0 Refer to SCALEXIO Firmware on page 139.
SYNECT	2.7	2.8	2.9	2.10 Refer to SYNECT on page 145.

Product	dSPACE Release			
	2019-A	2019-B	2020-A	2020-B
SystemDesk	5.3	5.4	5.4	5.5 Refer to SystemDesk on page 153.
TargetLink	4.4	5.0	5.0	5.1 Refer to TargetLink on page 167.
Variable Editor[3)]	2.4	2.4	2.4	2.4
VEOS	4.4	4.5	5.0	5.1 Refer to VEOS on page 227.

1) Including the standard I/O blocksets.
2) Including the RTI Gigalink Blockset.
3) The Variable Editor is not part of the dSPACE Release DVD. It is available at https://www.dspace.com/go/requestreleasedownload.

If you have not performed regular updates, refer to the *New Features and Migration* documents for the dSPACE Releases listed above for information about the new features and required migration steps.

New Key Product Features

Introduction

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

AutomationDesk

The new key features of AutomationDesk are:

- Enhancements to the SYNECT interface providing access to SYNECT Test Management projects.
- Support of the Message Reader API. You can use the API to read out log messages of the dSPACE Log, for example, in C# applications and Python scripts. The API also lets you filter log messages by dSPACE product and by time, for example.

For more information on the new features, refer to New Features of AutomationDesk 6.4 on page 31.

Bus Manager (stand-alone)

The new key feature of the Bus Manager (stand-alone) is:

- Support of AUTOSAR system description files based on AUTOSAR Classic Platform Release R19-11

For more information, refer to New Features of the Bus Manager (Stand-Alone) 6.6 on page 71.

ConfigurationDesk

The new key feature of ConfigurationDesk is:

- New **FuSa System Monitoring** function block type to support MicroAutoBox III.

For more information, refer to ConfigurationDesk on page 73.

ControlDesk

The new key features of ControlDesk 7.3 are:

Instrument enhancements

- 3-D Viewer: Selecting the coordinate system
- 3-D Viewer: Importing MAT files for roads
- 3-D Viewer: Dynamic item labeling
- 3-D Viewer: Rotating the camera via mouse if a camera target is selected

For more information, refer to New Instrument Features (ControlDesk 7.3) on page 84.

Measurement and recording enhancements

- Configuring data logging on SCALEXIO systems (only for systems based on a DS6001 Processor Board)
- Exporting MF4 files without path and block group information

For more information, refer to New Measurement and Recording Features (ControlDesk 7.3) on page 86.

Message handling enhancements

- Support of the Message Reader API

For more information, refer to New Message Handling Features (ControlDesk 7.3) on page 86.

Calibration and data set management enhancements

- Exporting data sets to CDFX 2.1

Refer to New Calibration and Data Set Management Features (ControlDesk 7.3) on page 86.

Bus Navigator enhancements

- Replaying Ethernet communication
- Support of the DS6336-PE Ethernet Board and DS6336-CS Ethernet Board
- Support of the AUTOSAR CP R19-11 system template

For more information, refer to New Bus Navigator Features (ControlDesk 7.3) on page 87.

Automation enhancements

- Automated playback of recorded data via the time cursor

For more information, refer to New Automation Features (ControlDesk 7.3) on page 87.

dSPACE AUTOSAR Compare

dSPACE AUTOSAR Compare is the new dSPACE product for comparing and merging AUTOSAR files. Refer to dSPACE AUTOSAR Compare on page 91.

dSPACE FlexRay Configuration Package

The new key feature of the dSPACE FlexRay Configuration Package is:

- Support of AUTOSAR Classic Platform R19-11.

For more information on the new features, refer to New Features of dSPACE FlexRay Configuration Package 4.6 on page 95.

dSPACE XIL API

The new key features of dSPACE XIL API and the Platform Management API are:

- Enhanced MAPort platform support for VEOS and SCALEXIO
- Support of the Message Reader API

For more information on the new features, refer to New Features of dSPACE XIL API .NET 2020-B on page 99.

ECU Interface Manager

The new key features of the ECU Interface Manager are:

- ARM Cortex-R4/R5: Integrating control logics
- Reading XCP time stamps

For more information on the new feature, refer to New Features of ECU Interface Manager 2.8 on page 101.

Firmware Manager

The new key features of the Firmware Manager are:

- Support of new SCALEXIO Processing Units.
- Support of the KVM Hypervisor for SCALEXIO Processing Units.
- Support of SCALEXIO systems based on real-time Linux®.
- Support of the Message Reader API.

For more information on the new hardware, refer to New Features of the SCALEXIO Firmware 5.0 on page 139.

For more information on the new features, refer to New Features of Firmware Manager 3.2 on page 105.

MicroAutoBox III firmware

The new key features of the MicroAutoBox III firmware are:

- Support of new functional safety monitoring features.

For more information on the new features, refer to New Features of the MicroAutoBox III Firmware 5.0 on page 107.

Model and Sensor Interface Blockset

You must use the Model and Sensor Interface Blockset for Sensor Simulation.

The blockset calculates and handles the simulation, status, feedback, and failure data between the simulation and the connected systems and applications. For example, you can connect, MotionDesk, the SensorSim application, and the Environment Sensor Interface Unit.

MotionDesk visualizes the simulation and the SensorSim application creates the sensor raw data.

The Environment Sensor Interface Unit splits the sensor raw data from the SensorSim application and inserts it into the relevant sensor hardware.

You can also send failure data to sensors connected to the Environment Sensor Interface Unit. The blockset can also receive feedback information from the Environment Sensor Interface Unit and status information from the other connected systems or applications.

For more information on the new features, refer to New Features of Model and Sensor Interface Blockset 1.0 on page 109.

Model Compare

The new key features of Model Compare are:

- You can now dump and compare single subsystems of interest instead of complete models. This reduces complexity and significantly enhances performance.
- A new demo comparison session shows you how to use a hook to limit the analysis of a model or subsystem to a specified number of levels. In applicable use cases, this reduces comparison efforts.
- Differences in properties and their values are now colored in the same way in reports as they are in the Property Inspector. This makes it easier to detect the relevant divergences.
- Models and subsystems that contain Referenced Subsystems are handled appropriately.

For more information on the new features, refer to New Features of Model Compare 3.1 on page 111.

Model Interface Package for Simulink

The new key features of Model Interface Package for Simulink are:

- Support of MATLAB R2020b.
- Support of the int64/uint64 data type for model port blocks and for Simulink Inport and Outport blocks on the root level of a behavior model.

For more information on the new features, refer to Model Interface Package for Simulink on page 115.

ModelDesk

The new key features of ModelDesk are:

- Road creation: Improvement of the OpenDRIVE import.
- Plotting: Supporting the **ASMSignalInterface** block that allows collecting signals of different subsystems of a model.

For more information on the new features, refer to New Features of ModelDesk 5.4 on page 113.

MotionDesk	The new key features of MotionDesk are: ▪ **Installation**: MotionDesk is installed in its own folder structure in the selected installation path. ▪ **Bird's-eye view**: The bird's-eye view can be enabled for the selected view. This is a two-dimensional view above the scene. ▪ **Scene editing**: You can edit the scene without the dedicated scene edit mode. For example, you can freely move, resize, or rotate an object in the scene view. You can also copy single and groups of objects in the scene and paste them to another location. ▪ **3-D object library vehicles**: A number of new vehicles and trailers have been added to the 3-D library. ▪ **Animated characters**: The speed of the movements of the animated characters changes with the changes in the simulation. ▪ **Scene generation**: The generation of the road network geometries supports OpenDRIVE roads. For more information on the new features, refer to MotionDesk Features (MotionDesk 4.7) on page 118.
Python Extensions	The Python Extension 3.4 does not have new features.
RTI, RTI-MP, and RTLib	The new key features of RTI, RTI-MP, and RTLib are: ▪ Support of MATLAB R2020b. ▪ The dialogs in RTI and RTI-MP to specify the behavior of the first simulation steps for avoiding task overruns have been enhanced. For more information on the new features, refer to New Features of RTI/RTI-MP and RTLib on page 123.
RTI Bypass Blockset	The new key feature of the RTI Bypass Blockset is: ▪ On-target bypassing on virtual ECUs on SCALEXIO. For more information on the new feature, refer to New Features of the RTI Bypass Blockset 3.15 on page 127.
RTI CAN MultiMessage Blockset	The new key features of the RTI CAN MultiMessage Blockset are: ▪ Support of AUTOSAR Classic Platform R19-11. ▪ Support of secure onboard communication for dynamic container IPDUs with secured PDU header. For more information on the new features, refer to New Features of the RTI CAN MultiMessage Blockset 5.5 on page 131.

RTI FPGA Programming Blockset

The new key features of the RTI FPGA Programming Blockset 3.10 are:

- Extended Xilinx® software support.
- The SCALEXIO and the MicroAutoBox III frameworks now support subchannels for the processor communication.

For more information on the new features, refer to New Features of the RTI FPGA Programming Blockset 3.10 on page 133.

RTI LIN MultiMessage Blockset

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

- Support of AUTOSAR Classic Platform R19-11.

For more information on the new feature, refer to New Features of the RTI LIN MultiMessage Blockset 3.5 on page 137.

SCALEXIO firmware

The new key features of the SCALEXIO firmware are:

- New basic operating system based on Linux®
- Support of the DS6336-CS Ethernet Board and DS6336-PE Ethernet Board
- Support of a new high parallel performance real-time PC

For more information on the new features, refer to New Features of the SCALEXIO Firmware 5.0 on page 139.

Sensor Simulation

Sensor Simulation lets you validate camera, fish-eye, laser, lidar, and radar sensors.

A number of performance optimizations have been added to the SensorSim application for road shapes, lane markings, and fish-eye sensors for sensor simulation.

For more information on the new product, refer to New Features of Sensor Simulation 1.4 on page 143.

SYNECT

The new key features of SYNECT 2.10 are:

- Evaluation functions support variant-dependent parameterization.
- New Workflow Management SVN CM add-on.

For more information on the new features, refer to New Features of SYNECT 2.10 on page 146.

SystemDesk

The new key features of SystemDesk 5.5 are:

- Support of the AUTOSAR 19-11 release.
- Support for the new VECU container file format.
- Changed handling of V-ECUs.
- New API for the specification of RTE interventions.

For more information on the new features, refer to New Features of SystemDesk 5.5 on page 154.

TargetLink

The new key features of TargetLink 5.1 are:

- Simulink® functions
  - Defining and calling global Simulink functions in a TargetLink subsystem.
- Referenced subsystems:
  - Using referenced subsystems created in Simulink for code generation with TargetLink.
- AUTOSAR Curve/MAP calibration:
  - Preparing look-up tables for measurement and calibration.
- dSPACE Merge Tools:
  - Merging DD files via DD three-way merge in TargetLink.
  - Using other tools in combination with TargetLink, such as Model Compare or dSPACE AUTOSAR Compare, in branch-merged-based workflows.

    Refer to Model Compare Guide and dSPACE AUTOSAR Compare Manual.
- Inheritance of TargetLink properties from Simulink masks:
  - Setting mask parameters for TargetLink block properties.
- Enhanced Adaptive AUTOSAR support:
  - SIL simulation of Adaptive AUTOSAR components.
  - Modeling of select parts of a service-based communication and accessing persistent memory as described by ara::per.

For more information on all the new features, refer to New Features of TargetLink 5.1 on page 168.

For more information on the TargetLink migration aspects (TargetLink, TargetLink AUTOSAR Module, TargetLink Data Dictionary), refer to Migrating to TargetLink 5.1 and TargetLink Data Dictionary 5.1 on page 183.

VEOS

The new key features of VEOS are:

- Running offline simulations on Linux.
- Support for the new VECU container file format
- Using VEOS via command line.
- Support for AUTOSAR R19-11
- Support for the Ocu driver module of the microcontroller abstraction layer

For more information on the new features, refer to New Features of VEOS 5.1 on page 227.

Aspects of Migrating from Previous Releases

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

Migrating to dSPACE Release 2020-B

Introduction	After you install Release 2020-B, some additional steps might be required.
Migrating from dSPACE Release 2020-A	**Product-specific migration steps** Product-specific migration steps are generally performed automatically. For exceptions, refer to the product-specific migration descriptions.
Migrating from dSPACE Release 2019-B or earlier	To migrate from dSPACE Release 2019-B or earlier to Release 2020-B, you also have to perform the migration steps of the intervening dSPACE Releases. All of the required migration steps can be performed using the software from dSPACE Release 2020-B. For more information on the required migration steps, refer to the *New Features and Migration* documents of the intervening dSPACE Releases.
Previous release documents	The PDF files of previous Releases are called `NewFeaturesAndMigrationxx.pdf`, where xx stands for the Release number.

You can find the *New Features and Migration* files for previous Releases in the following locations:

- In the installation folder of the current dSPACE Help. Refer to `C:\Program Files\Common Files\dSPACE\Help 2020-B\Print\PreviousReleases`.
- On the dSPACE DVDs. Refer to `\Doc\PreviousReleases`.
- At www.dspace.com/go/migration for download. Here, you can also find *New Features and Migration* documents for very early Releases.

AutomationDesk

Where to go from here

Information in this section

New Features of AutomationDesk 6.4

General enhancements

Enhanced SYNECT interface For the creation of elements or the update of modified elements, you can synchronize AutomationDesk custom libraries with SYNECT test cases via the push and pull operations. If there are elements in AutomationDesk that you do not want to synchronize because they are not relevant for use in SYNECT, or not fully implemented, you can now specify which elements to ignore during synchronization. The ignore state is displayed in the element's tooltip. The ignore flags in AutomationDesk can be combined with the ignore options in the SYNECT ECXML file for the AutomationDesk plug-in.

Ignore options set by SYNECT ECXML files in AutomationDesk 6.3 projects are automatically migrated to the ignore flags.

For more information, refer to:

- SYNECT (🕮 AutomationDesk Basic Practices)
- **`Set Ignore Flag`** (🕮 AutomationDesk Basic Practices)
- **`Clear Ignore Flag`** (🕮 AutomationDesk Basic Practices)
- **`Set Ignore Flags via ECXML`** (🕮 AutomationDesk Basic Practices)

Enhancement to the File data object The files of File data objects in a custom library can now be managed in a custom library's `attachments` folder. Such a file is specified in the following format.

```
<CustomLibraryName>::<FileName>
```

For more information, refer to File (📖 AutomationDesk Basic Practices)

Enhanced message handling Support of the Message Reader API. You can use the API to read out log messages of the dSPACE Log, for example, in C# applications and Python scripts. The API also lets you filter log messages by dSPACE product and by time, for example.

For more information, refer to Message Reader API (📖 AutomationDesk Basic Practices).

Enhancements to the COM API

The AutomationDesk COM API provides the following enhancements:

- The feature to ignore elements for synchronizing with SYNECT is supported by the following new properties:
  - `Ignored`

    To set or get the ignore flag of the current object and its child elements.
  - `IsIgnored`

    To get the ignore state of the current object.
- The *Clear Values* feature in AutomationDesk is now supported by the following methods:
  - `ClearValue`

    To clear the value of the current data object.
  - `ClearValues`

    To clear the specified values recursively in the current structure element.

For more information, refer to 📖 AutomationDesk Automation.

Migrating to AutomationDesk 6.4

General migration aspects

If you open an AutomationDesk project with a later AutomationDesk version, the software automatically detects whether a migration is required. For AutomationDesk projects that are saved in the legacy format, i.e., that were created with AutomationDesk 6.1 or earlier, you must click **OK** in the message dialog to start the migration. The migration of AutomationDesk projects in the new XML format do not have to be confirmed. Save the migrated project to another path or name.

An AutomationDesk project loaded and saved with AutomationDesk 6.3 and later can be opened in AutomationDesk 6.2. The different schema versions are then displayed in a warning message. It might not be possible to execute the project with the earlier AutomationDesk version.

Note

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

- You must create backups of the project and of the linked custom libraries.
- AutomationDesk must be running properly. The Log Viewer must not display any error messages.
- The built-in libraries, required custom libraries, and other packages must be loaded properly.

If you use a version control system, there are some preconditions for successful migration. Refer to How to Migrate Projects or Custom Libraries Under Version Control (🕮 AutomationDesk Basic Practices).

For more information, refer to Migrating AutomationDesk (🕮 AutomationDesk Introduction And Overview).

Migration to the new serialization format introduced with AutomationDesk 6.2

If you open an AutomationDesk project or a custom library created with AutomationDesk 6.1 or earlier, the data is automatically migrated to the new XML format. The migration considers the standard serialization format, exported projects and custom libraries in ZIP format, and exported projects and custom libraries in the legacy XML format.

If you use a version control system, you can get and open projects and custom libraries in the legacy formats. They are automatically migrated to the new serialization format and lose their connections to the version control system. You must add the migrated projects and custom libraries to new version control projects to avoid mixing the formats in the repository. For more information, refer to How to Migrate Projects or Custom Libraries Under Version Control (🕮 AutomationDesk Basic Practices).

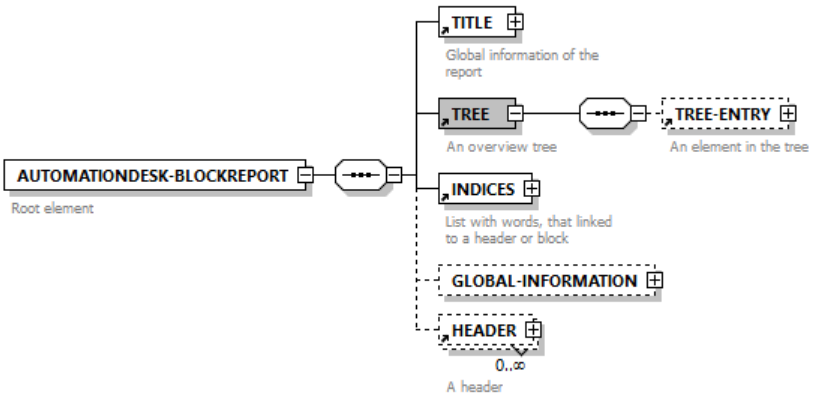
Project import restricted as of Release 2021-A

As of the dSPACE Release 2021-A, AutomationDesk will support the direct import only of projects last saved with one of the previous seven AutomationDesk versions.

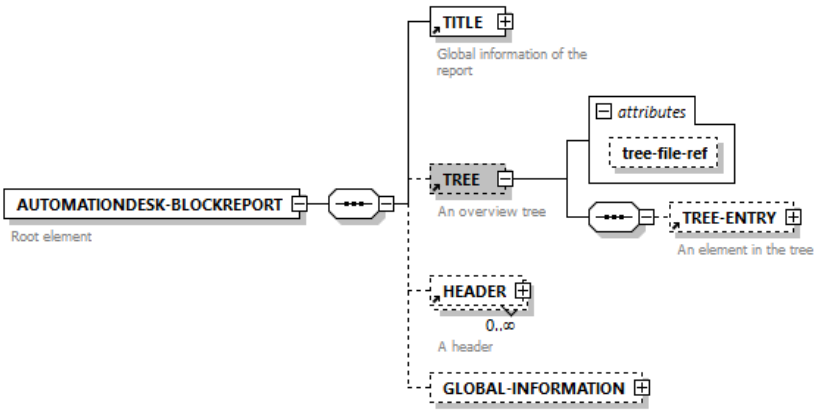
Migrating custom stylesheets

If you use custom stylesheets for your HTML or PDF reports, and you use the `TREE` element for the navigation elements, you must adapt them.

Up to and including AutomationDesk 6.3, the XML schema for the `AUTOMATIONDESK-BLOCKREPORT` element contained five elements. The `TREE` element was directly filled with content by `TREE-ENTRY` elements.



As of AutomationDesk 6.4, the XML schema for the `AUTOMATIONDESK-BLOCKREPORT` element contains only four elements. The `INDICES` element is removed. The `TREE` element now provides an attribute for specifying a separate file with the content of the required tree entries.



If you do not migrate the `TREE` elements, the navigation in the generated HTML report or the PDF bookmarks are not available or set to default values.

For an example, refer to the custom stylesheets available in the AutomationDesk demo folder.

Discontinued libraries and blocks

If you open a project containing discontinued elements in AutomationDesk 6.4, the discontinued data objects are replaced by Discontinued data object data objects and the discontinued blocks are replaced by Discontinued block blocks during the automatic project update. This lets you load your projects and search for blocks and data objects to be migrated. If you execute a project containing elements of a discontinued library via AutomationDesk or an API script, project execution will stop with an exception.

For more information on migration aspects, refer to Migrating AutomationDesk (🕮 AutomationDesk Introduction And Overview).

Canceled discontinuations

MATLAB Access library The announced discontinuation of the HideApplication, ShowApplication, MinimizeCommandWindow, and RestoreCommandWindow automation blocks has been canceled.

Automotive Simulation Models (ASM)

Where to go from here

Information in this section

All ASM Products

Changes in All ASM Demo Models

New model structure

All ASM demo models are now designed in a new, more modular structure to meet the requirements of today's model-based development. For detailed information, refer to ASM Model Structure (🕮 ASM User Guide).

ASM Base InCylinder

Where to go from here

Information in this section

New Features of ASM Base InCylinder Blockset 2.7

AMBIENT_CONDITIONS block

This block is new. You can use it for switching the ambient temperature and pressure between constant conditions, measurement data, and calculated values based on altitude.

Migrating to ASM Base InCylinder Blockset 2.7

New data type

The data type of the State_Engine[0|1|2|3|4] inport is changed from *uint8* to *double*.

This applies to the following blocks:

- ILDE_SPEED_CONTROL
- EGR_RATE_CONTROL
- RAIL_CONTROL
- THROTTLE_CONTROL
- START_STOP

ENGINE_TORQUE_SET_INTERVENTION block

The electric machine torque from the belt-integrated starter generator is now taken into consideration for the combustion torque request during battery charging.

Related topics

Basics

Migrating ASM Models (📖 ASM User Guide)

ASM Diesel Engine

Where to go from here

Information in this section

New Features of ASM Diesel Engine Blockset 2.8

AMBIENT_CONDITIONS block

This block is new. You can use it for switching the ambient temperature and pressure between constant conditions, measurement data, and calculated values based on altitude.

Changes in the ASM Diesel Engine Demo Model

P0 hybrid with belt-integrated starter generator

A belt-integrated starter generator for simulating a hybrid vehicle with a P0 topology has been introduced to the model. You can parameterize the different driving modes and torque request maps in ModelDesk.

Maneuver types

The *Offline Manual* and *Online Manual* maneuver types (MATLAB parameter CPT.MDL_SW.Sw_Maneuver) have been merged.

The following table shows you the new maneuver types and the corresponding values of the CPT.MDL_SW.Sw_Maneuver parameter:

New Maneuver Types	Former Maneuver Types
1 Manual	1 Offline Manual 2 Online Manual
3 Stimulus	3 Stimulus Maneuver
4 Test Cycle[1)]	4 Driver Maneuver

[1)] This maneuver type has been renamed, but its functionality has not changed.

When you work in Simulink, you can now control maneuver signals, such as the accelerator or brake pedal, via new dashboard instruments in: **/Environment/Plant/UserInterface/PAR_Plant/Manual_Controller**.

Migrated ASM models are not affected by these changes.

Migrating to ASM Diesel Engine Blockset 2.8

New data type

The data type of the **State_Engine[0|1|2|3|4]** inport is changed from *uint8* to *double*.

This applies to the following blocks:

- ILDE_SPEED_CONTROL
- EGR_RATE_CONTROL
- RAIL_CONTROL
- RAIL_CONTROL_CRANKBASED
- START_STOP

ENGINE_TORQUE_SET_INTERVENTION block

The electric machine torque from the belt-integrated starter generator is now taken into consideration for the combustion torque request during battery charging.

Related topics

Basics

Migrating ASM Models (🕮 ASM User Guide)

ASM Diesel Exhaust

Where to go from here

Information in this section

Changes in the ASM Diesel Exhaust Demo Model

Non-air SCR demo model

The non-air SCR demo model has been discontinued. If your SCR supply system is not air-assisted, i.e., if there is pure AdBlue injection into the exhaust system, use the following demo model instead: ASM_DieselExhaust_lib/Demos/Exhaust/ExhaustSystem_DOC_DPF_SCR. In this case, you must deactivate the air path, e.g., by switching off the AIR_NON_RETURN_VALVE and/or THROTTLE components. This lets you use the same model for several variants of the SCR supply system.

Migrating to ASM Diesel Exhaust Blockset 2.1.11

THROTTLE block

This block has a new parameter for switching the valve on and off: Sw_State_Valve.

There is also a new inport: T_Ambient[K]. It remains active during migration from an older Release. The behavior of the block does not change.

AIR_NON_RETURN_VALVE block

This block has a new parameter for switching the valve on and off: Sw_State_Valve.

There is also a new inport: T_Ambient[K]. It remains active during migration from an older Release. The behavior of the block does not change.

Related topics

Basics

Migrating ASM Models (📖 ASM User Guide)

ASM Diesel InCylinder

Changes in the ASM Diesel InCylinder Demo Model

P0 hybrid with belt-integrated starter generator

A belt-integrated starter generator for simulating a hybrid vehicle with a P0 topology has been introduced to the model. You can parameterize the different driving modes and torque request maps in ModelDesk.

Maneuver types

The *Offline Manual* and *Online Manual* maneuver types (MATLAB parameter CPT.MDL_SW.Sw_Maneuver) have been merged.

The following table shows you the new maneuver types and the corresponding values of the CPT.MDL_SW.Sw_Maneuver parameter:

New Maneuver Types	Former Maneuver Types
1 Manual	1 Offline Manual 2 Online Manual
3 Stimulus	3 Stimulus Maneuver
4 Test Cycle[1)]	4 Driver Maneuver

[1)] This maneuver type has been renamed, but its functionality has not changed.

When you work in Simulink, you can now control maneuver signals, such as the accelerator or brake pedal, via new dashboard instruments in: /Environment/Plant/UserInterface/PAR_Plant/Manual_Controller.

Migrated ASM models are not affected by these changes.

ASM Drivetrain Basic

Where to go from here

Information in this section

New Features of ASM Drivetrain Basic Blockset 5.4

SOFTECU_BSG block

This is a new block for simulating a belt-integrated starter generator. Different hybrid modes are selected based on pedal positions. The electric torque request depends on mode-specific maps.

Road components

The Road subsystem has been upgraded with new blocks to calculate the vehicle position and ambient conditions.

Changes in the ASM Drivetrain Basic Demo Model

Maneuver types

The *Offline Manual* and *Online Manual* maneuver types (MATLAB parameter CPT.MDL_SW.Sw_Maneuver) have been merged.

The following table shows you the new maneuver types and the corresponding values of the CPT.MDL_SW.Sw_Maneuver parameter:

New Maneuver Types	Former Maneuver Types
1 Manual	1 Offline Manual 2 Online Manual
3 Stimulus	3 Stimulus Maneuver
4 Test Cycle[1)]	4 Driver Maneuver

[1)] This maneuver type has been renamed, but its functionality has not changed.

When you work in Simulink, you can now control maneuver signals, such as the accelerator or brake pedal, via new dashboard instruments in: /Environment/Plant/UserInterface/PAR_Plant/Manual_Controller.

Migrated ASM models are not affected by these changes.

VEHICLE_POSITION block	This block is new. It calculates the driven distance and altitude.
AMBIENT block	This block is new. It calculates the ambient temperature and pressure as a function of the altitude.
ESP_FAST_TORQUE_SET block	This block is new. It is used for basic ESP intervention.
ENGINE_OPERATION_BASIC block	The block sets the correct engine state immediately when the start button is stimulated with the *starter on* state.
ESP Intervention block	The new ESP_FAST_TORQUE_SET block has been added to the Engine Soft ECU model but is not actively used.

Migrating to ASM Drivetrain Basic Blockset 5.4

SWITCHES_DRIVETRAINBASIC block	The global Goto block has been replaced by an outport.
SIMPLE_GEAR block	The Omega_In[rpm] outport is added to the ASMSignalBus.
LONGITUDINAL_CONTROL block	The data type of the State_Engine[0|1|2|3|4] inport has been changed from *uint8* to *double*. The block has a new inport: Enable_v_Vehicle_Init. To keep the model behavior unchanged, it is connected to a constant block during migration.
GEAR_SHIFTER block	The data type of the State_Engine[0|1|2|3|4] inport has been changed from *uint8* to *double*. The block has a new inport: Enable_v_Vehicle_Init. To keep the model behavior unchanged, it is connected to a constant block during migration.
START_BUTTON block	The data type of the State_Engine[0|1|2|3|4] inport has been changed from *uint8* to *double*.

CLUTCH_ENGAGEMENT_ CONTROL block	The block has a new inport: Enable_v_Vehicle_Init. To keep the model behavior unchanged, it is connected to a constant block during migration.
SHIFT_STRATEGY block	The block has a new inport: Enable_v_Vehicle_Init. To keep the model behavior unchanged, it is connected to a constant block during migration.
TORQUE_INTERVENTION_ CONTROL block	The block has a new inport: Enable_v_Vehicle_Init. To keep the model behavior unchanged, it is connected to a constant block during migration.
LOCKUP_CLUTCH_CONTROL block	The unit of the control signal has been changed from [0_1] to [%] and a first-order-dynamics filter has been added to the control signal to restore the previous behavior.
ENGINE block	There is a new outport for engine-indicated torque: Trq_Ind_Engine. The outport is terminated during migration.
ENGINE_OPERATION_BASIC block	The block has new inports. To keep the model behavior unchanged, the new inports are connected to constant blocks during migration.
IDLE_SPEED_CONTROL_ ENGINE_BASIC block	The block has a new outport for set point of idle speed: n_Engine_Idle_Set. The outport is terminated during migration.
TORQUE_INTERVENTION_ ENGINE_BASIC block	The block is shifted to the *former versions*. The successor is the TORQUE_INTERVENTION_BASIC block in: `ASM_DrivetrainBasic_lib/SoftECU/SoftECU_Engine/` that covers the same functionality. During migration the link to the TORQUE_INTERVENTION_BASIC block is changed to the former version in `ASM_DrivetrainBasic_lib/SoftECU/SoftECU_Engine/FormerVersions/TORQUE_INTERVENTION_BASIC_1_0`.

ENGINE_TORQUE_SET_ INTERVENTION block	The block is renamed to SHIFT_TORQUE_SET. You can use it to implement torque intervention during a gearshift.

Related topics

Basics

Migrating ASM Models (🕮 ASM User Guide)

ASM Electric Components

Where to go from here

Information in this section

New Features of ASM Electric Components Blockset 3.11

RC_BATTERY block

This block is new. It simulates an RC circuit.

The battery-equivalent circuit includes a number of RC circuits to simulate the dynamic voltage drop effects of a battery.

The RC_BATTERY block simulates one RC circuit consisting of a resistor and capacitor in parallel connection and provides a dynamic voltage drop for the BATTERY_MULTICELL model. If required, you can use several of the RC_BATTERY blocks to achieve a higher accuracy in the simulation of the loss voltage.

Note

The resistance and the capacitance of the RC circuit can depend on the temperature and the state of charge of the battery

BATTERY_MULTICELL block

The block has a new inport for external voltage drop. You can use it to simulate the dynamic voltage drop with several external RC circuits.

BRAKE_CONTROL_BEV block

You can now add a minimum velocity parameter to activate recuperation only if the vehicle velocity is higher than this minimum velocity.

A rate limitation for the recuperation torque request to smooth hard steps in the torque request has been added.

SOFT_ECU_CHARGING_STATION block

A new charging station voltage set mode has been added, where the vehicle requests the voltage and the current. The soft ECU sets the voltage demand with a current limitation according to the vehicle request.

	There is a new parameter to set the maximum energy of the charging station: E_Max[Wh].
TRQ_REQUEST_ COORDINATION_BEV block	A cross-fade between recuperation and acceleration torque, and a first-order delay for the request torque have been added to avoid abrupt torque request steps and provide a smoother torque request handling.
KEY_SIGNALS_ICE block	The block has a new inport to hide starter requests if the state of the DC link is *zero*: State_DCLink[0Off|1On]
ELECTRIC_MACHINE_BASIC block	A new I_DCLink[A] outport has been added. Because the machine voltage is usually lower than the battery (or the DC link) voltage, the battery (or the DC link) current is calculated according to the power balance.

Changes in the ASM Electric Components Demo Model

Battery Electric Vehicle Demo (BEV) model	Drivetrain and driver components are now automatically initialized for scenarios with an initial vehicle speed. This includes engaging the selector lever and initializing the different drivetrain speeds accordingly to ensure a smooth scenario startup.
Vehicle Dynamics Hybrid demo	A low-voltage battery has been added that supplies the electrical power to the low-voltage loads, such as the fan of the air conditioning system. The low-voltage battery is charged with energy applied by the high-voltage battery system. As the electrical load, an air conditioning system has been added. It consists of an electrically driven compressor, fan, and thermal interior model components (the refrigerant circuit, the air duct, and the cabin). The charging station model of a DC charging system has been added. The charging station controller receives a battery charging current demand from the vehicle and calculates a desired DC charging voltage to deliver the requested current.
Engine Gasoline Hybrid demo	A low-voltage battery has been added that supplies the electrical power to the low-voltage loads, such as the fan of the air conditioning system. The low-voltage battery is charged with energy applied by the high-voltage battery system.

As the electrical load, an air conditioning system has been added. It consists of an electrically driven compressor, fan, and thermal interior model components (the refrigerant circuit, the air duct, and the cabin).

The charging station model of a DC charging system has been added. The charging station controller receives a battery charging current demand from the vehicle and calculates a desired DC charging voltage to deliver the requested current.

Vehicle Electrical System demo

For each of the two high-voltage batteries that are connected in parallel, the dynamic voltage drop is now simulated with three external RC circuits to achieve a higher accuracy in the simulation of the voltage drop. The resistor and capacity value of the RC circuit can depend on the battery temperature and state of charge.

Migrating to ASM Electric Components Blockset 3.11

BRAKE_CONTROL block

For the *ICE only* drive mode, the Trq_Request_Brake signal is now set.

To ensure the former simulation behavior, during migration of an older model the link to the library is changed to the former implementation in: FormerVersion/BRAKE_CONTROL_4_0.

To use the new implementation, drag the BRAKE_CONTROL block from the ASM Electric Components Library to your model and adapt the inports, outports, and parameters to your requirements.

TRQ_REQUEST_ COORDINATION_BEV block

A cross-fade between recuperation and acceleration torque and a first-order delay for the request torque has been added to avoid abrupt torque request steps and provide a smoother torque request handling.

To ensure the former simulation behavior, during migration of an older model, the, link to the library is changed to the former implementation in: FormerVersion/TORQUE_REQUEST_COORDINATION_4_0.

To use the new implementation, drag the TORQUE_REQUEST_COORDINATION block from the ASM Electric Components Library to your model and adapt the inports, outports, and parameters to your requirements.

BRAKE_CONTROL_BEV block

To ensure the former simulation behavior, during migration of older models, the link to the library is changed to the former implementation in: FormerVersion/BRAKE_CONTROL_BEV_2_0.

	To use the new implementation, drag the BRAKE_CONTROL_BEV block from the ASM Electric Components Library to your model and adapt the inports, outports, and parameters to your requirements.
RLC_LOAD_SERIES block	The setting of the initial capacitance voltage via the block inport V_Init[V] has been corrected.
KEY_SIGNALS_ICE block	There is a new inport: State_DCLink[0Off|1On]. You can use it to hide the starter request if the state of the DC link is *zero*. The new inport is connected with a constant block with the value *one* during migration.
SOFT_ECU_CHARGING_STATION block	The block has two new parameters: Sw_CtrlMode and Const_E_Max. During migration, the new parameters are set to useful values to maintain the previous behavior.
ALTERNATOR block	The I_Terminal[A] signal has been added to the ASMSignalBus.
BATTERY block	There have been some internal adaptations. These have no effect on the simulation result or functionality of the component.
BATTERY_MULTICELL block	There is a new inport: V_Loss_External[V]. It is connected to a constant block with the value *zero* during migration.
ELECTRIC_MACHINE_BASIC block	There is a new inport: I_DCLink[A]. It is connected to a Terminator block during migration. According to the power balance $P_{DCLink} = P_{In,\ EM}$, you should use this new outport instead of the I[A] outport. Therefore, you have to connect the I[A] outport to a Terminator block.
Related topics	Basics Migrating ASM Models (🕮 ASM User Guide)

ASM Environment

Where to go from here

Information in this section

New Features of ASM Environment Blockset 4.13

DRIVETRAIN_INIT block

This block is new. You can use it to trigger drivetrain initialization for scenarios with an initial vehicle speed.

Migrating to ASM Environment Blockset 4.13

GEAR_SHIFTER block

The block has a new inport: `Enable_v_Vehicle_Init`. To keep the model behavior unchanged, it is connected to constant block during migration.

The new inport is used to enable drivetrain initialization for scenarios with an initial vehicle speed.

LONGITUDINAL_CONTROL block

The block has a new inport: `Enable_v_Vehicle_Init`. To keep the model behavior unchanged, it is connected to constant block during migration.

The new inport is used to enable drivetrain initialization for scenarios with an initial vehicle speed.

SPEED_PROFILER block

The block has a new inport: `Enable_v_Vehicle_Init`. To keep the model behavior unchanged, it is connected to constant block during migration.

The new inport is used to enable drivetrain initialization for scenarios with an initial vehicle speed.

Related topics

Basics

Migrating ASM Models (📖 ASM User Guide)

ASM Gasoline Engine Basic

Migrating to ASM Gasoline Engine Basic Blockset 2.2.6

New data type

The data type of the State_Engine[0|1|2|3|4] inport is changed from *uint8* to *double*.

This applies to the following blocks:

- ILDE_SPEED_CONTROL
- THROTTLE_CONTROL
- START_STOP

ENGINE_TORQUE_SET_ INTERVENTION block

The electric machine torque from the belt-integrated starter generator is now taken into consideration for the combustion torque request during battery charging.

Related topics

Basics

Migrating ASM Models (🕮 ASM User Guide)

ASM Gasoline Engine

Where to go from here

Information in this section

New Features of ASM Gasoline Engine Blockset 4.2

AMBIENT_CONDITIONS block

This block is new. You can use it for switching the ambient temperature and pressure between constant conditions, measurement data, and calculated values based on altitude.

Changes in the ASM Engine Gasoline Demo Model

P0 hybrid with belt-integrated starter generator

A belt-integrated starter generator for simulating a hybrid vehicle with a P0 topology has been introduced to the model. You can parameterize the different driving modes and torque request maps in ModelDesk.

Maneuver types

The *Offline Manual* and *Online Manual* maneuver types (MATLAB parameter CPT.MDL_SW.Sw_Maneuver) have been merged.

The following table shows you the new maneuver types and the corresponding values of the CPT.MDL_SW.Sw_Maneuver parameter:

New Maneuver Types	Former Maneuver Types
1 Manual	1 Offline Manual 2 Online Manual
3 Stimulus	3 Stimulus Maneuver
4 Test Cycle[1)]	4 Driver Maneuver

[1)] This maneuver type has been renamed, but its functionality has not changed.

When you work in Simulink, you can now control maneuver signals, such as the accelerator or brake pedal, via new dashboard instruments in: **/Environment/Plant/UserInterface/PAR_Plant/Manual_Controller**.

Migrated ASM models are not affected by these changes.

Migrating to ASM Gasoline Engine Blockset 4.2

New data type

The data type of the **State_Engine[0|1|2|3|4]** inport is changed from *uint8* to *double*.

This applies to the following blocks:

- ILDE_SPEED_CONTROL
- EGR_RATE_CONTROL
- RAIL_CONTROL
- INJECTOR_MODE
- THROTTLE_CONTROL
- IGNITION_SET
- RAIL_CONTROL_CRANKBASED
- START_STOP

ENGINE_TORQUE_SET_INTERVENTION block

The electric machine torque from the belt-integrated starter generator is now taken into consideration for the combustion torque request during battery charging.

Related topics

Basics

Migrating ASM Models (📖 ASM User Guide)

ASM Gasoline InCylinder

Where to go from here

Information in this section

Changes in the ASM Gasoline InCylinder Demo Model

P0 hybrid with belt-integrated starter generator

A belt-integrated starter generator for simulating a hybrid vehicle with a P0 topology has been introduced to the model. You can parameterize the different driving modes and torque request maps in ModelDesk.

Maneuver types

The *Offline Manual* and *Online Manual* maneuver types (MATLAB parameter CPT.MDL_SW.Sw_Maneuver) have been merged.

The following table shows you the new maneuver types and the corresponding values of the CPT.MDL_SW.Sw_Maneuver parameter:

New Maneuver Types	Former Maneuver Types
1 Manual	1 Offline Manual 2 Online Manual
3 Stimulus	3 Stimulus Maneuver
4 Test Cycle[1)]	4 Driver Maneuver

[1)] This maneuver type has been renamed, but its functionality has not changed.

When you work in Simulink, you can now control maneuver signals, such as the accelerator or brake pedal, via new dashboard instruments in: /Environment/Plant/UserInterface/PAR_Plant/Manual_Controller.

Migrated ASM models are not affected by these changes.

Migrating to ASM Gasoline InCylinder Blockset 2.8

New data type

The data type of the State_Engine[0|1|2|3|4] inport is changed from *uint8* to *double*.

This applies to the following blocks:

- INJECTOR_MODE
- IGNITION_SET

Related topics

Basics

Migrating ASM Models (📖 ASM User Guide)

ASM Pneumatics

New Features of ASM Pneumatics Blockset 2.0.12

Demo model access

In earlier Releases, the ASM Pneumatics Library contained prepared pneumatic subsystems in the **Models** subsystem. With these subsystems, an ASM Pneumatics Model had to be created manually based on the ASM Truck Trailer demo model.

These prepared subsystems are removed. Now, an executable ASM Pneumatics Model is available instead.

For information on access to the model, refer to How to Download the Pneumatics Demo Models (📖 ASM Pneumatics Reference).

ASM Traffic

Where to go from here

Information in this section

New Features of ASM Traffic Blockset 4.2

FELLOW_POS_VEL block	You can now simulate realistic motion effects of fellow vehicles used as trailer and/or motorbike.
Scenarios with initial vehicle speed	Drivetrain and driver components are now automatically initialized for scenarios with an initial vehicle speed. This includes setting the correct gear or selector lever and initializing the different drivetrain speeds accordingly to ensure a smooth scenario startup.

Changes in the ASM Traffic Demo Model

FELLOW_POS_VEL block	This subsystem is new. You can now define trailers and/or motorcycles as fellows. You can do this with UserSignal 2 of the TRAFFIC_SCHEDULER block.
Drivetrain model	Initial values of the integrators are now connected to nonzero values to enable correct initialization for scenarios with an initial vehicle speed. A new flag is used to indicate this feature.
SHIFT_TORQUE_SET block	This block is new. It enables torque intervention during gearshift.
ENGINE_OPERATION_BASIC block	The block sets the correct engine state immediately when the start button is stimulated with the *starter on* state.

Migrating to ASM Traffic Blockset 4.2

SOFT_ECU_ACC block	This block has been extended by two additional outports in reference to the new model structure. The new outports are used as the Front Collision Warning (FCW) flag and Autonomous Emergency Brake (AEB) flag. These flags were available in the ASMSignalBus in earlier Releases.
COLLISION_SENSOR block	This block has been extended by a new inport: Object_States. It is used for an improved initialization process.
OBJECT_SENSOR_2D_ CALCULATION block	This block has been extended by a new inport: Object_States. It is used for an improved initialization process.
OBJECT_SENSOR_3D_ CALCULATION block	This block has been extended by a new inport: Object_States. It is used for an improved initialization process.
OSI_GROUNDTRUTH_ INTERFACE block	This block has been extended by center lines as defined by the Open Simulation Interface (OSI). Refer to https://github.com/OpenSimulationInterface/open-simulation-interface. There is a center line for each driving lane. This information is available via the GroundTruth_Interface_Data[] outport.
FELLOW_PARAMETERS block	The internal data structure has been extended to support trailers and/or motorcycles as fellows.

Related topics

Basics

Migrating ASM Models (📖 ASM User Guide)

ASM Trailer

Where to go from here

Information in this section

New Features of ASM Trailer Blockset 2.6.8

Scenarios with initial vehicle speed

Drivetrain and driver components are now automatically initialized for scenarios with an initial vehicle speed. This includes setting the correct gear or selector lever and initializing the different drivetrain speeds accordingly to ensure a smooth scenario startup.

Changes in the ASM Trailer Demo Model

Drivetrain model

Initial values of the integrators are now connected to nonzero values to enable correct initialization for scenarios with an initial vehicle speed. A new flag is used to indicate this feature.

SHIFT_TORQUE_SET block

This block is new. It enables torque intervention during gearshift.

ENGINE_OPERATION_BASIC block

The block sets the correct engine state immediately when the start button is stimulated with the *starter on* state.

Migrating to ASM Trailer Blockset 2.6.8

SENSOR_POS_TRAILER block	The library link of the cross-product has been corrected.
DOLLY_MASS block	The calculation of the inertia tensor has been simplified.
TRAILER_MOVEMENT_INFO block	There is a new inport: SW_Trailer. During migration, it is connected to constant blocks to keep the behavior unchanged.
TRAILER_DYNAMICS_ VARIANT_SWITCHES block	An outport is terminated during migration.
HITCH_POS_DOLLY block	The signal bus has been renamed to DollyHitch_Position to have a unique name.

Related topics

Basics

Migrating ASM Models (📖 ASM User Guide)

ASM Truck

Where to go from here

Information in this section

New Features of ASM Truck Blockset 3.1.4

Scenarios with initial vehicle speed

Drivetrain and driver components are now automatically initialized for scenarios with an initial vehicle speed. This includes setting the correct gear or selector lever and initializing the different drivetrain speeds accordingly to ensure a smooth scenario startup.

Changes in the ASM Truck Demo Model

Drivetrain model

Initial values of the integrators are now connected to nonzero values to enable correct initialization for scenarios with an initial vehicle speed. A new flag is used to indicate this feature.

SHIFT_TORQUE_SET block

This block is new. It enables torque intervention during gearshift.

ENGINE_OPERATION_BASIC block

The block sets the correct engine state immediately when the start button is stimulated with the *starter on* state.

Migrating to ASM Truck Blockset 3.1.4

VEHICLE_FLEXIBLE_MASS_AND_ADD_LOADS block	The Pos_x_CoG_Vehicle_Total[m] and Inertia_z_Vehicle_Total[kgm2] signals have been added to the ASMSignalBus.

Related topics

Basics

Migrating ASM Models (📖 ASM User Guide)

ASM Turbocharger

Migrating to ASM Turbocharger Blockset 3.2.7

POSTTURBHPMAN block	The starting value for air mass calculations is now based on the gas constant for fresh air instead of exhaust.
POS_DISPL_COMPRESSOR block	A unit conversion of the compressor shaft speed is added for torque calculations. The old torque calculation was wrong. This has been fixed. The signal output has changed. The resulting torque may be higher. This block is normally not contained in a demo and requires extensive model adaptions to be used.

Related topics

Basics

Migrating ASM Models (🕮 ASM User Guide)

ASM Utils

Where to go from here

Information in this section

New Features of ASM Utils 4.2

ASMSignalInterface

The ModelDesk PlotSignalCollector has been renamed to *ASMSignalInterface*. It is now possible to use several instances of the block in the model. Each ASM module contains one separate ASMSignalInterface block. For more information, refer to ASMSignalInterface (🕮 ASM User Guide).

ASMSignalInterface Manager

The ASMSignalInterface Manager replaces the Toggle ModelDesk Plotting block to support the multi-instance capability of the ASMSignalInterface. For more information, refer to ASMSignalInterface Manager (🕮 ASM User Guide).

Migrating to ASM Utils 4.2

ModelDesk PlotSignalCollector

The PlotSignalCollector block is replaced by the new ASMSignalInterface. After migration, you must update the ASMSignalInterface.

After migration of the model, the MATLAB Command Window shows a message containing a link. Click the link to update the ASMSignalInterface.

Toggle ModelDesk Plotting block

The Toggle ModelDesk Plotting block has been replaced by the ASMSignalInterface Manager. No further action is necessary.

ASM Vehicle Dynamics

Where to go from here

Information in this section

New Features of ASM Vehicle Dynamics Blockset 4.2

Scenarios with initial vehicle speed

Drivetrain and driver components are now automatically initialized for scenarios with an initial vehicle speed. This includes setting the correct gear or selector lever and initializing the different drivetrain speeds accordingly to ensure a smooth scenario startup.

Adams2ASM Converter

The Adams2ASM converter has been expanded to include new functionalities. In addition to the parameterization for the suspension kinematics and compliance, it is now possible to import parameters of the entire vehicle into ASM. These include the mass, center of gravity, inertia tensor of the entire vehicle as well as the engine map and the configuration of the drivetrain.

Changes in the ASM Vehicle Dynamics Demo Model

Drivetrain model

Initial values of the integrators are now connected to nonzero values to enable correct initialization for scenarios with an initial vehicle speed. A new flag is used to indicate this feature.

SHIFT_TORQUE_SET block

This block is new. It enables torque intervention during gearshift.

ENGINE_OPERATION_BASIC block

The block sets the correct engine state immediately when the start button is stimulated with the *starter on* state.

Migrating to ASM Vehicle Dynamics Blockset 4.2

ASMSignalBus	New signals have been added to the ASMSignalBus. This applies to the following blocks: ▪ VEHICLE_MOVEMENT_INFO_CAR ▪ WHEEL_SPEED ▪ BRAKE_DISK ▪ VEHICLE_DYNAMICS_VARIANT_SWITCHES
CLUTCH_ENGAGEMENT_ CONTROL block	The block has a new inport: Enable_v_Vehicle_Init. To keep the model behavior unchanged, it is connected to constant block during migration. The new inport is used to enable drivetrain initialization for scenarios with an initial vehicle speed.
SHIFT_STRATEGY block	The block has a new inport: Enable_v_Vehicle_Init. To keep the model behavior unchanged, it is connected to constant block during migration. The new inport is used to enable drivetrain initialization for scenarios with an initial vehicle speed.
TORQUE_INTERVENTION_ CONTROL block	The block has a new inport: Enable_v_Vehicle_Init. To keep the model behavior unchanged, it is connected to constant block during migration. The new inport is used to enable drivetrain initialization for scenarios with an initial vehicle speed.
LOCKUP_CLUTCH_CONTROL block	The unit of the control signal has been changed from [0_1] to [%] and a first-order-dynamics filter has been added to the control signal to restore the previous behavior.
ENGINE block	There is a new outport for engine-indicated torque: Trq_Ind_Engine. The outport is terminated during migration.
ENGINE_OPERATION_BASIC block	The block has new inports. To keep the model behavior unchanged, the new inports are connected to constant blocks during migration.
IDLE_SPEED_CONTROL_ ENGINE_BASIC block	The block has a new outport for set point of idle speed: n_Engine_Idle_Set. The outport is terminated during migration.

TORQUE_INTERVENTION_ SLOW_ENGINE_BASIC block	The block is renamed to TORQUE_INTERVENTION_BASIC.
TORQUE_INTERVENTION_ FAST_ENGINE_BASIC block	The block is renamed to ESP_FAST_TORQUE_SET.
TORQUE_CONVERTER block	The grounds inside the block are replaced with constants during migration to avoid problems with underspecified signal dimensions.
GEARBOX_MT block	The block has a new inport: Enable_v_Vehicle_Init. To keep the model behavior unchanged, it is connected to constant block during migration. The new inport is used to enable drivetrain initialization for scenarios with an initial vehicle speed.
GEARBOX_MT_Rigid block	The block has a new inport: Enable_v_Vehicle_Init. To keep the model behavior unchanged, it is connected to constant block during migration. The new inport is used to enable drivetrain initialization for scenarios with an initial vehicle speed.
GEARBOX_AT block	The block has a new inport: Enable_v_Vehicle_Init. To keep the model behavior unchanged, it is connected to constant block during migration. The new inport is used to enable drivetrain initialization for scenarios with an initial vehicle speed.
GEARBOX_AT_Rigid block	The block has a new inport: Enable_v_Vehicle_Init. To keep the model behavior unchanged, it is connected to constant block during migration. The new inport is used to enable drivetrain initialization for scenarios with an initial vehicle speed.

Related topics

Basics

Migrating ASM Models (🕮 ASM User Guide)

Bus Manager (Stand-Alone)

New Features of the Bus Manager (Stand-Alone) 6.6

New supported AUTOSAR release	The Bus Manager now supports AUTOSAR system description files based on AUTOSAR Classic Platform Release R19-11.

ConfigurationDesk

Introduction

With ConfigurationDesk you can implement real-time applications for the SCALEXIO hardware or the MicroAutoBox III hardware.

Where to go from here

Information in this section

New Features of ConfigurationDesk 6.6

New names for the ConfigurationDesk software products

For a clearer separation of the ConfigurationDesk software products, the product names were changed as follows as of dSPACE Release 2020-B:

- ConfigurationDesk Implementation Version ⇒ ConfigurationDesk
- ConfigurationDesk Configuration Version ⇒ ConfigurationDesk for RapidPro

New operating system

With dSPACE Release 2020-B, dSPACE introduces a new basic operating system for SCALEXIO processing hardware. This operating system is based on real-time Linux and customized by dSPACE. The migration to Linux provides the following advantages:

- Faster deployment of new dSPACE features
- Improved real-time execution
- More straightforward third-party software support

The migration to Linux has an impact on binaries built for dSPACE Release 2020-A and earlier. Refer to Migrating to ConfigurationDesk 6.6 on page 79.

New V-ECU implementation features

New container format ConfigurationDesk supports the new V-ECU implementation container format that is introduced with dSPACE Release 2020-B. The new container format has the following benefits:

- It can contain data packages for different simulation platforms. This lets you reuse the same VECU implementation on different simulation platforms without any modifications.
- V-ECU implementation containers that contain TargetLink code already include the DSFXP library. You do not have to link it to the ConfigurationDesk application manually.

The file name extension of the new V-ECU implementation container format is VECU.

Support of additional CAN and LIN features for MCAL ConfigurationDesk now supports the following additional features for Microcontroller Abstraction Layer (MCAL) modules:

- CAN FD
- LIN Slaves
- Polling-based and interrupt-based communication for CAN

For more information, refer to Working with V-ECU Implementations (🕮 ConfigurationDesk Real-Time Implementation Guide).

Support of int64/uint64 ports

ConfigurationDesk now supports Simulink models, Simulink implementation container files (SIC files), and bus simulation container files (BSC files) that have model ports with the int64/uint64 data type.

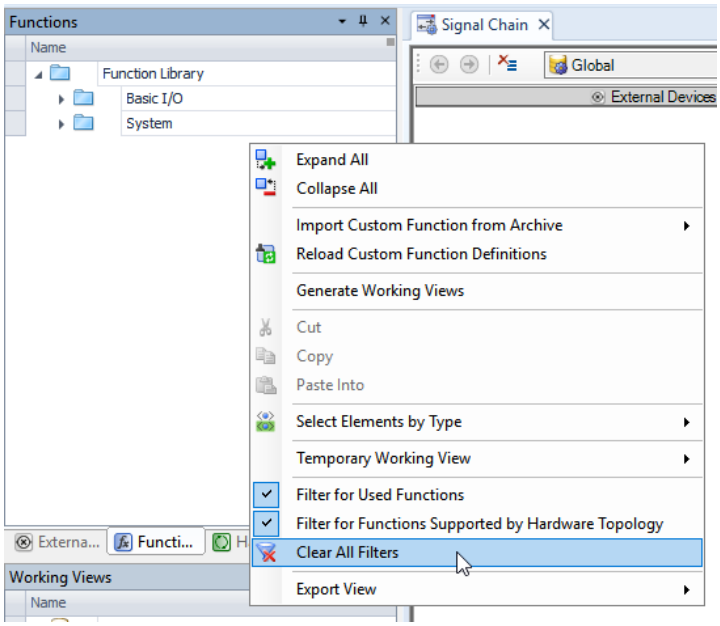
New features for ECU interfacing with SCALEXIO and MicroAutoBox III

ConfigurationDesk now supports XCP time stamps for ECU interfacing with SCALEXIO and MicroAutoBox III. If an ECU application with integrated XCP service supports time-stamping, the **ECU Interface Configuration** function block provides **TimestampRead** function ports when you import the related EIC file. If you map the **TimestampRead** function ports to model ports, the ECU transmits time stamps to real-time hardware.

For more information, refer to Overview of Ports and Basic Properties (ECU Interface Configuration) (🕮 ConfigurationDesk I/O Function Implementation Guide).

User interface improvements

Clear All Filters ConfigurationDesk now offers the Clear All Filters command that lets you disable all currently active filters in a pane.



Recommendations for model communication

The ConfigurationDesk Real-Time Implementation Guide now provides recommendations for setting up model communication between model port blocks with structured data ports. For optimum run-time performance of the model communication, specific mappings of structured data ports are recommended. For more information, refer to Model Communication Recommendations for Model Port Blocks with Structured Data Ports (🕮 ConfigurationDesk Real-Time Implementation Guide).

New Features Concerning I/O Functionality and Hardware Support

New function block types

The table below shows the new function block types:

Function Block	Desription	Supported Hardware	Channel Types	Further Information
FuSa System Monitoring	The **FuSa System Monitoring** function block works as a hardware-related monitor to enhance functional safety in your application. If you use this function block, a Functional Safety (FuSa) error can be triggered directly by any activated monitor, for example, if the internal operating temperature of the MicroAutoBox III exceeds a defined upper limit or the operating voltage is out of a defined range.	DS1403 (MicroAutoBox III)	-	FuSa System Monitoring (📖 ConfigurationDesk I/O Function Implementation Guide)

Enhanced function block types

SENT In and SENT Out The SENT function block types now support user-defined protocols to multiplex fast and/or serial messages. For more information, refer to Basics on Multiplexing SENT Messages (📖 ConfigurationDesk I/O Function Implementation Guide).

Digital Encoder In The **Digital Encoder In** function block type now supports the master APU functionality also for the **Digital In/Out 5** channel type on the DS6202 Digital I/O Board. The master APU functionality provides the angular position values to be used by other function blocks. Refer to Configuring the Basic Functionality (Digital Incremental Encoder In) (📖 ConfigurationDesk I/O Function Implementation Guide)

Ethernet Switch The **Ethernet Switch** function block type now lets you power down the PHY of Ethernet switch ports at run time, for example, for failure simulation. In addition, function ports provide information on the power status and the link speed. For more information, refer to New Features Concerning I/O Functionality and Hardware Support on page 76.

Function block types supporting the DS6202 Digital I/O Board For the **Digital In/Out 5** channel type (on the DS6202) now you can select the **Differential out** interface type.

The **Differential out** interface type has the following characteristics: The electrical interface of the function block works as a fully differential ouput. The function block provides the **Inverted Signal** and the **Signal** ports. These ports together represent the electrical connection points of a logical signal with two complementary potentials carrying the information in the voltage difference between these signals. Both signal ports (**Signal** and **Inverted Signal**) have identical electrical characteristics against the ground potential of the dSPACE real-time hardware. For a circuit diagram, refer to Digital In/Out 5 (📖 ConfigurationDesk I/O Function Implementation Guide).

With dSPACE Release 2020-B the following function block types support the new **Differential out** interface type setting:

- Multi Bit Out
- PWM/PFM Out
- Digital Incremental Encoder Out
- Waveform Digital Out
- Wavetable Digital Out
- Angular Wavetable Digital Out

New supported hardware

ConfigurationDesk supports the following new SCALEXIO hardware:

- DS6336-PE Ethernet Board

  The DS6336-CS Ethernet Board is a PCIe x4 board that provides two independent LAN ports for connecting external devices to a SCALEXIO AutoBox/LabBox via an Ethernet connection with up to 10 Gbit/s.

  For more information, refer to DS6336-PE Ethernet Board (🕮 SCALEXIO Hardware Installation and Configuration).
- DS6336-CS Ethernet Board

  The DS6336-CS Ethernet Board is a PCIe x4 board that provides two independent LAN ports for connecting external devices to a SCALEXIO AutoBox/LabBox via an Ethernet connection with up to 10 Gbit/s.

  For more information, refer to DS6336-CS Ethernet Board (🕮 SCALEXIO Hardware Installation and Configuration).
- High Parallel Performance Real-Time PC

  The new real-time PC is a mainboard with an Intel® Xeon® processor 6208U at 2.9 GHz with 16 cores (15 cores are available for the real-time model and hypervisor extension).
- DS2502 IOCNET Link Board

  The DS2502 IOCNET Link Board has been revised to support all cores of the new real-time PC.

Custom function blocks: Including external artifacts in RTA

For custom function blocks, you can now include external artifacts such as libraries, drivers, or any other custom file in the real-time application.

Refer to File Types and Directories for Custom Function Blocks (🕮 ConfigurationDesk Custom I/O Function Implementation Guide).

New Features of the Bus Manager in ConfigurationDesk

New supported AUTOSAR release

The Bus Manager now supports AUTOSAR system description files based on AUTOSAR Classic Platform Release R19-11.

Supported Container File Versions

Supported SIC file versions

ConfigurationDesk 6.6 supports SIC file versions as listed below:

SIC Files Created With ...	SIC Version
dSPACE Release 2020-B: ▪ Model Interface Package for Simulink 4.4 ▪ TargetLink 5.1	1.9
dSPACE Release 2020-A: ▪ Model Interface Package for Simulink 4.3	1.8
dSPACE Release 2019-B: ▪ Model Interface Package for Simulink 4.2 ▪ TargetLink 5.0	1.7
dSPACE Release 2019-A: ▪ Model Interface Package for Simulink 4.1	1.6

Note

You can use only SIC files that were generated with the Model Interface Package for Simulink for the `dsrt.tlc` system target file for building real-time applications with ConfigurationDesk. SIC files generated for the `dsrt64.tlc` system target file can only be used in ConfigurationDesk to generate BSC files for VEOS running on a Linux operating system.

Supported BSC file versions

ConfigurationDesk 6.6 supports BSC files of version 1.9.

Supported V-ECU implementation container versions

ConfigurationDesk 6.6 supports V-ECU implementation container versions as listed below:

V-ECU Implementations Created With...	V-ECU Implementation Version
dSPACE Release 2020-B: ▪ SystemDesk 5.5 ▪ TargetLink 5.1	3.0 As of version 3.0, *VECU* is the new V-ECU implementation file format.
dSPACE Release 2020-A: ▪ SystemDesk 5.4	2.10
dSPACE Release 2019-B: ▪ SystemDesk 5.4 ▪ TargetLink 5.0	2.10
dSPACE Release 2019-A: ▪ SystemDesk 5.3	2.9

Supported Functional Mock-up Unit versions

ConfigurationDesk 6.6 supports Functional Mock-up Units (FMUs) that comply with the following versions of the FMI standard:

- 2.0
- 2.0.1

Supported EIC file versions

ConfigurationDesk 6.6 supports EIC file versions as listed below:

EIC Files Created With ...	EIC Version
dSPACE Release 2020-B (ECU Interface Manager 2.8)	4.0.0
dSPACE Release 2020-A (ECU Interface Manager 2.7)	4.0.0
dSPACE Release 2019-B (ECU Interface Manager 2.6)	4.0.0
dSPACE Release 2019-A (ECU Interface Manager 2.5)	3.0.0
dSPACE Release 2018-B (ECU Interface Manager 2.4)	3.0.0
dSPACE Release 2018-A (ECU Interface Manager 2.3)	2.0.0
dSPACE Release 2017-B (ECU Interface Manager 2.2)	1.0.0
dSPACE Release 2017-A (ECU Interface Manager 2.1)	1.0.0
dSPACE Release 2016-B (ECU Interface Manager 2.0p1)	1.0.0

However, MicroAutoBox III systems only support EIC files as of version 4.0.0.

Migrating to ConfigurationDesk 6.6

Migrating to the Linux-based operating system on SCALEXIO

With dSPACE Release 2020-B, dSPACE changes the SCALEXIO firmware from a QNX-based to a Linux-based distribution. The following items built for dSPACE Release 2020-A and earlier are no longer compatible with the SCALEXIO system and must be (re-)built from source code based on dSPACE Release 2020-B:

- Real-time applications
- Binary libraries contained in model containers (i.e., SIC, BSC, FMU, and CTLGZ files)
- Binary libraries referenced by Simulink models
- Binary libraries referenced by ConfigurationDesk applications via custom code settings or custom I/O functions

Source code is expected to be reusable in most cases. Cases that might require an adaptation of C code are related to custom code that uses special features (e.g., OS-specific functions) created by your company or third-party suppliers.

BSC, FMU, and SIC files that were precompiled using the `SCALEXIO` platform identifier must be precompiled using the `SCALEXIO_LNX` identifier. As of dSPACE Release 2020-B, you must always use the `SCALEXIO_LNX` identifier to create precompiled BSC, FMU, and SIC files.

Project import restricted as of Release 2021-A

As of the dSPACE Release 2021-A, ConfigurationDesk will support the direct import only of projects last saved with one of the previous seven ConfigurationDesk versions.

Discontinuation of SCALEXIO Ethernet Solution

The SCALEXIO Ethernet Solution is discontinued as follows:

- The end-of-life date is January 31, 2021.
- Customers with a Software Maintenance Service contract who work with dSPACE Release 2018-B will be automatically migrated to the new ConfigurationDesk UDP/TCP function blocks.

For new projects (using dSPACE Release 2018-A and later), we recommend that you use the new UDP/TCP function blocks that are natively integrated in ConfigurationDesk. They provide additional and new options such as IPv6, UPD Multicast support, and enhanced TCP status information.

Note: The dedicated license is required for using the new UDP/TCP function blocks in ConfigurationDesk.

Changes to the I/O Ethernet communication

The following changes in dSPACE Release 2020-B involve the I/O Ethernet communication.

Receiving UDP messages The Received Bytes function port of the UDP Receive function block now continuously provides the number of valid data bytes that are provided by the Data Vector function port. Until Release 2020-A, the Received Bytes function port provides the number of valid data bytes only during the model step in which the function block provides new data.

Unsupported I/O Ethernet feature A SCALEXIO real-time application can suppress Internet group management protocol (IGMP) messages for I/O Ethernet communication.

As of dSPACE Release 2020-B, dSPACE introduces a Linux-based operating system for the SCALEXIO Processing Unit and DS6001 Processor Board. The Linux-based operating system does not support the suppression of IGMP messages.

For more information on the new SCALEXIO operating system, refer to New Features of the SCALEXIO Firmware 5.0 on page 139.

FPGA custom function blocks with APU functionality	With dSPACE Release 2018-B, the angle range handling of the angular processing unit (APU) changed. FPGA custom function blocks that use the APU in the 360° angle range are incompatible if they are built with the FPGA Programming Blockset 3.5 or earlier. To resolve the incompatibility, use the FPGA model/code of the incompatible FPGA custom function block and build a new FPGA custom function block with the RTI FPGA Programming Blockset 3.6 or later. The RTI FPGA Programming Blockset automatically migrates the framework of the FPGA model/code to the current version.
Changes to the tool automation interface that might lead to code malfunctions	The automation names of some function block properties were changed. You must adjust the automation names of these properties in existing scripts for them to work correctly. For more information, refer to New Features and Changes to the Automation Interface for Release 2020-B (🕮 ConfigurationDesk Automating Tool Handling).

ControlDesk

Where to go from here

Information in this section

New Features of ControlDesk 7.3

Where to go from here

Information in this section

New Instrument Features (ControlDesk 7.3)

3-D Viewer: Selecting the coordinate system

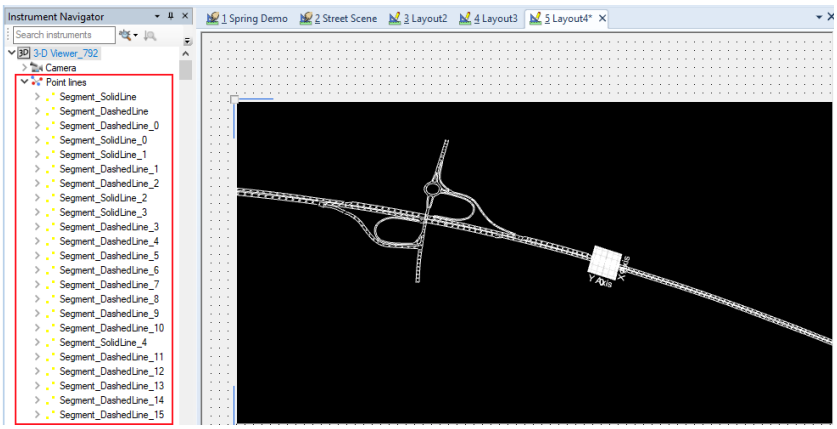
The ControlDesk 3-D Viewer now also supports the *right-handed* coordinate system. This system complies with the automotive coordinate system according to the DIN 70000 standard.

For more information, refer to Selecting the Coordinate System (🕮 ControlDesk Instrument Handling).

3-D Viewer: Importing MAT files for roads

The automation interface of the 3-D Viewer now lets you import road networks from MAT files created with the ModelDesk Road Generator. The elements of the road network are imported to point lines of the 3-D Viewer.

The following illustration shows an example:



For more information, refer to Adding Roads via MAT File Import (🕮 ControlDesk Instrument Handling).

3-D Viewer: Dynamic item labeling

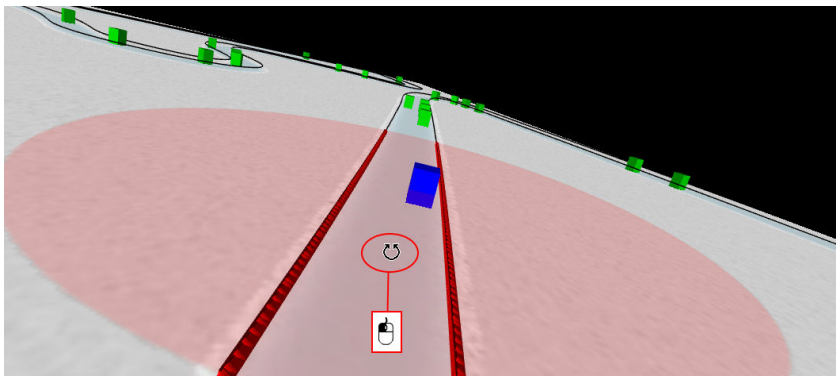
You can now connect a variable to the text label of an item in the 3-D Viewer. This allows you to display the current value of the variable, for example.

For more information, refer to Items and Templates Properties (🕮 ControlDesk Instrument Handling).

3-D Viewer: Rotating the camera via mouse if a camera target is selected

If you specified a camera target, for example, the blue car in the illustration below, you can now use the mouse to rotate the view:

Hold the left mouse button and move the pointer. You can also use the mouse wheel to zoom the view.



For more information, refer to Basics of Handling the 3-D Viewer (🕮 ControlDesk Instrument Handling).

New Calibration and Data Set Management Features (ControlDesk 7.3)

Exporting data sets to CDFX 2.1

ControlDesk now also lets you export data sets to the ASAM Calibration Data Format version 2.1 (CDFX 2.1).

For more information, refer to Data Set Management Page (🕮 ControlDesk Calibration and Data Set Management).

New Measurement and Recording Features (ControlDesk 7.3)

Configuring data logging on SCALEXIO systems (only for systems based on a DS6001 Processor Board)

ControlDesk now also lets you configure data logging on a SCALEXIO system based on a DS6001 Processor Board to perform data recording without connection to a host PC. The logged data is stored in the ASAM MDF (MF4) file format on a USB mass storage device connected to the SCALEXIO system.

Refer to General Information on Data Logging (🕮 ControlDesk Measurement and Recording).

Exporting MF4 files without path and block group information

ControlDesk and the Measurement Data API now let you configure the export of measurement data in the ASAM MDF (MF4) file format. You can specify whether to export variables with or without path and block group information.

Exporting MF4 files *without* this information can be useful if you postprocess the exported MF4 file in third-party tools.

Refer to Measurement Files Page (🕮 ControlDesk Measurement and Recording) and Class Description (MDFFormatOption) (🕮 ControlDesk Measurement Data API).

New Message Handling Features (ControlDesk 7.3)

Support of the Message Reader API

ControlDesk now supports the *Message Reader API*. You can use the API to read out log messages of the dSPACE Log, for example, in C# applications and Python scripts. The API also lets you filter log messages by dSPACE product and time, for example.

For more information, refer to Reading dSPACE Log Messages via the Message Reader API (🕮 ControlDesk Message Handling).

Related topics

Basics

Reading dSPACE Log Messages via the Message Reader API (📖 ControlDesk Message Handling)

New Bus Navigator Features (ControlDesk 7.3)

Replaying Ethernet communication

The ControlDesk Bus Navigator now lets you replay Ethernet communication and monitor the data that is being replayed. You can replay Ethernet communication with an Ethernet Bus Monitoring device in connection with a bus interface of SCALEXIO, MicroAutoBox III or VEOS, or in connection with a PC-based interface.

For more information, refer to Basics on Replaying CAN and Ethernet Bus Communication (📖 ControlDesk Bus Navigator).

Support of the DS6336-PE Ethernet Board and DS6336-CS Ethernet Board

The ControlDesk Bus Navigator now supports SCALEXIO systems with the DS6336-PE Ethernet Board and the DS6336-CS Ethernet Board.

The following features are supported:

- Monitoring and logging Ethernet communication

  For more information, refer to Basics on Monitoring, Logging, and Filtering Bus Communication (📖 ControlDesk Bus Navigator).
- Replaying Ethernet communication

  For more information, refer to Basics on Replaying CAN and Ethernet Bus Communication (📖 ControlDesk Bus Navigator).

Support of the AUTOSAR CP R19-11 system template

The ControlDesk Bus Navigator now also supports the AUTOSAR Classic Platform (CP) R19-11 system template.

For a complete list of all the supported versions, refer to Variable Descriptions Supported by ControlDesk (📖 ControlDesk Variable Management).

New Automation Features (ControlDesk 7.3)

Automated playback of recorded data via the time cursor

The automation interface of the time cursor now provides the `Playback` property to play back the visualization of recorded data.

You can specify the following playback properties:

- The start and stop time (or the duration) of the playback in the recording

- The playback acceleration factor
- Automatic repetition of the playback

To control the playback, the `Playback` property provides a `Start` and a `Stop` method.

The following listing shows how to configure and start the playback of recorded data as an example:

```
Application.TimeCursorManagement.TimeCursor.Active = 1
[...]
Application.TimeCursorManagement.TimeCursor.Playback.StartPosition = 6.0
Application.TimeCursorManagement.TimeCursor.Playback.Duration = 1
Application.TimeCursorManagement.TimeCursor.Playback.AccelerationFactor = 0.5
Application.TimeCursorManagement.TimeCursor.Playback.AutoRepeat = 1
Application.TimeCursorManagement.TimeCursor.Playback.Start()
[...]
Application.TimeCursorManagement.TimeCursor.Playback.Stop()
```

Refer to TimeCursor / IXaTimeCursor <<Interface>> (🕮 ControlDesk Automation).

Migrating to ControlDesk 7.3

Where to go from here

Information in this section

Discontinuations in ControlDesk

Discontinuations in ControlDesk (dSPACE Release 2021-B and later)

Project import supported for the last seven ControlDesk versions As of dSPACE Release 2021-A, ControlDesk will support the direct import of projects *last saved with one of the last seven ControlDesk versions*.

Up to and including ControlDesk 7.3, the direct import of projects *last saved with any ControlDesk version ≥ 4.0* was supported.

Tip

To use a ControlDesk project *last saved with a version earlier than the last seven ones*, import the project into one or more intermediate ControlDesk versions, i.e., import it *indirectly*.

Support for the Calibration Hub As of dSPACE Release 2021-A, ControlDesk no longer supports the Calibration Hub.

For new projects, you are recommended to use the DCI-CAN2 or DCI-CAN/LIN1.

End of software support for discontinued dSPACE hardware

For information on the end of software support for discontinued dSPACE hardware, refer to Discontinuations on page 18.

Migrating to ControlDesk 7.3

Introduction

To migrate from ControlDesk 7.2 to ControlDesk 7.3 and reuse existing experiments, you do not have to carry out additional migration steps.

Note

To migrate to ControlDesk 7.3 from versions earlier than 7.2, you also have to perform the migration steps of the intervening ControlDesk versions.

Migrating from earlier ControlDesk versions

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

Related topics

Basics

Basics on Migrating from Prior Versions of ControlDesk (📖 ControlDesk Introduction and Overview)

dSPACE AUTOSAR Compare

About dSPACE AUTOSAR Compare

Introduction

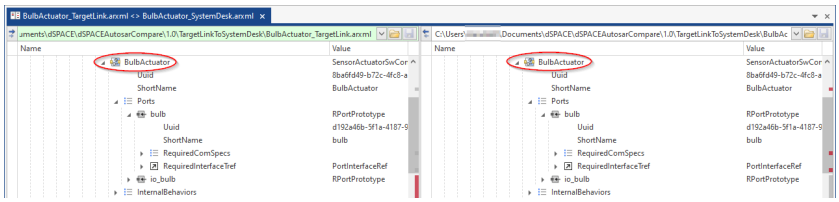
dSPACE AUTOSAR Compare lets you compare and merge AUTOSAR files.

Working principle

dSPACE AUTOSAR Compare compares and copies AUTOSAR elements, in contrast to text-based or binary comparison tools.

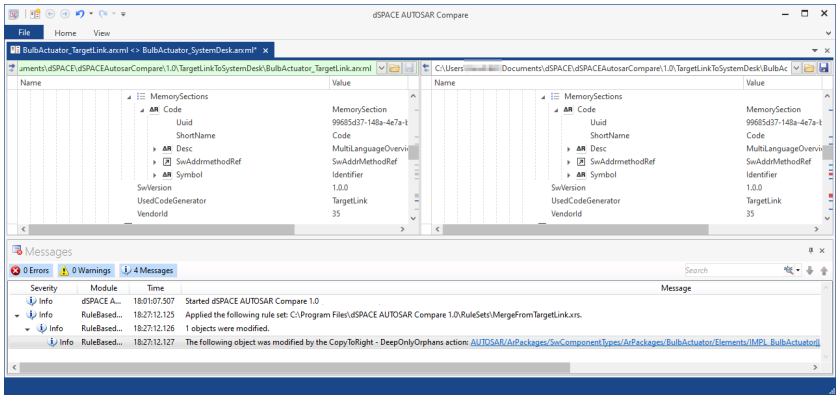
Create a session and load files dSPACE AUTOSAR Compare adds a comparison page for a session to the working area. The page lets you load a left-hand file and a right-hand file, and displays the contents of the files.

Align elements Because the files are compared element-wise, dSPACE AUTOSAR Compare automatically aligns AUTOSAR elements in the loaded files according to the split key of the elements. You can change the automatic alignment and match AUTOSAR elements that reside in different branches of the AUTOSAR tree, e.g., in different AUTOSAR packages.



Displaying differences The user interface of dSPACE AUTOSAR Compare highlights different elements, orphan elements, and elements that differ only in the value of an AUTOSAR property that you specified as a minor difference property. You can also show only the element differences of a specific category, such as, e.g., only orphans of the left-hand file.

Merging differences For merging, you can copy selected differing elements, including their subelements and AUTOSAR properties, from one file to the other. dSPACE AUTOSAR Compare also provides rule configurations that let you specify and execute a rule-based merge process for differences.



Using dSPACE AUTOSAR Compare

dSPACE AUTOSAR Compare can be used in different ways.

- It provides a user interface that lets you inspect differences and copy elements to merge the files.
- It provides a logical XML language to perform rule-based merging. This is helpful for complex tasks and large AUTOSAR models.
- It provides a command line interface for integration in a tool chain. This lets you use it as a comparison tool in SystemDesk for AUTOSAR master files and in the Container Manager for ARXML files of SWC containers. The command line interface lets you automate workflows in which AUTOSAR files have to be merged.

Comparison tool for version control systems

Version control systems, such as PTC or Git, let you manage file versions. For AUTOSAR files, you can use dSPACE AUTOSAR Compare to identify changes, such as added or removed elements. You can also find modified elements and skip elements that differ only in the values of specific properties, such as UUID, date, or time.

The intelligent merging of files lets you copy elements from branches to a master file version, including dependent elements that are related by AUTOSAR references, such as the reference of a port prototype to an interface and the data types of the interface elements.

SystemDesk-TargetLink round trip

For ECU software development according to AUTOSAR, SystemDesk acts as a *system-level design tool*. For *behavior modeling* according to AUTOSAR, dSPACE provides the TargetLink AUTOSAR Module.

dSPACE AUTOSAR Compare helps you seamlessly integrate both tools into a tool chain by intelligent merging of AUTOSAR files. It lets you copy only the elements that are modeled by TargetLink, such as the internal behavior and implementation of software components, to architecture ARXML files. The

AUTOSAR elements and properties that are not supported by TargetLink are preserved in the architecture ARXML files.

Further reading

Refer to 🕮 dSPACE AUTOSAR Compare Manual.

dSPACE FlexRay Configuration Package

New Features of dSPACE FlexRay Configuration Package 4.6

FlexRay Configuration Package

Support of AUTOSAR Classic Platform R19-11 The FlexRay Configuration Tool supports the format of AUTOSAR System Templates based on AUTOSAR Classic Platform Release R19-11 for describing FlexRay networks. However, no new features of AUTOSAR Classic Platform Release R19-11 are supported.

Refer to Communication Cluster Files Usable for Configuration (📖 FlexRay Configuration Tool Guide).

dSPACE Installation Manager

Where to go from here

Information in this section

New Features of dSPACE Installation Manager 5.6

New CodeMeter Runtime version for managing licenses

dSPACE Installation Manager now contains the CodeMeter Runtime Version 7.10a. It is automatically installed with dSPACE Installation Manager 5.6. It contains an important security update that eliminates vulnerabilities in the CodeMeter Runtime that Wibu-Systems classified as potential security risks. For detailed information, go to www.dspace.com/go/wsp.

Migrating to dSPACE Installation Manager 5.6

Using CmDongles

If you want to work with licenses on CmDongles in combination with dSPACE Installation Manager 5.6, for example, to activate, deactivate, or update licenses, the dongles must have firmware version 4.10.

To use CmDongles shipped for Releases earlier than dSPACE Release 2019-A, a firmware update is required. CmDongles shipped for dSPACE Release 2019-A and later contain the required firmware version.

dSPACE Installation Manager checks if the firmware of a connected dongle matches the required firmware version and displays if an upate is necessary.

For instructions on updating the firmware, refer to How to Update the Firmware of a CmDongle (🕮 Working with CodeMeter Licensing Technology).

dSPACE XIL API .NET

Where to go from here

Information in this section

New Features of dSPACE XIL API .NET 2020-B

New features

The dSPACE XIL API.NET 2020-B has the following new feature:

- Support of the Message Reader API. You can use the API to read out log messages of the dSPACE Log, for example, in C# applications and Python scripts. The API also lets you filter log messages by dSPACE product and by time, for example.

Migrating to dSPACE XIL API .NET 2020-B

Migrating dSPACE EESPort configuration files

As of dSPACE Release 2020-A, the DCI-CAN1 and the dSPACE CAN API 1.0 are no longer supported. As a result, dSPACE XIL API .NET no longer supports dSPACE EESPort configuration files that reference the DCI-CAN1, the dSPACE CAN API 1.0, or both.

- Instead of the DCI-CAN1, use the DCI-CAN2 or the DCI-CAN/LIN1.
- Instead of the dSPACE CAN API 1.0, use the dSPACE CAN API 2.0.

The following listing is an excerpt of a dSPACE EESPort configuration file that references both the dSPACE CAN API 1.0 and the DCI-CAN1:

```
<Driver ID="2" DriverType="DSCanApi1.0">
   <InterfaceType>dSPACE CAN Interface</InterfaceType>
   <CardIndex>15</CardIndex>
   <ControllerIndex>1</ControllerIndex>
   <SerialNumber>0</SerialNumber>
</Driver>
```

The following listing shows how to migrate the excerpt above:

```
<Driver ID="2" DriverType="DSCanApi2.0">
   <VendorName>dSPACE</VendorName>
   <InterfaceName>DCI-CAN2</InterfaceName>
   <ChannelIdentifier>1</ChannelIdentifier>
   <SerialNumber>0</SerialNumber>
</Driver>
```

The migrated listing references the dSPACE CAN API 2.0 and the DCI-CAN2.

ECU Interface Manager

Where to go from here

Information in this section

New Features of ECU Interface Manager 2.8

ARM Cortex-R4/R5: Integrating control logics

You can now integrate the following types of control logics in an ARM Cortex-R4/R5 ECU application:

- Control execution
- Register write back

The integration of Disable execution control logics in an ARM Cortex-R4/R5 ECU application has been supported since ECU Interface Manager 2.7.

For more information, refer to Basics on Controlling and Disabling Code Items via Control Logics (🕮 ECU Interface Manager Manual).

Reading XCP time stamps

Real-time applications that are connected to an ECU application with integrated XCP service can now read time stamps from the ECU via XCP.

If an ECU application with integrated XCP service supports time-stamping, i.e., if the ECU application A2L file contains the `TIMESTAMP_SUPPORTED` entry, and

when you import the related EIC file in ConfigurationDesk, the **ECU Interface Configuration** function block in ConfigurationDesk provides the following:

- One **TimestampRead** function port for each function access that is configured to read ECU data.
- One **TimestampRead** function port for each ECU-synchronous data access of the **Read from ECU** data access configuration type.

Time stamps are transmitted from the ECU to the real-time hardware only if required, i.e., only if the related function port is connected to a port of a model port block.

For more information on the function port, refer to Overview of Ports and Basic Properties (ECU Interface Configuration) (🕮 ConfigurationDesk I/O Function Implementation Guide).

Compatibility of ECU Interface Manager 2.8

Compatibility in general

dSPACE recommends using only software products from the same dSPACE Release. This ensures maximum run-time compatibility.

Compatibility between EIC files and ConfigurationDesk

The following table shows the compatibility between EIC files and ConfigurationDesk:

	EIC Files Created with ECU Interface Manager ...								
	Version 2.0p1[1), 2)]	**Version 2.1**[3), 2)]	**Version 2.2**[4), 2)]	**Version 2.3**[5), 2)]	**Version 2.4**[6), 2)]	**Version 2.5**[7), 2)]	**Version 2.6**[8)]	**Version 2.7**[9)]	**Version 2.8**[10)]
ConfigurationDesk 6.6[10)]	✓	✓	✓	✓	✓	✓	✓	✓	✓
ConfigurationDesk 6.5[9)]	✓	✓	✓	✓	✓	✓	✓	✓	–
ConfigurationDesk 6.4[8)]	✓	✓	✓	✓	✓	✓	✓	–	–
ConfigurationDesk 6.3[7)]	✓	✓	✓	✓	✓	✓	–	–	–
ConfigurationDesk 6.2[6)]	✓	✓	✓	✓	✓	–	–	–	–
ConfigurationDesk 6.1[5)]	✓	✓	✓	✓	–	–	–	–	–
ConfigurationDesk 6.0[4)]	✓	✓	✓	–	–	–	–	–	–
ConfigurationDesk 5.7[3)]	✓	✓	–	–	–	–	–	–	–
ConfigurationDesk 5.6 SP1[1)]	✓	✓	–	–	–	–	–	–	–

1) dSPACE Release 2016-B
2) To perform external ECU interfacing with MicroAutoBox III, the EIC file must be created with ECU Interface Manager 2.6 or later. EIC files created with ECU Interface Manager 2.5 or earlier are not supported.
3) dSPACE Release 2017-A

	EIC Files Created with ECU Interface Manager ...								
	Version 2.0p1[1), 2)]	Version 2.1[3), 2)]	Version 2.2[4), 2)]	Version 2.3[5), 2)]	Version 2.4[6), 2)]	Version 2.5[7), 2)]	Version 2.6[8)]	Version 2.7[9)]	Version 2.8[10)]

4) dSPACE Release 2017-B
5) dSPACE Release 2018-A
6) dSPACE Release 2018-B
7) dSPACE Release 2019-A
8) dSPACE Release 2019-B
9) dSPACE Release 2020-A
10) dSPACE Release 2020-B

Migrating to ECU Interface Manager 2.8

Automatic migration of projects

You can reuse projects in the ECU Interface Manager 2.8 if the projects were last saved with the ECU Interface Manager 2.0 p1 or later.

When you open the projects in the ECU Interface Manager 2.8, they are migrated automatically.

Note

In the ECU Interface Manager 2.8, you cannot reuse projects that were last saved with ECU Interface Manager 2.0 or earlier.

Additional migration steps in some cases

To migrate to the ECU Interface Manager 2.8 from versions earlier than the ECU Interface Manager 2.7, you might also have to perform the migration steps of the intervening ECU Interface Manager versions.

Firmware Manager

New Features of Firmware Manager 3.2

Enhanced platform support

The Firmware Manager supports the firmware update of the following dSPACE hardware:

- Support of the new SCALEXIO Processing Unit with 6208U Processor.
- Support of the KVM Hypervisor for:
  - SCALEXIO Processing Unit with E5-2640v3 Processor
  - SCALEXIO Processing Unit with 6208U Processor
  - SCALEXIO Processing Unit with E3-1275v6 Processor
- Support of SCALEXIO systems with Linux operating system. The Linux-based firmware archive is now used by default. To use the QNX firmware archive, you must select it.

Support of the Message Reader API. You can use the API to read out log messages of the dSPACE Log, for example, in C# applications and Python scripts. The API also lets you filter log messages by dSPACE product and by time, for example.

MicroAutoBox III Firmware

New Features of the MicroAutoBox III Firmware 5.0

DS1403 Processor Board

The functional safety monitoring of the DS1403 Processor Board supports the following new features:

- Monitoring the operating voltage of the MicroAutoBox III.
- Monitoring the internal operating temperature of the MicroAutoBox III.
- Monitoring an external signal to integrate an external device in the FuSa concept of the MicroAutoBox III.

  The external signal must be connected to the SAFETY In pin of the power input connector.

For more information, refer to FuSa System Monitoring (🕮 ConfigurationDesk I/O Function Implementation Guide).

Model and Sensor Interface Blockset

New Features of Model and Sensor Interface Blockset 1.0

Model and Sensor Interface Blockset

The Model and Sensor Interface Blockset (MSI Blockset) must be used to simulate with sensors in a virtual validation or in a hardware-in-the-loop environment.

In Sensor Simulation, the blockset is required for the calculation of simulation data and transmitting the data to the MotionDesk PC for visualization and to SensorSim applications that produce the sensor composition and raw data for connected sensors.

The data is transmitted in data streams of sequenced data elements using the TCP/IP protocol. In the Simulink model, the blockset must be connected to Ethernet functions to communicate with the selected supported simulation platform and to Sensor Simulation systems and applications using the TCP/IP protocol.

For each of the moving objects in the simulation, simulation data objects can be defined in kinematic chains in single object blocks using a parent-child relationship. Blocks can also be added to retrieve status and statistical information on the connected system.

You can connect an Environment Sensor Interface Unit. This splits the sensor raw data from the SensorSim application and inserts it into the relevant sensor hardware, for example, into imaging sensor interfaces of camera sensors. Additional blocks can be added to the Simulink model to receive feedback from the Environment Sensor Interface Unit on the connected sensors. You can also configure sensor failures on specific connected sensors, for example, a pixel error on a camera sensor.

If you adapted ASM Models using the Model and Sensor Interface Solution, you can migrate these models to use the Model and Sensor Interface Blockset.

Related topics

Basics

Model and Sensor Interface Blockset Manual
MotionDesk Features (MotionDesk 4.7)..... 118
Sensor Simulation Overview

Model Compare

Where to go from here

Information in this section

New Features of Model Compare 3.1

Focusing on subsystems

You can now dump and compare single subsystems instead of complete models. This reduces complexity and significantly enhances performance.

Related documentation

- How to Create Comparison Sessions (Model Compare Guide)
- How to Create XML Dump Files from MATLAB (Model Compare Guide)
- New Session (Model Compare Reference)

Focusing on specified levels of models or subsystems

New demo comparison sessions show you how to use hooks to limit the analysis of a model or subsystem to a specified number of levels. This reduces comparison efforts in applicable use cases.

Related documentation

- Basics on XML Dump Files (Model Compare Guide)
- How to Customize the XML Dump via Hooks (Model Compare Guide)
- The demo hook to focus the analysis on a specified range of model levels is available in `<DocumentsFolder>\Demos\hooks\fuelsys_5_levels_hook`.

Additional improvements

Model Compare 3.1 also provides the following improvements:

- Differences in properties and their values are now colored in the same way in reports as in the Property Inspector. This makes it easier to detect the relevant divergences.
- Models and subsystems that contain Referenced Subsystems are handled appropriately.

Migration to Model Compare 3.1

No adaptation necessary

You can migrate from Model Compare 3.0 to Model Compare 3.1 without adaptations.

ModelDesk

Where to go from here

Information in this section

New Features of ModelDesk 5.4

Custom object library

You can manage custom object libraries via tool automation.

Road creation

OpenDRIVE import When OpenDRIVE files are imported, the start positions of road segments are set according to the specified values. Usually, ModelDesk sets the start positions of road segments to the end positions of the preceding road segments.

Shapes export and import You can export and import shapes via tool automation.

Direction of travel You can get the direction of travel of a route via tool automation.

Traffic Object Manager

You can manage traffic objects via tool automation.

Plotting

ModelDesk supports the plotting of signals from ASMSignalInterface blocks. This block can be used several times in a model. In ModelDesk, the signals are

	listed in the **Signal Selector**. The **Signal Selector** has a tab page for each **ASMSignInterface** block used in the model.
Testing	The validation functions can use signals connected to different **ASMSignalInterface** blocks.

Migration to ModelDesk 5.4

Project migration	As of ModelDesk 5.4, you can migrate only projects created with ModelDesk version 4.4 (dSPACE Release 2016-B) and later.
Using an ASM of Release 2020-B in a migrated project	When you migrate a ModelDesk project and the corresponding ASM model, all signals used in the layouts for plotting are still connected. If you exchange the migrated ASM model with a ASM model created with Release 2020-B, you must reconnect the signals. ASM models created with Release 2020-B use **ASMSignalInterface** blocks which can be used in different parts of the model.
Tool automation for plotting	As of ModelDesk 4.4, ModelDesk has new plotters, and the tool automation for plotting changes. To reuse scripts for plotting, you must adapt scripts written for ModelDesk 4.3 and earlier.
Triggering of plots	As of ModelDesk 4.6, plotting is triggered by the simulation model. Previously, ModelDesk triggered plotting. The plots are usually identical but can differ in some cases. **Tip** To compare measurements, it is useful to use the XY Plotter and use the maneuver time as a signal for the x-axis.

Model Interface Package for Simulink

Where to go from here

Information in this section

New Features of the Model Interface Package for Simulink 4.4

Support of the int64/uint64 data type

The Model Interface Package for Simulink now supports the int64/uint64 data type for model port blocks and for Simulink Inport and Outport blocks on the root level of a behavior model.

Support of Simulink bus element blocks

The Model Interface Package for Simulink now supports In Bus Element blocks and Out Bus Element blocks on the root level of a behavior model. This lets you simplify the interfaces of Simulink behavior models and subsystems. Bus element blocks are supported with MATLAB R2019b and later. Refer to 🕮 Model Interface Package for Simulink - Modeling Guide.

Adding unspecific files to an SIC file

You can add unspecific files, i.e., files without a file type to a Simulink implementation container via the `dsrt_addfiles()` API command. Refer to Adding External Files to a Simulink Implementation Container (🕮 Model Interface Package for Simulink - Modeling Guide).

Migrating to the Model Interface Package for Simulink 4.4

Data migration of older versions

If you open a Simulink model that was created with an earlier version of the Model Interface Package for Simulink, messages for data migration are displayed in the MATLAB Command Window or in the Simulink Diagnostics Viewer. The data of the model is then migrated.

Migrating to the Linux-based operating system on SCALEXIO

With dSPACE Release 2020-B, dSPACE changes the SCALEXIO firmware from a QNX-based to a Linux-based distribution. Therefore, binaries that were built for dSPACE Release 2020-A and earlier are no longer compatible with the current SCALEXIO system and must be (re-)built from source code based on dSPACE Release 2020-B. For your work with the Model Interface Package for Simulink, note the following points:

- For SIC files that contain binaries, you must recompile the binaries for Linux and add them to the SIC files using the `SCALEXIO_LNX` platform identifier.
- Binaries that are contained in Simulink models must be recompiled for the current SCALEXIO platform.

Unsupported new features of MATLAB R2020b

The Model Interface Package for Simulink does not support the following new feature of MATLAB R2020b:

- As of MATLAB R2020b, code generation settings for Simulink signals and Data Store Memory blocks can be configured only via the Code Mappings Editor. Settings of the Code Mappings Editor are not copied to the models created during model separation. You must configure the code generation settings via the Code Mappings Editor also in the separated models.
- In Variant Subsystem blocks, Variant Sink blocks, and Variant Source blocks, you can specify variants for Simulink simulation and code generation. If model port blocks are used in such variants, only the active variant of a model port block is recognized during model analysis. The code generation variant is recognized during code generation and the creation of SIC files. Therefore, use model port blocks in such variants only for generating SIC files.

MotionDesk

New Features of MotionDesk 4.7

Where to go from here

Information in this section

MotionDesk Features (MotionDesk 4.7)

Installation

MotionDesk is installed in its own folder structure in the selected installation path. It is not installed in the RCP and HIL folder.

3-D scene features

The following MotionDesk features for use in the 3-D scene are available:

- Bird's-eye view

  The bird's-eye view can be enabled for the selected view. This is a two-dimensional view above the scene.

  If an observer is fixed to a movable object in the selected view, the bird's-eye view follows the object through the scene.
- Scene editing

  You can edit the scene without the dedicated scene edit mode. For example, you can freely move, resize, or rotate an object in the scene view. You can also copy single and groups of objects in the scene and paste them to another location.
- 3-D object library vehicles

  A number of new vehicles and trailers are added to the 3-D library:
  - Mercedes AMG
  - BMW Mini
  - VW Touran
  - VW Up
  - Truck for semitrailer
  - Semitrailer
  - Car trailer
  - NCAP Global Vehicle Target (GVT)
- Animated characters

  The movement of animated characters, for example, a person walking or running is synchronized with ASM maneuver time. The speed of the movements of the animated characters changes with the changes in the simulation. This ensures a realistic determination of sensor data in a sensor

simulation. If a simulation is stopped, the animated characters also stop moving.

- Road objects and scene generation

  A number of scene generation improvements have been added for road objects, for example, for height maps, shapes, and lane markings.

  MotionDesk also asks the user if a road object clean-up can be performed. The clean-up removes roads, trajectories, shapes, and terrain resources from the experiment folder that are no longer in use.

  The generation of the road network geometries adapts to the start segment parameters configured in ModelDesk to support OpenDRIVE roads.

Tool automation for the 3-D library

The path to the dSPACE and Customer Objects libraries can be accessed using methods in the `LibraryManager` class.

Related topics

Basics

New Features of Sensor Simulation 1.4 143

Migrating to MotionDesk 4.7

Discontinuation of scene edit mode

As of MotionDesk 4.7, the scene edit mode is no longer available. You can edit the scene, for example, to move, resize, or rotate an object without the dedicated scene edit mode. The undo and redo funtions of the dedicated scene edit mode are no longer supported.

The 2-D view control tab is also no longer supported. The Bird's-eye view must be used. Refer to MotionDesk Features (MotionDesk 4.7) on page 118.

Discontinuation of the ReplaceScene tool automation function

As of MotionDesk 4.7, the ReplaceScene tool automation function in the SceneManager class is discontinued.

If you try to use the function, a warning message is displayed.

Using endless ground plate and horizon

In MotionDesk 4.0 and earlier, the virtual world of a scene was built using ground plate and dome 3-D objects. If you want to use the endless ground plate and endless sky in an old scene, you first have to delete these 3-D objects.

Using advanced lighting mode

In advanced lighting mode, the static objects used for domes are not suitable for building the virtual world. Use the endless sky of the environment instead.

Migrating 3-D custom objects	If you want to use 3-D custom objects in the VRML2 format that you used in MotionDesk 2.2.1 or earlier, you have to convert the VRML2 files to COLLADA format files. You can convert the files at any time using the 3-D Library Manager.
Migrating from previous Releases	MotionDesk supports experiments from the previous four versions of MotionDesk.

Real-Time Testing

Where to go from here

Information in this section

New Features of Real-Time Testing 4.4

Platform support

The internal Python interpreter of SCALEXIO platforms is migrated to version 3.6.4.

Migrating to Real-Time Testing 4.4

Platform support

As of Real-Time Testing 4.3, the DS1005 PPC Board is no longer supported.

Incompatible BCG files

The BCG files generated with Real-Time Testing 4.0 or earlier cannot be used for Real-Time Testing 4.4. You must create the BCG file of the RTT sequence again.

Only for SCALEXIO as of Real-Time Testing 4.4 and VEOS as of Real-Time Testing 4.2: The internal Python interpreter version changed from 2.7.11 to 3.6.4 To use older scripts in the syntax of Python 2.7.11, you must migrate them to the syntax of Python 3.6.4. For more information on migrating Python scripts, refer to http://www.dspace.com/go/Python36Migration.

C# demo scripts

As of Real-Time Testing 4.2, demo scripts in C# are not longer included. The required internal COM interface is discontinued.

Related topics

Basics

Creating and Starting RTT Sequences in Python Scripts (📖 Real-Time Testing Guide)

RTI/RTI-MP and RTLib

Where to go from here

Information in this section

New Features of RTI/RTI-MP and RTLib

New features of RTI/RTI-MP

RTI and RTI-MP have the following new features:

- Support of MATLAB R2020b.
- The dialogs in RTI and RTI-MP for configuring a step size increase of the first simulation steps to avoid task overruns have been enhanced. You are now asked to select a configuration set if multiple configuration sets are available.
  - RTI: Refer to Configure Initial Step Size Dialog (📖 RTI and RTI-MP Implementation Reference).
  - RTI-MP: Refer to Configure Initial Step Size Dialog (Multiprocessor Setup Dialog) (📖 RTI and RTI-MP Implementation Reference).

Unsupported new features of MATLAB R2020b

RTI/RTI-MP does not support the following new feature of MATLAB R2020b:

- The new `codebuild` function used to configure and start the compilation of code is not supported.

Migration Aspects of RTI/RTI-MP and RTLib

Modified features in later MATLAB versions

Switching to a later MATLAB version If you install a new MATLAB version, some settings are adopted from previously installed MATLAB versions. To prevent unexpected behavior by the Simulink models when you switch to a later MATLAB version or dSPACE Release, always reset the MATLAB and Simulink preferences to their default values before you start using the models.

If you change the MATLAB version and/or the dSPACE Release, configuration sets stored in a MAT file of an earlier version might cause problems. Therefore, you are recommended to create these configuration sets again when you change the Release version.

Note

Configuration sets stored in MAT files using dSPACE Release 2013-B or earlier cannot automatically be migrated to dSPACE Release 2019-B or later with MATLAB R2019b or later. When you load these MAT files, some settings might be lost.

Code generation setting in MATLAB R2020b As of MATLAB R2020b, code generation settings for Simulink signals and Data Store Memory blocks can be configured only via the Code Mappings Editor. Settings of the Code Mappings Editor are not copied to the models created during RTI-MP model separation. You must configure the code generation settings via the Code Mappings Editor also for the separated models.

Discontinued compiler options

Up to and including dSPACE Release 2019-B, you configured the behavior of the initial simulation steps with the `-DFIRST_SIMSTEP_INCREASEMENT` and `-DNUM_INCREASED_SIMSTEPS` compiler options. These compiler options were discontinued with dSPACE Release 2020-A. If you open or load an RTI model created with Release 2019-B or earlier, the compiler options are automatically migrated to the new coefficients. For RTI-MP models, you also have to open the Multiprocessor Setup block to start the automatic migration.

For new models, you can specify the behavior of the initial simulation steps via the Configure Initial Step Size Dialog and the Configure Initial Step Size Dialog (Multiprocessor Setup Dialog).

End of software support for discontinued dSPACE hardware

For information on the end of software support for discontinued dSPACE hardware, refer to Discontinuations on page 18.

Discontinuation of DS1005 PPC Board If you want to use a model configured for RTI1005, which was discontinued with Release 2020-A, with another RTI platform, you have to specify the system target file for the new RTI platform, e.g., `rti1007.tlc`. Switching to another platform sets the code generation options in the configuration set, e.g., the RTI general build options, to the platform's default values. You must therefore adapt the settings on the pages of the Code Generation dialog manually.

If you want to use a multiprocessor model that was configured for DS1005, with another RTI-MP-supporting platform, e.g., a DS1007, you have to delete the **Multiprocessor Setup** block configured for DS1005 and replace it with a new **Multiprocessor Setup** block. Note that switching to another RTI-MP platform sets the RTI-MP-specific options to the platform's default values.

RTI Bypass Blockset

Where to go from here

Information in this section

New Features of the RTI Bypass Blockset 3.15

RTI Bypass Blockset

On-target bypassing on virtual ECUs on SCALEXIO The RTI Bypass Blockset supports on-target, service-based bypassing on virtual ECUs (V-ECUs) on the SCALEXIO system.

On-target, service-based bypassing on virtual ECUs with the RTI Bypass Blockset is based on the dSPACE Internal Bypassing Service. The V-ECU's A2L file must contain an accordingly adapted `IF_DATA dSPACE_INTERNAL_BYPASS` entry. Refer to Interface Description Data for Internal Bypassing (📖 Interface Description Data Reference).

RTI Bypass Function block: Support of 64-bit variables The RTI Bypass Blockset now supports variables of the Int64, UInt64, and Float64 (Double) data type in RTI Bypass Function blocks.

RTI Bypass Blockset MATLAB API

Support of changes made to RTI Bypass Blockset The RTI Bypass Blockset MATLAB API supports all changes made to the RTI Bypass Blockset.

Refer to the 📖 RTI Bypass Blockset MATLAB API Reference.

Migrating to RTI Bypass Blockset 3.15

Models containing a DCI-GSI1-based interface

As of dSPACE Release 2020-A, the DCI-GSI1 is no longer supported. As a consequence, the RTI Bypass Blockset no longer supports the following DCI-GSI1-based bypass interface types:

- DCI_GSI1
- dSPACE_on_JTAG_NEXUS
- dSPACE_on_JTAG_OCDS
- dSPACE_on_JTAG_SDI
- dSPACE_on_NBD_AUD
- dSPACE_on_NEXUS_READI
- cPATCH_on_JTAG_NEXUS
- cPATCH_on_JTAG_OCDS
- cPATCH_on_JTAG_SDI
- cPATCH_on_NBD_AUD
- cPATCH_on_NEXUS_READI

When you open a model with one or more of these DCI-GSI1-based bypass interfaces specified in the imported database files in the RTI Bypass Blockset 3.14 or later, a message informing you about the discontinued support is displayed, and the DCI-GSI1-based interfaces are removed from the Setup block. You can then continue working with the model as follows:

- If you did not select any of the listed interfaces in the model, you can continue working with the model as usual. No migration steps are required.
- If you selected one of the above interfaces, you must select a different bypass interface type offered in the Setup block and reconfigure all affected blocks accordingly.
- If you selected one of the above interfaces but no bypass interface type is offered for selection in the Setup block, you must import another database file with suitable interface definitions (`IF_DATA` entries) in the Setup block.

For more information, refer to Migrating Models for DCI-GSI1-Based Interfaces (📖 RTI Bypass Blockset Reference).

Working with models from earlier RTI Bypass Blockset versions 3.x and 2.x

The current Release contains RTI Bypass Blockset 3.15, which is compatible with earlier blockset versions 3.x and 2.x. However, there are some points to note:

- *Working with models from RTI Bypass Blockset 2.5 or earlier*

  Data management was changed from the prior RTI Bypass Blockset versions. If you have a Simulink model built with RTI Bypass Blockset 2.5 or earlier and you open it with RTI Bypass Blockset 3.15, the old Data Dictionary file (with the file name extension .dd) is replaced by a new Data Dictionary file (.vdb) using the information stored in the Setup block. This step is performed automatically when you open and close the Setup block dialog by clicking **OK**,

or when you open the Read, Write, Upload, or Download block dialog and click **Fill Variable Selector** on the Variables page.

If you have a model that was saved with RTI Bypass Blockset 3.15 and want to use it with RTI Bypass Blockset 2.5 or earlier, the model's Data Dictionary file required for blockset version 2.5 or earlier (file name extension .dd) is created. This step is performed when you update the A2L files in the Setup block, or when you open the Read, Write, Upload, or Download block and click **Fill Variable Selector** on the Variables page. The Data Dictionary file created under RTI Bypass Blockset 3.15 (.vdb) remains on the disk.

To enable the RTI Bypass Blockset to create the Data Dictionary again, the database files specified in the Setup block must be unchanged and accessible at the specified location.

- *Working with models from RTI Bypass Blockset 2.6 up to and including RTI Bypass Blockset 3.14*

  If a Simulink model was built with RTI Bypass Blockset 2.6 up to RTI Bypass Blockset 3.14, and you open it with RTI Bypass Blockset 3.15, the old Data Dictionary file is replaced by a new Data Dictionary file. However, the new Data Dictionary file cannot be used in earlier RTI Bypass Blockset versions. If you want to reuse the model with RTI Bypass Blockset 2.6 up to and including RTI Bypass Blockset 3.14, you have to create a suitable database in the earlier RTI Bypass Blockset version by reimporting the database files (A2L files) specified in the Setup block.

RTI CAN MultiMessage Blockset

Where to go from here

Information in this section

New Features of the RTI CAN MultiMessage Blockset 5.5

Support of AUTOSAR Classic Platform R19-11

The RTI CAN MultiMessage Blockset supports the AUTOSAR System Template based on AUTOSAR Classic Platform Release R19-11 for describing CAN networks.

Refer to General Settings Page (RTICANMM MainBlock) (🕮 RTI CAN MultiMessage Blockset Reference).

Support of secure onboard communication (SecOC) for dynamic container IPDUs with secured PDU header

The RTI CAN MultiMessage Blockset now supports SecOC for container IPDUs with a dynamic container layout by using secured PDU headers. Securing dynamic container IPDUs is supported both for secured IPDUs configured as cryptographic IPDUs and for secured IPDUs not configured as cryptographic IPDUs.

Refer to Aspects of Miscellaneous Supported AUTOSAR Features (🕮 RTI CAN MultiMessage Blockset Reference).

Migrating to RTI CAN MultiMessage Blockset 5.5

Working with models from earlier RTI CAN MultiMessage Blockset versions

To reuse a model created with an earlier RTI CAN MultiMessage Blockset version, you must update the S-functions for all the RTICANMM blocks and save the model before modifying the CAN configuration.

To create new S-functions for all the RTICANMM blocks in a model in one step, you can perform one of the following actions after opening the model:

- In the MATLAB Command Window, enter `rtimmsu_update('System', bdroot)`.

  For more information on the command and its options, enter `help rtimmsu_update` in the MATLAB Command Window.
- Select the **Create S-Function for all CAN Blocks** command from the **Options** menu of the RTICANMM GeneralSetup block.

For more information, refer to Limitations with RTICANMM (🕮 RTI CAN MultiMessage Blockset Reference).

Additional migration steps in some cases

To migrate to the RTI CAN MultiMessage Blockset 5.5 from versions earlier than the RTI CAN MultiMessage Blockset 5.4 and reuse existing models, you might have to carry out additional migration steps. For more migration information, refer to Migration (🕮 RTI CAN MultiMessage Blockset Reference).

RTI FPGA Programming Blockset

Where to go from here

Information in this section

New Features of the RTI FPGA Programming Blockset 3.10

Extended Xilinx® support

The RTI FPGA Programming Blockset now supports the following products and versions of the Xilinx design tools:

Xilinx Design Tools Version	MATLAB Version[1)]	Operating System
Vivado 2020.1[2)]	▪ MATLAB R2019a ▪ MATLAB R2019b ▪ MATLAB R2020a	Windows operating system that is supported by the RCP and HIL software of the current Release. For a list of supported operating systems, refer to Operating System on page 236. The listed Windows Server 2016 edition is not officially supported by Xilinx, but tested by dSPACE.

1) The Processor Interface sublibrary of the RTI FPGA Programming Blockset also supports MATLAB R2020b.
2) The Vivado HL WebPACK Editions of the Xilinx design tools also support the DS2655 (7K160) and DS6601 FPGA base boards. A separate license for the Xilinx System Generator for DSP is required for modeling FPGA applications with the RTI FPGA Programming Blockset.

Framework enhancements

Support of subchannels The SCALEXIO and the MicroAutoBox III frameworks now support subchannels to implement the communication between a task of the processor application and the FPGA application.

With subchannels, you can use a single buffer to exchange data via several Simulink buses. This gives you the flexibility of registers and the performance of buffers with the support of Simulink buses.

For more information, refer to Using Subchannels for Data Exchange (📖 RTI FPGA Programming Blockset Guide).

Related topics

Basics

Migrating to the RTI FPGA Programming Blockset 3.10 134

Migrating to the RTI FPGA Programming Blockset 3.10

Introduction

There are various ways to migrate an existing model, depending on the blockset version used.

Migrating from RTI FPGA Programming Blockset 1.1 and higher to 3.10

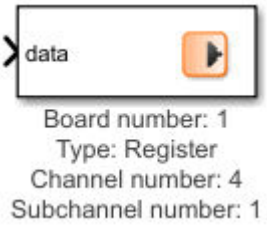
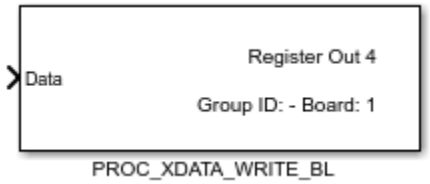
If you implemented an FPGA application with the RTI FPGA Programming Blockset Version 1.1 and later and want to use it with the RTI FPGA Programming Blockset 3.10, the framework automatically updates itself to the current framework version.

The update affects all the subsystems in the model/subsystem. The parameters of the blocks stay the same after updating to the current framework version.

The migration of processor interface blocks can also be done with a script to migrate RTI processor models without the FPGA model, for example. For more information, refer to MigrateToModelPortBlocks (📖 RTI FPGA Programming Blockset Script Interface Reference).

Display of migrated processor interfaces With the RTI FPGA Programming Blockset 3.4 ... 3.8, you modeled the processor interface of a SCALEXIO/MicroAutoBox III with processor interface blocks of the Processor Interface sublibrary.

If you migrate a model with processor interface blocks of the Processor Interface sublibrary, the update process migrates these blocks to the model port blocks of the Model Interface Package for Simulink. The following illustrations provide an example.

Display After Migration	Display Before Migration
	

ConfigurationDesk custom functions incompatible with current dSPACE Release

FPGA custom function block types that are not built with the RTI FPGA Programming Blockset 3.10 might be incompatible with the current ConfigurationDesk version.

FPGA Programming Blockset 3.5 or earlier With dSPACE Release 2018-B, the angle range handling of the angular processing unit (APU) changed. FPGA custom function blocks that use the APU in the 360° angle range are incompatible if they are built with the FPGA Programming Blockset 3.5 or earlier.

To resolve the incompatibility, use the FPGA model/code of the incompatible FPGA custom function block and build a new FPGA custom function block with the RTI FPGA Programming Blockset 3.6 or later. The RTI FPGA Programming Blockset automatically migrates the framework of the FPGA model/code to the current version.

RTI FPGA Programming Blockset 2.5 An FPGA custom function block generated with RTI FPGA Programming Blockset 2.5 from dSPACE Release 2013-A and the real-time applications containing the FPGA custom function block are incompatible with the current dSPACE Release. To produce a usable custom function, you have to rebuild the FPGA model with the current RTI FPGA Blockset.

Using different dSPACE hardware

Using an FPGA model on different dSPACE hardware requires some model modifications. Refer to Migrating to Different FPGA Hardware (📖 RTI FPGA Programming Blockset Guide).

RTI LIN MultiMessage Blockset

Where to go from here

Information in this section

New Features of the RTI LIN MultiMessage Blockset 3.5

Support of AUTOSAR Classic Platform R19-11

The RTI LIN MultiMessage Blockset supports the AUTOSAR System Template based on AUTOSAR Classic Platform Release R19-11 for describing LIN networks.

Refer to General Settings Page (RTILINMM MainSetup) (📖 RTI LIN MultiMessage Blockset Reference).

Migrating to RTI LIN MultiMessage Blockset 3.5

Working with models from earlier RTI LIN MultiMessage Blockset versions

To reuse a model created with an earlier RTI LIN MultiMessage Blockset version, you must update the S-functions for all the RTILINMM blocks and save the model before modifying the LIN configuration.

To create new S-functions for all the RTILINMM blocks in a model in one step, you can perform one of the following actions after opening the model:

- In the MATLAB Command Window, enter `rtimmsu_update('System', bdroot)`.

  For more information on the command and its options, enter `help rtimmsu_update` in the MATLAB Command Window.

- Select the **Create S-Function for all LIN Blocks** command from the **Options** menu of the RTILINMM GeneralSetup block.

For more information, refer to Limitations of RTI LIN MultiMessage Blockset (🕮 RTI LIN MultiMessage Blockset Reference).

SCALEXIO Firmware

Where to go from here

Information in this section

New Features of the SCALEXIO Firmware 5.0

New operating system

With dSPACE Release 2020-B, dSPACE introduces a new operating system for SCALEXIO processing hardware. This operating system is based on real-time Linux and customized by dSPACE. The migration to Linux provides the following advantages:

- Faster deployment of new dSPACE features
- Improved real-time execution
- More straightforward third-party software support

The migration to Linux has an impact on binaries built for dSPACE Release 2020-A and earlier. Refer to Migrating to SCALEXIO Firmware 5.0 on page 140.

Hypervisor Extension

The new SCALEXIO Hypervisor Extension adds a new hypervisor to the new Linux-based operating system (refer to New operating system on page 139) . It can be run on the following Processing Units:

- SCALEXIO Processing Unit with E3-1275v6 Processor
- SCALEXIO Processing Unit with E5-2640v3 Processor
- SCALEXIO Processing Unit with 6208U Processor

The new hypervisor is based on KVM and lets you create one or more virtual machines that run under any Linux distribution (Ubuntu 18.04 is tested and recommended).

The new hypervisor provides the following advantages:

- Creation of virtual machines via configuration files.
- Parallel operation of virtual machines.
- Free distribution of memory and processing power between operating systems.
- Dynamic assignment of resources to virtual machines.
- Straightforward managing of Linux installations (as simple as with PCs).

New supported hardware

The SCALEXIO firmware supports the following new hardware:

- High Parallel Performance SCALEXIO Process Unit (SCALEXIO Processing Unit HPP 2.0)

  The new real-time PC is a mainboard with an Intel® Xeon® processor 6208U at 2.9 GHz with 16 cores (15 cores are available for the real-time model and hypervisor extension).
- DS2502 IOCNET Link Board

  The DS2502 IOCNET Link Board has been revised to support all cores of the SCALEXIO Processing Unit HPP 2.0.
- DS6336-CS Ethernet Board

  The DS6336-CS Ethernet Board is a PCIe x4 board that provides two independent LAN ports for connecting external devices to a SCALEXIO AutoBox/LabBox via an Ethernet connection with up to 10 Gbit/s.
- DS6336-PE Ethernet Board

  The DS6336-PE Ethernet Board is a PCIe v3.0 x4 board that provides two independent LAN ports for connecting external devices to a SCALEXIO Processing Unit via an Ethernet connection with up to 10 Gbit/s.

DS6001 Processor Board

The DS6001 supports data logging, i.e., you can record without connection to a host PC. The logged data is stored on a USB mass storage device connected to the DS6001 in the ASAM MDF (MF4) file format.

Related topics

Basics

DS6342 CAN Board (📖 SCALEXIO Hardware Installation and Configuration)
DS6651 Multi-I/O Module (📖 SCALEXIO Hardware Installation and Configuration)

Migrating to SCALEXIO Firmware 5.0

Migrating to the Linux-based operating system

With dSPACE Release 2020-B, dSPACE changes the SCALEXIO firmware from a QNX-based to a Linux-based distribution. The following items built for dSPACE

Release 2020-A and earlier are no longer compatible with the SCALEXIO system and must be (re-)built from source code based on dSPACE Release 2020-B:

- Real-time applications
- Binary libraries contained in model containers (i.e., SIC, BSC, FMU, and CTLGZ files)
- Binary libraries referenced by Simulink models
- Binary libraries referenced by ConfigurationDesk applications via custom code settings or custom I/O functions

Source code is expected to be reusable in most cases. Cases that might require an adaptation of C code are related to custom code that uses special features (e.g., OS-specific functions) created by your company or third-party suppliers.

Hypervisor extension

If the new SCALEXIO Hypervisor Extension is installed, SCALEXIO real-time applications as well as Linux real-time and non-real-time applications created for dSPACE Release 2020-A and older can no longer be used. You must rebuild the SCALEXIO real-time applications (refer to Migrating to the Linux-based operating system on page 140) and adopt Linux real-time and non-real-time applications to the new hypervisor, especially regarding interrupts, shared memories, and the assignment of hardware resources.

Sensor Simulation

New Features of Sensor Simulation 1.4

SensorSim application

Performance optimizations The following performance optimizations are added to the SensorSim application for sensor simulation.

Road and scenery performance optimization:

- CPU performance
- Fish-eye sensors
- Repeated and continuous shape objects
- Road lane markings

Animated characters The movement of animated characters, for example, a person walking or running is synchronized with ASM maneuver time. The speed of the movements of the animated characters changes with the changes in the simulation. The movements pause when the simulation stops. This ensures a realistic determination of sensor data in a sensor simulation. If a simulation is stopped, the animated characters also stop moving.

Related topics

Basics

MotionDesk Features (MotionDesk 4.7)..... 118
MotionDesk Sensor Simulation Control
Sensor Simulation Manual
Sensor Simulation Overview

SYNECT

Where to go from here

Information in this section

New Features of SYNECT 2.10

Where to go from here

Information in this section

New General Features of SYNECT

Improved OData service

The OData service now supports changes of the SYNECT data model without the requirement to restart the SYNECT server.

Limited support for system model containers

SYNECT model management supports the integration of system models and lets you build system models for virtual validation in VEOS. A system model contains different components that can be imported to SYNECT from container files.

As of dSPACE Release 2020-B, the container format for V-ECU implementation containers changed. The new VECU format improves the data exchange between ECU software suppliers and OEMs and provides multiplatform support.

SYNECT does not support VECU containers. You cannot integrate system models with dSPACE products of dSPACE Release 2020-B. This affects the import of the dSPACE-specific VECU, SIC, and BSC containers and the exchange of system models with VEOS using SMC containers. However, the FMU 2.0 support and the support for the dSPACE-specific model containers of earlier dSPACE Releases has not changed.

New Features of Test Management

Improved support for synchronization with AutomationDesk

The AutomationDesk add-on lets you synchronize SYNECT test management projects with custom libraries to support the teamwork of test managers and test developers.

You can now synchronize SYNECT test management projects with custom libraries that you opened in AutomationDesk. This lets you synchronize test cases with sequences in multiple open custom libraries and reduces potential errors that stem from loading custom libraries from files one after another.

In AutomationDesk, you can annotate sequences, libraries, and data objects to be ignored for synchronization. Refer to New Features of AutomationDesk 6.4 on page 31.

Support for parameterizing evaluation functions

You can now add parameters to evaluation functions and assign parameter values to prepare evaluations for variant-dependent execution.

SYNECT provides similar support as for parameterizing test cases, i.e., you can:

- Assign variant dependencies to evaluation functions.
- Select variant configurations to configure evaluation variants.
- Map data pool item values to parameters to prepare variant-dependent evaluations.
- Validate the variant dependency of evaluation functions when SYNECT prepares evaluations.

Further reading

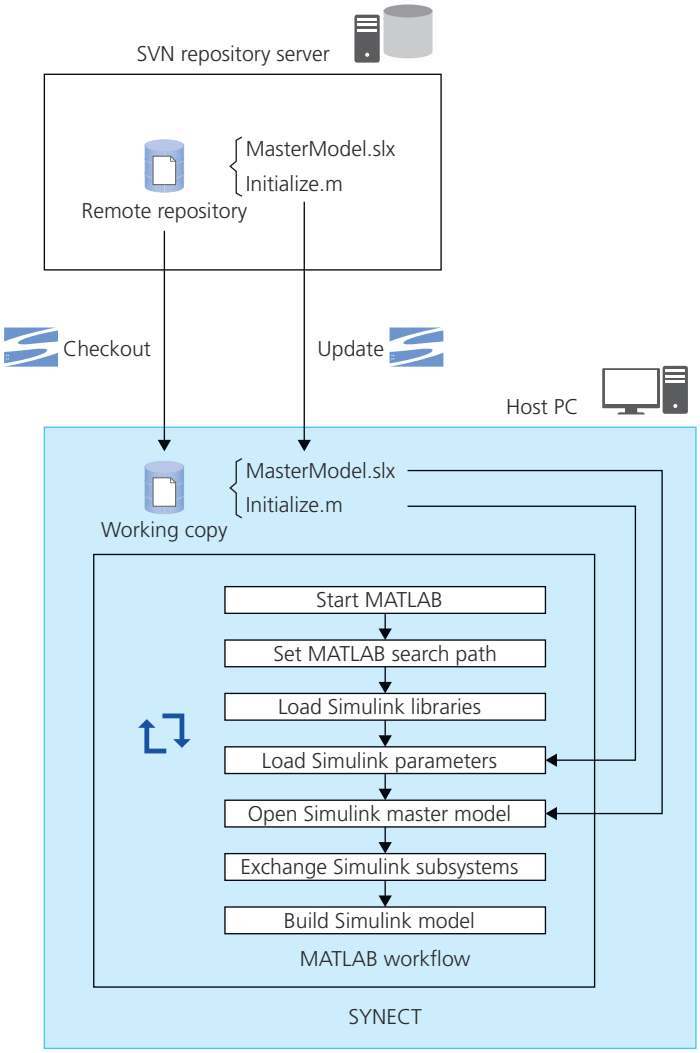
Refer to Managing Tests (📖 SYNECT Guide).

New Features of Workflow Management

Workflow Management SVN CM add-on

Workflow management now provides the Workflow Management SVN CM add-on, which lets you work with SVN repositories in SYNECT via the Workflow Management Python API.

Use case A typical use case is the continuous integration of a variant-dependent Simulink model in a SYNECT workflow. If you control the version of the model files via a remote SVN repository, you can use the SVN CM add-on to integrate SVN tasks into your workflow, such as checking out a remote repository and updating files from it. Refer to the following illustration.



SVN workflow steps and parameters The SVN CM add-on provides Python workflow steps for basic SVN operations, such as checkout, update, add, commit, and revert. You can use workflow management parameters to specify variant-dependent information, such as file locations or the repository folders, that is required to perform SVN operations.

Run condition of workflow step instances

Workflow step instances now have the Run Condition attribute, which lets you configure the execution of workflow step instances in relation to the overall verdict of the workflow, i.e., the summarized verdict of previously executed workflow steps and their fail actions.

You can select one of the following options:

- Only if previous steps were successful is the default run condition.
  Steps with this run condition are executed if the overall verdict of the workflow is passed, i.e., if all previous workflow steps were successful or failed workflow steps have the Ignore fail action.

The run condition lets you prevent time consuming workflow steps or workflow steps that block a hardware resource from being executed if a previous workflow step already failed.

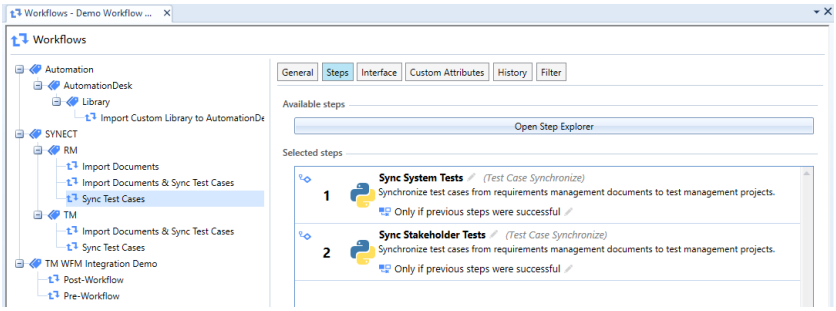
- Steps with the **Even if previous steps failed and workflow was not aborted** run condition are executed regardless of the overall verdict of the workflow. However, if a previously failed workflow step has the **Abort** fail action the steps are not executed.

  You can use the run condition for general workflow steps without specific tasks or requirements.
- Steps with the **Always** run condition are executed regardless of the overall verdict of the workflow and the fail action of previous steps.

  You can use the run condition for workflow steps, which always have to be executed, such as workflow steps that clean-up temporary execution directories of a workflow.
- Steps with the **Only if previous steps failed** run condition are executed if the overall status of the workflow is failed, i.e., if not all previous workflow steps were successful and at least one failed workflow step has the **Abort** or **Continue** fail action.

  The run condition lets you use workflow steps to send job report E-mails of non-successful workflows.

You can edit the attribute on the Steps (Workflows) page. Refer to the following illustration.



Further reading

Refer to Managing Workflows (🕮 SYNECT Guide).

Migrating to SYNECT 2.10

Where to go from here

Information in this section

Migrating Databases

Introduction

To use the data from previous SYNECT versions with SYNECT 2.10, you have to migrate the SYNECT database.

To migrate databases from SYNECT Versions 2.0 - 2.9 to SYNECT 2.10, use the **Database Migrator** of SYNECT 2.10.

Note

Contact dSPACE Support if you want to migrate SYNECT versions earlier than SYNECT 2.0. Refer to www.dspace.com/go/supportrequest.

For basic information and instructions on migrating databases, refer to Migrating Databases from Previous SYNECT Versions (🕮 The SYNECT Server Guide).

Migrating from SYNECT 2.9

Migrating workflows

Workflow steps now have a run condition that can disable their execution if previous workflow steps result in an error.

You have to migrate workflows whose steps use the **Continue** fail action. To reproduce the behavior of SYNECT 2.9, you have to select a run condition other than **Only if previous steps were successful**, which is the default run condition.

Migrating the ECXML files of the AutomationDesk plug-in

With SYNECT 2.10, the AutomationDesk plug-in supports the variant configuration of evaluations. As a consequence, you must change the following property keys of the execute adapter:

Old Name	New Name
TestItemVariantConfigDictName	ItemVariantConfigDictName
TestEnvironmentVariantConfigDictName	EnvironmentVariantConfigDictName

Data Model Changes From SYNECT 2.9 to SYNECT 2.10

Introduction

Some parts of the SYNECT data model have been changed from SYNECT 2.9 to SYNECT 2.10.

Deleted

Item types, attributes, or reference types were not deleted.

New

Item types were not added.

Attributes The following attributes were added:

Domain	Item Type	Attributes
Workflow Management	Workflow Step Instance (🕮 SYNECT Data Model Reference)	Run Condition

References The following reference types were added:

Name	Source	Target
Variant Dependencies/Referencing Evaluation Functions	Evaluation Function (🕮 SYNECT Data Model Reference)	Variant Dependency (🕮 SYNECT Data Model Reference)
Evaluation Environment Variant Configuration/Referencing Evaluations (Evaluation Environment)	Evaluation (🕮 SYNECT Data Model Reference)	Variant Configuration (🕮 SYNECT Data Model Reference)
Evaluation Item Variant Configuration/Referencing Evaluations (Evaluation Item)	Evaluation (🕮 SYNECT Data Model Reference)	Variant Configuration (🕮 SYNECT Data Model Reference)
Selected Evaluation Item Variants/Referencing Evaluations (Evaluation Item)	Evaluation (🕮 SYNECT Data Model Reference)	Variant (🕮 SYNECT Data Model Reference)
Selected Evaluation Environment Variants/Referencing Evaluations (Evaluation Environment)	Evaluation (🕮 SYNECT Data Model Reference)	Variant (🕮 SYNECT Data Model Reference)
Dependent Variants/Referencing Evaluation Function Results	Evaluation Function Result (🕮 SYNECT Data Model Reference)	Variant (🕮 SYNECT Data Model Reference)

Renamed Item types and attributes were not renamed.

References The following references were renamed:

New Name	Source	Target
Item Variant Dependencies/Referencing Execute Queues (Item)[1)]	Execute Queue (📖 SYNECT Data Model Reference)	Variant Dependency (📖 SYNECT Data Model Reference)
Environment Variant Dependencies/Referencing Execute Queues (Environment)[2)]	Execute Queue (📖 SYNECT Data Model Reference)	Variant Dependency (📖 SYNECT Data Model Reference)

1) Old name: Test Item Variant Dependencies/Referencing Execute Queues (Test Item)
2) Old name: Test Environment Variant Dependencies/Referencing Execute Queues (Test Environment)

SystemDesk

Where to go from here

Information in this section

New Features of SystemDesk 5.5

Where to go from here

Information in this section

New General Features

Classic Platform support by SystemDesk 5.5

AUTOSAR release for modeling SystemDesk lets you model Classic Platform software and system architectures with a data model according to the AUTOSAR 19-11 Release. However, SystemDesk also lets you exchange data of other AUTOSAR releases.

Data exchange support SystemDesk supports AUTOSAR 19-11, 4.4.0, 4.3.1, 4.3.0, 4.2.2, 4.2.1, 4.1.3, 4.1.2, 4.1.1, 4.0.3, and 4.0.2 for data exchange.

Adaptive Platform support

SystemDesk now supports AUTOSAR 19-11 for developing Adaptive Platform software. For exchanging data, AUTOSAR 19-11 and 19-03 are supported.

Support for the new V-ECU container file format 3.0

This SystemDesk version supports the new V-ECU container file format 3.0 for V-ECU implementation containers. The new format uses the file name extension .vecu instead of the previously used .ctlgz.

The new V-ECU container file format provides the following improvements:

- Simplified data exchange between ECU software suppliers and OEMs: A V-ECU container consists of *packages*, in which the code files are organized. Suppliers can therefore update containers by simply exchanging packages.
- Multi-platform support: You can integrate simulation code for various simulation platforms in one V-ECU container file, including descriptions of the relevant platform dependencies.

Simplified creation and handling of V-ECUs

To facilitate the creation of V-ECUs, the SystemDesk V-ECU Manager now lets you create and handle single V-ECUs without having to create a simulation system.

The revision of the **V-ECU Manager** also reflects the new V-ECU container file format for exchanging V-ECU implementations:

- The top-level organizing element is a *package* ().

  To increase efficiency, the revised **V-ECU Manager** provides several new commands for creating packages, for example, **New - Package from Folder**.
- V-ECU port listings of classic V-ECUs are no longer displayed. You can access V-ECU ports via **V-ECU - Edit Ports** on the V-ECU ribbon.

SystemDesk can now generate build scripts for VEOS automatically. If you use VEOS 5.1, you can utilize this feature to initiate test builds of V-ECUs from within SystemDesk. This lets you detect errors before loading a V-ECU to a real simulation target. You can also adjust the build scripts, for example, to use them in automated continuous integration scenarios.

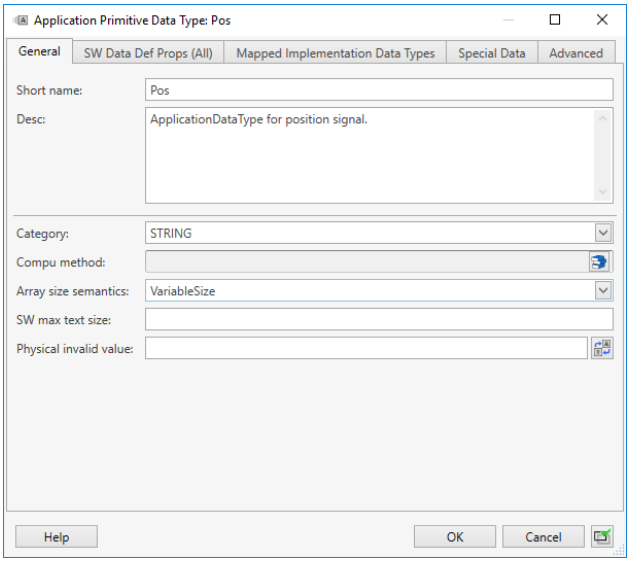
Migration of projects and scripts The simplified creation and handling of V-ECUs has an impact on the migration of SystemDesk projects and automation scripts. Refer to Migrating to SystemDesk 5.5 on page 159.

Generating Code for the Run-Time Environment (RTE)

Improved support for application data types

You can now generate RTE for V-ECUs that use the following application data types:

- Application primitive data types with `STRING`, `CURVE`, and `MAP` category.
- Application array data types with a variable size profile set to `VSA_LINEAR`. Refer to the following illustration.



Generating RTE code for background events

The RTE generation now supports the code generation for `BackgroundEvents`.

Configuring ECUs

Additional support for generating V-ECUs

SystemDesk provides additional basic software modules for generating a V-ECU from ECU software.

Output compare unit driver (Ocu) You can now add an output compare unit driver module of the microcontroller abstraction layer to a V-ECU to trigger actions on the basis of the comparison of a threshold with a ring counter value.

On a real ECU, the OCU compares a threshold value that can be set via an API of the driver with the value of a hardware clock. The unit triggers an action if the compared values match. The OCU driver can trigger software actions, such as notifications via callback functions, and the OCU hardware can trigger events, such as direct memory events or analog-to-digital converter events. The value of an output pin can also be set as the comparison result action at a specific time.

On a virtual ECU, the OCU driver from dSPACE provides an API that lets you set the threshold value for the comparison. The driver references the global simulation time instead of a hardware clock. You can specify the clock frequency as required. The notification with callback functions can also be configured.

The hardware events of the OCU are supported in simulations as follows. A DAP signal for the output pin can be switched when the compared values match. dSPACE provides the sample and hold unit to store signal values at a specifc time for converting them to ADC signals later on. dSPACE does not support simulating direct memory access events but provides the possibility to call callback functions instead.

Sample and hold unit (Shu) The sample and hold unit is a dSPACE-specific module that lets you sample and hold physical inputs of specific points in time.

The module lets you read physical inputs at a specific point in time and hold them for later use. This allows, e.g., to sample physical inputs triggered by an OCU event and hold the input to be read by the ADC module.

Signal-to-frame converter (Sfc) You can now add the signal-to-frame converter module to an ECU, which lets you connect V-ECUs with modeled bus communication to environment models or other V-ECUs that use signal-based communication, such as SIC and FMU files, or buses that are not supported. The bus signals can be transferred. However, bus timings cannot be simulated using the signal-to-frame converter.

The signal-to-frame converter module can transfer bus protocol data units (PDUs) via DAP signals of the dSPACE-specific data access points module. You do not have to adapt the basic software of the ECU to simulate.

Further reading Refer to Configuring ECUs (🕮 SystemDesk Manual).

Bypassing software component runnables

You can now perform virtual bypassing of software component runnables.

For this, you have to add RTE intervention points and related services for the runnables, you want to bypass. To facilitate the RTE intervention specification and to generate a bypass service for the connection to rapid prototyping hardware, you can add specific basic software modules to the ECU configuration that were designed for this use case. You can use the dSPACE RTI Bypass Blockset to implement rapid prototyping bypass functions.

The following dSPACE-specific modules enable virtual bypassing.

Rapid prototyping access (RptAccess) The module provides an editor that lets you select the runnables to bypass from the application software components that are mapped to the ECU configuration. You can automatically configure the module and generate the code of the bypass service.

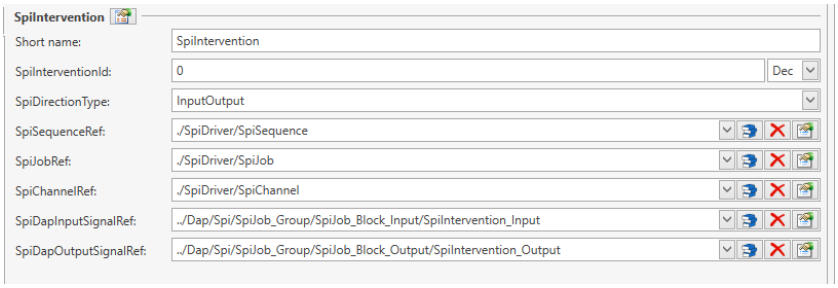
Rapid prototyping service interface (RptSi) Platform-dependent module to configure the rapid prototyping access to V-ECUs.

Futher reading For an overview of the rapid prototyping process of V-ECUs, refer to Rapid Prototyping Access to V-ECU-Internal Variables and Functions (🕮 Virtual Validation Overview).

SPI interventions

You can now use the SPI intervention mechanism that is a dSPACE-specific extension of the SPI handler/driver for direct communication with the simulation system via DAP signals. One use case is the connection of SPI channels with plant models to simulate SPI devices.

The Spi module configuration lets you specify that data, which is prepared for SPI bus communication is accessible for reading and/or writing as DAP signals. This lets you validate ECU software with access to SPI devices.



RTE interventions API

The API of SystemDesk now provides interfaces to automate the configuration of RTE interventions in a similar workflow than via the user interface. You can use interfaces that get possible intervention points and create services for specific elements. It is possible to access RTE intervention services and enable signals and configure their properties. This improves the working with RTE interventions using the API of SystemDesk.

Further reading Refer to Automating the Configuration of RTE Interventions (🕮 SystemDesk Manual).

Related topics

Basics

Rapid Prototyping Access to V-ECU-Internal Variables and Functions (🕮 Virtual Validation Overview)

Migrating to SystemDesk 5.5

Where to go from here

Information in this section

Migrating to SystemDesk 5.5

Automatic migration of projects

SystemDesk 5.5 automatically migrates SystemDesk 5.3, and 5.4 SDP project files when it starts.

Note

You are recommended to install the most recent patch for SystemDesk 5.3 or 5.4. Then, save the SDP project files you want to migrate before opening them in SystemDesk 5.5.

Migrating projects with simulation systems In SystemDesk 5.5, you can directly model a V-ECU without having to create a simulation system. Therefore, information related to simulation systems and all the build options of classic V-ECUs are dropped when you migrate a SystemDesk project from a former dSPACE Release to SystemDesk 5.5. Refer to Migrating SystemDesk Project Files on page 160.

Migrating scripts for automating SystemDesk

The SystemDesk API was changed as of SystemDesk 5.5. Some interfaces were added compared to SystemDesk 5.4 and certain interfaces were changed.

For more information, refer to *API Changes from SystemDesk 5.4 to SystemDesk 5.5* in the *SystemDesk API Reference* that is only available in dSPACE Help.

Migrating scripts for automating V-ECU generation and build In addition to the API changes due to the support of AUTOSAR 19-11 the API reflects also the changed handling of V-ECUs. Refer to Migrating Scripts on page 162.

Migrating SystemDesk Project Files

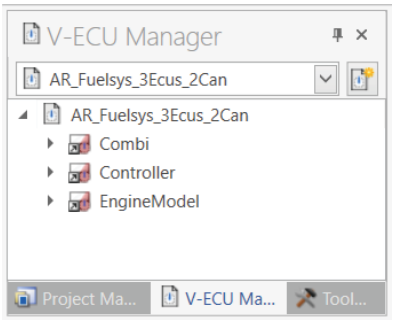
Introduction

In SystemDesk 5.5, you can directly model a V-ECU without having to create a simulation system. The new V-ECU container file format 3.0 is introduced.

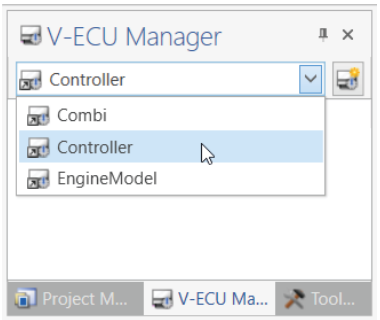
Changes in the V-ECU Manager

SystemDesk is the dSPACE software for generating V-ECU implementations. You can integrate V-ECU implementations in simulation systems with dSPACE VEOS. To simplify the workflow, the **V-ECU Manager** has been redesigned. It now focuses on the creation of V-ECUs. Therefore, the simulation system element has been removed. Refer to the following illustration.

SystemDesk 5.4



SystemDesk 5.5



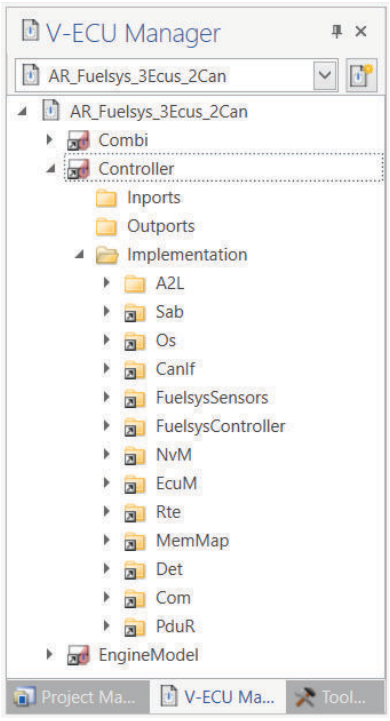
Furthermore, the presentation of V-ECUs in the **V-ECU Manager** is now adapted to the new V-ECU container file format:

- The **V-ECU Manager** now displays the implementation files directly below the V-ECU node. They are organized in packages that are represented by the icon.

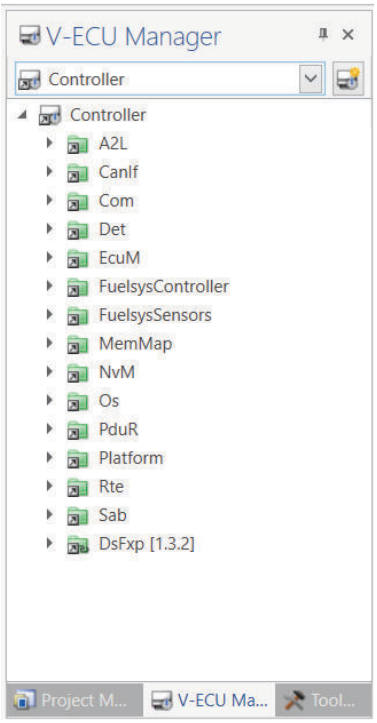
  A package can contain an arbitrary hierarchy of folders and subfolders.

The following illustration compares the V-ECU Manager in SystemDesk 5.4 and SystemDesk 5.5.

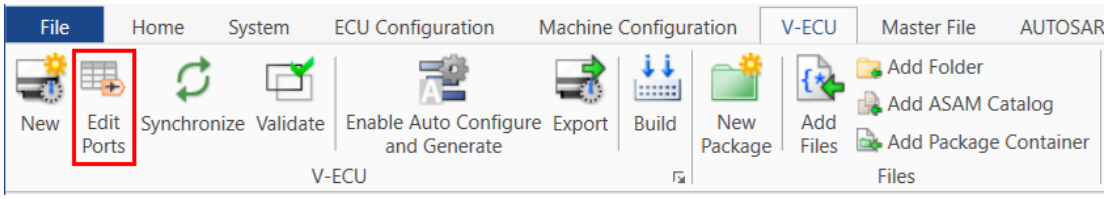
SystemDesk 5.4



SystemDesk 5.5

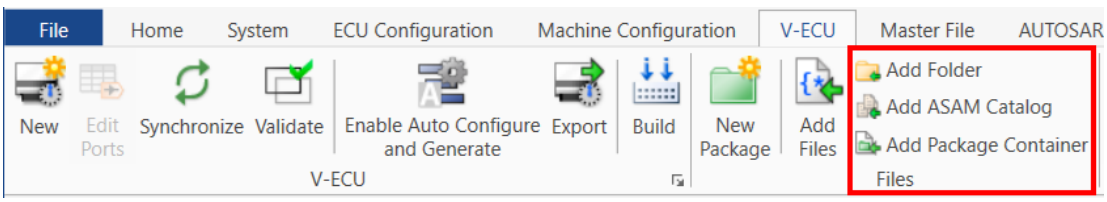


- To edit VPU ports, you can now access the Dap module configuration directly via the Edit Ports button on the V-ECU - V-ECU ribbon. For clarity, VPU ports are no longer displayed in the V-ECU Manager.



The Dap module configuration contains the configuration of signals that access V-ECUs directly, i.e., without modeled bus communication, such as signals of I/O hardware abstraction and RTE interventions.

New functionalities The V-ECU ribbon and the V-ECU context menu contain new functionalities for creating packages from folders, ASAM catalog files and package containers. Refer to the following illustration.



Creating V-ECUs SystemDesk now provides a unified dialog for creating V-ECUs. The new **Create V-ECU** dialog lets you choose in one step which kind of V-ECU to create.

For model-based V-ECUs, you can directly select the ECU configuration to be used as the basis for the V-ECU implementation.

Building V-ECUs

The revised **V-ECU Manager** lets you generate a Python build script that uses the new VEOS command line interface to perform a test build of a V-ECU. You can also use the script as a starting point to create your own build scripts with build options tailored to your use case. This lets you ensure that a V-ECU can be built and integrated into a simulation system without errors. Refer to Performing Test Builds of V-ECUs (SystemDesk Manual).

You can import the generated V-ECU implementation container into VEOS to build and integrate it in a simulation system. Refer to Integrating the Simulation System (VEOS Manual).

Migrating SystemDesk projects

In SystemDesk 5.5, you can directly model a V-ECU without having to create a simulation system. Therefore, information related to simulation systems and all the build options of classic V-ECUs are dropped when you migrate a SystemDesk project from a former dSPACE Release to SystemDesk 5.5.

To specify build options and reintegrate simulation systems, use dSPACE VEOS. Refer to Integrating the Simulation System (VEOS Manual).

You can still use SystemDesk to perform test builds of V-ECUs in conjunction with the VEOS command line interface. Refer to Performing Test Builds of V-ECUs (SystemDesk Manual).

Migrating Scripts

Introduction

You have to migrate scripts regarding the creation and build of simulation systems.

Accessing V-ECUs from the IProject interface

The *IProject* interface now provides access to V-ECUs. Due to the removal of the *ISimulationSystem* interface, you can now directly access collections of the *IVecu*, *IAdaptiveVecu*, and *IClassicVecu* interfaces from the *IProject* interface.

You can no longer access VPUs that are not V-ECUs via the SystemDesk API. Use the VEOS API instead.

The following example from the AR Fuelsys demo Python scripts of SystemDesk 5.4 and SystemDesk 5.5 provide examples for adapting scripts.

Old listing The V-ECUs are accessed from the simulation system in the following listing for SystemDesk 5.4.

```
def ExportVEcuImplementations():
    """
    Exports V-ECU Implementation files (*.ctlgz) for all V-ECUs
    in a simulation system.
    """
    activeProject = SdApplication.ActiveProject
    simSystem = activeProject.SimulationSystems.Item(Options.ProjectName)

    for vEcu in simSystem.ClassicVEcus.Elements:
        expDir = vEcu.ResolvedContainerDirectory
        ctlgzAbsPath = os.path.abspath(os.path.join(expDir, vEcu.Name + ".ctlgz"))
        print(("Exporting V-ECU implementation " + ctlgzAbsPath))
        vEcu.ExportContainer(exportDir)
```

New listing You can now access V-ECUs from the *IProject* interface as shown in the following listing for SystemDesk 5.5.

```
def ExportVEcuImplementations():
    """
    Exports V-ECU Implementation files (*.vecu) for all V-ECUs
    in a simulation system.
    """
    project = SdApplication.ActiveProject

    oldBatchMode = SdApplication.BatchMode
    try:
        SdApplication.BatchMode = True
        for vEcu in project.ClassicVEcus.Elements:
            vecuAbsPath = GetVEcuExportPath(vEcu)
            print(("Exporting V-ECU implementation " + vecuAbsPath))
            vEcu.ExportContainer(vecuAbsPath)
    finally:
        SdApplication.BatchMode = oldBatchMode
```

Interface changes The following table provides an overview of changes to the *IProject* interface.

Interface	Changes
IProject	▪ Property (removed): SimulationSystems { get; } : ISimulationSystems ▪ Property (added): AdaptiveVEcus { get; } : IAdaptiveVEcus ▪ Property (added): ClassicVEcus { get; } : IClassicVEcus ▪ Property (added): VEcus { get; } : IVEcus

Specifying build options

The build option properties of V-ECUs were removed. You can use SystemDesk to generate a build script for the VEOS command line tool and adapt it to your requirements. Refer to Performing Test Builds of V-ECUs (🕮 SystemDesk Manual).

The build() method of V-ECUs is still available. However, the build is now performed via the VEOS command line tool. The returned *IBuildResult* interface only provides information on the last build and not on the build status of the OSA, as for earlier SystemDesk versions.

Old listing In SystemDesk 5.4 you were able to configure the VEOS build as shown in the following listing.

```
# Configure V-ECU controller
    vEcuContrl = simSystem.ClassicVEcus.Item("Controller")
    vEcuContrl.ContainerDirectory.Value = "<ProjectDir>\\<SimulationSystemName>"
    vEcuContrl.BuildDirectory.Value = "<ProjectDir>\\..\\VEOS\\<SimSys>\\<EcuInst>"
    vEcuContrl.BuildOptions.AssignXcpPortAutomatically = False
    vEcuContrl.BuildOptions.XcpServicePort = 30200
    debugCompilerConfig = SystemDeskEnums.CompilerConfigurationEnum.Debug
    vEcuContrl.BuildOptions.CompilerConfiguration = debugCompilerConfig
```

Interface changes The following table provides an overview of the changes to the *IVecu* interface. The interfaces for classic and adaptive V-ECUs also changed.

Interface	Changes
IVEcu	▪ Property (removed): BuildDate { get; } : DateTime ▪ Property (removed): BuildDirectory { get; } : IFileNameMacroSupport ▪ Property (removed): BuildInformation { get; } : IBuildInformation ▪ Property (removed): BuildState { get; } : BuildStateEnum ▪ Property (removed): ContainerDirectory { get; } : IFileNameMacroSupport ▪ Property (removed): OsaId { get; } : String ▪ Property (removed): ResolvedBuildDirectory { get; } : String ▪ Property (removed): ResolvedContainerDirectory { get; } : String ▪ Method (removed): Clean() : Void

Automating test builds

SystemDesk 5.5. now provides a generated build script that uses the VEOS command line tool for test builds of V-ECUs. You can generate and parameterize the build script via the SystemDesk automation interface.

New listing You can configure the test build in SystemDesk 5.5 as follows.

```
vEcu = SdApplication.ActiveProject.ClassicVEcus.Item("Controller")
buildParams = vEcu.GetNewBuildScriptGenerationParameters()

buildParams.OutputFile = "<ProjectDir>\\..\\VEOS\\<SystemName>\\<SystemName>.osa"
buildParams.ContainerFile = "<ProjectDir>\\<SystemName>\\<VEcuName>.vecu"
buildParams.BuildDirectory = "<ProjectDir>\\..\\VEOS\\<SystemName>\\<VEcuName>"
buildParams.XcpPort = 30200
buildParams.Configuration = "Debug"

vEcu.GenerateBuildScriptFromParameters(buildParams)
```

Interface changes The following table provides an overview of the changes that were made to support test build automation using the VEOS command line tool.

Interface	Changes
IVEcu	▪ Property (added): BuildScript { get; } : IFileNameMacroSupport ▪ Property (added): BuildScriptTemplateFile { get; } : IFileNameMacroSupport

Interface	Changes
	▪ Property (added): DefaultBuildDirectory { get; } : IFileNameMacroSupport ▪ Property (added): DefaultBuildScriptFile { get; } : IFileNameMacroSupport ▪ Property (added): DefaultVEcuContainerFile { get; } : IFileNameMacroSupport ▪ Method (added): GenerateBuildScript(String targetFile, String templateFile, String ContainerFile, String BuildDirectory) : Boolean ▪ Method (added): GenerateBuildScriptFromParameters(IBuildScriptGenerationParameters parameters, String targetFile, String templateFile) : Boolean ▪ Method (added): GetNewBuildScriptGenerationParameters() : IBuildScriptGenerationParameters

Support of the new V-ECU container file format

Due to the support of the new V-ECU container file format, the *IVecuImplementation* interface changed. These changes apply to scripts that specify the implementation of code-based V-ECUs.

The following specialized collections, replace the removed IVecuFolders collection:

- IVEcuPackages: To group implementation files.
- IVEcuModulePackages: To group BSW module configurations for platform-specific modules.
- IVEcuRequiredPackages: To add required libraries.

Old listing The following old listing of the Lunar Lander demo shows how code files were added in SystemDesk 5.4.

```
# Add code files to the V-ECU implementation.
    folder = vEcuImplementation.Folders.AddNew("Autopilot")
    codeDir = os.path.join(demoDir, "Autopilot_" + Options.Platform)
    for implementationFile in implementationFiles:
        codeFile = folder.Files.Add(os.path.join(codeDir, implementationFile))
        codeFile.Category = "Code.Component"
```

New listing The following listing of the Lunar Lander demo shows how to add code files in SystemDesk 5.5.

```
# Add code files to the V-ECU implementation.
    package = vEcuImplementation.Packages.AddNew("Autopilot")
    codeDir = os.path.join(demoDir, "Autopilot_" + Options.Platform)
    for implementationFile in implementationFiles:
        package.Files.Add(os.path.join(codeDir, implementationFile))
```

You no longer have to specify the MIME type and category of files that you add to code-based V-ECUs.

Interface changes The following table provides an excerpt of the changes on the IVecuImplementation interface.

Interface	Changes
IVEcuImplementation	▪ Property (removed): Folders { get; } : IVEcuFolders ▪ Property (added): ModulePackages { get; } : IVEcuModulePackages ▪ Property (added): Packages { get; } : IVEcuPackages ▪ Property (added): RequiredPackages { get; } : IVEcuRequiredPackages ▪ Method (added): GetAllPackages() : IVEcuPackage[]

Further reading

For more information, refer to *API Changes from SystemDesk 5.4 to SystemDesk 5.5* in the *SystemDesk API Reference* that is only available in dSPACE Help.

TargetLink

Where to go from here

Information in this section

Information in other sections

TargetLink New Features and Migration Guide
Provides information on new features, migration steps, discontinuations and code changes of the different TargetLink releases.

New Features of TargetLink 5.1

Where to go from here

Information in this section

Modeling in Simulink or Stateflow

Where to go from here

Information in this section

Referenced Subsystems

Referenced subsystems

TargetLink 5.1 supports Simulink® Subsystem Reference. You can use Subsystem Reference for component-based modeling to reference a reuseable group of blocks.

Via Subsystem Reference you can save the contents of a subsystem in a separate SLX file. You can reference this subsystem by using a **Subsystem Reference** block.

Related documentation

- How to Make Referenced Subsystems TargetLink-Compliant (TargetLink Preparation and Simulation Guide)

Related topics

HowTos

How to Make Referenced Subsystems TargetLink-Compliant (TargetLink Preparation and Simulation Guide)

Array-of-Struct Support

SIL simulation

TargetLink 5.1 supports the access of array-of-struct variables during simulation via signal injection and signal tunneling.

Related documentation

- Basics on Injecting or Tunneling Signals During Simulation (TargetLink Preparation and Simulation Guide)

Dynamic access via Custom Code (type II) block

TargetLink 5.1 supports the dynamic access to array-of-struct variables that are specified at data stores via **Custom Code (type II) blocks**.

Related documentation

- Example of Dynamically Accessing Array-of-Struct Variables via Data Store Blocks and Custom Code (Type II) Blocks (TargetLink Preparation and Simulation Guide)

Support of Simulink Functions

You can now define global Simulink Function subsystems in a TargetLink subsystem that can be called from **Function Caller blocks** in the same code generation unit.

Related documentation Working With Simulink Function Subsystems in TargetLink (TargetLink Preparation and Simulation Guide)

Probe Block

Newly supported TargetLink simulation block: Probe block

TargetLink now supports the Probe block to output the width and/or signal dimensions of the input signal.

Related documentation

- Probe Block Functionality (TargetLink Model Element Reference)
- Probe block (TargetLink Limitation Reference)

MATLAB Code

MATLAB Code Improvements

Constant variables in MATLAB code

TargetLink 5.1 lets you use constant variables in MATLAB code to increase code readability. In addition, you can declare a constant local MATLAB variable as a macro or constant C variable.

Related documentation

- Basics on Constant Variables in MATLAB® Code (TargetLink Code Generation Guide for MATLAB® Code in Simulink® Models)

Related topics

Basics

Basics on Constant Variables in MATLAB® Code (TargetLink Code Generation Guide for MATLAB® Code in Simulink® Models)

Adaptive AUTOSAR

Enhanced Support for Adaptive AUTOSAR

Enhanced Adaptive AUTOSAR support

TargetLink 5.1 supports additional features of Adaptive AUTOSAR.

The following additional features are supported:

- Import of Adaptive AUTOSAR ARXMLs with elements defined by ara::per
- Modeling of select parts of a service-based communication as described by ara::com:
  - Sending and receiving events as defined in Adaptive AUTOSAR release 18-10.
  - SIL simulation of Adaptive AUTOSAR components.
- Modeling of select parts of access to persistent memory as described by ara::per:
  - Read and write access to key-value pairs with an **AdaptivePlatformType** from key-value storages via Data Store blocks.

A demo model that showcases the different modeling styles for Adaptive AUTOSAR communication is provided. Refer to AAR_COMMUNICATION (📖 TargetLink Demo Models).

Additionally, a new guide for the user documentation is available. Refer to 📖 TargetLink Adaptive AUTOSAR Modeling Guide.

Related documentation

- 📖 TargetLink Adaptive AUTOSAR Modeling Guide
- AAR_COMMUNICATION (📖 TargetLink Demo Models)

Classic AUTOSAR

Where to go from here

Information in this section

Dynamically Accessing Array-of-Struct Variables Specified at Data Stores

TargetLink now lets you dynamically access array-of-struct variables that are specified at data stores via **Custom Code** (type II) blocks.

This is possible for the following Classic AUTOSAR data prototypes:

- Data elements
- Interrunnable variables

- NvData elements
- Static memories

Related documentation

- Dynamically Accessing AUTOSAR Data Stores via Custom Code (Type II) Blocks (🕮 TargetLink Classic AUTOSAR Modeling Guide)
- AR_ARRAY_OF_STRUCT_DATA (🕮 TargetLink Demo Models)

General Enchancements and Changes for Classic AUTOSAR

KPR.2014.09.23.004 resolved

With TargetLink 5.1, the known problem report KPR.2014.09.23.004 is resolved.

It is now possible to compile the production code for AUTOSAR models that use the fixed-point library regardless of the module names used.

The handling of data types used for the fixed-point library has changed. As a result, the `tl_types.h` header file is no longer required and thus no longer generated.

Related documentation

- Migration Aspects Regarding Custom Code on page 195

AUTOSAR Import and Export Improvements

Support of new data types

TargetLink now lets you import ARXML files that use the following data types into the Data Dictionary:

- `Int64` and `UInt64` whose encoding is `None` or `2C`
- Data types whose Category is `DATA_REFERENCE` (Pointer)
- `Optimized` integer data types as described in the `Platform_Types` header

The export of all these data types is also supported.

Note

You cannot use these data types in code generation.

Look-up-table-related elements

TargetLink now lets you import and export the following:

- Primitive application data types of the categories `CURVE`, `MAP`, and `COM_AXIS`.
- Elements that are related to the calibration of look-up tables.

You can use these when modeling look-up tables for AUTOSAR and to prepare measurement and calibration.

Related documentation

- Basics on Preparing Look-Up Tables for Measurement and Calibration (Classic AUTOSAR) (🕮 TargetLink Classic AUTOSAR Modeling Guide)
- Example of Preparing a Look-Up Table for Measurement and Calibration (🕮 TargetLink Classic AUTOSAR Modeling Guide)

Tool Chain Support

Improved Container Export

Improvements to FMU containers

With TargetLink 5.1, the following improvements to FMU containers are available:

- FMU containers that include the following can be generated:
  - Source code only
  - Binaries only (new)
  - Source code and binaries
- Binaries for the Linux 64-bit platform can be built. This requires a suitable compiler, which you must provide.
- By default, the source code of the generated FMUs is compatible with the following platforms:
  - If system files are not included: all 32-bit and 64-bit platforms (QNX, Windows and Linux)
  - If system files are included: all 32-bit and 64-bit platforms (QNX, Windows and Linux) with little endian byte order

Related documentation

- Basics on Exporting FMUs from TargetLink (🕮 TargetLink Interoperation and Exchange Guide)
- tl_generate_fmu
- TargetLink FMU Manager (🕮 TargetLink Tool and Utility Reference)

SIC compatibility

With TargetLink 5.1, the SIC files generated by TargetLink are compatible with the following dSPACE platforms:

- Host PC: Windows 32-bit, Windows 64-bit and Linux 64-bit (VEOS)
- SCALEXIO: QNX/Linux 32-bit (ConfigurationDesk)
- MicroAutoBox III: Linux 32-bit (ConfigurationDesk)

Related documentation

- TargetLink SIC Manager (🕮 TargetLink Tool and Utility Reference)
- tlGenerateSic (🕮 TargetLink API Reference)

V-ECU implementation container

With TargetLink 5.1, the V-ECU containers generated by TargetLink comply with V-ECU Version 3.0 and are compatible with the following dSPACE platforms:

- Host PC: Windows 32-bit, Windows 64-bit and Linux 64-bit (VEOS)
- SCALEXIO: QNX/Linux 32-bit (ConfigurationDesk)
- MicroAutoBox III: Linux 32-bit (ConfigurationDesk)

Related documentation

- Basics on Interoperating with Other dSPACE Tools for Virtual Validation (TargetLink Interoperation and Exchange Guide)
- tl_generate_vecu_implementation

V-ECU implementation with fixed-point library

With TargetLink 5.1, the dSPACE fixed-point library is included in a V-ECU implementation container if required. Therefore, it is no longer necessary to export the library as a ZIP archive when using V-ECU implementation containers in ConfigurationDesk.

Related documentation

- TargetLink V-ECU Manager (TargetLink Tool and Utility Reference)
- tl_generate_vecu_implementation

Related topics

References

tl_generate_fmu (TargetLink API Reference)
tl_generate_vecu_implementation (TargetLink API Reference)

Code Generation Core Functionality

Where to go from here

Information in this section

Unicode

Unicode support

TargetLink 5.1 supports Unicode in the following areas:

- Descriptions in Simulink blocks.
- Descriptions in TargetLink blocks.
- Descriptions in **TargetLink Data Dictionary** entries.
- Comments in A2L files.
- Descriptions in AUTOSAR XML files.

You can specify a character encoding via the **CharacterSet** property in the Data Dictionary Manager. With TargetLink 5.1 the default is changed from **LocalDefault** to **UTF-8**.

Related documentation

- Overview of the Supported Character Sets (🕮 TargetLink Interoperation and Exchange Guide)

C99 Support Enhancements

Fixed-width integer data types as defined by C99

TargetLink 5.1 supports fixed-width integer data types with exact widths as defined by the language standard C99. This support includes signed and unsigned types. You can use the data types via the **Generic C99** Code Generation target.

Related documentation

- Basics on Generating Code Compliant with C99/C11/C++ (🕮 TargetLink Customization and Optimization Guide)

Floating-Point Limit Handling

Macros and DD objects for floating-point limits

With TargetLink 5.1, floating-point type limits are handled via macros in the generated code. These macros are defined in separate, platform-specific header files that can be automatically generated again if required. This ensures that the generated code that uses the floating-point limits can be compiled for different platforms.

TargetLink now provides predefined DD **Variable** objects for the floating-point limits of the `Float32` and `Float64` base types. These objects can be used to model the use offloating-point limits.

Related documentation

- Basics on Floating-Point Limits in TargetLink (📖 TargetLink Preparation and Simulation Guide)

Improved Data Flow Analysis

Introduction

TargetLink performs optimizations based on data flow analysis. Examples are:

- Elimination of intermediate variables.
- Scope reduction of variables.
- Removal of unnecessary computations.
- Moving code into conditionally executed control flow branches.

Improved data flow analysis

With TargetLink 5.1, data flow analysis is improved in the following ways:

- Consistent treatment of the influence of the DD **VariableClass** object's **Optimization** property values, especially `MOVABLE`.
- Consistent treatment of variables and functions accessible across code generation unit (CGU) boundaries.
- Consistent scheduling of less-used custom-code-section code.
- Scope reduction of variables used in the initialization of other variables.
- Interprocedural data flow analysis for cyclic function call relationships.
- Increased analysis depth for interprocedural data flow analysis.
- Utilization of data flow properties of arguments of function calls and the newly introduced user assertions for Data Store variables of the **Custom Code** (type II) block.
- Improved utilization of alias relationships for accesses via pointer dereference.

This leads to several major and minor changes in generated code, usually contributing either to code efficiency or matching user expectations with regard to consistent optimization behavior or to correct code with respect to data flow across function, CGU, and external functionality borders. For details, see Interprocedural Analysis on page 207 in Code Changes Between TargetLink 5.0 and TargetLink 5.1 on page 200.

Target Simulation (PIL)

Changes in the Target Simulation Modules

New and discontinued compiler versions

The following table shows the compiler versions that are now supported by TargetLink 5.1. Refer to the New and No changes columns. Compiler versions that are no longer supported are listed in the Discontinued column.

Microcontroller Family	Compiler	New	No Changes	Discontinued
ARM CortexM3	Keil	—	5.2	—
C16x	TASKING	—	8.6	—
MPC57xxVLE	Diab	—	5.9	—
	GreenHill	—	2019	—
RH850	GreenHill	—	2019	—
S12X	Cosmic	—	4.8	—
	Metrowerk	—	5.1	—
SH2	Renesas	—	9.3	—
SH2A-FPU	Renesas	—	9.4	—
TriCore17xx	TASKING	—	3.2	—
TriCore2xx	TASKING	—	6.3	—
	GCC	—	4.9	—
TMS570 (ARM)	CCS	—	7.0	—
V850	GreenHill	—	2019	—
XC22xx	TASKING	—	3.0	—

For more information on the evaluation boards supported by TargetLink, refer to Combinations of Evaluation Boards and Compilers (📖 Evaluation Board Reference).

Note

For more PIL support combinations that are part of a valid Software Maintenance Service (SMS) contract, refer to the dSPACE TargetLink PIL Support website at the TargetLink Product Support Center.

Usability

Where to go from here

Information in this section

Property Manager

Usability improvements

TargetLink 5.1 provides several usability improvements in the Property Manager, for example:

- It is now possible to use the breadcrumb navigation to copy, paste, or edit the path of a model element.
- There are several improvements concerning the display of hints and tooltips in the Property Manager.

Related documentation

- Display of Validation Errors (TargetLink Preparation and Simulation Guide)
- Breadcrumb (TargetLink Tool and Utility Reference)
- Reset View Set (TargetLink Tool and Utility Reference)
- Search (Property View) (TargetLink Tool and Utility Reference)

Data Dictionary

Usability improvements

TargetLink 5.1 provides several usability improvements in the Data Dictionary Manager. For example:

- In the **Edit Matrix** dialog, you can now use the keys and key combinations known from spreadsheet applications.
- Properties containing a list of file paths can now be edited in the **StringList** dialog.
- An improved design of the property value list improves orientation and ease of use.

Related documentation

- Edit Matrix (TargetLink Data Dictionary Manager Reference)
- Edit StringList (TargetLink Data Dictionary Manager Reference)

DD three-way merge

TargetLink 5.1 provides a DD three-way merge that can compare and merge two different versions of a DD file derived from a common ancestor.

Related documentation

- Basics on Comparing and Merging DD Files via DD Three-Way Merge (🕮 TargetLink Data Dictionary Basic Concepts Guide)

Code Generator Options

New Code Generator Options

Overview of new Code Generator options

The following new Code Generator options are available with TargetLink 5.1:

- DefaultCppHeaderFileExtension

  Lets you define the default header file name extension for modules whose programming language is set to C++. You can set the programming language of a module via the Language property of the DD ModuleInfo object.
- DefaultCppSourceFileExtension

  Lets you define the default source file name extension for modules whose programming language is set to C++. You can set the programming language of a module via the Language property of the DD ModuleInfo object.
- UsePlatformTypeAbstractionLayer

  Lets you generate an additional, central header file for mapping typedef names to platform data types.
- MapTypeLimitMacrosToTargetConfigurationValues

  Lets you use values that are defined in the `TargetConfig.xml` for the macros for floating-point limits in the production code.
- ReportDataStoreUserAssertionsForCustomCode

  Lets you generate a report that lists the user assertions specified for a Data Store access via Custom Code (type II) block.
- AssumePossibleSideEffectsForReadAccessToNonErasableVariables

  Lets you specify whether TargetLink has to assume potential side effects on read accesses to non-ERASABLE variables.

Related documentation

- DefaultCppHeaderFileExtension (🕮 TargetLink Model Element Reference)
- DefaultCppSourceFileExtension (🕮 TargetLink Model Element Reference)
- UsePlatformTypeAbstractionLayer (🕮 TargetLink Model Element Reference)
- MapTypeLimitMacrosToTargetConfigurationValues (🕮 TargetLink Model Element Reference)

- ReportDataStoreUserAssertionsForCustomCode (📖 TargetLink Model Element Reference)
- AssumePossibleSideEffectsForReadAccessToNonErasableVariables (📖 TargetLink Model Element Reference)

For reference information on all Code Generator options, refer to Alphabetical List of Code Generator Options (📖 TargetLink Model Element Reference).

Migration aspects of Code Generator options

For more information, refer to Migration Aspects Regarding Code Generator Options on page 192.

API Functions and Hook Scripts

New API Functions

List of new API functions

API Function	Purpose
`tlPromoteProperty`	Provides TargetLink properties in TargetLink block masks for Simulink® promote mechanism. Refer to Basics on Adding Mask Parameters for TargetLink Block Properties (📖 TargetLink Preparation and Simulation Guide).

Related topics

References

tlPromoteProperty (📖 TargetLink API Reference)

Other

Where to go from here

Information in this section

General Enhancements and Changes

Programming language of a module

With TargetLink 5.1, you can set the programming language of a module via the Language property of the DD ModuleInfo object. By default, the programming language depends on the code generation mode. For Adaptive AUTOSAR, it is C++, otherwise C.

Related documentation

- Basics on Generating Code Compliant with C99/C11/C++ (🕮 TargetLink Customization and Optimization Guide)

Adding mask parameters for TargetLink block properties

With TargetLink 5.1, you can provide mask parameters at TargetLink blocks for the Simulink® promote mechanism..

Related documentation

- Basics on Adding Mask Parameters for TargetLink Block Properties (🕮 TargetLink Preparation and Simulation Guide)

Dynamically assigning new values to data store variables

With TargetLink 5.1, TargetLink can detect block patterns that are used to dynamically assign new values to data store variables.

Related documentation

- Basics on Dynamically Assigning New Values to Data Store Variables (🕮 TargetLink Preparation and Simulation Guide)

Support of Docker for Windows

With TargetLink 5.1, TargetLink can be used for tasks without user interaction on a Docker for Windows container.

For more information, contact dSPACE Support (www.dspace.com/go/supportrequest) for more information.

TargetLink Demos

New demo models

New Demo Model	Description
AAR_COMMUNICATION (🕮 TargetLink Demo Models)	The model shows you how to model communication according to Adaptive AUTOSAR.
ENUM_BLINKER (🕮 TargetLink Demo Models)	The model shows you how to implement a turn signal by using Simulink enumerations.

Modified demo models

Changed Demo Model	Description
AR_COLLISION_DETECTION (📖 TargetLink Demo Models)	The model is extended with a model part concerning `Rte_result` error handling.
AR_ARRAY_OF_STRUCT_DATA (📖 TargetLink Demo Models)	The model now uses a generic modeling pattern which allows TargetLink to generate direct accesses to the RTE variables, or the correct RTE API functions and access points.
PRELOOKUP_USR (📖 TargetLink Demo Models)	The model is extended with the specification of table data via table input ports.

Migrating to TargetLink 5.1 and TargetLink Data Dictionary 5.1

Upgrade process

Carefully read all of the following information and modify the tool chain accordingly.

Where to go from here

Information in this section

General Migration Information

Upgrading Models, Libraries, and Data Dictionaries

Where to go from here

Information in this section

Basics on Migrating Between TargetLink Versions

Automatic upgrade from TargetLink 3.1 or later

TargetLink 5.1 automatically upgrades models and TargetLink-compliant libraries if they were created with TargetLink 3.1 or later.

The automatic upgrade includes all the steps required by the intervening TargetLink versions. For example, an automatic upgrade from TargetLink 4.0 to TargetLink 5.1 comprises the steps 4.0 to 4.1 to 4.2 to 4.3 to 4.4.

Note

Check the TargetLink migration documentation of the different TargetLink versions to see whether user interaction is required.

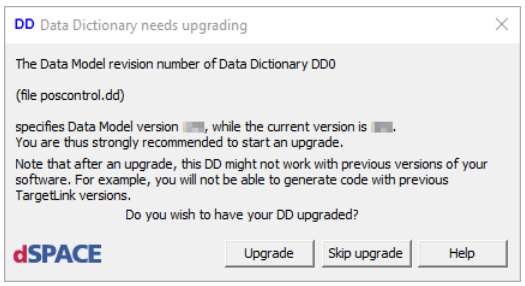
User interaction required In the following cases, for example, the automatic upgrade requires additional user interaction:

- Libraries must be TargetLink-compliant. Otherwise, no upgrade can be performed.
- Style sheets for code generation are version-specific and subject to change from one TargetLink version to another. Thus, modified style sheets of older TargetLink versions have to be updated to match the current version (reapplying the modifications as intended). Refer to Basics on Code Formatting (🕮 TargetLink Customization and Optimization Guide).
- Custom code S-functions built with 32-bit TargetLink versions do not work with 64-bit versions of TargetLink.

  Initiate a rebuild of all custom code S-functions using the `tlUpgrade('Model',<MyModel>,'CheckModel','FixIssues')` API function.

Upgrading Data Dictionaries

When you open a Data Dictionary created with a TargetLink version 2.x or later with TargetLink 5.1 for the first time, you are prompted for the upgrade. For example:



To upgrade DD files with included DD files, refer to How to Upgrade a Data Dictionary with Included DD Files on page 186.

Making new libraries TargetLink-compliant

Libraries that you create from scratch and that consist of TargetLink blocks must be made upward compatible so that you can upgrade them to a newer TargetLink version in the future. Otherwise, no upgrade can be performed.

Note

A library does not automatically become a TargetLink library if it contains TargetLink blocks. The library itself must be TargetLink-compliant.

Refer to How to Make User Libraries Upgrade-Capable (📖 TargetLink Orientation and Overview Guide).

Making existing libraries TargetLink-compliant

The following two approaches let you make libraries created with earlier TargetLink versions compliant with the current TargetLink version 5.1:

The earlier TargetLink version is available Use the TargetLink version with which the library was created to make the library TargetLink-compliant. Refer to the TargetLink documentation of the earlier TargetLink version. You can then use this library with all later TargetLink versions because TargetLink automatically performs an upgrade. The library can still be used with TargetLink versions earlier than TargetLink 5.1 because the automatic upgrade does not save a library in the newer TargetLink version.

Only the current TargetLink version 5.1 is available Use TargetLink version 5.1 and the `tlUpgrade` API command to make the library TargetLink-compliant. Refer to How to Manually Upgrade Libraries and Models via the API on page 188. If you follow the instructions, the library is saved in TargetLink version 5.1. Therefore, it cannot be used with TargetLink versions earlier than TargetLink 5.1.

Manual upgrade from TargetLink 2.x or 3.0.x

Models and libraries created with TargetLink versions 2.x or 3.0.x have to be upgraded manually to the latest TargetLink version 3.x (3.1...3.5) you have. Afterwards, automatic upgrade is possible.

No backward compatibility

You cannot use models. libraries and Data Dictionaries in the format of newer TargetLink versions in earlier TargetLink versions.

Data model filter rule files

Existing data model filter rule files can contain invalid elements because the data model of the TargetLink Data Dictionary changed. The following files that were shipped with previous TargetLink versions can be affected:

- `DD_Filter_Admin.xml`
- `DD_Filter_AR_User.xml`
- `DD_Filter_NonAR_NonRTOS_User.xml`

You can check filter rule files via the API in the MATLAB Command Window:

Checking a Single File	Checking Filter Rule Sets[1)]
`dsdd_free;`	`dsdd_free;`
`dsdd('ReadFilterRuleSet', 'file', '<myFile>.xml');`	`dsdd('ReloadFilterRuleSets');`
`ds_error_register(dsdd('GetMessageList'));`	`ds_error_register(dsdd('GetMessageList'));`
`ds_msgdlg('update');`	`ds_msgdlg('update');`

1) All the files contained in the directory defined in Data Dictionary - Filter Rules in the Preferences Editor.

TargetLink informs you about errors in the TargetLink Message Browser. Each error contains the following information so that you can fix it in any XML-capable editor:

- File name
- Row number
- Column number

Related topics

Basics

Basics on Code Formatting (📖 TargetLink Customization and Optimization Guide)

HowTos

How to Make User Libraries Upgrade-Capable (📖 TargetLink Orientation and Overview Guide)
How to Manually Upgrade Libraries and Models via the API 188

References

tlUpgrade (📖 TargetLink API Reference)

How to Upgrade a Data Dictionary with Included DD Files

Precondition

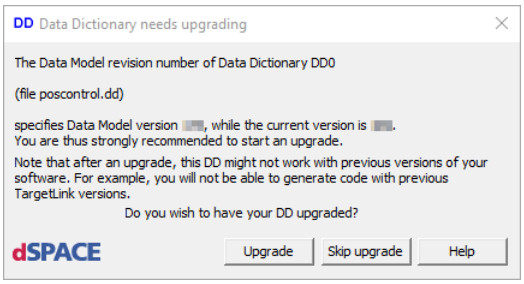
In the main DD file to be loaded, the AutoLoad property of the DDIncludeFiles objects is set to on.

Method

To upgrade a Data Dictionary with included DD files

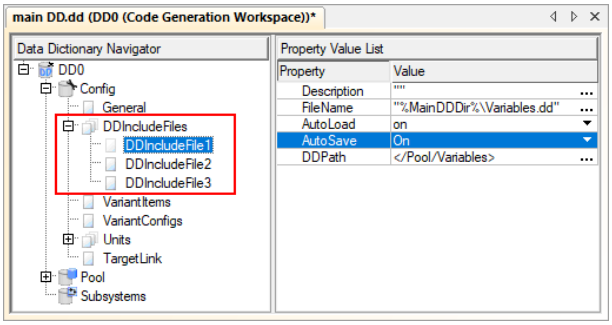
1 Open the Data Dictionary Manager and the main DD via File - Open.

The Data Dictionary needs upgrading dialog automatically opens if an earlier data model revision is used.



2 Select Upgrade. The main DD and all included DD files are upgraded to the latest data model revision.

3 Set the AutoSave property of all the `/Config/DDIncludeFiles/<DDIncludeFile>` objects to on to save all included DD files together with the main DD.



If you do not want to save all included DD files together with the main DD, specify their AutoSave property as required.

4 Save the main DD file.

Result

You upgraded the main DD file and all included DD files. TargetLink adjusted the revision number of every upgraded DD file to the latest data model revision.

Related topics

Basics

Basics on Opening and Handling DD Files (TargetLink Data Dictionary Basic Concepts Guide)

HowTos

How to Include Partial Data Dictionary Files (TargetLink Data Dictionary Basic Concepts Guide)

References

Point of Inclusion (TargetLink Data Dictionary Manager Reference)

How to Manually Upgrade Libraries and Models via the API

Objective

To prepare a central upgrade of libraries and models in a tool chain scenario with several users, for example.

Preconditions

The model or library files are available on the MATLAB search path but they are not open.

The required and upgraded DD project file has been opened, for example, via **`dsdd_manage_project('Open','<name>.dd')`**. DD project files can be upgraded via **`dsdd('Upgrade'[,<DD_Identifier>])`**.

Method

To manually upgrade libraries and models via the API

1 Type the following API command in the MATLAB Command Window:

```
tlUpgrade('Model', '<Model|Library>.slx',
'CheckModel','FixIssues')
```

The model or library is upgraded.

Note

When you upgrade models and libraries, first upgrade models or libraries that do not reference any other libraries, i.e., the blocks and subsystems they contain have no links to other libraries. Start with the bottom library and then upgrade the libraries above it in ascending order.

2 Save the upgraded model or library files, e.g., `Library.slx`.

3 Repeat steps 1 and 2 for all other models or libraries.

Result

You upgraded the models and libraries.

Related topics

References

tlUpgrade (📖 TargetLink API Reference)

Migrating Data Dictionaries to CodeDecorationSets

Introduction of CodeDecorationSet and CodeDecoration objects

Since TargetLink 4.3 DD CodeDecorationSet and CodeDecoration objects are introduced.

Additionally, several properties were removed from the Data Dictionary data model:

DD Object	Change	Replacement
FunctionClass	Removal of the DeclarationStatements and SectionName properties.	The DeclarationStatements and SectionName properties of the DD CodeDecoration.Settings object.
VariableClass		
VariableClassTemplate.Filter	Removal of the WidthSpec property.	The WidthSpec property of the DD CodeDecoration.Filter object.

Automatic upgrade by TargetLink

Limitation TargetLink no longer supports width-specific type prefixes for variable classes. The automatic upgrade of the Data Dictionary fails if the original Data Dictionary contains variable class templates used to derive variable classes that have width-specific type prefixes.

Use declaration statements instead.

When you open a Data Dictionary whose data model is older than the latest revision, TargetLink prompts you to perform an automatic upgrade.

Object Kind	Trigger	Upgrade Action
VariableClass FunctionClass	DeclarationStatements or SectionName properties are set.	1. Creating a DD CodeDecorationSet object. 2. Creating a single DD CodeDecoration object for each DD CodeDecorationSet object. The settings of the CodeDecoration object and its child objects match the settings of the original objects. 3. Referencing the CodeDecorationSet object at the original object.
SubStructTemplate	Filter.VariableClass is set.	Transfer the values of the following properties from the variable class to the SubStructTemplate object's filter: ▪ DeclarationStatements ▪ SectionName ▪ TypePrefix

Object Kind	Trigger	Upgrade Action
VariableClassTemplate	▪ Filter.FilterCondition property is set to ALL_TRUE. ▪ Settings.VariableClass references a DD VariableClass object whose DeclarationStatements or SectionName properties are set. ▪ The Filter.WidthSpec property is set for this DD VariableClassTemplate object or for another VariableClassTemplate object whose Filter.VariableClassSpec property has the same value.	1. Create a new DD VariableClass object in /Pool/VariableClasses/Templates. 2. Create a new DD CodeDecorationSet object in /Pool/CodeDecorations/Templates. 3. For each VariableClassTemplate object with the same value at the Filter.VariableClassSpec property, adding a CodeDecoration object to the CodeDecorationSet object. 4. Specifying the CodeDecoration object as required. 5. Referencing the CodeDecorationSet object at the VariableClass object created in step 1. 6. Referencing the VariableClass object created in step 1 via the VariableClassTemplate.Settings.VariableClass property.

Special considerations for variable class templates

If you specified DD VariableClassTemplate objects whose Filter.FilterCondition property is set to ALWAYS or NEVER, TargetLink deletes the object's Filter.WidthSpec property during the upgrade without replacement.

If you want to keep the property value, set the DD VariableClassTemplate object's Filter.FilterCondition property to ALL_TRUE before upgrading the Data Dictionary.

Limitation TargetLink does not upgrade DD VariableClassTemplate objects whose Filter.FilterCondition property is set to ONE_OR_MORE or ALL_FALSE.

Cleaning

The automatic upgrade retains the functionality that was specified in the previous Data Dictionary. You can clean it manually to reduce the number of objects in the new Data Dictionary.

Merging width-specific variable classes If the previous Data Dictionary contained width-specific VariableClassTemplate/VariableClass objects, the new Data Dictionary still contains all these variable classes.

Because the width-specific information is now stored in DD CodeDecoration objects, you can manually reduce the number of VariableClass objects in the Data Dictionary. For example, if you used variable classes in the form of <Name>_<Width>, you can replace them by a single <Name> variable class that references a suitable code decoration set.

Two methods are possible:

- Merging code decoration sets:
  1. Copy all the DD CodeDecoration objects that were generated during the upgrade for each variable class called <Name>_<Width> to a single CodeDecorationSet object.
  2. Make each CodeDecoration object width-specific via its filter.
  3. Reference the resulting CodeDecorationSet object at the <Name> variable class.

- Using a code decoration set created for variable class templates:
  1. If the original `<Name>_<Width>` variable classes were referenced by variable class templates, the DD upgrade automatically creates a width-specific code decoration set in `/Pool/CodeDecorationSets/Templates`.
  2. You can reference this code decoration set at the resulting variable class called `<Name>`.

Note

Replace references from model elements to the variable classes called `<Name>_<Width>` with references to `<Name>`.

Retarget variable class templates After you merged the previous width-specific variable classes, you can use them again as the target of the variable class templates. You can then delete all the variable classes contained in `/Pool/VariableClasses/Templates` that were created during the upgrade.

Simplifying user-specified scope reduction chains (SRC) If you used a user-specified SRC to specify declaration statements or section names for variables with specific scopes, you can do the following:

1. Adjust the Filter.ScopeSpec property of the code decoration that belongs to the set referenced by the first variable class in the SRC (highest scope) as required.
2. Delete the other variable classes of the SRC.
3. If you also used the SRC to prevent static local variables, you can now use the AvoidStaticLocalScope Code Generator option instead.

Remove obsolete variable class templates Find DD VariableClassTemplate objects with the same value of the Filter.VariableClassSpec property and delete all but one.

Changes in the generated production code

Changes in CodeDecoration objects can influence the generated production code mainly in the following respects:

- Changed code comments (📖 TargetLink New Features and Migration Guide)
- Sorting of variable definitions (📖 TargetLink New Features and Migration Guide)

Refer to Code Changes Between TargetLink 4.3 and TargetLink 4.4 (📖 TargetLink New Features and Migration Guide).

Migrating from TargetLink 5.0 to 5.1

Where to go from here

Information in this section

Code Generator Options

Migration Aspects Regarding Code Generator Options

User interface

The **OptimizedBoolType** Code Generator option has been removed from the **Advanced** page of the TargetLink **Main Dialog** block. It is still available via the **All options...** button on this page.

Related documentation

- OptimizedBoolType (📖 TargetLink Model Element Reference)

Basics on changed defaults

The settings of the Code Generator options are stored with the model (model-based option storage). In addition, you can store user-defined sets of Code Generator options in DD **CodegenOptionSet** objects (DD-based option storage). You can use DD **CodegenOptionSet** objects as a central source for overwriting and replacing the model-based option settings that were used since TargetLink 4.1.

If a model-based option value equals the old default value, it is automatically changed to the new default value during the upgrade. If a DD-based option value equals the old default value, it is not changed to the new default value during the upgrade but keeps the old value.

Option value = old default If Code Generator options were set to default values in the earlier TargetLink version, and the new TargetLink version uses modified default values, note the following points:

- Model-based option:

  If you want to keep the old default values, you must reset them manually.
- DD-based option:

  If you want to use the new default values, you must adjust them manually.

The following table is an example describing the impact of a TargetLink upgrade ($TargetLink_{Old}$ to $TargetLink_{New}$) on three option values: 9, 11, and 13. The table illustrates two basic migration scenarios:

- Scenario #1: New default = old default

  The default value of a Code Generator option has not changed in the new TargetLink version, i.e., the default value remains 9.

  None of the option values is changed.
- Scenario #2: New default ≠ old default

  The default value of a Code Generator option changed with the new TargetLink version, i.e., the default value changed to 11.

Option Storage	**Option Value ($TargetLink_{Old}$)**	**Option Value (≤ $TargetLink_{New}$)**	
	Default = 9	**Default = 9 (Scenario #1)**	**Default = 11 (Scenario #2)**
Model-based	9[1)]	9[1)]	11[2)]
	11	11	11[1)]
	13	13	13
DD-based	9	9	9[3)]
	11	11	11
	13	13	13

1) The option value is not stored with the model because it equals the default.
2) Manual reset might be necessary.
3) Manual adjustment might be necessary.

Option value = new default If the Code Generator options were not set to default values in the former TargetLink version (A) but are in the new TargetLink version (B), TargetLink assumes that you intentionally specified the default value in the new TargetLink version. The same applies if the default changes again in the next TargetLink version (C).

Note

Upgrading $TargetLink_A \Rightarrow TargetLink_B \Rightarrow TargetLink_C$ and upgrading $TargetLink_A \Rightarrow TargetLink_C$ can cause different option values. Refer to the following table.

If the default values for TargetLink versions A, B, and C are 9, 11, and 13, and an option was set to 11 in version A, an upgrade to version C changes the option value as follows:

Upgrade Strategy	Option Value $TargetLink_A$ Default = 9	Option Value $TargetLink_B$ Default = 11	Option Value $TargetLink_C$ Default = 13
A ⇒ B ⇒ C	11 (≠ default)	11 (= default)[1)]	13 (= default)[1)]
A ⇒ C	11 (≠ default)	—	11 (≠ default)

[1)] The option value is not stored with the model because it equals the default.

New Code Generator options

For more information on new Code Generator options, refer to New Code Generator Options on page 179.

AUTOSAR

Migration Aspects Regarding AUTOSAR

Syntax of AUTOSAR versions

The syntax for the specification of the AutosarVersion property of DD Config objects found under `/Pool/Autosar` has been adjusted to the naming convention of AUTOSAR: From AUTOSAR Standard R19-11 onward, the AUTOSAR Standard RYY-MM is specified as `YY-MM`.

With AUTOSAR Standard R19-11 `Rte_TransformerError` changes into `Std_TransformerError`.

Related documentation:

- Basics on Data Transformation (📖 TargetLink Classic AUTOSAR Modeling Guide)

Optional DD RteEvents and DD Runnables objects

With TargetLink 5.1, the following child objects of DD SoftwareComponent objects are optional:

- DD RteEvents object
- DD Runnables object

These objects will no longer be automatically created when a DD SoftwareComponent object is created. Because neither RteEvents nor Runnables are relevant in Adaptive AUTOSAR, the related objects are superfluous in this context.

Custom scripts that create DD SoftwareComponent objects and expect the existence of the now optional child objects must be adjusted accordingly.

Existing objects and DD projects remain unchanged.

Custom Code

Migration Aspects Regarding Custom Code

tl_types.h no longer generated

The `tl_types.h` header file is no longer generated. If your custom code uses the TargetLink base types and contains an include statement for `tl_types.h`, replace it as follows:

Code Generation Mode	Replacement
Standard/RTOS	Include statement for the `tl_basetypes.h` header file.
Classic AUTOSAR	Include statement for the `Rte_Types.h` header file.
Adaptive AUTOSAR	Replace the TargetLink base types with the fixed-width data types of Adaptive AUTOSAR, such as `int8_t`.

API Functions and Hook Scripts

Changes in TargetLink and TargetLink Data Dictionary API Functions

tlCodeCoverage

New commands

Command	Purpose
`GenerateCombinedReport` (refer to tlCodeCoverage('GenerateCombinedReport', propertyName, propertyValue, ...) (🕮 TargetLink API Reference))	Generates a combined report from all CTC files in and below the specified folders. Refer to How to Measure Code Coverage Over Multiple Simulation Application Builds via Third-Party Tools (Testwell CTC) (🕮 TargetLink Preparation and Simulation Guide).
`MoveCTCFiles` (refer to tlCodeCoverage('MoveCTCFiles', propertyName, propertyValue, ...) (🕮 TargetLink API Reference))	Moves CTC files created during the build process and simulation to the specified folder. These files can later be used to create a combined report of code coverage analysis for multiple builds and simulations. By default, each time an application is (re)built, the existing CTC files are moved from the TargetLink build folder to the `.\CTCRESULTS\TLBuild_<BuildTimeStamp>` folder.

tldoc

Removed properties The **Encoding** property value pair is removed. UTF-8 encoding is used.

tl_generate_vecu_implementation

Removed properties The **ExportFixedPointLibrary** property value pair is removed because it is no longer required.

Refer to Improved Container Export on page 173.

tlRebuildFixedPointLibrary

New properties

Property	Description
Assembler	Code generation target setting. Default: 'off'
CodeOpt	Code generation target setting. Default: 'Generic ANSI-C'

New output parameters

Parameter	Description
bError	Error flag
msgList	Error message

Removed output parameters The following output parameters have been removed:

- succeededNr
- failedNr
- upToDateNr

Messages

Changes in TargetLink Messages

A20449

The following messages changed their type to **Warning** in TargetLink 5.1 because of changes in the semantics of the Optimization property of DD VariableClass objects:

Old Message Number	New Message Number
A20449	W20449

For more information on the changed semantics, refer to Migration Aspects Regarding Optimization on page 199.

Reserved Identifiers

Migration Aspects regarding reserved identifiers

New reserved identifiers

With TargetLink 5.1, a number of prefixes in identifiers are reserved by TargetLink. The use of these identifiers leads to messages of different severity:

Identifier Use	Used Prefix	Result
▪ Function name ▪ Variable name ▪ Macro ▪ Enum value ▪ Type name	▪ `tl_dsfxp_` ▪ `tl_sim_` ▪ `tl_std_`	Error
	▪ `tl_`	Warning
▪ Struct tag ▪ Struct component	▪ `tl_`	Advice

Adjust the models accordingly.

Related documentation

- Basics on TargetLink Base Types, Typedefs, and Header Files (🕮 TargetLink Customization and Optimization Guide)

Reserved identifiers for limit macros

With TargetLink 5.1, additional identifiers for macros that represent floating-point limits are reserved:

`tl_FLOAT32MAX`	`tl_sim_FLOAT32MAX`	`tl_dsfxp_FLOAT32MAX`
`tl_FLOAT32NMIN`	`tl_sim_FLOAT32NMIN`	`tl_dsfxp_FLOAT32NMIN`
`tl_FLOAT64MAX`	`tl_sim_FLOAT64MAX`	`tl_dsfxp_FLOAT64MAX`
`tl_FLOAT64NMIN`	`tl_sim_FLOAT64NMIN`	`tl_dsfxp_FLOAT64NMIN`

In addition, the identifiers for fixed-point limit macros for the Fixed-Point Library and the simulation frame are reserved as well. These identifiers consist of one of the prefixes `tl_sim_` or `tl_dsfxp_`, the base type, and one of the `MAX` or `NMIN` suffixes, e.g., `tl_dsfxp_INT8MAX`.

Adjust the model accordingly.

Related documentation

- Basics on Floating-Point Limits in TargetLink (🕮 TargetLink Preparation and Simulation Guide)

Other Migration Aspects

Various Migration Aspects

DD ModuleTemplate objects

With TargetLink 5.1, DD ModuleTemplate objects have been removed.

If you used DD ModuleTemplate objects to change the file name extension of modules for Adaptive AUTOSAR modules to `cpp`, you can now set the programming language of a module via the Language property of the DD ModuleInfo object.

By default, the Language property sets the programming language of modules depending on the code generation mode. For Adaptive AUTOSAR, it is C++, otherwise C. You can define the default file name extension for header files and source files containing C++ code via the following Code Generator options:

- DefaultCppHeaderFileExtension (🕮 TargetLink Model Element Reference)
- DefaultCppSourceFileExtension (🕮 TargetLink Model Element Reference)

TargetLink Autodoc Customization block

With TargetLink 5.1, the Autodoc Customization block that was introduced with TargetLink 3.3 (green) is the only Autodoc Customization block that is supported. This block is available via the TargetLink block library.

The Autodoc Customization block from TargetLink versions earlier than TargetLink 3.3 (yellow) is no longer supported and must be replaced.

Void BaseTypeRename

With TargetLink 5.1, the `Void` TargetLink base type can be renamed only via a BaseTypeRename if the `CodedType` property in the `TargetConfig.xml` for the selected code generation target is not specified as `Use standard C void type`.

Leaf struct components at Data Store Memory blocks

If you reference a leaf struct component of a struct variable whose Scope property is set to `ref_param`, TargetLink now displays E20009.

With TargetLink versions earlier than 5.1, the error was displayed only if a plain variable or a struct variable whose Scope property was set to `ref_param` was referenced at a Data Store Memory block.

Updating version-specific style definition file and style sheets

During code generation, TargetLink 5.1 checks if the version of the used code output style definition file and code output style sheets matches the current TargetLink version to avoid unexpected behavior when using old versions of these files. If it does not match, TargetLink displays an error. Compare and update your currently used code output style definition file and code output style

sheets, e.g., via a diff and merge tool, with the original files provided by TargetLink 5.1. In addition, make sure that the correct TargetLink version entry is specified.

Related documentation

- Basics on Code Formatting (📖 TargetLink Customization and Optimization Guide)

Optimization

Migration Aspects Regarding Optimization

Optimization

If you set the **Optimization** property at the DD **VariableClass** object to MOVABLE, ERASABLE, or SCOPE_REDUCIBLE, you guarantee that there are no unknown variables accesses, i.e., there are no variable accesses outside of the CGU or variable accesses from external functions unless they have been specified at the interfaces.

Related documenation:

- Basics on Optimizing Variables (📖 TargetLink Customization and Optimization Guide)

Code Changes Between TargetLink 5.0 and TargetLink 5.1

Where to go from here

Information in this section

AUTOSAR

Optimizing comparisons of RTE API calls

TargetLink now optimizes not equal comparisons of identical RTE API calls.

TargetLink ≤ 5.0	TargetLink 5.1
`if(Rte_IRead_Run_NvSRP_Deep_01()->e.b != Rte_IRead_Run_NvSRP_Deep_01()->e.b) { ... }`	`if (0) { ... }` (and subsequent removal)

Reason Code efficiency

Auxiliary Variables

Auxiliary variables in sqrt function calls

In calls of the `sqrt` function, TargetLink now uses an auxiliary variable if the operand of the function is an expression that is more complex than a simple variable access. This prevents that the expression is calculated twice:

TargetLink ≤ 5.0

```
if (((Ca6_F64Index1 + ((Float64) Ca6_F32Index1)) - Ca6_F64Index1) >= 0.) {
   AUX_var_d = (Int32) sqrt((Ca6_F64Index1 + ((Float64) Ca6_F32Index1)) - Ca6_F64Index1);
} else {
   AUX_var_d = (Int32) (-sqrt(-((Ca6_F64Index1 + ((Float64) Ca6_F32Index1)) - Ca6_F64Index1)));
}
```

TargetLink 5.1

```
Aux_ = (Ca6_F64Index1 + ((Float64) Ca6_F32Index1)) - Ca6_F64Index1;
if (Aux_ >= 0.) {
   AUX_var_d = (Int32) sqrt(Aux_);
} else {
   AUX_var_d = (Int32) (-sqrt(-Aux_));
}
```

This change occurs mainly in Stateflow or MATLAB code.

Additionally, when generating unoptimized code, the names of the auxiliary variables might change.

Reason Code efficiency

Auxiliary variables for min, max and unary minus

In `min`, `max`, and `unary minus` operations, TargetLink now uses an auxiliary variable if the operand is an expression that is more complex than a simple variable access. This prevents that the expression is calculated twice:

TargetLink ≤ 5.0

```
if ((((Int32) Ca1_I16In1) + 5) < (((Int32)
   Ca1_I16In2) - 12)) {
      Ca1_I32Out = ((Int32) Ca1_I16In1) + 5;
} else {
   Ca1_I32Out = ((Int32) Ca1_I16In2) - 12;
}
```

TargetLink 5.1

```
Aux_ = ((Int32) Ca1_I16In1) + 5;
Aux__a = ((Int32) Ca1_I16In2) - 12;
if (Aux_ < Aux__a) {
   Ca1_I32Out = Aux_;
} else {
   Ca1_I32Out = Aux__a;
}
```

Reason Code efficiency

Dimension of auxiliary matrix variables

TargetLink now might downgrade the dimension of 64-Bit auxiliary matrix variables.

TargetLink ≤ 5.0	TargetLink 5.1

`for(i = 0; ...) {`
`  F_I64MUL..(Const, &Aux_Hi[0][i], &Aux_LO[0][i]);`
`  ...`
`}`
	`for(i = 0; ...) {`
	`  F_I64MUL..(Const, &Aux_a, &Aux_b);`
	`  ...`
	`}`

Reason Code efficiency

Basetype-Related Code Changes

Macros for minimum and maximum values for floating-point types

With TargetLink 5.1, you can use predefined DD Variable objects to model the use of floating-point limits. This will result in the use of macros in the production code. Refer to Basics on Floating-Point Limits in TargetLink (🕮 TargetLink Preparation and Simulation Guide). The macros are named as follows:

- tl_FLOAT32MAX
- tl_FLOAT32NMIN
- tl_FLOAT64MAX
- tl_FLOAT64NMIN

In addition to user-specified uses of the macros, the macros occur in the exception handling of the following operations:

- division with floating-point result
- `RSqrt ( = 1/sqrt)`
- `atanh`
- `log`, $\log_{10}$

Example for the division:

TargetLink ≤ 5.0

```
if (Sa1_InPort1 != 0) {
   Sa1_NoMinNoMax = ((Float32) Sa1_InPort) / ((Float32)
Sa1_InPort1);
} else {
   if (((Float32) Sa1_InPort) < 0.F) {
      Sa1_NoMinNoMax = -3.402823466e+38F;
   } else {
      Sa1_NoMinNoMax = 3.402823466e+38F;
   }
}
```

TargetLink 5.1

```
if (Sa1_InPort1 != 0) {
   Sa1_NoMinNoMax = ((Float32) Sa1_InPort) / ((Float32)
Sa1_InPort1);
} else {
   if (((Float32) Sa1_InPort) < 0.F) {
      Sa1_NoMinNoMax = tl_FLOAT32NMIN;
   } else {
      Sa1_NoMinNoMax = tl_FLOAT32MAX;
   }
}
```

Reason Improved platform independence of the production code

Migration issue With this change, the names of the macros and the corresponding macros for simulation frame and Fixed-Point library are treated as identifiers reserved for use by TargetLink. Refer to Migration Aspects regarding reserved identifiers on page 197.

Comment-Related Code Changes

Omitting identical comments TargetLink now omitts identical comment parts for each statement comment.

TargetLink ≤ 5.0

```
/* # combined # BusOutport: Subsystem/Runnable/Bus Outport
   # combined # Gain: Subsystem/Runnable/Gain
   # combined # BusOutport: Subsystem/Runnable/Bus Outport
*/
Rte_Call_ClientPort_Setter(Scalar * 5,
(const sint16 *) Vector, (const RootType *) &Struct_a);
```

TargetLink 5.1

```
/* # combined # BusOutport: Subsystem/Runnable/Bus Outport
   # combined # Gain: Subsystem/Runnable/Gain */
Rte_Call_ClientPort_Setter(Scalar * 5,
(const sint16 *) Vector, (const RootType *) &Struct_a);
```

Reason

- Bug fix
- Readability
- Matches user expectations

Sorting operation comment parts TargetLink now transfers the comment part of the outermost operation to the first position.

TargetLink ≤ 5.0

```
/* Abs: subsystem/ABS
   # combined # TargetLink outport: subsystem/OutPort */
Sa1_OutPort
= (Int8) __satb(((__sat) __abs((Int16) (((Int16)
Sa1_InPort) - 158))) - ((__sat) 12));
```

TargetLink 5.1

```
/* TargetLink outport: subsystem/OutPort
   # combined # Abs: subsystem/ABS */
Sa1_OutPort
= (Int8) __satb(((__sat) __abs((Int16) (((Int16)
Sa1_InPort) - 158))) - ((__sat) 12));
```

Reason

- Bug fix
- Easier matching of model and code

Unifying block type, block path and code comment

TargetLink now adds the block type and block path for the Merge and Rate Limiter block.

TargetLink ≤ 5.0	TargetLink 5.1
`/* Reference of merge block: Merge subsystem/Merge */`	`/* Merge: subsystem/Merge */`
`/* subsystem/Rate Limiter: output */`	`/* Rate Limiter: subsystem/Rate Limiter: output */`

For the Assignment, Custom Code, and the dynamic Selector block, TargetLink now adds the seperator **:** between the block path and the code comment.

TargetLink ≤ 5.0	TargetLink 5.1
`/* Assignment: subsystem/Assignment - output initialization */`	`/* Assignment: subsystem/Assignment: output initialization */`
`/* Custom code: subsystem/CustomCode << output code >> */`	`/* Custom code: subsystem/CustomCode: << output code >> */`
`/* Selector: subsystem/Selector - init phase */`	`/* Selector: subsystem/Selector: init phase */`

Reason

- Bug fix
- Easier matching of model and code
- Matches user expectations

Slashes in include statements

TargetLink now uses slashes (/) instead of backslashes (\) in include statements.

TargetLink ≤ 5.0	TargetLink 5.1
`#include "builtin_functions\standard_mathematics.h`	`#include "builtin_functions/standard_mathematics.h`

Reason MISRA C compliance

List of charts and statemachines in header comments

The header comments of generated files changed regarding the list of Stateflow statemachines and charts. With TargetLink 5.1 this list contains only those elements that are accessible by the code generation unit (CGU) for that the file was generated.

TargetLink ≤ 5.0

`*** SF-NODE  CORRESPONDING STATEFLOW NODE  DESCRIPTION`
`*** Cd0  MyLibNoExternGraphFcn`
`*** Cd1  MyLibWithExternGraphFcn`
`*** Cd2  SomeModel`
`*** Cd3  SomeSubsystem/Chart`

TargetLink 5.1

```
*** SF-NODE        CORRESPONDING STATEFLOW NODE                  DESCRIPTION
*** Cd1            MyLibWithExternGraphFcn
*** Cd2            SomeModel
*** Cd3            SomeSubsystem/Chart
```

Reason

- Easier matching of model and code
- Matches user expectations

Control Flows

Merging control flows

TargetLink now merges the control flows in opposite, mirrored, and `#if` conditions.

TargetLink ≤ 5.0	TargetLink 5.1

`if (a >= b) {`
`  //1`
`} else {`
`  //2`
`}`
`if (a < b) {`
`  //3`
`}`
	`if (a >= b) {`
	`  //1`
	`} else {`
	`  //2`
	`  //3`
	`}`

`if (a >= 0) {`
`  //1`
`}`
`if (0 <= a) {`
`  //2`
`}`
	`if (a >= 0) {`
	`  //1`
	`  //2`
	`}`

`if (a < b) {`
`  //1`
`  if (b > a)) {`
`  //2`
`  }`
`} else {`
`  //3`
`  if (b <= a) {`
`  //4`
`  }`
`}`
	`if (a < b) {`
	`  //1`
	`  //2`
	`} else {`
	`  //3`
	`  //4`
	`}`

`#if defined a && a >= 199`
`  //1`
`#endif`
`#if defined a && a >= 199`
`  //2`
`#else`
`  //3`
`#endif`
	`#if defined a && a >= 199`
	`  //1`
	`  //2`
	`#else`
	`  //3`
	`#endif`

TargetLink ≤ 5.0	TargetLink 5.1

`#if MODE == 3`
`  //1`
`  #if MODE != 3`
`  //2`
`  #else`
`  //3`
`  #endif`
`#endif`
	`#if MODE == 3`
	`  //1`
	`  //3`
	`#endif`

Reason Code efficiency

Migration issue You can control this optimization via the CombineControlFlowStatements Code Generator option.

Custom Code

Custom code sections and control flow

TargetLink sorts custom code sections in the production code in the following order:

- Common before not common
- Top before unqualified or bottom
- Output before update

TargetLink now also considers data flow on sorting custom code sections which can lead to the following consequences:

- Custom code output variables are not removed if a bottom output section is present, because the bottom output section potentially changes the value of the output variable after the value is read.
- Variables that are custom code input variables and for whose input direct feedthrough is specified, may not be removed when using a top output section or may not be reduced in scope to automatic storage duration if the calculation of these variables is performed after the top output section as the top output section potentially accesses the input variable.
- Variables that are custom code input variables and for whose input direct feedthrough is not specified are described earlier when using a top update section in order to prevent uninitialized read accesses.
- Update sections that are not qualified as bottom section are not placed after the other state sections. Instead these update sections are placed at the position where the state update of another block would be placed.

In some situations, the position of custom code sections in the production code can change if the data flow requires it.

Reason Bug fix

Function Inlining

Changed variable suffixes

A function might contain several calls to the same function:

```
void func() {
   Int16 Sum;
   Sum = expr;
}

void main() {
   func();
   ...
   func();
}
```

These function calls are now always inlined in the same order in which they are located in the code. This can lead to changed variable suffixes:

TargetLink ≤ 5.0	TargetLink 5.1

`main() { ...`
`Sum_a = expr;`
`...;`
`Sum = expr; }`
	`main() { ...`
	`Sum = expr;`
	`...;`
	`Sum_a = expr; }`

Reason

- Bug fix
- Matches user expectations

Interprocedural Analysis

Different statement order

In rare cases, TargetLink now might move some statements differently due to improved interprocedural and alias data flow analysis.

Reason Bug fix

AccessFunction patterns

TargetLink now uses different AccessFunction code patterns for the following situations:

- Multiple definitions before one use.
- Multiple uses before one definition.
- One use before multiple definitions.

TargetLink ≤ 5.0	TargetLink 5.1
Multiple Definitions Before One Use	

`if (cond) {`
`  Aux_S16 = ...;`
`} else {`
`  Aux_S16 = ...;`
`}`
`...`
`... = ... Aux_S16...;`
`SetA(Aux_S16);`
	`if (cond) {`
	`  Aux_S16 = ...;`
	`} else {`
	`  Aux_S16 = ...;`
	`}`
	`SetA(Aux_S16);`
	`...`
	`... = ... Aux_S16...;`

Multiple Uses Before One Definition	

`Aux_S16 = GetA();`
`... = ... Aux_S16...;`
`...`
`... = ... Aux_S16...;`
`...`
 	
`Aux_S16 = ...;`
`SetA(Aux_S16);`
	`Aux_S16 = GetA();`
	`... = ... Aux_S16...;`
	`...`
	`... = ... Aux_S16...;`
	`...`
	
	`SetA(...);`

One Use Before Multiple Definitions	

`Aux_S16 = GetA();`
`... = ... Aux_S16...;`
`...`
 	
`if (cond) {`
`  Aux_S16 = ...;`
`} else {`
`  Aux_S16 = ...;`
`}`
`SetA(Aux_S16);`
	`... = ... GetA() ...;`
	`...`
	
	`if (cond) {`
	`  Aux_S16 = ...;`
	`} else {`
	`  Aux_S16 = ...;`
	`}`
	`SetA(Aux_S16);`

Reason

- Bug fix
- Code efficiency

Adding auxiliary variables when inlining for global CallByValue function parameters

Based on TargetLink's internal data flow analysis during code generation, TargetLink now adds auxiliary variables when inlining for global function arguments if the global variable might have write accesses via this function call. If a variable is passed into a function both by value and as a pointer an auxiliary variable is used when inlining, too. In many situations the auxiliary variables are removed during the optimization process.

Reason Bug fix

Write accesses of global variables in conditionally executed control flow branches

TargetLink now keeps the write accesses of global variables out of conditionally executed control flow branches in any of the following cases:

- If there may be additional accesses to the variable to be propagated outside the current CGU or in external functions.

- If the global variable has a DD VariableClass whose Optimization property does not containing MOVABLE.

TargetLink ≤ 5.0	TargetLink 5.1

foo() {
 	
  if (cond) {
  Out = <...>
  }
 	
}
bar() {
  foo();
  ... = Out
}
	foo () {
	  static int X;
	
	  if (cond) {
	  X = <...>
	  }
	
	  Out = X;
	}
	bar() {
	  foo();
	  ... = Out
	}

Reason Bug fix

Condition assignment for Boolean variables

TargetLink now optimizes Boolean variables that are specified with the predefined variable class FCN_REF_ARG.

TargetLink ≤ 5.0	TargetLink 5.1

if (Cond == 1) {
*PtrVar = 1;
}
else {
*PtrVar = 0;
}
	*PtrVar = Cond == 1;

Reason Bug fix

Migration issue You can avoid this optimization via the AssignmentOfConditions Code Generator option, at the cost of losing optimization in other contexts that are targeted by this option.

Consecutive assignments to dereferenced pointers

Some TargetLink code patterns, e.g., for Stateflow graphical functions, require potentially unnecessary assignments to variables to make sure that values are assigned on all execution paths. TargetLink removes any unnecessary previous assignments. This optimization is now also possible if the value assignment takes place via a dereferenced pointer.

TargetLink ≤ 5.0	TargetLink 5.1

pA = &a;
*pA = 5;
*pA = 4;
	pA = &a;
	*pA = 4;

Reason Bug fix

Correct data flow relationships for static local variables in cyclic function calls

TargetLink now correctly considers interprocedural data flow via cyclic function calls for variables with `local` scope and `static` storage duration that have write accesses, because they might be a state of the complete cycle:

- Seemingly unnecessary write accesses can be removed only if there are no intervening calls to a function belonging to the cycle.
- Reduction of the storage duration to `auto` is only possible if no cyclic function calls occur between write and read accesses.

The following table shows a case where the seemingly unnecessary assignment `Ca1_x = 1.;` is retained:

TargetLink ≤ 5.0

```
if (Ca1_x == 1.) {
   Ca1_a = 10.;
}
else {
   Ca1_a = -10.;

   /* call of function: Subsystem/Chart2 */
   Ca3_Chart2();
   Ca1_x = 2.;
   Ca1_out = Ca1_a;
}
```

TargetLink 5.1

```
if (Ca1_x == 1.) {
   Ca1_a = 10.;
}
else {
   Ca1_a = -10.;
   Ca1_x = 1.;

   /* call of function: Subsystem/Chart2 */
   Ca3_Chart2();
   Ca1_x = 2.;
   Ca1_out = Ca1_a;
}
```

Reason Bug fix

Static local variables prevent expression movement

TargetLink no longer moves function calls that directly or indirectly contain accesses to the same static local variable past each other.

Reason Bug fix

Interfaces of global functions

In TargetLink 5.1, global functions with function reuse and Stateflow® exported graphical functions can be called outside the current code generation unit (CGU) and effectively become part of the CGU's interface. Consequently, TargetLink does not remove seemingly unused output variables because calls outside of the current CGU might consume them. Moreover, for exported graphical functions unnecessarily modeled input variables are also preserved in order to prevent compiler or linker errors. . If some of the inputs or outputs are truly unnecessary and the functions are not to be called outside the current CGU, consider removing them from the respective function interface via changing the model.

Reason Matches user expectations

Optimizing movable variables

TargetLink now interprets the presence of `MOVABLE` in the Optimization property of a DD VariableClass object in the same way as `ERASABLE` and `SCOPE_REDUCIBLE` with respect to assumptions about data flow:

If a variable is `MOVABLE` then the modeled accesses in the current CGU are all accesses that take place, i.e., there are neither accesses in other CGUs nor in

external functionality called by TargetLink unless they have been specified as part of the respective function or Custom Code interface. This affects the moving of accesses to global variables as follows:

- Past calls to functions with unknown content.
- Into conditionally executed control flow branches.

This usually leads to generated code that is more efficient.

In addition to the previously mentioned effects, TargetLink sometimes does the following:

- Eliminates more or different intermediate variables.
- Performs more control flow optimizations.

Refer to Basics on Optimizing Variables (🕮 TargetLink Customization and Optimization Guide).

Reason Code efficiency

Migration issue Refer to Migration Aspects Regarding Optimization on page 199.

Optimization in guaranteed write before read situations

For function calls or custom code, variables are now optimized as follows: Variables whose **Write before read** user assertion is set are treated the same way as variables whose **Write only** user assertion is set.

For information on the different user assertions, refer to Basics on Using TargetLink Data Stores with Custom Code (Type II) Blocks (🕮 TargetLink Preparation and Simulation Guide).

Example

```
f(&b /* Write Before Read */);
c = b;
```

The variable `b` can be reduced to automatic storage duration and eliminated in favour of variable `c`. If variable `b` has an assignment before the call, the assignment can be removed.

Reason Code efficiency

Saturated Abs operations

Previously, if the `Abs` operation had an unecessary saturation and this saturation was implemented in different branches of a control flow, TargetLink generated a comment in each branch. Now, TargetLink tests earlier for unecessary saturations of `Abs` operations. This results in less comments:

TargetLink ≤ 5.0

```
if(I16In >= 0) {
   /* <Block>: Omitted upper saturation
   <Block>: Omitted lower saturation */
   U16Out = I16In;
} else {
   /* <Block>: Omitted upper saturation */
   U16Out = C_U16FITI16_SATl(I16In, -32768);
}
```

TargetLink 5.1

```
/* <Block>: Omitted upper saturation */
if(I16In >= 0) {
   /* <Block>: Omitted lower saturation */
   U16Out = I16In;
} else {
   U16Out = C_U16FITI16_SATl(I16In, -32768);
}
```

Additionally, the unecessary saturation might be removed from individual branches. In the following example, the saturation of the lower bound in the `then` branch is omitted:

TargetLink ≤ 5.0

```
/* Abs: Inp_Int32_Outp_Int8/Abs_14 */
if (Sh1_InPort_14 >= 0) {
   Sh1_OutPort_14 = C__I8SHRI32C6_LT24_SATb(Sh1_InPort_14, 23, 1065353216,
    -1073741824);
} else {
   /* Inp_Int32_Outp_Int8/Abs_14: Omitted lower saturation */
   C__I64NEGI32(Sh1_InPort_14, Aux_S32, Aux_U32);

   Sh1_OutPort_14 = C__I8SHRI64C6_LT32_SATu(Aux_S32, Aux_U32, 23, 9, 0,
    1065353216);
}
```

TargetLink 5.1

```
/* Abs: Inp_Int32_Outp_Int8/Abs_14 Inp_Int32_Outp_Int8/Abs_14: Omitted lower saturation */
if (Sh1_InPort_14 >= 0) {
   Sh1_OutPort_14 = C__I8SHRI32C6_LT24_SATu(Sh1_InPort_14, 23, 1065353216);
} else {
   C__I64NEGI32(Sh1_InPort_14, Aux_S32, Aux_U32);
   Sh1_OutPort_14 = C__I8SHRI64C6_LT32_SATu(Aux_S32, Aux_U32, 23, 9, 0, 1065353216);
}
```

Reason Code efficiency

Reducing variable scopes

In previous TargetLink versions, variables whose addresses were used in the initialization of other variables were not reduced in scope.

In TargetLink 5.1, the Code Generator can reduce variables whose addresses are used in the initialization of other variables down to static local, if the initialization allows for it. For example:

- Reuse substructures
- Axes and map in a look-up table map structure

If referenced and referencing variables are created on the same level, i. e., in the same file, function, or compound statement in the function body, the variables are defined in relation to each other. This means, that the variables might be moved further upwards or change their relative order if necessary.

Reason MISRA C compliance, Matches user expectations.

Migration issue To avoid the scope reduction of variables whose addresses are used in the initialization of other variables, you can use one of the following:

- Disable the AvoidStaticLocalScope Code Generator option.
- Reference a DD VariableClass object whose Optimization property does not contain `SCOPE_REDUCIBLE`.

More inlining in conditionally compiled code

TargetLink now estimates the costs of functions that are encapsulated by preprocessor directives such as `#if` in a better way. This results in more inlined

function in the context of conditional compilation. In other contexts, less inlining might occur in rare cases.

Reason Code efficiency

Optimizing read accesses of non erasable symbols

TargetLink can now optimize ineffectual read access of symbols with a DD VariableClass object whose Optimization property is not set to ERASABLE. While the symbols cannot be removed, some of their read access may be removed. *Ineffectual* read access do not affect the result of a computation, usually due to constant operands that determine the value of an expression.

Additionally, read access of macros with a DD VariableClass object whose Optimization property is not set to ERASABLE can now be removed only if they are constant. Otherwise they are considered to have possible side-effects. This can prevent optimizations that were performed in previous versions of TargetLink.

Examples Consider the following examples:

```
x = a * 0;
y = b || 1;
z = c && 0;
```

a, b, and c do not contribute to the value of x, y, or z, respectively, and can be removed. Note that for logical AND and OR operations, the operand order determines whether an access is ineffectual or unreachable.

```
Aux = MY_EXTERNAL_MACRO;

if (Value > 1)
{
  Output = Aux * 2.;
} else {
  Output = Aux * 0.5;
}
```

Given the externally defined macro MY_EXTERNAL_MACRO, the accesses to the variable Aux in the if and else branches will not be replaced by MY_EXTERNAL_MACRO, because MY_EXTERNAL_MACRO is neither constant nor has a variable class with the ERASABLE property set.

Reason

- Matches user expectations
- Code efficiency

Migration issue To prevent the optimized read access, you have to enable the AssumePossibleSideEffectsForReadAccessToNonErasableVariables Code Generator option.

No Stateflow machine data variables optimization

TargetLink no longer optimizes Stateflow® machine data variables because they can be accessed anywhere in the respective model and are not limited to one CGU.

Reason Bug fix

Initialization

Optimization of initial and constant values

When optimizig initial values or constant values, TargetLink now preserves data type information by generating a cast that preceeds the propagated value. This ensures that optimized and unoptimized code behaves in the same way.

Take this code as an example:

```
UInt32 var = 1;
if ( var << 31 >= 2 ) {
//...
}
```

The following table shows the optimzed code:

TargetLink ≤ 5.0	TargetLink 5.1

`if ( 1 << 31 >= 2 ) {`
`//...`
`}`
	`if ( ((UInt32) 1) << 31 >= 2 ) {`
	`//...`
	`}`

No cast is generated in the following contexts:

- Direct assignments
- Function parameters
- Function return values
- Macros (OmitCastOfMacroValue property set to `true`)
- The replaced variable has a floating point type.

The following side effects might rarely occur:

- Casts are omitted for constants that have special semantics.
- Additional comments for constant folding might be generated.

Reason Bug fix

Initial value for floating point variables specified at block and in Data Dictionary

The code changed, if all of the following conditions apply:

- The block references a a DD Variable object whose Type property references a floating point data type.
- An initial value is specified both at the block and at the DD Variable object.
- The initial values specified at the block and at the DD Variable differ within allowed tolerance (epsilon comparison).

TargetLink ≤ 5.0	TargetLink 5.1
The initial value was chosen from Simulink or from the Data Dictionary.	The initial value of the DD Variable object is used.

Reason

- Bug fix
- Increased consistency

Initialization code for Assignment blocks in iterized subsystems

The TargetLink code pattern for the initialization code of Assignment blocks in iterized subsystems changed. In rare circumstances, the old cold pattern could result in false code. With TargetLink 5.1 the initialization is now performed within a conditional control flow that tests an explicit variable (`FirstRun`).

This can result in the following changes:

- The suffixes in the names of local variables might change during optimization.
- The `FirstRun` variable remains in production code, if any of the following conditions holds:
  - The Assignment block resides in a conditionally executed subsystem.
  - The code generated for the Assignment block lies in a branch of a conditionally executed control flow.
- In while-iterator subsystems, the statement order changes: the initilalization of the assignment is now placed between the initialization of the loop variable and the begin of the `while` or `do` loop.

The following code example shows the `FirstRun` variable that now remains in the code:

TargetLink ≤ 5.0

```
for (Sa3_For_Iterator_it = 1; Sa3_For_Iterator_it <= Aux_; Sa3_For_Iterator_it++)

{
    if (Sa3_For_Iterator_it == 1) {
        for (Aux__a = 0; Aux__a < VvwY0PropOn; Aux__a++)
         {
           /* Assignment: Subsystem/For/Y: output initialization */
           Sa3_Y[Aux__a] = DD_Sa3_Y0[Aux__a];
         }
      }

         //...

   }
```

TargetLink 5.1

```
Sa3_Y_FirstRun = 1;

for (Sa3_For_Iterator_it = 1; Sa3_For_Iterator_it <= Aux_; Sa3_For_Iterator_it++)

{
    if (Sa3_Y_FirstRun > 0) {
        for (Aux__a = 0; Aux__a < VvwY0PropOn; Aux__a++)
         {
            /* Assignment: Subsystem/For/Y: output initialization */
            Sa3_Y[Aux__a] = DD_Sa3_Y0[Aux__a];
         }
         Sa3_Y_FirstRun = 0;
      }

          //...

   }
```

Reason Bug fix

Logical Expressions

Optimizing logical expressions

TargetLink now optimizes logical expressions with the type AND or OR that contain operand relations which express identical or inverse relations in arbitrary order. Prior to this, only identically structured expressions were considered as identical relations.

TargetLink ≤ 5.0	TargetLink 5.1
`c = x < 5 || 5 > x`	`c = x < 5`
`c = x < 5 || x >= 5`	`c = 1`
`c = x < 5 && x >= 5`	`c = 0`

Reason Code efficiency

Range Propagation

Improved UInt64 code

The TargetLink worst-case range calculation for unsigned 64-Bit integer code was improved. This can result in better code:

TargetLink ≤ 5.0	TargetLink 5.1

`F__U64MULI32I32(1999999999, 1999999999,`
`  &Aux_U32_b, &Aux_U32_c);`
`C__U64ADDU64U64(Aux_U32, Aux_U32_a, Aux_U32_b, Aux_U32_c,`
`  Aux_U32, Aux_U32_a);`
`Sa1_OutUpLowSat = (Int16) C__I32FITU64_SATu(Aux_U32,`
`  Aux_U32_a, 2147483647);`
	`F__U64MULI32I32(1999999999, 1999999999,`
	`  &Aux_U32_b, &Aux_U32_c);`
	`C__U64ADDU64U64(Aux_U32, Aux_U32_a, Aux_U32_b, Aux_U32_c,`
	`  Aux_U32, Aux_U32_a);`
	`Sa1_OutUpLowSat = (Int16) 2147483647;`

Reason Bug fix

Worst-case range propagation

In prior versions, TargetLink sometimes propagated worst-case ranges for variables of global scope or static storage duration. Because, in rare cases, this might have led to false code, propagation of worst-case ranges is now restricted to variables of local scope that do not have static storage duration.

This can result in the following code changes:

- Unnecessary saturations
- Unnecessary comparisons
- Operations calculated in a width greater than actually needed

The following table shows an example:

TargetLink ≤ 5.0	TargetLink 5.1
`b = C__I16SATI16_SATb(Sa1_InPort, 15, -1);` `c = C__I16SATI16_SATu(b, 1);`	`b = C__I16SATI16_SATb(Sa1_InPort, 15, -1);` `c = C__I16SATI16_SATb(b, 1, -15);`

In some cases, the resulting code might improve as shown in the following table:

TargetLink ≤ 5.0	TargetLink 5.1

`Sa1_OutPort4 = Sa1_InPort2;`
`Sa1_OutPort4 = Sa1_OutPort4 * 5;`
`Sa1_OutPort4 = (Int16) (((UInt16) Sa1_OutPort4) + 2);`
	`Sa1_OutPort4 = Sa1_InPort2;`
	`Sa1_OutPort4 = Sa1_OutPort4 * 5;`
	`Sa1_OutPort4 = (Int16) (Sa1_OutPort4 + 2);`

Reason Bug fix

Shift Operations

Code pattern for the Shift Arithmetic block

TargetLink now uses the shift operation code pattern for the Shift Arithmetic block.

TargetLink ≤ 5.0	TargetLink 5.1
`Sa1_U16LShift = Sa1_U16In << 2;`	`Sa1_U16LShift = (UInt16) (Sa1_U16In << 2);`

Reason Improved platform independence of the production code

Code patterns for shifting numerical values

TargetLink now creates special code patterns for shifting numerical values. Closing casts are also inserted if the shift value is a variable.

TargetLink ≤ 5.0	TargetLink 5.1
`Ca1_UnscaledI32Tmp = (Int32) (10 << 1);`	`Ca1_UnscaledI32Tmp = (Int32) (((UInt32) 10) << 1);`

Reason

- Improved platform independence of the production code
- MISRA C compliance

Stateflow and MATLAB Code

Different order of extracting parts from complex expressions

In Stateflow and MATLAB code, TargetLink extracts different parts from complex expressions, e.g., function calls, matrix multiplications, or casts.

Consider the following code as example:

```
out = 4 + f() + 5 + g();
```

TargetLink ≤ 5.0	TargetLink 5.1

Extracting from left to right:
`aux = g();`
`aux_a = f();`
`out = 4 + aux_a + 5 + aux;`
	Extracting from right to left:
	`aux = f();`
	`aux_a = g();`
	`out = 4 + aux + 5 + aux_a;`

Reason

- Matches user expectations
- Increased consistency
- Readability

Improved detection of constant values for Stateflow/MATLAB code

During code generation, TargetLink now detects more constant elements in Stateflow or MATLAB code. This can result in code changes like the following:

- Different or simpler code for MATLAB `for` loops.
- Concatenations might now be implemented as a single vector/matrix assignment using an auxiliary constant instead of several individual assignments.
- More frequent folding of local MATLAB variables, e.g.:

```
localMATLABVar = 5;
out = 4 + localMATLABVar;
```

becomes:

```
out = 9;
```

Reason Code efficiency

Counter variables for Stateflow temp logic more efficient

The counter variable generated for Stateflow constructs such as `after`, `at`, `before`, and `any` is generated differently:

TargetLink ≤ 5.0	TargetLink 5.1

TargetLink always uses `Float64` (constrained to `UInt32`) as data type:
`if (Ca1_Chart_ctr < 4294967295U) {`
`  Ca1_Chart_ctr++;`
`}`
	TargetLink determines the data type from the Data Dictionary, e.g., `Int16`:
	`if (Ca1_Chart_ctr < 32767) {`
	`  Ca1_Chart_ctr++;`
	`}`

Reason

- Code efficiency
- Precision

For loops in Stateflow

With TargetLink 5.1, the conditions for a For loop generation in Stateflow are changed.

TargetLink ≤ 5.0

```
for (Ca1_i = 0; ((Int32) Ca1_i) < (((Int32) Ca1_LoopSize) * 5); Ca1_i = Ca1_i + 1)
{
   Ca1_F64Out = Ca1_F64Out + (Sa1_InPort1 + 42);
}
```

TargetLink 5.1

```
Ca1_i = 0;
while (((Int32) Ca1_i) < (((Int32) Ca1_LoopSize) * 5))
{
   Ca1_F64Out = Ca1_F64Out + (Sa1_InPort1 + 42);
   Ca1_i = (Int16) (Ca1_i + 1);
}
```

Reason Bug fix

Migration issue For the changed conditions, refer to For Loop (📖 TargetLink Preparation and Simulation Guide).

Binary bitwise operations in Stateflow

TargetLink now considers the TargetLink code generation target on binary bitwise operations in Stateflow®, i. e., TargetLink changes `U16Var` and `I16Var` to `Int32` if the code generation target is 32 bits.

TargetLink ≤ 5.0	TargetLink 5.1
`if ((UInt16) (Ca1_tlI16Var | Ca1_tlU16Var) > 0) { …`	`if ((Int32) (((Int32) Ca1_tlI16Var) | ((Int32) Ca1_tlU16Var)) > 0) { …`

Reason Bug fix

Migration issue When performing MIL-SIL comparisons, make sure you specify the same properties for `int` datatypes in Simulinks model configuration parameters on the Hardware Implementation page. This results in target-specific code. If generic code is required, avoid binary bitwise on 16 bit operands with mixed signedness. Remember that bitwise operations with signed operands are a MISRA-violation anyhow.

Variable Optimization

Optimized vector and matrix assignments

TargetLink now optimizes iterated vector and matrix access where the assigning loop copies a part of a larger vector or matrix variable to the intermediate variable.

TargetLink ≤ 5.0	TargetLink 5.1

`for (Aux_ = 0; Aux_ < 5; Aux_++)`
`{`
`  /* Selector: Sample/Selector */`
`  Sa1_Selector[Aux_] = Sa1_In1[Aux_ + 0];`
`}`
 	
`/* Rescaler: Sample/Data Type Conversion */`
`Sa1_Data_Type_Conversion = Sa1_In2 != 0;`
`for (Aux__a = 0; Aux__a < 5; Aux__a++)`
`{`
`  /* Logical: Sample/Logical Operator */`
`  Sa1_Logical_Operator[Aux__a] =`
`  (Sa1_Selector[Aux__a]) && Sa1_Data_Type_Conversion;`
`}`
 	
`for (Aux__b = 0; Aux__b < 5; Aux__b++)`
`{`
`  /* TargetLink outport: Sample/Out1 */`
`  Sa1_Out1[Aux__b] = Sa1_Logical_Operator[Aux__b];`
`}`
	`Sa1_Data_Type_Conversion = Sa1_In2 != 0;`
	`for (Aux_S32 = 0; Aux_S32 < 5; Aux_S32++)`
	`{`
	`  /* TargetLink outport: Sample/Out1`
	`  # combined # Logical: Sample/Logical Operator`
	`  # combined # Sum: Sample/Sum`
	`  # combined # Selector: Sample/Selector */`
	`  Sa1_Out1[Aux_S32] = (Sa1_In1[Aux_S32]) &&`
	`  Sa1_Data_Type_Conversion;`
	`}`

Reason Bug fix

Variable Scope

Changed initial scope

TargetLink now sets the initial scope to `global` for the following variables:

- Custom Code parameter variables
- Custom Code work variables

Subsequent optimization usually reduces the scope to the same or even lower scope as in previous TargetLink versions.

Reason Bug fix

Other

Removing includes from tl_defines_a.h

TargetLink now removes includes of `tl_defines_a.h` if the header file only defines variable class macros that are not used in production code.

Reason Bug fix

Dynamically assigning new values to data store variables

TargetLink now can dynamically assign new values to data store variables.

In these situations, the code patterns are changed as follows:

- Without code optimization

TargetLink ≤ 5.0

```
for (Aux__a = 0; Aux__a < 10; Aux__a++)
   {
      /* Assignment:
Subsystem/Run/Subsystem/Subsystem/Assignment: output
initialization */
      Sa3_Assignment[Aux__a] =
Sa3_Data_Store_Read[Aux__a];
   }

   /* Assignment:
Subsystem/Run/Subsystem/Subsystem/Assignment: output
calculation */
   Sa3_Assignment[Sa3_Rescaler1 - 1] = Sa3_Rescaler;
   for (Aux__b = 0; Aux__b < 10; Aux__b++)
   {
      /* Data store write: Subsystem/Run/Subsystem/Data
Store Write */
      Sa2_Data_Store_Memory[Aux__b] =
Sa5_Assignment[Aux__b];
   }
```

TargetLink 5.1

```
/* Assignment: Subsystem/Run/Subsystem/Assignment: output
calculation */
   Sa3_Assignment[Sa3_Rescaler1 - 1] = Sa3_Rescaler;

   /* Data store write: Subsystem/Run/Subsystem/Data
Store Write */
   Sa2_Data_Store_Memory[Sa3_Rescaler1 - 1] =
Sa3_Assignment[Sa3_Rescaler1 - 1];
```

- With code optimization

TargetLink ≤ 5.0

```
/* SLStaticLocal: Default storage class for static local
variables | Width: 16 */
   static SInt16 Sa4_Assignment[10] =
   {
      /* [0..9] */ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
      /* 0., 0., 0., 0., 0., 0., 0., 0., 0. */
   };

   for (Sa3_For_Iterator_it = 1; Sa3_For_Iterator_it
<= 3; Sa3_For_Iterator_it++)
   {
      /* Assignment:
Subsystem/Run/Subsystem/Subsystem/Assignment: output
calculation
         # combined # Rescaler:
Subsystem/Run/Subsystem/Rescaler */
      Sa4_Assignment[Sa3_For_Iterator_it - 1] = 5;
      for (Aux_S32 = 0; Aux_S32 < 10; Aux_S32++)
      {
         /* Data store write:
Subsystem/Run/Subsystem/Data Store Write */
         Sa2_Data_Store_Memory[Aux_S32] =
Sa4_Assignment[Aux_S32];
      }
   }
```

TargetLink 5.1

```
for (Sa3_For_Iterator_it = 1; Sa3_For_Iterator_it <= 3;
Sa3_For_Iterator_it++)
   {
      /* Data store write: Subsystem/Run/Subsystem/Data
Store Write
         # combined # Assignment:
Subsystem/Run/Subsystem/Assignment: output calculation
         # combined # Rescaler:
Subsystem/Run/Subsystem/Rescaler */
      Sa2_Data_Store_Memory[Sa3_For_Iterator_it - 1] = 5;
   }
```

Reason

- Code efficiency
- Code size

Migration issue To dynamically assign new values to data store variables, refer to Basics on Dynamically Assigning New Values to Data Store Variables (🕮 TargetLink Preparation and Simulation Guide).

Changed scope reduction of structure components

TargetLink now does not reduce the scope of structure components if any access to one of the parent structures of the component is located in the same function.

TargetLink ≤ 5.0	TargetLink 5.1
`Aux_S16 = 3;` `S4 = S3;`	`S3.bli = 3;` `S4 = S3;`

Reason Bug fix

Discontinuations as of TargetLink 5.1

Where to go from here

Information in this section

Discontinued TargetLink Features

DD ModuleTemplate

With TargetLink 5.1, DD ModuleTemplate objects have been removed. Existing DD ModuleTemplate objects are removed from the Data Dictionary without replacement.

The removal of DD ModuleTemplate objects was announced with TargetLink 5.0.

Demo models

The following demo models are discontinued with TargetLink 5.1:

- SF_USER_FUNCTION
- BLACKBOX

Obsolete Limitations

Array-of-struct	Signal injection is not supported for array-of-struct variables. Signal tunneling is not supported for array-of-struct variables.
Compatibility of SICs with different platforms	SICs built for Linux 64-bit (Generic_x86_64_Linux/GCC) are not compatible with dSPACE platforms.. TargetLink does not inform at SIC generation time if the generated SIC is compatible with at least one of the dSPACE simulation platforms consuming SIC files.

Changes in Future TargetLink Versions

Where to go from here

Information in this section

Features to Be Discontinued

RTOS/OSEK code generation mode	Support for the TargetLink RTOS/OSEK code generation modes will be discontinued in a future TargetLink version.
Target Optimizations Module (TOM)	Support for the TargetLink Target Optimizations Module (TOM) will be discontinued in a future TargetLink version.
Clean code and Do not log anything	Variables selected for logging cannot be fully optimized. When generating code with the Global logging option `Do not log anything` or `Log according to block data`, TargetLink does not fully optimize the code to facilitate testing. This means the code differs only with regard to the log macros. This contrasts the Clean code checkbox on the Code Generation page of the TargetLink Main Dialog block, which always activates full code optimization. The special `Do not log anything` behavior will be removed in a future TargetLink version.
Simulink classic initialization mode	Support for the Simulink classic initialization mode will be discontinued in a future TargetLink version.
Dynamic components	Support for specifying dynamic components for DD Variable objects will be discontinued ina future TargetLink version.
Automatic interpretation of Boolean	The automatic interpretation of certain integer data types as Boolean by TargetLink will be discontinued in a future TargetLink version.

Unit Delay Reset Enabled	The TargetLink support of the Unit Delay Reset Enabled block will be discontinued in a future TargetLink version.
Model change detection via checksum	The TargetLink support of model change detection via checksum will be discontinued in a future TargetLink version.
Sample blocks	The TargetLink support of Sample blocks will be discontinued in a future TargetLink version.

API Functions to Be Discontinued

Discontinued API functions

The following API functions are deprecated and will be removed in a future TargetLink version:

Function	Deprecated Since	Replacement Function
`tl_compare_fcn_signature`	TargetLink 5.0	-

Deprecated Code Generator Options

Deprecated Code Generator options

The following Code Generator options are deprecated and will be removed in future TargetLink versions:

- AllowStructAssignments (📖 TargetLink Model Element Reference)

.

VEOS

Where to go from here

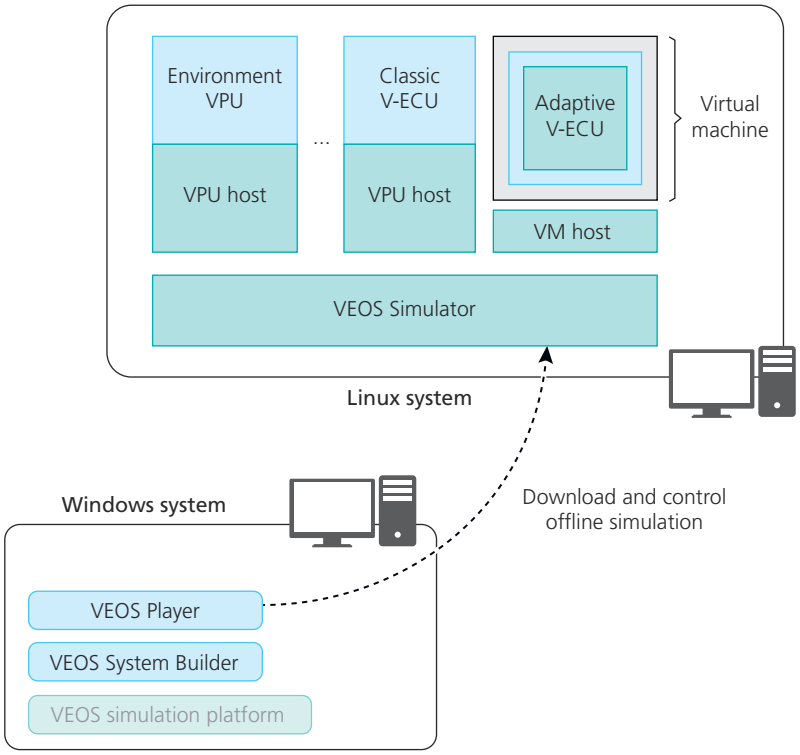
Information in this section

New Features of VEOS 5.1

Running offline simulations on Linux

You can now also run offline simulations with VEOS on Linux.

You can control an offline simulation on a Linux system either locally via command line or remotely via VEOS Player from a Windows system. Refer to the following illustration.



The option to build offline simulation applications for execution on a Linux operating system was already introduced with VEOS 5.0 (dSPACE Release 2020-A).

For more information, refer to Introduction to VEOS on Linux (🕮 VEOS Manual).

Support for the new VECU container file format

VEOS 5.1 supports the new *VECU container file format* for V-ECU implementation containers.

The VECU container file format is the successor to the CTLGZ format.

- The VECU file format *supports IP protection between ECU software suppliers and OEMs* because it lets suppliers integrate precompiled object and library files. Adding precompiled ECU code to a VECU container file also has the advantage that build times in the subsequent workflow are shortened.
- The VECU file format *provides multi-platform support*, i.e., it lets the user integrate simulation code for various simulation platforms in one VECU file, including descriptions of the relevant platform dependencies.

Using VEOS via command line

VEOS now provides a command line interface that lets you do the following without having to use the VEOS Player:

- Building VPUs and integrating them in an offline simulation application
- Loading and controlling an offline simulation application

For more information, refer to Basics on the VEOS Command Line Interface (🕮 VEOS Manual).

Support for AUTOSAR R19-11

VEOS 5.1 supports AUTOSAR R19-11. R19-11 is a combined release for the Classic and Adaptive Platform.

Support for the Ocu driver module of the microcontroller abstraction layer

VEOS 5.1 lets you build and simulate V-ECUs that contain a configuration and implementation of the Output Compare Unit (`Ocu`) driver module of the microcontroller abstraction layer (MCAL).

The `Ocu` driver module lets you compare the value of a counter with a defined threshold, and responds automatically when the threshold is reached.

Refer to *AUTOSAR_SWS_OCUDriver.pdf*. The document is available at www.autosar.org.

Related topics

Basics

Basics on the VEOS Command Line Interface (VEOS Manual)
Introduction to VEOS on Linux (VEOS Manual)

Compatibility of VEOS 5.1

General compatibility

dSPACE recommends using only software products from the same dSPACE Release. This ensures maximum run-time compatibility.

Supported compiler versions

For information on supported compiler versions, refer to Basics on Integrating the Simulation System (VEOS Manual).

Supported operating systems

Windows For information on the Windows operating systems supported by VEOS, refer to Operating System on page 236.

Linux For information on the Linux distribution recommended for VEOS, refer to Introduction to VEOS on Linux (VEOS Manual).

BSC file compatibility

VEOS 5.1 is compatible with bus simulation container (BSC) files created with the Bus Manager of dSPACE Release 2020-B (BSC version 1.9).

- If a BSC file was generated *without an SIC file*, or *with an SIC file that was generated for the dsrt.tlc system target file*, you can integrate the BSC file in a VEOS simulation on Windows.
- If a BSC file was generated *with an SIC file that was generated for the dsrt64.tlc system target file*, you can integrate the BSC file in a VEOS simulation on Linux.

CTLGZ/VECU file compatibility

The following table shows the compatibility between VEOS 5.1 and V-ECU implementation container (CTLGZ/VECU) files:

V-ECU Implementations Created With...	V-ECU Implementation Version
dSPACE Release 2020-B: ▪ SystemDesk 5.5 ▪ TargetLink 5.1	3.0[1)] As of version 3.0, *VECU* is the new V-ECU implementation file format.
dSPACE Release 2020-A: ▪ SystemDesk 5.4	2.10[2)]
dSPACE Release 2019-B: ▪ SystemDesk 5.4 ▪ TargetLink 5.0	2.10[2)]
dSPACE Release 2019-A: ▪ SystemDesk 5.3	2.9[3)]

1) You can integrate a VECU file of this version in a VEOS simulation on Windows and in a VEOS simulation on Linux.
2) You can integrate a CTLGZ file of this version in a VEOS simulation on Windows and in a VEOS simulation on Linux.
3) You can integrate a CTLGZ file of this version in a VEOS simulation on Windows, but not in a VEOS simulation on Linux.

FMU file compatibility

- VEOS supports Functional Mock-up Units (FMUs) that comply with the FMI 2.0 standard.
- VEOS supports only the *FMI for Co-Simulation interface*, but not the FMI for Model Exchange interface.

You can integrate an FMU file in a VEOS simulation on Windows and in a VEOS simulation on Linux.

For detailed and up-to-date compatibility information on the dSPACE FMI support, refer to:

http://www.dspace.com/go/FMI-Compatibility.

OSA file compatibility

The following table shows the compatibility between VEOS 5.1 and offline simulation application (OSA) files:

OSA Files Created with Products Of ...	OSA Version
dSPACE Release 2020-B	5.1
dSPACE Release 2020-A	5.0[1)]

OSA Files Created with Products Of ...	OSA Version
dSPACE Release 2019-B	4.5[1)]
dSPACE Release 2019-A	4.4[1)]

1) You cannot modify the properties of VPUs contained in an OSA file if you open the OSA file in a later VEOS version than the version with which the OSA file was originally created. However, port and network connections can be edited. As a consequence, it is recommended to rebuild the binary OSA files from existing model implementation container files (CTLGZ, SIC, BSC, FMU) when you migrate from one VEOS version to another.

SIC file compatibility

The following table shows the compatibility between VEOS 5.1 and Simulink implementation container (SIC) files:

SIC Files Created With ...	SIC Version
dSPACE Release 2020-B: ▪ Model Interface Package for Simulink 4.4 ▪ TargetLink 5.1	1.9[1), 2)]
dSPACE Release 2020-A: ▪ Model Interface Package for Simulink 4.3	1.8[1), 2)]
dSPACE Release 2019-B: ▪ Model Interface Package for Simulink 4.2 ▪ TargetLink 5.0	1.7[1), 2)]
dSPACE Release 2019-A: ▪ Model Interface Package for Simulink 4.1	1.6[1), 3)]

1) If the SIC file is created with a previous dSPACE Release and if the SIC file contains an ASM model, you cannot simulate the model in VEOS 5.1 (dSPACE Release 2020-B). For more information, refer to Migrating ASM Models (🕮 VEOS Manual).

2) If an SIC file of this version was generated for the `dsrt.tlc` system target file, you can integrate it in a VEOS simulation on Windows. If an SIC file of this version was generated for the `dsrt64.tlc` system target file, you can integrate it in a VEOS simulation on Linux. Refer to How to Generate a Simulink Implementation Container (🕮 Model Interface Package for Simulink - Modeling Guide).

3) You can integrate an SIC file of this version in a VEOS simulation on Windows, but not in a VEOS simulation on Linux.

SMC file compatibility

The following table shows the compatibility between VEOS 5.1 and system model container (SMC) files:

SMC Files Created With ...	SMC Version
dSPACE Release 2020-B: ▪ SYNECT 2.10 ▪ VEOS 5.1	1.2
dSPACE Release 2020-A: ▪ SYNECT 2.9 ▪ VEOS 5.0	1.2
dSPACE Release 2019-B: ▪ SYNECT 2.8 ▪ VEOS 4.5	1.1

SMC Files Created With ...	SMC Version
dSPACE Release 2019-A: ▪ SYNECT 2.7 ▪ VEOS 4.4	1.1

You also have to consider the following compatibility restrictions of the individual container files contained in the SMC file to be imported: If the SMC file contains a container of an unsupported version, VEOS 5.1 imports neither the unsupported container nor the connections to the application process based on the unsupported container.

Real-Time Testing compatibility

To use RTT in connection with VEOS and ControlDesk, the Real-Time Testing (RTT) version used by the VEOS Simulator that runs the simulation system and the RTT version that is active on the PC must be identical.

The following table shows the VEOS Simulator version and the corresponding RTT version:

VEOS Simulator	RTT Version
... from VEOS 5.1	Real-Time Testing Version 4.4
... from VEOS 5.0	Real-Time Testing Version 4.3
... from VEOS 4.5	Real-Time Testing Version 4.2
... from VEOS 4.4	Real-Time Testing Version 4.1
... from VEOS 4.3	Real-Time Testing Version 4.0
... from VEOS 4.2	Real-Time Testing Version 3.4
... from VEOS 4.1	Real-Time Testing Version 3.3
... from VEOS 4.0	Real-Time Testing Version 3.2

ControlDesk 7.3 automatically uses the VEOS Simulator of VEOS 5.1. You can therefore use RTT in connection with VEOS and ControlDesk if RTT 4.3 is active on the PC.

Simulation of adaptive V-ECUs

For compatibility information regarding the simulation of adaptive V-ECUs in VEOS, refer to Compatibility Requirements for the Simulation of Adaptive V-ECUs (🕮 VEOS Manual).

Related topics

Basics

Compatibility Requirements for the Simulation of Adaptive V-ECUs (🕮 VEOS Manual)

Migrating to VEOS 5.1

Introduction

To migrate from VEOS 5.0 to VEOS 5.1, you might have to perform certain migration steps.

Note

To migrate to VEOS 5.1 from versions earlier than 5.0, you might also have to perform the migration steps of the intervening VEOS versions.

Migrating ASM models

You cannot simulate an ASM model on VEOS if the model is contained in an OSA or SIC file that was created with a dSPACE Release earlier than the one to which your VEOS installation belongs.

To simulate an ASM model that was last saved with a dSPACE Release earlier than the dSPACE Release to which your VEOS version belongs, perform the following steps:

1. Migrate the ASM model to the dSPACE Release to which your VEOS version belongs.

   For information on migrating ASM models, refer to Migrating ASM Models (🕮 ASM User Guide).
2. Generate a Simulink implementation container (SIC) file on the basis of the ASM model by using the *Model Interface Package for Simulink*.

   For instructions, refer to Generating Simulink Implementation Containers (🕮 Model Interface Package for Simulink - Modeling Guide).
3. Import the SIC file to the VEOS Player of your VEOS version.

   For instructions, refer to How to Import Simulink Implementations (🕮 VEOS Manual).

Migrating from earlier VEOS versions

To migrate from earlier VEOS versions and reuse existing offline simulation applications, you might have to carry out additional migration steps. For more information on the migration steps, refer to Migrating from Prior Versions of VEOS (🕮 VEOS Manual).

Compatibility Information

Where to go from here

Information in this section

Supported MATLAB Releases

MATLAB®/Simulink®

Working with various dSPACE products requires that MATLAB is installed on your host PC.

Tip

For system requirements of MathWorks® software, refer to http://www.mathworks.com/support/sysreq.html.

MATLAB Release...	...Is Supported by dSPACE Release 2020-B					
	RCP and HIL Software [1), 2)]	AutomationDesk 6.4 [3)]	TargetLink 5.1	Model Compare 3.1	dSPACE Python Extensions 3.4 [4)]	XIL API .NET MAPort 2020-B
R2020b	✓ [5)]	✓	✓	✓	✓	✓
R2020a	✓	✓	✓	✓	✓	✓
R2019b	✓	✓	✓	✓	✓	✓
R2019a	✓	✓ [6)]	✓	✓ [6)]	✓ [7)]	✓

1) 'RCP and HIL software' is a generic term for a software package containing several dSPACE software products, for example, ASM, RTI, ConfigurationDesk, and ModelDesk. These software products are installed in a common folder.
2) MATLAB/Simulink Student Suite does not support Automotive Simulation Models (ASM).
3) The AutomationDesk MATLAB Access Library requires MATLAB.
4) matlablib2 of dSPACE Python Extensions requires MATLAB.
5) R2020b is not supported by the RTI FPGA Programming Blockset – FPGA Interface.
6) R2019a is only supported by Model Compare and the MATLAB Access Library in AutomationDesk if at least R2019a Update 5 is used.
7) R2019a is only supported by the matlablib2 if at least R2019a Update 5 is used.

For up-to-date information on additional MATLAB releases that can be used in combination with dSPACE software, refer to http://www.dspace.com/go/MATLABCompatibility.

Operating System

Operating system on host PC

The dSPACE products of dSPACE Release 2020-B support the following operating systems:

- The following editions, channels, and servicing options of Windows 10:
  - Windows 10 Professional, Education, and Enterprise (64-bit versions)

The Windows 10 Home, Mobile, and Windows 10 S editions are not supported.

- Long-Term Servicing Branch: LTSB 2016
- Long-Term Servicing Channel: LTSC 2019
- Semi Annual Channel (formerly known as Current Branch (CB)): The compatibility statement of Microsoft applies. This means that newer versions released in this channel should be compatible with all previous versions. dSPACE used the 1909 version of the Semi Annual Channel for testing.

- Windows Server 2016 Standard and Datacenter edition, each with the Desktop Experience installation option

  Only the listed editions are supported. The Windows Server 2016 Essentials, MultiPoint Premium Server editions are not supported.
- Windows Server 2019 Standard and Datacenter editions, each with the Desktop Experience installation option

  Only the listed editions are supported. The Windows Server 2019 Essentials edition is not supported.

Some limitations apply when you use dSPACE software in conjunction with features of Windows. Refer to Limitations for Using Windows Features on page 244.

Linux operating system on host PC

The dSPACE products of dSPACE Release 2020-B with Linux compatibility support the following operating systems:

- Ubuntu 18.04 LTS with the General Availability Kernel in the Desktop, Server, and Cloud version

Some limitations apply when you use dSPACE software in conjunction with features of Linux. Refer to Limitations for Using Linux Features on page 245.

Using MicroAutoBox Embedded PC as host PC

ControlDesk can also be installed on:

- MicroAutoBox Embedded PC 6th Gen. Intel® Core™ i7-6822EQ Processor, running on Microsoft® Windows® 10 IOT Enterprise, LTSB 2016, 64-bit version
- MicroAutoBox III Embedded PC, running on

  Microsoft® Windows® 10 IoT Enterprise LTSC 2019, 64-bit version

Operating system on SYNECT server

The SYNECT server supports the following operating systems:

- The same operating systems as listed above for all dSPACE products of dSPACE Release 2020-B.
- Windows Server 2012, Windows Server 2012 R2

Operating system on server for floating network licenses

If you purchased floating network licenses, you have to specify one of the network PCs as a license server. Every PC with CodeMeter Runtime software can be used as a license server.

Valid for servers without dSPACE software dSPACE tests license servers only with Microsoft Windows operating systems in combination with protected dSPACE software.

Note

Non-Windows operating systems, such as Ubuntu Linux, are not tested. You can use them at your own risk. dSPACE does not provide support in this case.

Valid for servers with dSPACE Installation Manager dSPACE Installation Manager supports the same operating systems as the other dSPACE software products described above.

Allowing communication

Installing of additional firewall rules Additional Windows firewall rules are installed during the installation of various dSPACE software products. For example, one rule allows communication with a dSPACE expansion box, such as AutoBox. Another rule allows MotionDesk to receive motion data from a network channel. These example rules are created by the following commands:

- `netsh advfirewall firewall add rule name="dSPACE Net Service" service=any dir=in action=allow profile=any protocol=icmpv4:0, any description="Allow the dSPACE Net Service to connect to a dSPACE expansion box via network."`
- `netsh advfirewall firewall add rule name="dSPACE MotionDesk" program=<main installation path>\dSPACERCPHIL2020-B\MotionDesk\Bin\MotionDesk.exe" dir=in action=allow profile=any description="Allow dSPACE MotionDesk to receive motion data via network."`

Required open TCP/IP network ports If you are using third-party firewall software on your host PC, ensure that the TCP/IP communication of dSPACE software is not blocked:

- VEOS requires the following open TCP/IP network ports: 111 (TCP and UDP), 3702 (UDP), 7214 (TCP and TCP6), 8090 (TCP), 9923 (UDP), 15000 (UDP), 49152 ... 65535 (TCP, TCP6 and UDP)
- dSPACE Installation Manager and CodeMeter licensing software require the following open TCP/IP network ports:
  - 22350 (TCP and UDP) for communication in a LAN network (if not changed from the default setting).
  - 22352 (TCP and UDP): To access CodeMeter WebAdmin via http.
  - 22353 (TCP and UDP): To access CodeMeter WebAdmin via https.
- dSPACE Help requires an open TCP/IP network port for interprocess communication between its components. The default port number is 11000. If this port number is already being used, another free port is used automatically. The related processes can be identified via the following prefixes: `HelpAbsLayer<xxx>`, `HelpInstaller<xxx>`.

Using dSPACE Software on Virtual Machines (VMs)

Introduction

As of dSPACE Release 2019-A, you can operate several dSPACE products installed on virtual machines. However, some dSPACE products support VMs only with limitations, and other dSPACE products cannot be operated on VMs at all.

Usage restrictions

Note

The dSPACE End User License Agreement (EULA) prohibits:

- Using a virtual machine for circumventing license protection mechanisms, for multiple use of an acquired license or for use outside the use determined by the license type.
- Accessing dSPACE software via Internet or network applications (e.g., Citrix, Microsoft Remote Desktop or other terminal/device servers) or to grant such access to third parties.

If you have any questions or encounter any problems, contact dSPACE Support (www.dspace.com/go/supportrequest).

Recommended virtual machine software

dSPACE tests the functionality of dSPACE software products with current VMware products and VM hardware compatibility version 10 and version 13.

You can use Windows, Linux, or macOS® as the host operating system.

Support of dSPACE software on virtual machines

Note

The following table shows the compatibility for all dSPACE products. For products that support VMs with limitations, the known limitations are listed. For these products, further limitations might apply depending on the use case.

Product	Full Support	Support with Known Limitations	No Support
ASM	✓	—	—
AutomationDesk	—	✓ Known limitations: ▪ Access to DS1006 modular systems via dSPACE link boards is not possible. ▪ Access to DS1006 modular systems via Ethernet connection and slot CPU: Communication and therefore performance is very low. ▪ Access to DS1104 R&D Controller Boards is not possible.	—
Bus Manager	✓	—	—
ConfigurationDesk for RapidPro	✓	—	—
ConfigurationDesk	✓	—	—

Product	Full Support	Support with Known Limitations		No Support
Container Manager	✓	—		—
ControlDesk	—	✓	Known limitations: ▪ Access to DS1006 modular systems via dSPACE link boards is not possible. ▪ Access to DS1006 modular systems via Ethernet connection and slot CPU: Communication and therefore performance is very low. ▪ Access to DS1104 R&D Controller Boards is not possible.	—
Data Dictionary Manager	✓	—		—
DCI-GSI Configuration Package	✓	—		—
dSPACE AUTOSAR Compare	✓	—		—
dSPACE Installation Manager	✓	—		—
ECU Flash Programming Tool	✓	—		—
ECU Interface Base Package	✓	—		—
ECU bypassing target compiler	✓	—		—
Failure Simulation Package	—	✓	Supported only in combination with the VEOS platform. Combinations with other platforms are not tested and therefore not released for use on VMs.	—
Firmware Archives Firmware Manager	—	✓	Known limitations: ▪ Access to DS1006 modular systems via dSPACE link boards is not possible. ▪ Access to DS1006 modular systems via Ethernet connection and slot CPU: Communication and therefore performance is very low. ▪ Access to DS1104 R&D Controller Boards is not possible.	—
FlexRay Configuration Tool	✓	—		—
Model Compare	✓	—		—
ModelDesk	—	✓	Known limitations: ▪ The Traffic Object Manager cannot show custom sensor points in the preview. ▪ Plotting occasionally does not start if a start trigger is used.	—
Model Interface Package for Simulink	✓	—		—
MotionDesk	—	—		✓ [1)]
Platform API Package	—	✓	Known limitations: ▪ Access to DS1006 modular systems via dSPACE link boards is not possible. ▪ Access to DS1006 modular systems via Ethernet connection and slot CPU: Communication and therefore performance is very low. ▪ Access to DS1104 R&D Controller Boards is not possible.	—

Product	Full Support	Support with Known Limitations		No Support
Real-Time Testing	—	✓	Known limitations: ▪ Access to DS1006 modular systems via dSPACE link boards is not possible. ▪ Access to DS1006 modular systems via Ethernet connection and slot CPU: Communication and therefore performance is very low. ▪ Access to DS1104 R&D Controller Boards is not possible.	—
RTI Blocksets (Real-Time Interface)	—	✓	Known limitations: ▪ Access to DS1006 modular systems via dSPACE link boards is not possible. ▪ Access to DS1006 modular systems via Ethernet connection and slot CPU: Communication and therefore the performance is very low. ▪ Access to DS1104 R&D Controller Boards is not possible.	—
Sensor Simulation	—	—		✓[1]
SYNECT	✓	—		—
SYNECT Server	✓	—		—
SYNECT License Server	✓	—		—
SystemDesk	✓	—		—
TargetLink	✓	—		—
VEOS	✓[2]	—		—

[1] VMs do not fulfill the requirements for graphics adapters.

[2] If you want to simulate adaptive AUTOSAR V-ECUs, refer to Compatibility Requirements for the Simulation of Adaptive V-ECUs (🕮 VEOS Manual).

Required knowledge for setting up a virtual machine

To set up a virtual machine, you must have knowledge about the technology of VMs.

In virtual environments, significantly higher latencies and lower network performance (network throughput) must be expected compared to physical PCs. dSPACE has no influence on this.

Using virtual machines in parallel If you use multiple VMs simultaneously on one PC, sharing of host resources such as CPU, network and disk I/O bandwidth can cause timing issues. dSPACE recommends to use a physical PC if high performance is required by an application.

Using the 'Revert to snapshot' feature

NOTICE

Using the 'Revert to snapshot' feature causes invalid licenses.

If you use the 'Revert to snapshot' feature in a VM, all software-based CmContainers on your PC (dSPACE Activation Container and/or dSPACE Borrow Container) become invalid and the contained licenses are lost. This is not the case if the license information is stored on CmDongles.

- Do not use the 'Revert to snapshot' feature for VMs containing software-based CmContainers with activated licenses.

Using a virtual machine on the host PC

System requirements PCs that host virtual machines with dSPACE software, must meet at least the requirements listed in Appendix: System Requirements (🕮 Installing dSPACE Software). You are recommended to use a PC with more resources so that the software runs smoothly on a VM, because the VM software itself uses up some of the resources:

- The CPU speed and RAM size must be sufficient to run the operating system and the software on the host PC as well as the guest operating system and the application software on the VM.
- You also require sufficient free disk space to install the VM software and the software you want to run, just as you would if you were installing it directly on your PC.

Connecting dongle-based devices If you use dongle-based single-user licenses to use dSPACE software, you first have to connect your CmDongle to the host PC. Then you have to connect the **WIBU-Systems CodeMeter-Stick** device to the virtual machine on the host PC. For instructions, refer to the documentation of the VM software you use.

Using floating network licenses If you use floating network licences, the virtual machine requires access to the dSPACE License Server. For further instructions, refer to How to Set up a Connection Between Client and Server (🕮 Working with CodeMeter Licensing Technology).

Optimal display of dSPACE Help For an optimal display of the content in dSPACE Help, you have to activate the **ClearType** setting in the VM (= default setting).

You can access this setting via the Windows Start menu (**Start – Control Panel – Appearance and Personalization – Display – Adjust ClearType text**).

Run-Time Compatibility of dSPACE Software

Definition

Run-time compatibility means that:

- dSPACE products can be used in parallel after software installation, even if they are installed in different folders.

- dSPACE products without interaction can run independently of each other.

Compatibility of products in dSPACE Release 2020-B

dSPACE recommends using only software products from the same dSPACE Release. This ensures maximum run-time compatibility.

Observe the following points:

- Limitations regarding run-time compatibility in the dSPACE tool chain might occur if products from different dSPACE Releases are used together.

  If dSPACE products interact directly (through automation interfaces) or indirectly (through common file types like A2L), limitations might apply. For minor limitations, refer to the relevant product documentation. The major limitations are described in the following.

  In rare cases, an additional patch must be installed for a product to achieve run-time compatibility. For more information on the patch and whether a patch is required, refer to http://www.dspace.com/go/CompPatch.
- RCP and HIL software products (of Release 2020-B) cannot be used in combination with RCP and HIL software products from earlier dSPACE Releases.

Major limitation for working with a SCALEXIO system and with MicroAutoBox III The products for working with a SCALEXIO system and with MicroAutoBox III must be compatible. This is guaranteed only for products delivered with the same dSPACE Release. Contact dSPACE for more information.

Compatibility of real-time applications loaded to a DS1006, DS1104 or MicroAutoBox II platform If a real-time application is loaded to one of these platforms with a software product of dSPACE Release 2016-B or later, software products of dSPACE Release 2016-A (and earlier) do not detect that the loaded real-time application is the same as the real-time application stored on your host PC. In this case, you cannot work with the related software product without restrictions.

This also applies if you load a real-time application with a software product of dSPACE Release 2016-A or earlier and use software products of dSPACE Release 2016-B or later, for example, for experimenting.

Combining dSPACE products from earlier Releases

For more information and notes on the combined use of different products from and with earlier Releases, refer to http://www.dspace.com/go/ds_sw_combi.

Limitations for Using Windows Features

Motivation	Some limitations apply to using dSPACE software in conjunction with features of Windows.
Installing and running dSPACE software within the Windows service accounts	Non-service-based dSPACE software is not designed to be installed or run in the context of any predefined Windows service account (LocalService, NetworkService, LocalSystem).
Fast user switching not supported	dSPACE software does not support the fast user switching feature of Windows.
Closing dSPACE software before PC shutdown	The shutdown process of Windows operating systems might cause some required processes to be aborted although they are still being used by dSPACE software. To avoid a loss of data, it is recommended to close the dSPACE software manually before shutting down the PC.
User Account Control	It is recommended to disable the Windows User Account Control (UAC) during the installation of dSPACE software. If you cannot disable UAC, note the following Windows behavior: If UAC is enabled, the setup programs use the administrator account instead of the user account. Therefore, it is important that the administrator account has access to the required drives, particularly the required network drives.
USB devices	If you connect dSPACE USB devices that use cables with optoisolation to the PC for the first time, there might be a message that the device driver software was not installed successfully. However, the dSPACE device will work properly later on.
Using 4K monitors	The following dSPACE software products have limitations for working with 4K monitors: ▪ SYNECT: If you use 4K monitors, the SYNECT graphical user interface might not be displayed properly in some cases, but this does not cause functional limitations. ▪ Real-Time Testing: The Real-Time Test Manager, the user interface for handling RTT sequences, does not support working with 4K monitors. ▪ FlexRay Configuration Package: The FlexRay Configuration Tool does not support working with 4K monitors.

FIPS support

dSPACE software was not developed for or tested against the FIPS PUB 140-2 U.S. government computer security standard (Security Requirements for Cryptographic Modules). For more information on FIPS, refer to https://technet.microsoft.com/en-us/library/security/cc750357.aspx.

Long paths

dSPACE software does not support the long path syntax of the Windows API. If a path that exceeds 260 characters is used directly or indirectly, the behavior of the dSPACE software is not defined.

Enabling Windows 8dot3name creation option

Note

It is strongly recommended that the Windows 8dot3name creation option is enabled for all drives (drives used for installation and drives used for work) before you install third-party software, such as MATLAB®/Simulink®, and the dSPACE software.

If the option is disabled during software installation, serious errors can occur when you run the dSPACE software. For example, the build process might be aborted. To repair an installation that was installed while the 8dot3name creation option was disabled, you have to install dSPACE software and required third-party software again.

For instructions on checking the setting and enabling the option, refer to http://www.dspace.com/faq?346 or to the Microsoft Windows documentation.

Settings in Windows for user locale and system locale must match

MATLAB reads the user locale and system locale settings that are specified in Windows operating systems. The user locale and the system locale must match. If these settings are not the same, the system might not behave as expected when working with MATLAB and dSPACE software.

For instructions on checking and changing the settings, refer to https://www.mathworks.com/help/matlab/matlab_env/setting-locale-on-windows-platforms.html?s_tid=gn_loc_drop.

This affects all MATLAB versions and all Windows operating systems, that are supported by dSPACE.

Limitations for Using Linux Features

FIPS support

dSPACE software was not developed for or tested against the FIPS PUB 140-2 U.S. government computer security standard (Security Requirements for Cryptographic Modules). For more information on FIPS, refer to https://ubuntu.com/blog/fips-certification-ubuntu-18-04-lts.

Long paths	dSPACE software does not support the available path length of 4,096 characters. If a path that exceeds 260 characters is used directly or indirectly, the behavior of the dSPACE software is not defined.
Locale	dSPACE software was tested only on a system with a US English locale.