dSPACE Release

New Features and Migration

Release 2022-B – November 2022

dSPACE

How to Contact dSPACE

Mail:	dSPACE GmbH Rathenaustraße 26 33102 Paderborn Germany
Tel.:	+49 5251 1638-0
E-mail:	info@dspace.de
Web:	https://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 have to add your contact details manually.

If possible, always provide 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 the software updates and patches themselves and for more information, such as how to receive an automatic notification when an update or a patch is available for your dSPACE software.

Important Notice

Contents

About This Document

Content

This document informs you about the new features of all the dSPACE software products in Release 2022-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:
https://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.
	Follows 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 Windows folders

Windows-based software products use the following special folders:

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

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

or

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

Documents folder A standard folder for application-specific files that are used by the current user.

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

Local Program Data folder A standard folder for application-specific program 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 Windows-based dSPACE software, the documentation for the installed products is available in dSPACE Help and as 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**

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

dSPACE Help (Web) Independently of the software installation, you can access the Web version of dSPACE Help at https://www.dspace.com/go/help.

To access the Web version, you must have a *mydSPACE* account.
For more information on the mydSPACE registration process, refer to www.dspace.com/faq?097.

Overview of dSPACE Release 2022-B

Introduction

Gives you an overview of the new key features in Release 2022-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.

Providing legal notes for using third-party software

Several dSPACE software products 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

dSPACE Release 2022-B contains a Python 3.9 distribution with the following packages.

Package	Python 3.9 Release 2022-B
comtypes	1.1.11
Core	3.9.13
cycler	0.11.0
fonttools	4.33.3
future	0.18.2
grpcio	1.43.0
grpcio_tools	1.43.0
kiwisolver	1.4.3
lxml	4.9.0
matplotlib	3.5.2
numpy	1.23.0
packaging	21.3
pillow	9.1.1
pip	22.1.2
protobuf	3.19.3
pycparser	2.21
pyglet	1.5.26
pyparsing	3.0.9
pypubsub	4.0.3
Python-dateutil	2.8.2
pythonnet	2.5.3[1)]
pytz	2022.1
pywin32	302.10[1)]
scipy	1.8.1
six	1.16.0
wxPython	4.1.1
yapsy	1.12.2

[1)] This package contains some dSPACE-specific bugfixes. It should not be replaced by the package from the standard Python package servers.

Note

Notes when using AutomationDesk or ModelDesk

The dSPACE Release 2022-B contains a Python 3.9 distribution with the latest Python packages to provide bugfixes and security updates for your applications. In the new `numpy` package, some deprecated methods, e.g., the `alen` method and the `asscalar` method, have been removed.

The affected dSPACE products are AutomationDesk and ModelDesk with its testing feature. With dSPACE Release 2022-B, these products are prepared to use the new `numpy` package. However, if you installed the Python 3.9 distribution from dSPACE Release 2022-B and you use AutomationDesk or ModelDesk Testing from dSPACE Release 2021-A, 2021-B, or 2022-A, these products might throw an exception or stop with an error if one of the removed methods is called internally.

dSPACE will provide patches for the affected versions of AutomationDesk to solve this problem. Visit https://www.dspace.com/go/patches for the software updates and patches themselves and for more information, such as how to receive an automatic notification when an update or a patch is available for your dSPACE software.

If you need a patch for one of the above mentioned versions of ModelDesk, contact dSPACE Support.

New versioning scheme

The new versioning used for dSPACE products corresponds to the principles and recommendations of agile software development. The calender-based versioning scheme consists of the year and a consecutive number. For example, *22.3* stands for the third product version released in 2022.

For intermediate versions, the third digit of the version can be used. Patches and hotfixes are marked with the suffix *p* respectively *HF* with a following number. For example, the following product versions can appear: *22.2.1*, *22.2p1*, or *22.2HF12345*.

Release products additionally have the Release version as the main identifier of the product version, for example, *2022-B*. This allows the user to identify the dSPACE Release with which the current product version was first released.

The Release version is also used for the API of a Release product if version information is required. For example, to instantiate a not-registered COM server you have to call the COM server with a specific version, such as *ControlDesk.Application.2022-B*.

Platform support

SCALEXIO systems now have a 64-bit Linux operating system. This is supported by all related products.

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:

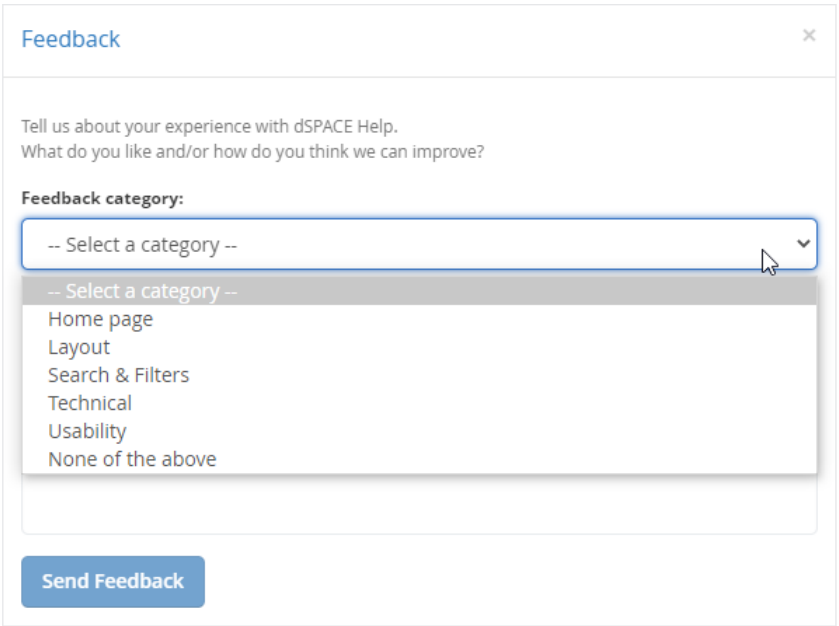
- MinGW (GNU Compiler Collection (GCC 6.3.0))
- Microsoft Visual Studio 2019 Professional

New Features of dSPACE Help

Submitting general feedback on dSPACE Help

You can submit general feedback on dSPACE Help via the Feedback button at the bottom of the screen.

Refer to the following illustration (for an animated graphic, refer to 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 https://www.dspace.com/go/discontinuation.

Discontinuation of software support

Operating system As of dSPACE Release 2022-B, Ubuntu 18.04 LTS is no longer supported.

dSPACE CAN API and further bus-specific APIs As of dSPACE Release 2022-B, the dSPACE CAN API and further bus-specific APIs are no longer available for 32-bit applications.

Support for the 64-bit versions of these APIs is not affected and will be continued.

Sensor simulation with radar, lidar, and laser sensors As of dSPACE Release 2022-B, Sensor Simulation based on MotionDesk with radar, lidar, and laser sensors is discontinued.

Planned discontinuation of software support

MotionDesk and Sensor Simulation

- Sensor Simulation based on MotionDesk

  Sensor Simulation based on MotionDesk for all sensor types will be discontinued after Release 2022-B. For more information, refer to https://www.dspace.com/en/pub/home/support/pli/elas/elassw/elasensim.cfm.

- MotionDesk

  MotionDesk will be discontinued after Release 2023-B. For more information, refer to https://www.dspace.com/go/elamd.

AURELION dSPACE AURELION is a new product and the successor of MotionDesk and Sensor Simulation for ADAS/AS simulations with camera, radar, and lidar sensors to validate driving functions and sensor output.

For more information and to prepare for your migration to AURELION, refer to https://www.dspace.com/en/pub/home/support/kb/faqs/faq433.cfm or contact dSPACE Support.

Discontinuation of dSPACE hardware

With Release 2022-A, software support is discontinued for the following product:

- SCALEXIO Processing Unit with the Intel® Core™ i7-860 mainboard (Real-Time PC Version 1.0).

  For new projects, we recommend that you use another variant of the SCALEXIO Processing Unit. Refer to Variants of the SCALEXIO Processing Unit (SCALEXIO Hardware Installation and Configuration 🕮).

Planned discontinuation of dSPACE hardware

PHS-bus hardware In 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, were discontinued. This also applies 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.

MicroAutoBox II MicroAutoBox II with all its variants (1401/1507, 1401/1511, 1401/1511/1514, 1401/1513, and 1401/1513/1514) will be discontinued at the end of 2027. You can buy the product until the end of 2024 and the software support will be continued up to dSPACE Release 2026-B.

For new projects, we recommend that you use MicroAutoBox III.

Planned discontinuation of legacy licensing technologies

As of dSPACE Release 2023-A, dSPACE software will no longer support legacy licensing technologies (WibuKey from Wibu-Systems for dongle licenses and FlexNet from Flexera for floating network licenses). This means:

- dSPACE Release 2023-A and later will support only the CodeMeter licensing technology introduced with dSPACE Release 2017-B.
- As of dSPACE Release 2023-A, the dSPACE Installation Manager will no longer provide the file download (keys.dsp and license.dsp) for legacy licensing based on CodeMeter licenses.
- dSPACE Release 2023-A cannot be installed on the same PC as dSPACE Release 2017-A or earlier.
- As of dSPACE Release 2023-A, dSPACE License Manager (Legacy) will no longer be shipped and installed with the dSPACE Installation Manager. The associated command line utilities are also discontinuated, including in particular the `IMLicUtil.exe` and the `InstallationReporter.exe` utilities.

Planned discontinuation of the dongle migration with dSPACE Installation Manager

As of dSPACE Release 2023-A, dSPACE Installation Manager will no longer support migration from old dongles (delivered for dSPACE Release 2017-A and earlier) to currently delivered CmDongles.

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			
	2021-A	2021-B	2022-A	2022-B[1)]
AutomationDesk	6.5	6.6	6.7	2022-B Refer to AutomationDesk on page 29.
Automotive Simulation Models	9.7	10.0	10.1	2022-B Refer to Automotive Simulation Models (ASM) on page 33.
Bus Manager (stand-alone)	6.7	6.8	22.1	2022-B Refer to Bus Manager (Stand-Alone) on page 45.
ConfigurationDesk for RapidPro	6.7	–[2)]	–	–

Product	dSPACE Release			
	2021-A	2021-B	2022-A	2022-B[1)]
ConfigurationDesk	6.7	6.8	22.1	2022-B Refer to ConfigurationDesk on page 51.
Container Manager	5.2	5.2	5.2	–
ControlDesk	7.4	7.5	7.6	2022-B Refer to ControlDesk on page 63.
DCI Configuration Tool	3.13	3.13	3.13	2022-B Refer to DCI Configuration Tool on page 79.
dSPACE AUTOSAR Compare	1.0	1.1	1.1	2022-B Refer to dSPACE AUTOSAR Compare on page 75.
dSPACE CAN API Package	4.0.6	4.0.7	4.0.8	2022-B
dSPACE ECU Flash Programming Tool	2.8	2.8	2.8	2022-B
dSPACE FlexRay Configuration Package	4.7	4.8	4.9	2022-B Refer to dSPACE FlexRay Configuration Package on page 81.
dSPACE Installation Manager	5.7	5.8	5.9	22.2 Refer to dSPACE Installation Manager on page 83
dSPACE Python Extensions	4.0	4.1	4.2	2022-B Refer to dSPACE Python Extensions on page 85.
dSPACE XIL API .NET	2021-A	2021-B	2022-A	2022-B Refer to dSPACE XIL API .NET on page 87.
ECU Interface Manager	2.9	2.10	2.11	2022-B Refer to ECU Interface Manager on page 91.
Firmware Manager	3.3	3.4	3.5	22.2 Refer to Firmware Manager on page 95.
FPGA Programming Blockset[3)]	3.11	3.12	3.13	2022-B Refer to FPGA Programming Blockset on page 97.
MicroAutoBox III firmware	5.1	5.2	6.0	22.2 Refer to MicroAutoBox III Firmware on page 103.
Model and Sensor Interface Blockset	1.1	1.2	1.3	2022-B Refer to Model and Sensor Interface Blockset on page 111.
Model Compare	3.1	3.2	3.2	2022-B Refer to Model Compare on page 105.
Model Container Utility	–	–	–	2022-B Refer to About Model Container Utility on page 109.
ModelDesk	5.5	5.6	5.7	2022-B Refer to ModelDesk on page 115.
Model Interface Package for Simulink	4.5	4.6	22.1	2022-B Refer to Model Interface Package for Simulink on page 117.

Product	dSPACE Release			
	2021-A	2021-B	2022-A	2022-B[1)]
MotionDesk	4.8	4.9	4.10	2022-B Refer to MotionDesk on page 119.
MotionDesk Blockset	2.6.3	2.6.4	2.6.5	2022-B Refer to MotionDesk on page 119.
Real-Time Testing	5.0	5.1	5.2	2022-B Refer to Real-Time Testing on page 121.
RTI[4)]	7.16	7.17	7.18	2022-B Refer to RTI/RTI-MP and RTLib on page 123.
RTI-MP[5)]	7.16	7.17	7.18	2022-B Refer to RTI/RTI-MP and RTLib on page 123.
RTI Bypass Blockset	3.16	3.17	3.18	2022-B Refer to RTI Bypass Blockset on page 125.
RTI CAN Blockset	3.4.12	3.4.13	3.4.14	2022-B
RTI CAN MultiMessage Blockset	5.6	5.7	5.8	2022-B Refer to RTI CAN MultiMessage Blockset on page 127.
RTI Electric Motor Control Blockset	1.4.3	1.4.4	1.4.5	2022-B
RTI Ethernet Blockset	1.2.5	1.2.6	1.2.7	2022-B
RTI Ethernet (UDP) Blockset	1.4.5	1.4.6	1.4.7	2022-B
RTI LIN MultiMessage Blockset	3.6	3.7	3.8	2022-B Refer to RTI LIN MultiMessage Blockset on page 129.
RTI RapidPro Control Unit Blockset	2.2.5	2.2.6	2.2.7	2022-B
RTI Synchronized Time Base Manager Blockset	1.4.2	1.4.3	1.4.4	2022-B Refer to RTI Synchronized Time Base Manager Blockset on page 131.
RTI USB Flight Recorder Blockset	1.2.4	1.2.5	1.2.6	2022-B
RTI Watchdog Blockset	2.1.3	2.1.4	2.1.5	2022-B
Sensor Simulation	1.5	1.6	1.7	2022-B Refer to Sensor Simulation on page 135.
SCALEXIO firmware	5.1	5.2	6.0	2022-B Refer to SCALEXIO Firmware on page 133.
SYNECT	2.11	2.12	2.13	2022-B Refer to SYNECT on page 137.
SystemDesk	5.5	5.6	5.6	2022-B Refer to SystemDesk on page 145.
TargetLink	5.1	5.2	5.2	2022-B Refer to TargetLink on page 153.
VEOS	5.2	5.3	5.4	2022-B Refer to VEOS on page 225.

[1] For information on the new versioning scheme of dSPACE products, refer to General Enhancements and Changes on page 13.

[2] As of dSPACE Release 2021-B, ConfigurationDesk for RapidPro is no longer part of the dSPACE Release. You can download it from the dSPACE website as a separate software package with its own setup program. Refer to https://www.dspace.com/go/CFDRP.

[3] RTI FPGA Programming Blockset up to Release 2022-A.

[4] Including the standard I/O blocksets.

[5] Including the RTI Gigalink Blockset.

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 feature of AutomationDesk is:

- Enhanced user documentation

For more information on the new feature, refer to New Features of AutomationDesk 2022-B on page 29.

Bus Manager (stand-alone)

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

- Triggering the transmission of multiplexed IPDUs by their static-part IPDUs
- Generating ISignals for update bits of ISignal groups
- Enhanced bus configuration tables

For more information, refer to New Features of the Bus Manager (Stand-Alone) 2022-B on page 45.

ConfigurationDesk

The new key features of ConfigurationDesk are:

- New DS6330M3 Automotive Ethernet Module support.
- Enhanced function block types to provide status information for data replay in a HIL environment.
- Support of binary FMUs with linux64/linux32 shared objects.
- Extended binary folder support according to the FMI 2.0 standard.
- Various enhancements of the Bus Manager for configuring bus communication for simulation, inspection, and manipulation purposes.

For more information, refer to ConfigurationDesk on page 51.

ControlDesk

The new key features of ControlDesk are:

General enhancements

- Improved navigation in Variables pane and Measurement Data Pool

For more information, refer to New User Interface Handling Features (ControlDesk 2022-B) on page 64.

Platform/device enhancements

- cmdloader: Clearing the complete flash memory of a platform

For more information, refer to New Features of Platform Management and Platforms/Devices (ControlDesk 2022-B) on page 65.

Variable management enhancements

- Uniform resource identifier for variables

For more information, refer to New Variable Management Features (ControlDesk 2022-B) on page 65.

Measurement and recording enhancements

- Improvement for data logging

For more information, refer to New Measurement and Recording Features (ControlDesk 2022-B) on page 66.

Bus Navigator enhancements

- Bus Navigator (stand-alone)
- Improvements to the Bus Navigator user interface
- Support of R21-11 ARXML files

For more information, refer to New Bus Navigator Features (ControlDesk 2022-B) on page 67.

Automation enhancements

- Writing individual function and axis values of variables connected to a Table Editor
- Switching the data source of layouts
- Specifying output parameters of calculated variables

For more information, refer to New Automation Features (ControlDesk 2022-B) on page 70.

dSPACE FlexRay Configuration Package

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

- Defining default values for source switch variables.

For more information on the new feature, refer to New Features of dSPACE FlexRay Configuration Package 2022-B on page 81.

dSPACE XIL API

The new key features of dSPACE XIL API are:

- Support of first features in the ECUPort, which is a combination of the ECUCPort and the ECUMPort defined in the ASAM XIL API standard
- Enhanced `VariableInfo` interface according to ASAM XIL API 2.2.0 in the MAPort and ECUPort implementation

For more information on the new features, refer to New Features of dSPACE XIL API .NET 2022-B on page 87.

ECU Interface Manager

The new key features of the ECU Interface Manager are:

- Preparing dynamic access to ECU variables
- XCP on CAN: Assigning specific CAN IDs
- XCP on CAN: Disabling PID transmission

For more information on the new features, refer to New Features of ECU Interface Manager 2022-B on page 91.

Firmware Manager

The new key feature of the Firmware Manager is:

- Enhanced support of automatic reboot after a firmware update

For more information, refer to New Features of Firmware Manager 22.2 on page 95.

FPGA Programming Blockset

The new key features of the FPGA Programming Blockset 2022-B are:

- Extended Xilinx® software support.
- Support of scaling subsystems for SCALEXIO and the MicroAutoBox III to preprocess and postprocess the processor signals.
- New container file format for the SCALEXIO and the MicroAutoBox III build results.

For more information on the new features, refer to New Features of the FPGA Programming Blockset 2022-B on page 97.

MicroAutoBox III firmware

The new key features of the MicroAutoBox III firmware are:

- The DS1403 Processor Board supports the use of a real-time-application with an activated data logging configuration.
- The DS1521 Bus Board supports full duplex RS485 mode.

For more information on the new features, refer to New Features of the MicroAutoBox III Firmware 22.2 on page 103.

Model and Sensor Interface Blockset

The new key features of the Model and Sensor Interface Blockset are:

- A TCP Server connection for VEOS for software-in-the-loop simulations.
- Simulink simulation in normal and accelerated mode is supported.
- The Model and Sensor Interface Blockset Animation Interface is included in the ASM Traffic model demo to support animation and sensor simulation with AURELION.

  For more information on the new features of the ASM Traffic Demo Model, refer to Changes in the ASM Traffic Demo Model on page 41.

For more information on the new features of the blockset, refer to New Features of Model and Sensor Interface Blockset 2022-B on page 111.

Model Compare	With Model Compare 2022-B several improvements take place. For detailed information on the new features, refer to New Features of Model Compare 2022-B on page 105.
ModelDesk	The new key feature of ModelDesk is: ▪ Road Generator: Specifying the position of traffic objects in absolute coordinates. For more information on the new feature, refer to New Features of ModelDesk 2022-B on page 115.
Model Interface Package for Simulink	The new key features of Model Interface Package for Simulink are: ▪ Enhancements for propagating configured bus communication to a Simulink model ▪ Support of MATLAB® R2022b For more information on the new features, refer to Model Interface Package for Simulink on page 117.
RTI, RTI-MP, and RTLib	The new key feature of RTI, RTI-MP, and RTLib is: ▪ Support of MATLAB® R2022b. For more information, refer to New Features of RTI/RTI-MP and RTLib on page 123.
RTI Synchronized Time Base Manager Blockset	The new key feature of the RTI Synchronized Time Base Manager Blockset is: ▪ New STBM_SYNCHRONOUS_TRIGGER block. For more information on the new feature, refer to New Features of the RTI Synchronized Time Base Manager Blockset 2022-B on page 131.
SCALEXIO firmware	The new key feature of the SCALEXIO firmware is: ▪ Support of the DS6330M4 Automotive Ethernet Module with integrated MACsec (media access control security) functionality. For more information on the new feature, refer to New Features of the SCALEXIO Firmware 22.2 on page 133.
SYNECT	The new key features of SYNECT 2022-B are: ▪ Modernized user interface framework ▪ Improved global search For more information on the new features, refer to New Features of SYNECT 2022-B on page 138.

SystemDesk

The new key features of SystemDesk 2022-B are:

- Support of the AUTOSAR R21-11 release.
- Support of the FMI 3.0 standard for V-ECU FMUs.
- Export of contract header files for generated basic software modules.

For more information on the new features, refer to New Features of SystemDesk 2022-B on page 146.

TargetLink

The new key features of TargetLink 2022-B are:

- n-D Interpolation block
  - Supporting look-up tables with up to five dimensions.
- AUTOSAR INOUT parameters
  - Supporting operation arguments of the ARGINOUT kind.
- Bus Creator block supports array-of-buses
  - Connecting different types of array-of-bus signals to bus-with-array-of-bus-as-payload signals.
- Adaptive AUTOSAR data types
  - Supporting C++ data types in code generation.
- Stateflow temporal count
  - Supporting the temporal logic operator `temporalCount`.
- Headless Linux support (separate from dSPACE Release)
  - Running TargetLink for code generation in Linux-based containers.

For more information on all new features, refer to New Features of TargetLink 2022-B on page 154.

For more information on the TargetLink migration aspects (TargetLink, TargetLink AUTOSAR Module, TargetLink Data Dictionary), refer to Migrating to TargetLink 2022-B and TargetLink Data Dictionary 2022-B on page 171.

VEOS

The new key features of VEOS are:

- Support of the FMI 3.0 release
- Model Container Utility support
- Co-simulation with VEOS
- V-ECU SDK
- Building V-ECUs and SICs on Linux (VEOS Build Console)
- ISO 26262 certification in the context of the SIMPHERA workflow
- Status signals for CAN/Ethernet data replay

For more information on the new features, refer to New Features of VEOS 2022-B on page 225.

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 2022-B

Introduction

After you install Release 2022-B, some additional steps might be required.

Migrating from dSPACE Release 2022-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 2021-B or earlier

To migrate from dSPACE Release 2021-B or earlier to Release 2022-B, you also have to perform the migration steps of the intervening dSPACE Releases. All of the required migration steps can be performed using software from dSPACE Release 2022-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 represents 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 2022-B\Print\PreviousReleases`.
- On the dSPACE DVDs. Refer to `\Doc\PreviousReleases`.

- At https://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 2022-B

General enhancements

Enhanced user documentation The AutomationDesk user documentation has been enhanced by several animated graphics to visualize the handling in the user interface.

Migrating to AutomationDesk 2022-B

General migration aspects

If you open an AutomationDesk project with a later AutomationDesk version, the software automatically detects whether migration is required. You must click OK in the message dialog to start the migration. Save the migrated project to another path or name.

Note

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

- You must create backups of the project and of the linked custom libraries.
- AutomationDesk must be running properly. 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 🕮).

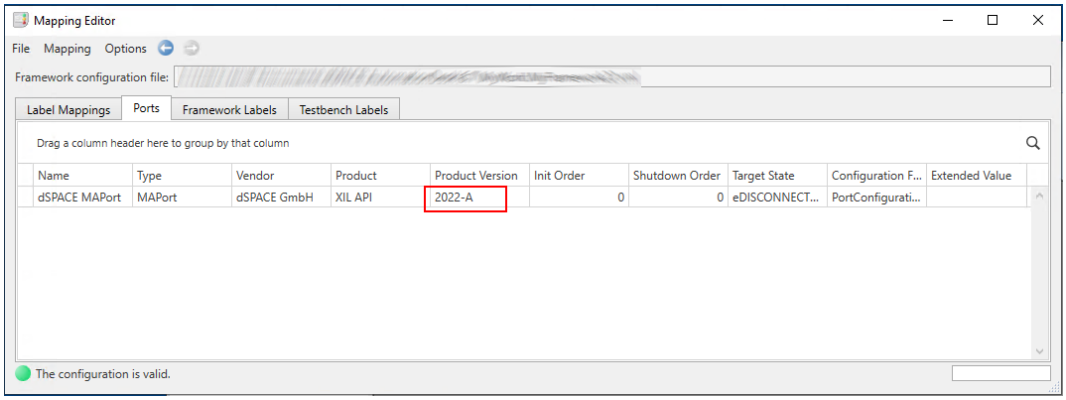
Project import restricted since Release 2021-A

Since dSPACE Release 2021-A, AutomationDesk has supported the direct import only of projects last saved with one of the previous seven AutomationDesk versions.

Migrating to XIL API 2.2.0

When you use blocks from the XIL API library or the XIL API Convenience library, you have to adapt some configurations.

Using the XIL API framework If you have not used the default vendor *dSPACE*, you have to migrate the port definition in the Mapping Editor. You have to specify the dSPACE implementation of Release 2022-A or later to switch the XIL API version to 2.2.0.



Using the XIL API testbench If you use the XIL API testbench, you have to migrate the Vendor data object to the new vendor identifier *dSPACE GmbH_XIL API_2022-A* or *dSPACE GmbH_XIL API_2022-B*.

Using STZ files

If you work with STZ files, no manual migration is required. If you execute the stimulus signals, the signal description is automatically temporarily converted to XIL API 2.2.0. The STZ file is saved in the new format only if the stimulus definition has been modified.

Planned discontinuations

Automation Server In December 2022, the *Automation Server* will be discontinued. New Releases of the AutomationDesk Automation Server were only available for customers with a Software Maintenance Service Contract until June 2022.

The *AutomationDesk Basic* license provides the technical equivalent to the Automation Server.

For more information, refer to https://www.dspace.com/go/discontinuation.

Automotive Simulation Models (ASM)

Where to go from here

Information in this section

All ASM Products

New Features of All ASM Blocksets

New ASM video

Several videos show how you can work with ASM and ModelDesk.

There is a new video explaining how to modify parameter values by means of automation.

You can access the videos in dSPACE Help and on the dSPACE website.

Changes in MATLAB M code of ASM

ASM-internal `find_system` calls now consider all variant choices of variant subsystems.

In the M code of ASM, e.g. for model migration, ASM Testbench, ASMSignalInterface, etc., several `find_system` calls are used to find blocks, lines, ports, etc.

Up to Release 2022-A, the search considered only the active variant choice in the variant subsystems.

From Release 2022-B on, all variant choices in the variant subsystems are considered during a search.

This is done to account for the changed handling of variants with `find_system` calls as of MATLAB Release R2022b. For more details, refer to the corresponding MATLAB documentation.

The changed behavior applies to ASM models used with all the MATLAB Releases supported by ASM.

ASM Battery

New Features of ASM Battery Blockset 2022-B

New ASM library with new demo model

The ASM Battery blockset is new.

The ASM Battery library contains blocks that allow modeling and simulating of a battery or a battery pack, and their thermal behavior. The library also contains blocks for monitoring and controlling the battery system. For descriptions of the blocks contained in the library. Refer to ASM Battery Reference 🕮.

Furthermore, there is a new demo project to develop and test a battery management system (BMS) which supervises and ensures a safe and efficient operation of a battery. For information on the demo model, refer to ASM Battery Model Description 🕮.

ASM Electric Components

Changes in the ASM Electric Components Demo Models

Battery Electric Vehicle with Traffic demo

There are the following changes to the Battery Electric Vehicle with Traffic demo model:

AURELION support The Battery Electric Vehicle with Traffic demo model now supports AURELION as an animation tool.

To support animation in AURELION, the MotionDeskInterface was replaced by the AnimationInterface in the demo model. The respective MotionDesk project is discontinued.

SCALEXIO platform support The new AnimationInterface requires a SCALEXIO hardware-specific configuration. Because of this, SCALEXIO real-time objects are not part of the Traffic demo models any more. Instead, the Traffic project now contains a SIC file and a Python script that you can use to easily create a real-time object for a specific SCALEXIO system.

Default simulation platform For the Traffic models, the default platform that is selected when you initialize the model with the go.m file has been changed from SCALEXIO to VEOS.

When you initialize a Traffic model with a call of the go.m file, it now prepares the model for simulation on a VEOS platform by default if you do not specify a platform argument.

ASM Environment

Where to go from here

Information in this section

New Features of ASM Environment Blockset 2022-B

MANEUVER_FCN_CALL_ GENERATOR block

The number of samples between successive calls can now be optionally provided using an inport.

Migrating to ASM Environment Blockset 2022-B

MANEUVER_FCN_CALL_ GENERATOR block

The new inports which are associated with the external number of samples are connected to constant values during the migration.

Related topics

Basics

Migrating ASM Models (ASM User Guide 🕮)

ASM Fuel Cell

Where to go from here

Information in this section

New Features of ASM Fuel Cell Blockset 2022-B

ELECTROCHEMISTRY_PHYSICAL block

The block now contains a new ModelDesk processing function to calculate the parameters that define the polarization curve of the fuel cell.

HYDROGEN_EJECTOR block

This is a new block which calculates mass and enthalpy flows through the ejector. The ejector performs the role of both hydrogen intake and recirculation of hydrogen from the anode side. The block also provides a processing function which adjusts default values of three parameters based on measured values of ejector mass flows and other measurements affecting the ejector.

Changes in the ASM Fuel Cell Demo Model

Recirculation mode parameter

A new Recirculation Mode parameter was added to the UserInterface inside `ASM_FuelCell/FuelCell/FuelCell_Slow/Plant`.

The parameter activates one of the two recirculation modes:

- The first one uses the injector valve for hydrogen intake and recirculation blower for recirculation of hydrogen.
- The second mode uses the hydrogen ejector for both roles.

Stationary mass power balance calculation

The efficiency calculation in the UserInterface inside `ASM_FuelCell/FuelCell/FuelCell_Slow/Plant` was modified. Currently only one efficiency is calculated, based on the lower heat value and also all losses and power consumption of the BoP components (e.g., air compressor) are taken into account. Also in the model, the ejector mass and enthalpy flows are now taken into account for calculation of mass and enthalpy balances if the ejector mode is activated.

MEA_BP_Casing_Assembly block	Mean values were added to the block (over all active segments) for the following: ▪ Membrane humidity. ▪ Liquid water volume fraction at cathode catalyst layer. ▪ Gas fraction of hydrogen at anode catalyst layer. ▪ Gas fraction of oxygen and water vapor at cathode catalyst layer.

Migrating to ASM Fuel Cell Blockset 2022-B

HYDROGEN SUPPLY CONTROL block	Block migration is performed because an ejector model is added to the model. The block now has an additional input for the current recirculation mode and an additional output for the ejector control signal. It now also contains two new parameters for the ejector controller.
COOLANT_PUMP and FAN blocks	Block migration is performed because a new parameter for calculation of power with volume flow has been added to the COOLANT_PUMP and FAN blocks.

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 2022-B

ASM_FCN_CALL_GENERATOR block

The number of samples between successive calls can now be optionally provided using an inport.

Migrating to ASM Utils 2022-B

ASM_FCN_CALL_GENERATOR block

The new inports which are associated with the external number of samples are connected to constant values during the migration.

ASM Traffic

Where to go from here

Information in this section

Changes in the ASM Traffic Demo Model

2-D SensorParameters block extension

The SensorParameters block for the 2-D sensors has been extended with a logic to determine parameterization cases in which it is not required to calculate the DISCRETE_OBJECTS block. A switch signal is provided to enable or disable the DISCRETE_OBJECTS block.

AURELION support

The Vehicle Dynamics with Traffic demo model now supports AURELION as an animation tool.

To support animation in AURELION, the MotionDeskInterface was replaced by the AnimationInterface in the demo model. The respective MotionDesk project is discontinued.

SCALEXIO platform support

The new AnimationInterface requires a SCALEXIO hardware-specific configuration. Because of this, SCALEXIO real-time objects are not part of the Traffic demo models any more. Instead, the Traffic project now contains a SIC file and a Python script that you can use to easily create a real-time object for a specific SCALEXIO system.

Default simulation platform

For the Traffic models, the default platform that is selected when you initialize the model with the go.m file has been changed from SCALEXIO to VEOS.

When you initialize a Traffic model with a call of the go.m file, it now prepares the model for simulation on a VEOS platform by default if you do not specify a platform argument.

Migrating to ASM Traffic Blockset 2022-B

DISCRETE_OBJECTS block

The block has been extended by a new inport which can be used to enable or disable the calculations of the DISCRETE_OBJECTS S-function. During migration, the new inport is connected with a constant value *1* to enable the block.

Related topics

Basics

Migrating ASM Models (ASM User Guide 🕮)

ASM Truck

Changes in the ASM Truck Demo Model

Pitman Arm Steering

The ASM Truck demos now use the new **STEERING_PITMAN_ARM** block for the steering system. The steering system is now based on pitman arm steering which is often used in the truck sector.

ASM Vehicle Dynamics

New Features of ASM Vehicle Dynamics Blockset 2022-B

STEERING_PITMAN_ARM block

The new STEERING_PITMAN_ARM block models a steering system with a pitman arm. The translation and the dynamic behavior of the steering system are simulated from the steering wheel to the coupling with the suspension. Internal and external power steering systems can be considered.

Bus Manager (Stand-Alone)

Where to go from here

Information in this section

New Features of the Bus Manager (Stand-Alone) 2022-B

Triggering the transmission of multiplexed IPDUs by static-part IPDUs

The Bus Manager (stand-alone) now lets you trigger the transmission of multiplexed IPDUs by their static-part IPDUs. For this purpose, timings that are specified in communication matrices for static-part IPDUs can be considered. You can also assign the **PDU Cyclic Timing Control** and **PDU Trigger** features to static-part IPDUs to trigger their transmission and the transmission of their related multiplexed IPDUs.

For more information, refer to Triggering the Transmission of Multiplexed IPDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮).

Generating ISignals for update bits of ISignal groups

The Bus Manager (stand-alone) now automatically generates ISignals for update bits of ISignal groups. If a communication matrix specifies such update bits, the ISignals are generated during the import of the communication matrix.

For more information, refer to Working with Communication Matrices (Bus Manager (Stand-Alone) Implementation Guide 🕮).

Enhanced bus configuration tables

The Bus Manager (stand-alone) now provides the following enhancements for the bus configuration tables:

- Additional columns are available. The columns let you access further elements, such as frame triggerings, initial ISignal values, or user signals of the PDU User Code feature.
- For a better overview, not all of the available columns are displayed by default. The columns that are not displayed in the tables are available in the Column Chooser. From the Column Chooser, you can add these columns to the tables via drag & drop.
- You can reset the tables to their default states by using the Reset Columns context menu command.
- To unambiguously identify each column, some columns are renamed.
- For each column, a tooltip is available on the table header. The tooltips provide brief information about each column and references to further information in the user documentation, if applicable.

For more information, refer to Bus Configuration Tables (Bus Manager (Stand-Alone) Implementation Guide 🕮).

Enhancements for implementing global time synchronization (GTS) in executable applications

The Bus Manager (stand-alone) now lets you trigger the transmission of individual PDUs synchronously to a global time. You can do this by using triggers that are provided by the RTI Synchronized Time Base Manager Blockset.

For more information, refer to Basics on Implementing Global Time Synchronization in Executable Applications (Bus Manager (Stand-Alone) Implementation Guide 🕮).

Enhanced SecOC Authenticator Invalidation feature

When you now add the SecOC Authenticator Invalidation feature to a secured IPDU, the Recalculate SecOC Information checkbox is now cleared by default. Because of this, the recalculation of the authentication information is disabled by default, which is required for most use cases.

Accessing parent elements via the ConfigurationDesk automation interface

You can now access the parent elements of elements that are selected in the Buses Browser or the bus configuration tables via the ConfigurationDesk automation interface. You can do this by using the new `BusParentElement` relation of the `ICaRelation` interface.

For more information, refer to New Features and Changes to the Automation Interface for Release 2022-B (ConfigurationDesk Automating Tool Handling 🕮).

Modernized user interface

The style of the user interface of the Bus Manager (stand-alone) is modernized. This does not affect the way you use the Bus Manager or any functionalities.

Enhancements for propagating configured bus communication to a Simulink model

Due to enhancements of the Model Interface Package for Simulink, you can now create Simulink.Bus objects for bus configuration elements when you propagate configured bus communication to a Simulink model. To do this, enter the following command in the MATLAB Command Window:

```
dsmpb_pref('Set', 'CreateBusObjectsDuringPropagation', 'true');
```

When you now propagate configured bus communication to a Simulink model, model port blocks and Simulink.Bus objects are created. The Simulink.Bus objects are created to the MATLAB workspace and are assigned as data type to the ports of the related model port blocks. For more information, refer to Basics of Data Port Blocks with Structured Data Ports (Model Interface Package for Simulink - Modeling Guide 🕮).

Migrating to the Bus Manager (Stand-Alone) 2022-B

Opening projects created with previous Bus Manager (stand-alone) versions

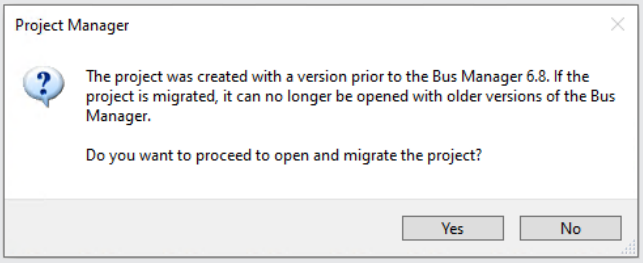
You can open a project created with a previous Bus Manager (stand-alone) version in the same way as a project from the current Bus Manager (stand-alone) version. The Bus Manager (stand-alone) migrates the project when opening it.

Note

- As of dSPACE Release 2021-A, the Bus Manager (stand-alone) supports the direct import only of projects last saved with one of the previous seven Bus Manager (stand-alone) versions.
- After you have migrated and saved a project, you can no longer open it with an earlier Bus Manager (stand-alone) version.

Note the following specifics regarding the changed project and application file formats with the Bus Manager (stand-alone) 6.8 (dSPACE Release 2021-B):

- You can still open project and application files and backups from earlier Bus Manager (stand-alone) versions. The Bus Manager (stand-alone) prompts you to confirm the migration when you open a project:



  Click Yes to migrate the project and all its applications to the new file formats. Afterwards, you can no longer open them with earlier Bus Manager (stand-alone) versions.
- The new project and application files are stored in the respective project and application folders. They are identified by a GUID. Do not move or rename them.
- If you are using version control software for ConfigurationDesk projects, be aware that the old project and application files are deleted and the new files might not automatically be known to your version control software when you check in your local copy.

  If you checked in and deleted your local copy after the migration, you can recreate the new project and application files by checking out the project with the old project and application files again and repeating the migration. Only the project and application files will be replaced, no other migration steps will be repeated.

Note the following specifics regarding the new project cache folder introduced with the Bus Manager (stand-alone) 22.1 (dSPACE Release 2022-A):

- The path of project and application elements must not exceed 260 characters. Refer to Limitations Concerning Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 🕮).

  Elements that are moved to the new project cache folder during project migration might exceed the character limit afterwards. To avoid this, you can change the project cache folder to a shorter path. Refer to Paths Page (ConfigurationDesk User Interface Reference 🕮).

Generated ISignals for update bits of ISignal groups not available in existing ConfigurationDesk applications

ISignals that are generated by the Bus Manager for update bits of ISignal groups are not available in ConfigurationDesk applications that were created with the Bus Manager (stand-alone) 22.1 or earlier. To make such ISignals available, you have to add another version of the related communication matrix, i.e., a communication matrix whose **Content hash** value differs, to the affected ConfigurationDesk application. Then, you have to replace the old communication matrix in the related bus configurations and manually assign the newly available ISignals, if required.

For more information on replacing communication matrices, refer to Replacing Assigned Communication Matrices (Bus Manager (Stand-Alone) Implementation Guide 🕮).

Changes to the tool automation interface that might cause code malfunctions

Some changes to the tool automation interface affect the data model and can cause code from previous Releases to malfunction. For more information, refer to New Features and Changes to the Automation Interface for Release 2022-B (ConfigurationDesk Automating Tool Handling 🕮).

Bus Manager (Stand-Alone) Discontinuations

Planned discontinuation of Excel export of ConfigurationDesk applications

The **Export Configuration** command for exporting the configuration data of the active ConfigurationDesk application to an Excel™ file (XLSX file) will be removed with dSPACE Release 2023-B.

The corresponding tool automation API command `ICaAlgorithms:ExportConfiguration` will also be removed.

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 General Features of ConfigurationDesk 2022-B

Support of Ethernet BSC files

ConfigurationDesk now supports bus simulation containers (BSC files) that were generated with the Ethernet Configuration Package. The benefits of model container workflows are now also available for real-time applications with Ethernet communication:

- Separation of modeling and configuration work from integration, simulation, and test.
- Shorter build times by using model containers that can be optionally precompiled with the `PrecompileBSC` automation command of ConfigurationDesk.
- IP protection through compiled container files without source code.

For more information, refer to Working with Bus Simulation Containers (BSC Files) (ConfigurationDesk Real-Time Implementation Guide 🕮).

Enhanced FMU support

Support of binary FMUs with linux64/linux32 shared objects ConfigurationDesk now supports binary FMUs that provide their implementation as Linux 64/Linux 32 shared objects. This has the following benefits:

- You can use FMUs that were exported in shared object binary form and that do not contain source code directly in ConfigurationDesk. Hypervisor-based solutions are no longer required.
- You can use the same FMU on different Linux-based simulators.
- IP protection scenarios can be achieved more easily.

Extended binary folder support The ConfigurationDesk support for FMUs that contain source code and additional binaries was extended to support the following folders according to the definition of the FMI 2.0 standard:

- `<FMU root>/binaries/linux64`
- `<FMU root>/binaries/linux32`

This increases the portability of the FMUs.

Support of FMUs that employ multiple processor cores ConfigurationDesk now supports FMUs that create multiple parallel threads on multiple processor cores to parallelize their model computations. ConfigurationDesk provides the new Number of required processor cores property that lets you specify the number of processor cores that are reserved for the application process to which the FMU is assigned.

For more information, refer to Working with Functional Mock-up Units (FMU Files) (ConfigurationDesk Real-Time Implementation Guide 🕮).

New demo with hand-coded FMU

ConfigurationDesk provides a new demo with a hand-coded FMU. The source code of the FMU and the modelDescription.xml file contain comments that make it easier for you to get started with the FMI standard. You can use this FMU as a starting point for integrating custom C-code into ConfigurationDesk real-time applications without using Simulink®. For more information, refer to CfgCustomFMUDemo Project: Getting Started with the FMI Standard (ConfigurationDesk Using Demo Projects 🕮).

New model implementation properties

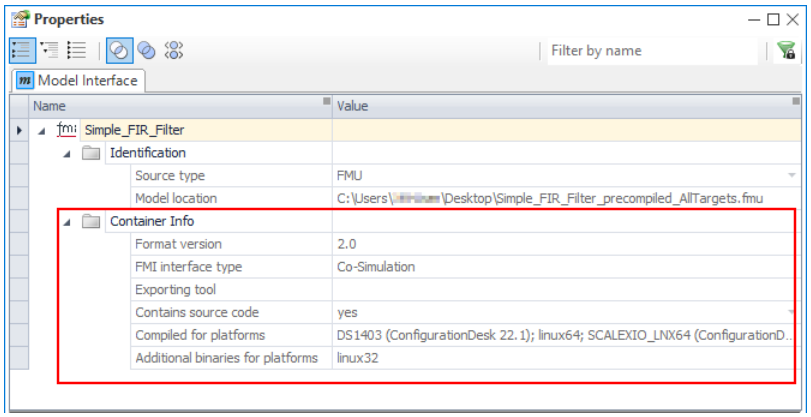
Model containers (FMUs, SIC files, BSC files) can provide their interface functions as C source code files and/or as binary files. In addition, the model containers can contain additional binaries. ConfigurationDesk now provides the following model implementation properties to display information on the contents of model containers:

Contains source code The Contains source code property displays whether an FMU, a BSC file, or an SIC file contains source code files or not.

Compiled for platforms The Compiled for platforms property displays the platforms the model container was compiled for. For model container files that have been compiled using the `PrecompileFMU`, `PrecompileSIC`, or `PrecompileBSC` command, this property lists the ConfigurationDesk version the FMU, BSC file, or SIC file was compiled with.

Additional binaries for platform The Additional binaries for platform property for FMUs, SIC files, and BSC files lists the target platforms with which the additional binaries contained in an FMU, SIC file, or BSC file are compatible.

The following illustration shows the properties for an example FMU:



Note

The former Precompiled for property of model containers was renamed to Compiled for platforms.

Display of application process build times

ConfigurationDesk now displays the build time and the build process status (succeeded or failed) for each application process in the Build Log Viewer. This lets you identify model implementations with long build times.

Hyperlinks to specific folders in Build Log Viewer

For an easier navigation to specific folders, ConfigurationDesk now outputs hyperlinks to the Simulink code generation folder, the Compile folder, and the Build Results folder in the Build Log Viewer during the build process.

New features of the ECU interfacing support

ConfigurationDesk now lets you implement *dynamic data accesses* in a real-time application. With dynamic data accesses, you can specify at run time which ECU variables are accessed by the real-time application. Dynamic data accesses must be specified in the ECU Interface Manager. When you import an EIC file that contains dynamic data accesses to an ECU Interface Configuration function block, function ports are available for each dynamic data access. Using the function ports, you can specify the ECU variables that are to be accessed by the real-time application.

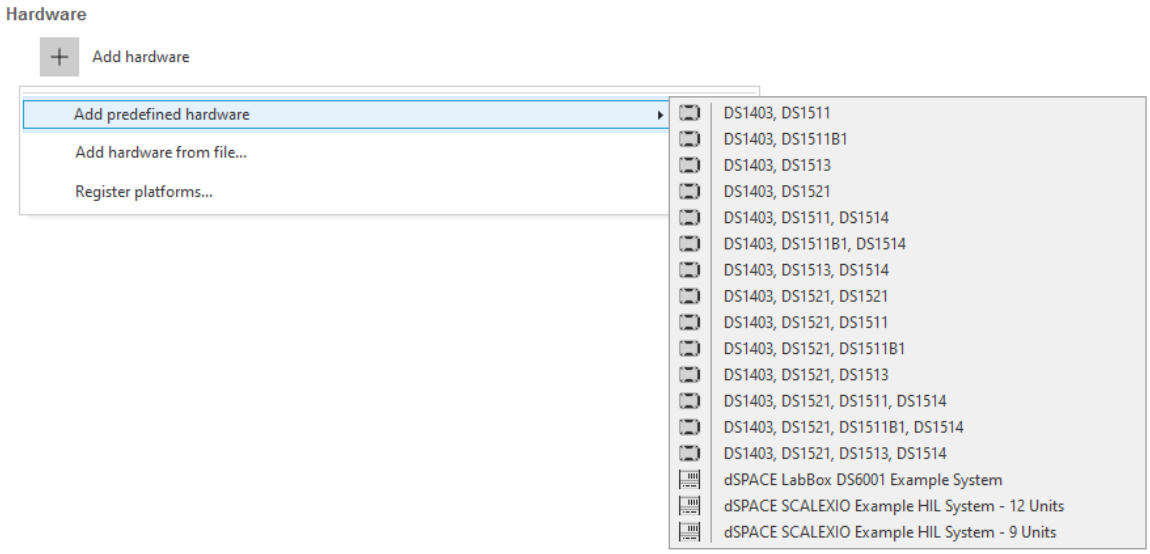
For more information, refer to Overview of Ports and Basic Properties (ECU Interface Configuration) (ConfigurationDesk I/O Function Implementation Guide 🕮).

User interface modernized

The style of ConfigurationDesk's graphical user interface has been modernized. This does not affect the way you use ConfigurationDesk or any functionalities.

Import predefined hardware from list

When you import hardware to a ConfigurationDesk application, you can now select one of multiple predefined hardware topologies from a list:



New features of the tool automation interface

The ConfigurationDesk automation interface supports additional ConfigurationDesk features. For more information, refer to New Features and Changes to the Automation Interface for Release 2022-B (ConfigurationDesk Automating Tool Handling 🕮).

New Features Concerning I/O Functionality and Hardware Support

Enhanced function block types

Enhanced data streaming support The Ethernet Setup and the CAN function block types now provide function ports with status information on the assigned data streaming channel.

The function ports provide the following information:

- State of the data streaming channel.
- Number of data frames that are forwarded to the assigned controller.
- Number of data frames that are not forwarded to the assigned controller due to invalid content.
- Number of data frames at which the transmission with the assigned controller fails.
- Number of data frames that are dropped after the stop time has been reached.

You can enable the function ports with the Channel status property.

SENT Out The SENT Out function block type now supports the output of the CRC checksum for the SENT message to be transmitted if no SENT protocol

is used. The CRC Value function outport provides the CRC checksum that is calculated at run time.

For more information, refer to Encoding SENT Messages in the Behavior Model (ConfigurationDesk I/O Function Implementation Guide 🕮).

New features concerning hardware support

SCALEXIO The Ethernet Switch function block type now supports the DS6330M3 Automotive Ethernet Module. The DS6330M3 module supports the 10BASE-T1S Ethernet standard. The 10BASE-T1S standard is a 2-wire automotive Ethernet network that supports the physical layer collision avoidance (PLCA) and allows a connection of several nodes with one bus line.

For more information, refer to Configuring the PHYs of I/O Ethernet Ports (ConfigurationDesk I/O Function Implementation Guide 🕮) .

New Features of the Bus Manager in ConfigurationDesk

Triggering the transmission of multiplexed IPDUs by static-part IPDUs

The Bus Manager now lets you trigger the transmission of multiplexed IPDUs by their static-part IPDUs. For this purpose, timings that are specified in communication matrices for static-part IPDUs can be considered. You can also assign the PDU Cyclic Timing Control and PDU Trigger features to static-part IPDUs to trigger their transmission and the transmission of their related multiplexed IPDUs.

For more information, refer to Triggering the Transmission of Multiplexed IPDUs (ConfigurationDesk Bus Manager Implementation Guide 🕮).

Generating ISignals for update bits of ISignal groups

The Bus Manager now automatically generates ISignals for update bits of ISignal groups. If a communication matrix specifies such update bits, the ISignals are generated during the import of the communication matrix.

For more information, refer to Working with Communication Matrices (ConfigurationDesk Bus Manager Implementation Guide 🕮).

Enhanced bus configuration tables

The Bus Manager now provides the following enhancements for the bus configuration tables:

- Additional columns are available. The columns let you access further elements, such as frame triggerings, initial ISignal values, or user signals of the PDU User Code feature.
- For a better overview, not all of the available columns are displayed by default. The columns that are not displayed in the tables are available in the Column Chooser. From the Column Chooser, you can add these columns to the tables via drag & drop.
- You can reset the tables to their default states by using the Reset Columns context menu command.
- To unambiguously identify each column, some columns are renamed.

- For each column, a tooltip is available on the table header. The tooltips provide brief information about each column and references to further information in the user documentation, if applicable.

For more information, refer to Bus Configuration Tables (ConfigurationDesk Bus Manager Implementation Guide 🕮).

Enhancements for implementing global time synchronization (GTS) in executable applications

The Bus Manager now lets you trigger the transmission of individual PDUs synchronously to a global time. You can do this by using triggers that are provided by the RTI Synchronized Time Base Manager Blockset.

For more information, refer to Basics on Implementing Global Time Synchronization in Executable Applications (ConfigurationDesk Bus Manager Implementation Guide 🕮).

General enhancements

The Bus Manager now provides the following general enhancements:

- When you add the **SecOC Authenticator Invalidation** feature to a secured IPDU, the **Recalculate SecOC Information** checkbox is cleared by default. Because of this, the recalculation of the authentication information is disabled by default, which is required for most use cases.
- In the **Buses** view set, the **Hardware Resource Browser** is now available by default.

Accessing parent elements via the ConfigurationDesk automation interface

You can now access the parent elements of elements that are selected in the **Buses Browser** or the bus configuration tables via the ConfigurationDesk automation interface. You can do this by using the new `BusParentElement` relation of the `ICaRelation` interface.

For more information, refer to New Features and Changes to the Automation Interface for Release 2022-B (ConfigurationDesk Automating Tool Handling 🕮).

Enhancements for propagating configured bus communication to a Simulink model

Due to enhancements of the Model Interface Package for Simulink, you can now create Simulink.Bus objects for bus configuration elements when you propagate configured bus communication to a Simulink model. To do this, enter the following command in the MATLAB Command Window:

```
dsmpb_pref('Set', 'CreateBusObjectsDuringPropagation', 'true');
```

When you now propagate configured bus communication to a Simulink model, model port blocks and Simulink.Bus objects are created. The Simulink.Bus objects are created to the MATLAB workspace and are assigned as data type to the ports of the related model port blocks. For more information, refer to Basics of Data Port Blocks with Structured Data Ports (Model Interface Package for Simulink - Modeling Guide 🕮).

Supported Container File Versions

Supported SIC file versions ConfigurationDesk 2022-B (22.2) supports SIC file versions as listed below:

SIC Files Created With ...	MATLAB Release
dSPACE Release 2022-B: ▪ Model Interface Package for Simulink 2022-B (22.2) ▪ TargetLink 5.3	R2022b, R2022a, R2021b, R2021a
dSPACE Release 2022-A: ▪ Model Interface Package for Simulink 22.1	R2022a, R2021b, R2021a, R2020b
dSPACE Release 2021-B: ▪ Model Interface Package for Simulink 4.6 ▪ TargetLink 5.2	R2021b, R2021a, R2020b, R2020a
dSPACE Release 2021-A: ▪ Model Interface Package for Simulink 4.5	R2021a, R2020b, R2020a, R2019b
dSPACE Release 2020-B: ▪ Model Interface Package for Simulink 4.4 ▪ TargetLink 5.1	R2019a
dSPACE Release 2020-A: ▪ Model Interface Package for Simulink 4.3	R2018b
dSPACE Release 2019-B: ▪ Model Interface Package for Simulink 4.2	R2018a
dSPACE Release 2019-A: ▪ Model Interface Package for Simulink 4.1	R2017b
dSPACE Release 2018-B: ▪ Model Interface Package for Simulink 4.0	R2017a
dSPACE Release 2018-A: ▪ Model Interface Package for Simulink 3.6	R2016b
dSPACE Release 2017-B: ▪ Model Interface Package for Simulink 3.5	R2016a
dSPACE Release 2017-A: ▪ Model Interface Package for Simulink 3.4	R2015b

Target platform compatibility of SIC files The following table shows the compatibility of SIC files and target platforms, and indicates which system target file you have to select for the generation of an SIC file:

SIC File Created With	System Target File	Target Platform
dSPACE Release 2022-A and later	dsrt.tlc	▪ SCALEXIO Linux 64-bit ▪ SCALEXIO Linux 32-bit (legacy support) ▪ MicroAutoBox III You have to select the target architecture for which the SIC file is generated depending on the target platform. Refer to Basics on Simulink Implementation Containers (Model Interface Package for Simulink - Modeling Guide 🕮).

SIC File Created With	System Target File	Target Platform
dSPACE Release 2021-B and earlier	dsrt64.tlc (available as of dSPACE Release 2019-B)	SCALEXIO Linux 64-bit
	dsrt.tlc	▪ SCALEXIO 32-bit (legacy support) ▪ MicroAutoBox III

Limitations for earlier SIC file versions in ConfigurationDesk scenarios SIC files created with the Model Interface Package for Simulink version 3.4 … 4.1 are not supported in the following ConfigurationDesk scenarios:

- In multimodel application processes.
- For building real-time applications that use Real-Time Testing.

Limitations for Simulink behavior models underlying earlier SIC files versions The following limitations apply to Simulink behavior models underlying SIC files created with the Model Interface Package for Simulink version 3.4 … 4.4:

- The Simulink behavior model must not contain blocks from the following blocksets:
  - Blocks of ASM

    Real-time applications that contain such SIC files cannot be used with ModelDesk.
  - FPGA Programming Blockset
  - MotionDesk Blockset

    Real-time applications that contain such SIC files cannot be used with MotionDesk.
  - Blocks of any dSPACE Solution.
- Compiled objects contained in SIC files must match the target platform. SIC files with compiled objects that were created at a time when the current target platforms did not exist are not supported. In this case, it is not possible to use compiled objects.

Supported BSC file versions

ConfigurationDesk 2022-B (22.2) supports BSC files that were generated with the Bus Manager of the current Release, i.e., BSC file version 1.11. or with the Ethernet Configuration Package.

Supported Functional Mock-up Unit versions

ConfigurationDesk 2022-B (22.2) supports Functional Mock-up Units (FMUs) that comply with the following versions of the FMI standard:

- 2.0.x

Supported EIC file versions

ConfigurationDesk 2022-B (22.2) supports EIC file versions as listed below:

EIC Files Created With ...	EIC Version
dSPACE Release 2022-B (ECU Interface Manager 2022-B)	5.0.0
dSPACE Release 2022-A (ECU Interface Manager 2.11)	5.0.0
dSPACE Release 2021-B (ECU Interface Manager 2.10)	4.0.0
dSPACE Release 2021-A (ECU Interface Manager 2.9)	4.0.0
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

Note

- For 64-bit target architectures, only EIC files as of version 5.0.0 are supported.
- For MicroAutoBox III systems, only EIC files as of version 4.0.0 are supported.

Migrating to ConfigurationDesk 2022-B

Opening projects created with previous ConfigurationDesk versions

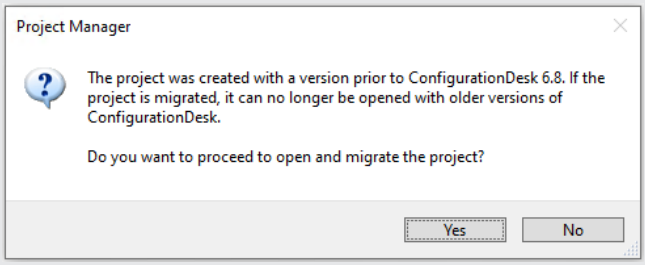
You can open a project created with a previous ConfigurationDesk version in the same way as a project from the current ConfigurationDesk version. ConfigurationDesk migrates the project when opening it.

Note

- As of dSPACE Release 2021-A, ConfigurationDesk supports the direct import only of projects last saved with one of the previous seven ConfigurationDesk versions.
- After you have migrated and saved a project, you can no longer open it with a previous ConfigurationDesk version.

Note the following specifics regarding the changed project and application file formats with ConfigurationDesk 6.8 (dSPACE Release 2021-B):

- You can still open project and application files and backups from previous ConfigurationDesk versions. ConfigurationDesk prompts you to confirm the migration when you open a project:



  Click Yes to migrate the project and all its applications to the new file formats. Afterwards, you can no longer open them with previous ConfigurationDesk versions.
- The new project and application files are stored in the respective project and application folders. They are identified via a GUID. Do not move or rename them.
- If you are using version control software for ConfigurationDesk projects, be aware that the old project and application files are deleted and the new files might not automatically be known to your version control software when you check in your local copy.

  If you checked in and deleted your local copy after the migration, you can recreate the new project and application files by checking out the project with the old project and application files again and repeating the migration. Only the project and application files will be replaced, no other migration steps will be repeated.

Note the following specifics regarding the new project cache folder introduced with ConfigurationDesk 22.1 (dSPACE Release 2022-A):

- The path of project and application elements must not exceed 260 characters. Refer to Limitations Concerning Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 🕮).

  Elements such as build artifacts that are moved to the new project cache folder during project migration might exceed the character limit afterwards. To avoid this, you can change the project cache folder to a shorter path. Refer to Paths Page (ConfigurationDesk User Interface Reference 🕮).

SCALEXIO target architecture compatibility of model implementation containers

Model implementation containers that include 32-bit compatible binaries are not compatible with the SCALEXIO Linux 64-bit target architecture. You must regenerate the model implementation container with Linux 64-bit compatible binaries, or specify a Linux 32-bit target architecture.

SIC files must must be regenerated with the appropriate **Target architecture** setting in the Simulink **Configuration Parameters** dialog or via the `dsrt_build()` API command.

Changes to the Bus Manager

Generated ISignals for update bits of ISignal groups not available in existing ConfigurationDesk applications ISignals that are generated by the Bus Manager for update bits of ISignal groups are not available in ConfigurationDesk applications that were created with the Bus Manager (stand-alone) 22.1 or earlier. To make such ISignals available, you have to add another version of the related communication matrix, i.e., a communication matrix whose **Content hash** value differs, to the affected ConfigurationDesk application. Then, you have to replace the old communication matrix in the related bus configurations and manually assign the newly available ISignals, if required.

For more information on replacing communication matrices, refer to Replacing Assigned Communication Matrices (Bus Manager (Stand-Alone) Implementation Guide 🕮).

Changes to the tool automation interface that might cause code malfunctions

Some changes to the tool automation interface affect the data model and can cause code from previous Releases to malfunction. For more information, refer to New Features and Changes to the Automation Interface for Release 2022-B (ConfigurationDesk Automating Tool Handling 🕮).

Changes to custom function folders

As of dSPACE Release 2022-A, ConfigurationDesk does not automatically add the project-specific custom function folder `<ProjectLocation>\<ProjectFolder>\CustomFunctions` to a new project. The lack of the project-specific custom function folder might affect the execution of automation scripts that copy custom functions to that folder.

For new projects, it is recommended to use search paths to add custom functions. Search paths let you use repositories and version control software, for example. For more information, refer to Managing Search Paths (ConfigurationDesk Custom I/O Function Implementation Guide 🕮).

However, ConfigurationDesk still supports project-specific custom function folders.

ConfigurationDesk Discontinuations

Discontinuation of V-ECU implementation container support	With ConfigurationDesk 2022-B (22.2), the support of V-ECU implementation containers (VECU files) was removed. If you open a ConfigurationDesk application that was created with an earlier dSPACE Release and that contains V-ECU implementation containers, the V-ECU implementation containers are removed automatically.
Planned discontinuation of Excel export of ConfigurationDesk applications	The Export Configuration command for exporting the configuration data of the active ConfigurationDesk application to an Excel™ file (XLSX file) will be removed with ConfigurationDesk 2023-B. The corresponding tool automation API command ICaAlgorithms:ExportConfiguration will also be removed.
Planned discontinuation of custom device properties	The possibility to add custom device properties to device topology elements will be removed with ConfigurationDesk 2023-B.
Planned discontinuation of external cable harness calculation	The possibility to calculate the representation of an external cable harness will be removed with ConfigurationDesk 2024-A.

ControlDesk

Where to go from here

Information in this section

New Features of ControlDesk 2022-B

Where to go from here

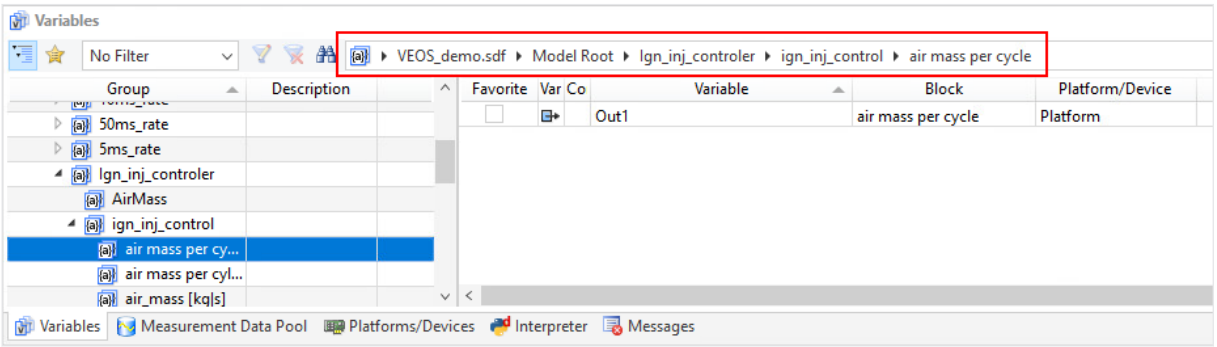
Information in this section

New User Interface Handling Features (ControlDesk 2022-B)

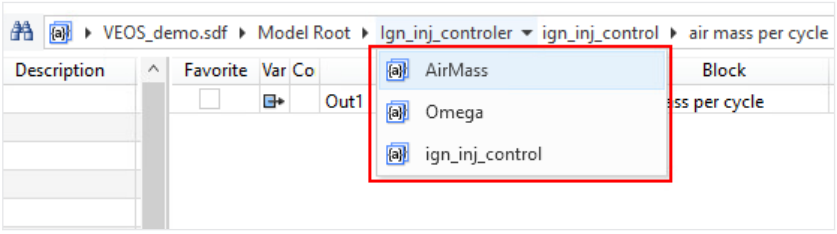
Improved navigation in Variables pane and Measurement Data Pool

The new breadcrumb control available in the Variables pane and in the Measurement Data Pool makes navigation easier, especially when dealing with variable descriptions with nested structures.

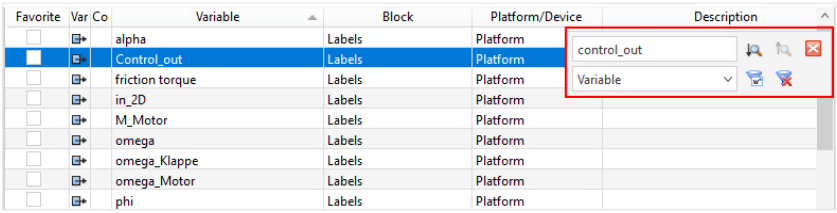
The following illustration shows the new control in the Variables pane as an example:



The breadcrumb control is active, i.e., when you can click an entry of the breadcrumb, you can access the subordinate entries as shown in the following illustration:



Click **Ctrl** + **F** to search or filter variables:



For more information, refer to:

- Basics of the Variables Pane (ControlDesk Variable Management 🕮)
- Basics on the Measurement Data Pool (ControlDesk Measurement and Recording 🕮)

New Features of Platform Management and Platforms/Devices (ControlDesk 2022-B)

cmdloader: Clearing the complete flash memory of a platform

The `cmdloader` command line tool now lets you clear the complete flash memory of a platform.

The following example shows you how to clear the complete flash memory of a platform named '`ds1403`':

```
cmdloader -clear_flash -p ds1403
```

For more information, refer to Registering Platforms and Handling Applications via cmdloader (ControlDesk Platform Management 🕮).

New Variable Management Features (ControlDesk 2022-B)

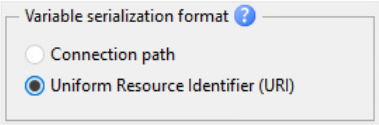
Uniform resource identifier for variables

In addition to the *connection path*, you can now select the *Uniform Resource Identifier (URI)* format for the textual representation of variables.

The format is used when you copy a variable to the Clipboard and paste it to an automation scripts, for example.

To select the URI format, you have two options:

- Use the **Variable serialization format** option on the **Variables** page of the **ControlDesk Options** dialog.



  For more information, refer to Variables Page (ControlDesk Variable Management 🕮).

- Enter the following Python code in the **Internal Interpreter**:

```
Application.VariablesManagement.VariableSerializationFormat = 1
```

  For more information, refer to VariablesManagement / IXaVariablesManagement <<Interface>> (ControlDesk Automation 🕮).

Related topics

References

VariablesManagement / IXaVariablesManagement <<Interface>> (ControlDesk Automation 🕮)

New Measurement and Recording Features (ControlDesk 2022-B)

Improvement for data logging

As of ControlDesk 2022-B, the data logging configuration is part of the related real-time application (RTA) file.

As a result, the ControlDesk PC no longer needs to be connected to the real-time hardware to load a data logging configuration, because you can load the RTA file (including the data logging configuration) to the real-time hardware, for example, by connecting a USB mass storage device containing that RTA file to the hardware.

This allows you to easily distribute data logging configurations.

Data logging workflow as of ControlDesk 2022-B:

1. In ControlDesk, open an experiment containing a supported platform and a variable description.
2. Configure data logging as described in Basics on Configuring Data Logging (ControlDesk Measurement and Recording 🕮).
3. Activate the data logging configuration.

   This loads the configuration to the real-time application (RTA) file.

4. To perform data logging, you have the following options:
   - In ControlDesk, load the RTA file to the flash memory or to the RAM of the connected real-time hardware. A data logging session starts automatically when the real-time application starts.

     Or:
   - Use the web interface of the real-time hardware to load the RTA file including the data logging configuration to the real-time hardware.

     Or:
   - (MicroAutoBox III only) Copy the RTA file to a USB mass storage device and connect the device to the MicroAutoBox III. The RTA file including the data logging configuration is loaded during the next start-up of the MicroAutoBox III, and data logging is started.
5. Stop data logging.

For more information, refer to Data Logging (ControlDesk Measurement and Recording 🕮).

Related topics

Basics

Basics on Configuring Data Logging (ControlDesk Measurement and Recording 🕮)

New Bus Navigator Features (ControlDesk 2022-B)

Bus Navigator (stand-alone)

The new Bus Navigator (stand-alone) lets you use the ControlDesk Bus Navigator functionalities with some restrictions. For example, going online and measuring is only supported for bus monitoring devices.

Using the Bus Navigator (stand-alone) requires a valid ControlDesk Bus Navigator Module (CONTROLDESK_BNV) license only. No license for the ControlDesk main version is required. You can upgrade to the full-featured main version of ControlDesk by adding the CONTROLDESK license later on.

The Bus Navigator (stand-alone) supports the following features:

- Monitoring and logging communication on CAN, Ethernet, and LIN buses by using bus monitoring devices.
- Replaying logged CAN and Ethernet communication.
- Instruments for handling CAN, LIN, and Ethernet messages, frames, and PDUs.
- Bus interfaces in connection with a bus monitoring device:
  - dSPACE and supported third-party PC-based bus interfaces.
  - Bus channels of SCALEXIO, MicroAutoBox III, or VEOS.

  For more information, refer to Supported Interfaces for Accessing a Communication Bus (ControlDesk Platform Management 🕮).

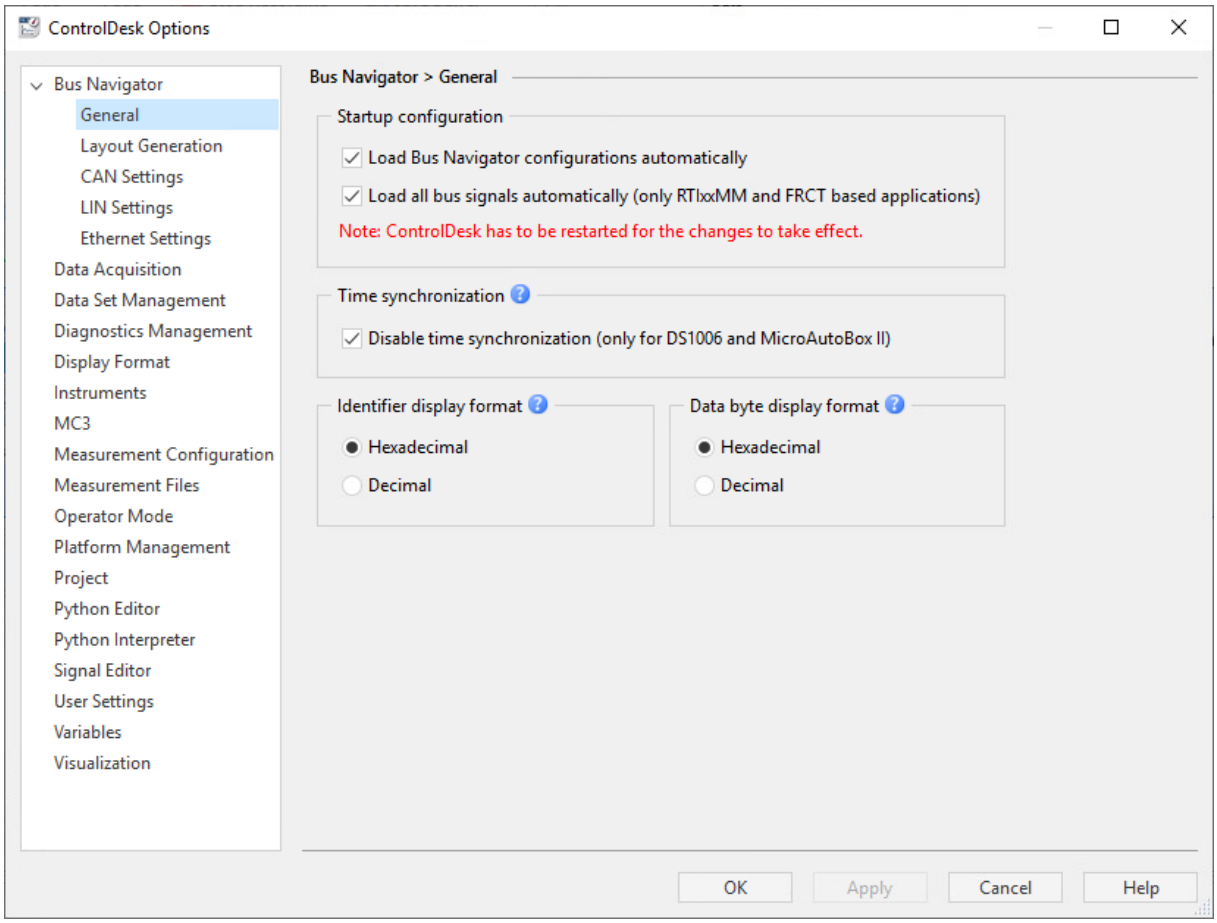
Note

As of ControlDesk 2022-B, a valid license for the Bus Navigator Module is required to access CAN-based dSPACE real-time hardware and PC-based CAN interfaces (e.g., DCI-CAN2) for handling CAN messages, and for monitoring and logging CAN bus communication. This functionality was moved from the main version license to the Bus Navigator Module license.

For more information on the Bus Navigator (stand-alone), refer to Using the Bus Navigator (Stand-Alone) (ControlDesk Bus Navigator 🕮).

Improvements to the Bus Navigator user interface

Revised Bus Navigator page (ControlDesk Options dialog) The Bus Navigator page, which is part of the ControlDesk Options dialog, was revised. The following illustration shows the new structure:



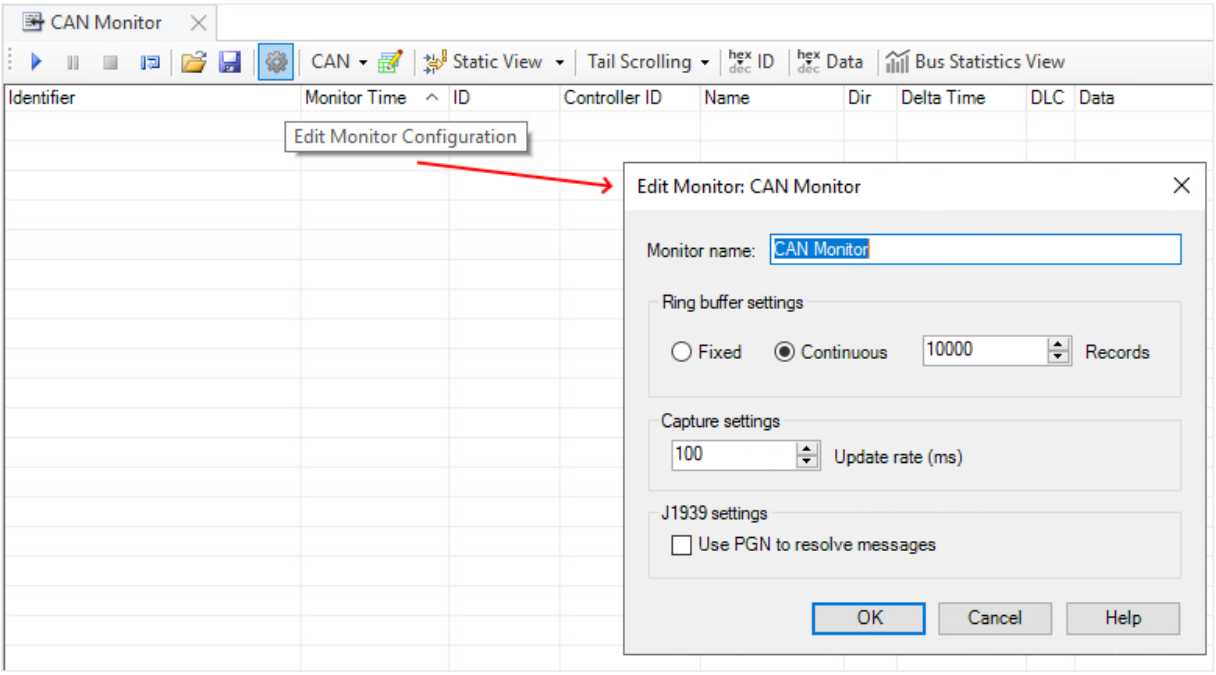
For more information, refer to Bus Navigator Page (ControlDesk Bus Navigator 🕮).

Display of further Bus Navigator element properties

- The ControlDesk **Properties** pane now also displays information on the following elements selected in the **Bus Navigator** pane:
  - Channel
  - Controller
  - Global time domain
- The **Source** of elements selected in the **Bus Navigator** pane, i.e., the name of the implementation software and/or the communication matrix file type now is displayed.
- PDU properties specific to container IPDUs are displayed.

For more information, refer to Properties Related to Bus Navigator Elements (ControlDesk Bus Navigator 🕮).

Improved access to Edit Monitor dialog The new ⚙ button in the toolbar of a monitoring list now lets you easily access the **Edit Monitor** dialog.



Refer to Add/Edit Monitor Dialog (ControlDesk Bus Navigator 🕮).

Support of R21-11 ARXML files

The ControlDesk Bus Navigator now also supports AUTOSAR system description (ARXML) files according to the R21-11 version of the AUTOSAR system template.

Refer to Variable Descriptions Supported by ControlDesk (ControlDesk Variable Management 🕮).

New Automation Features (ControlDesk 2022-B)

Writing individual function and axis values of variables connected to a Table Editor

- The `IViTableEditorInstrument` automation interface now provides the `SetSpecificFunctionValues` method to set individual function values of a variable connected to a Table Editor.

  For more information, refer to TableEditorInstrument / IViTableEditorInstrument <<Interface>> (ControlDesk Automation 🕮).
- The `IViValueAxis` automation interface now provides the `SetSpecificValue` method to set individual axis values of a variable connected to a Table Editor.

  For more information, refer to ValueAxis / IViValueAxis <<Interface>> (ControlDesk Automation 🕮).

Switching the data source of layouts

The automation interface of layouts now provides the `SwitchDataSource` property to switch the data sources of variable connections within a layout. This lets you compare values of different recordings and the current measurement.

For more information, refer to LayoutDocument / IXaLayoutDocument <<Interface>> (ControlDesk Automation 🕮).

Specifying output parameters of calculated variables

The `IXaCalculatedVariable` automation interface now provides the `OutputParameters` property to specify output parameters for calculated variables.

For more information, refer to CalculatedVariable / IXaCalculatedVariable <<Interface>> (ControlDesk Automation 🕮).

Finding variables in all variable descriptions

The `IXaVariablesManagement` automation interface now provides the `FindVariable` method that lets you search for variables in all variable descriptions of the currently loaded experiment.

For more information, refer to VariablesManagement / IXaVariablesManagement <<Interface>> (ControlDesk Automation 🕮).

Migrating to ControlDesk 2022-B

Migrating to ControlDesk 2022-B

Introduction

To migrate from ControlDesk 7.6 to ControlDesk 2022-B and reuse existing experiments, you might have to carry out the following migration steps.

Note

To migrate to ControlDesk 2022-B from versions earlier than 7.6, you also have to perform the migration steps of the intervening ControlDesk versions.

CONTROLDESK_BNV license required for handling CAN bus communication

As of ControlDesk 2022-B, the CONTROLDESK_BNV license for the ControlDesk Bus Navigator Module now is required to access CAN-based dSPACE real-time hardware and PC-based CAN interfaces (e.g., DCI-CAN2) for handling CAN messages, and for monitoring and logging CAN bus communication.

DSSIGCONV tool changes in ControlDesk 2022-B

No extraction of single elements of measurement arrays and value blocks The `/s:list` and `/sl:filename` parameters of the DSSIGCONV command line tool no longer support the specification of single elements of measurement arrays and value blocks, i.e., you can no longer extract single elements of measurement arrays and value blocks from the source measurement data file.

Removal of linear identity scalings Linear identity scalings (factor = 1.0, offset = 0.0) in the source destination file are removed during export.

Unsupported scaling formats The following scaling formats are not supported:

- %c
- %C
- %n
- %o
- %p
- %s
- %S
- %Z

The following formats are changed to the float format (%f).

- %a
- %A
- %e
- %E

- %g
- %G

For more information on the DSSIGCONV tool, refer to Using DSSIGCONV (ControlDesk Measurement and Recording 🕮).

Measurement Data API changes in ControlDesk 2022-B

ControlDesk 2022-B provides a new implementation of the Measurement Data API. As a consequence, you may have to adapt existing scripts.

Measurement object invalid after removing it from the Measurements collection As of ControlDesk 2022-B, when you remove a `Measurement` from the `Measurements` collection, the `Measurement` object becomes invalid. You get an exception when you try to access it.

Elements of measurement arrays and value blocks no longer part of the Signals Collection The content of `Signals` collections containing measurement arrays and value blocks, i.e., signals of the `vtMeasurementArray` and `vtValueBlock` type:

- Up to and including ControlDesk 7.6, a `Signals` collection containing measurement arrays and value blocks also contained the individual array elements.
- As of ControlDesk 2022-B, a `Signals` collection containing measurement arrays and value blocks no longer contains the individual array elements

As a consequence, you have to access the data of an individual array element *by its index*. The following code excerpt shows an example:

```
>>> Signal = Measurement.Signals[“PLATFORM/ArraySignal/5ms”]
>>> Signal.Data[1]
(0.01, 0.011, 0.012, 0.013, 0.014)
```

Scalings now optional The specification of signal scalings was changed in ControlDesk 2022-B.

- Up to and including ControlDesk 7.6, when you added a signal to the `Signals` collection, the specification of a scaling was mandatory. For signals without conversion, a linear identity scaling (factor = 1.0, offset = 0.0) was required.
- As of ControlDesk 2022-B, when you add a signal to the `Signals` collection, specify the `None` scaling for signals without conversion.

```
>>> SignalWithoutSourceData = Measurement.Signals.Add('Without Source Data', 'Without Source Data', 'Device', '',
SignalData, XAxis, None)
```

When you load a measurement data file, linear identity scalings are removed.

Adding overlapping table scalings no longer possible As of ControlDesk 2022-B, when you add a table scaling with overlapping table range specifications, you get an exception.

Rational function scalings now convert from source to converted The behavior of rational function scalings, i.e., scalings of the `stRationalFunction` type, was changed in ControlDesk 2022-B.

- Up to and including ControlDesk 7.6, rational function scalings were defined to convert measurement values *from converted to source*.

- As of ControlDesk 2022-B, rational function scalings are defined to convert measurement values *from source to converted*, which is consistent with the other scaling types.

Unique raster names As of ControlDesk 2022-B, the Measurement Data API requires unique names of the measurement rasters of a platform. Therefore, when you load a measurement data file, the Measurement Data API disambiguates raster names if necessary.

The following table shows a raster name as an example.

Up to and Including ControlDesk 7.6	As of ControlDesk 2022-B
`PLATFORM/Signal/5ms` (same name for two different rasters)	▪ `PLATFORM/Signal/5ms` ▪ `PLATFORM/Signal/5ms(2)` (after disambiguation)

As a consequence, the keys of the x-axes and signals may be different when you reuse scripts with the Measurement Data API of ControlDesk 2022-B. The keys of the rasters are available from the `Signal` objects.

Unsupported scaling formats The following scaling formats are not supported:

- %c
- %C
- %n
- %o
- %p
- %s
- %S
- %Z

The following formats are changed to the float format (%f).

- %a
- %A
- %e
- %E
- %g
- %G

GetDeviceOptions/SetDeviceOptions methods discontinued The following methods of the `Measurement` object are obsolete. As of ControlDesk 2022-B, they are no longer available:

- `GetDeviceOptions`
- `SetDeviceOptions`

Changed signal order in the Signals collection As of ControlDesk 2022-B, when you load a measurement data file, the order of `Signal` objects in the `Signals` collection differs from the order when loading the file in earlier versions of the Measurement Data API.

No multiple loading of the same measurement data file As of ControlDesk 2022-B, you get an exception when you load the same measurement data file more than once.

However, you can load a measurement data file with the same name *if it is located in another folder* and if you specify a different name for the `Measurement` object.

The following script excerpt from the `LoadSaveAndExport.py` demo script shows an example:

```
Measurement = Measurements.Load(SubFolderFilePath, Name = 'Measurement from sub-folder')
```

For more information on the Measurement Data API, refer to Basics on the Measurement Data API (ControlDesk Measurement Data API 🕮).

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 New Features and Migration 🕮).

Related topics

Basics

Basics on Migrating from Prior Versions of ControlDesk (ControlDesk New Features and Migration 🕮)

dSPACE AUTOSAR Compare

Where to go from here

Information in this section

New Features of dSPACE AUTOSAR Compare 2022-B

General improvements

The new version also supports AUTOSAR Release R21-11.

Related topics

Basics

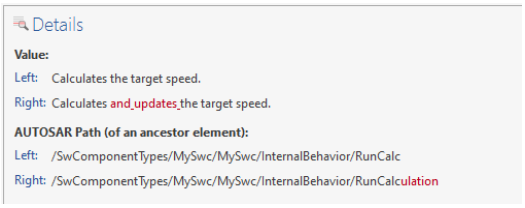
Basics on dSPACE AUTOSAR Compare (dSPACE AUTOSAR Compare Manual 🕮)

New Features of dSPACE AUTOSAR Compare 1.1

General improvements

- The new version also supports AUTOSAR Release R20-11.
- You can now specify an AUTOSAR path to go to elements on a comparison page.

- The added **Details** pane highlights the differences of selected AUTOSAR elements and displays the full AUTOSAR path.



Integration into version control systems

dSPACE AUTOSAR Compare can be integrated into version control systems, such as Git, to view differences between different revisions of ARXML files and to assist with merge conflicts. The following sections for *adding dSPACE AUTOSAR Compare*, *viewing differences*, and *solving merge conflicts* focus on Git.

Adding dSPACE AUTOSAR Compare

You can add dSPACE AUTOSAR Compare as a diff and a merge tool in Git. You can use dSPACE AUTOSAR Compare as the default tool for the aforementioned purposes.

Viewing differences

You can use the following command to inspect local changes to a file that has not been committed:

```
git difftool -t darc <file>
```

dSPACE AUTOSAR Compare opens a session in which you can compare the file contents between the last commit and your local changes.

You can omit '-t darc' if you set up dSPACE AUTOSAR Compare as the default Git diff tool. If you do not specify a file name, a session will be opened for all changed files.

Solving merge conflicts

When merging or rebasing branches, Git notifies you of any merge conflicts with this message:

```
Auto-merging <file>
CONFLICT (content): Merge conflict in <file>
Automatic merge failed; fix conflicts and then commit the result.
```

You can use the following command to solve the merge conflicts with dSPACE AUTOSAR Compare:

```
git mergetool -t darc <file>
```

You can omit '-t darc' if you set up dSPACE AUTOSAR Compare as the default Git merge tool.

If mergetool.prompt is set to 'true' in your Git configuration, Git now shows further information and lets you solve merge conclicts manually in dSPACE AUTOSAR Compare. Otherwise, the merge conclicts are solved automatically.

Related topics

Basics

Integration in Other Tools (dSPACE AUTOSAR Compare Manual 🕮)

DCI Configuration Tool

New Features of the DCI Configuration Tool 2022-B

Firmware version for DCI-GSI2 interfaces

For the DCI-GSI2 interfaces, the firmware version 1.5.5 is delivered with the DCI Configuration Tool 2022-B.

Note

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

dSPACE FlexRay Configuration Package

New Features of dSPACE FlexRay Configuration Package 2022-B

FlexRay Configuration Package

Default value of source switch variable Up to and including FlexRay Configuration Package 4.9, the source switch variables are always initially set to the Simulink model. The source switch variable values, which control the data source, can be changed at run time and only individually for each TRC file variable.

As of FlexRay Configuration Package 2022-B, you can specify the default value for the source switch variables in the FlexRay Configuration Tool, i.e., before code generation. On the **Generators** page of the **General Properties** dialog, you can select the global default value for all source switch variables, i.e., you can specify whether the initial data source is set to TRC variables or the Simulink model by default.

Refer to Generators Page (FlexRay Configuration Tool Reference 🕮).

dSPACE Installation Manager

Where to go from here

Information in this section

New Features of dSPACE Installation Manager 22.2

New features for managing installations

dSPACE Installation Manager can now check and display whether an update (such as a patch) is available for your installed dSPACE software. This can be done manually or automatically. In the latter case, dSPACE Installation Manager uses notifications to inform you if an update is available.

Note

Be aware of the following restrictions:

- Only updates for dSPACE software released with dSPACE Release 2020-A and later are displayed.
- For the following software, no updates are displayed, even if updates are available:
  - dSPACE Firmware Manager, dSPACE Firmware Archives
  - dSPACE Installation Manager
  - Third-party software provided with a dSPACE Release, for example, software from Wibu-Systems
  - dSPACE software that is not delivered within a dSPACE Release, for example, software products from the dSPACE Solutions collection or AURELION
- Hotfixes are not supported and therefore not displayed in general.

For more information, for example, on specific administrator use cases, refer to Getting Information on Available dSPACE Software Updates (Managing dSPACE Software Installations 🕮).

New features for managing licenses

Now you can also use Ubuntu Linux as the operating system on a server for floating network licenses in combination with protected dSPACE software that is running unter Microsoft Windows. Contact dSPACE Support (www.dspace.com/go/supportrequest) for more details.

Migrating to dSPACE Installation Manager 22.2

Using CmDongles

If you want to work with licenses on CmDongles in combination with dSPACE Installation Manager 22.2, for example, to activate, deactivate, or update licenses, the dongles must have at least 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 minimum firmware version.

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

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

dSPACE Python Extensions

New Features of dSPACE Python Extensions 2022-B

New features

The dSPACE Python Extensions do not have new features.

dSPACE XIL API .NET

Where to go from here

Information in this section

New Features of dSPACE XIL API .NET 2022-B

New features

The dSPACE XIL API.NET 2022-B has the following new features:

- The new ECUPort implementation provides the following first features:
  - Reading calibration values.
  - Writing calibration values.
  - Downloading parameter sets (CDFX).
  - Supported variables are scalars, arrays, curves, and maps of any data type.
  - Only V-ECUs are supported.
  - Available for Windows and Linux.

  For more information, refer to dSPACE XIL API ECUPort Implementation 🕮.
- To fullfill the requirements of the new ECUPort implementation, the `VariableInfo` interface has been enhanced.

  The following features are now available:
  - You can get the axis types of curve, matrix, and map variables.
  - You can get information on the conversion method.
  - You can get a description and a physical unit.

  These features are now also available for the MAPort implementation, because it also uses the `VariableInfo` interface.

Migrating to dSPACE XIL API .NET 2022-B

Migrating to XIL API 2.2.0

Changes in the source code To migrate your applications to the ASAM XIL API 2.2.0 standard, you have to modify the references:

- C#: In the Solution Explorer, open the Add Reference dialog to replace the DLL references by the references from version 2.2.

  Then, recompile your C# client application.

- Python: Modify the version string and the public key tokens. With ASAM XIL API 2.2.0, all assemblies have the same public key token.

  The updated references should look like the following example.

```
# Load ASAM assemblies from the global assembly cache (GAC)
clr.AddReference("ASAM.XIL.Implementation.TestbenchFactory, Version=2.2.0.0, Culture=neutral,
PublicKeyToken=bf471dff114ae984")
clr.AddReference("ASAM.XIL.Interfaces, Version=2.2.0.0, Culture=neutral, PublicKeyToken=bf471dff114ae984")
```

Changes in the framework configuration The framework configurations for XIL API 2.1.0 and 2.2 are not compatible. If an incorrect XML schema is used, an exception occurs, displaying that the XML file is not valid.

Because there is now a different way to configure the target state, the following examples show the entire framework configurations for the SCALEXIO platform. The modifications are marked.

XIL API 2.1.0

```
<?xml version="1.0" encoding="utf-8"?>
<FrameworkConfiguration xmlns="http://www.asam.net/XIL/FrameworkConfiguration/2.0">
  <MappingFileList>
    <MappingFile Name="DemoFrameworkLabels">..\FrameworkMappings\FrameworkLabels.xml</MappingFile>
    <MappingFile Name="DemoLabelMapping_SCALEXIO">..\FrameworkMappings\LabelMapping_SCALEXIO.xml</MappingFile>
  <MappingFile Name="DemoUnits">..\FrameworkMappings\Units.xml</MappingFile>
  </MappingFileList>
  <PortDefinitionList>
    <MAPortDefinition InstanceName="dSPACE SCALEXIO MAPort" InitOrder="0" ShutdownOrder="0" TargetState="eSIMULATION_RUNNING">
      <VendorName>dSPACE GmbH</VendorName>
      <ProductName>XIL API</ProductName>
      <ProductVersion>2021-B</ProductVersion>
      <PortConfigurationFile>..\..\..\MAPort\Common\PortConfigurations\MAPortConfigSCALEXIO.xml</PortConfigurationFile>
    </MAPortDefinition>
  </PortDefinitionList>
</FrameworkConfiguration>
```

XIL API 2.2.0

```
<?xml version="1.0" encoding="utf-8"?>
<FrameworkConfiguration xmlns="http://www.asam.net/XIL/FrameworkConfiguration/2.2.0">
    <MappingFileList>
        <MappingFile Name="DemoFrameworkLabels">..\FrameworkMappings\FrameworkLabels.xml</MappingFile>
        <MappingFile Name="DemoLabelMapping_SCALEXIO">..\FrameworkMappings\LabelMapping_SCALEXIO.xml</MappingFile>
        <MappingFile Name="DemoUnits">..\FrameworkMappings\Units.xml</MappingFile>
    </MappingFileList>
    <PortDefinitionList>
        <MAPortDefinition InstanceName="dSPACE SCALEXIO MAPort" InitOrder="0" ShutdownOrder="0">
            <VendorName>dSPACE GmbH</VendorName>
            <ProductName>XIL API</ProductName>
            <ProductVersion>2022-A</ProductVersion>
            <TargetState>
                <eSIMULATION_RUNNING>
                    <PortConfigurationFile>..\..\..\MAPort\Common\PortConfigurations\MAPortConfigSCALEXIO.xml</PortConfigurationFile>
                    <ForceConfig>false</ForceConfig>
                </eSIMULATION_RUNNING>
            </TargetState>
        </MAPortDefinition>
    </PortDefinitionList>
</FrameworkConfiguration>
```

Changes in the mapping file The mapping files for XIL API 2.1.0 and 2.2 are not compatible. If an incorrect XML schema is used, an exception occurs, displaying that the XML file is not valid.

You only have to change the following entry.

XIL API 2.1.0

```
<Mapping xmlns="http://www.asam.net/XILAPI/Mapping/2.1.0">
```

XIL API 2.2.0

```
<Mapping xmlns="http://www.asam.net/XIL/Mapping/2.2.0">
```

ECU Interface Manager

Where to go from here

Information in this section

New Features of ECU Interface Manager 2022-B

Preparing dynamic access to ECU variables

The ECU Interface Manager now lets you prepare *dynamic access to ECU variables*. As a result, you can change the number of variables to be read or written as well as their addresses, data types, and variable conversions during run time of the related real-time application.

For more information, refer to Basics on Configuring Dynamic Data Accesses (ECU Interface Manager Manual 🕮).

XCP on CAN: Assigning specific CAN IDs

For XCP on CAN, you can now specify a list of CAN IDs for the transmission of XCP messages with DAQ lists. Depending on whether static or dynamic DAQ lists are used, and on whether PID transmission is enabled or not, CAN IDs specified in the A2L file or in the ECU Interface Manager are used for XCP message transmission.

Refer to:

- Data Access (ECU Interface Manager Manual 🕮)
- Dynamic Data Access (ECU Interface Manager Manual 🕮)

- ECU Interface (ECU Interface Manager Manual 🕮)
- Function (ECU Interface Manager Manual 🕮)

XCP on CAN: Disabling PID transmission

The ECU Interface Manager now lets you disable packet identifier (PID) transmission in connection with the XCP on CAN ECU interface. This increases the DAQ data throughput since all the bytes of an XCP data transfer object (DTO) packet are used for DAQ data transfer.

Refer to:

- Data Access (ECU Interface Manager Manual 🕮)
- Dynamic Data Access (ECU Interface Manager Manual 🕮)
- Function (ECU Interface Manager Manual 🕮)

Compatibility of ECU Interface Manager 2022-B

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 2.9[1]	Version 2.10[2]	Version 2.11[3]	Version 2022-B[4]
ConfigurationDesk 2022-B[4]	✓[5]	✓[5]	✓[6]	✓[6]
ConfigurationDesk 22.1[3]	✓[5]	✓[5]	✓[6]	✓[6]
ConfigurationDesk 6.8[2]	✓	✓	–	–
ConfigurationDesk 6.7[1]	✓	✓	–	–

1) dSPACE Release 2021-A
2) dSPACE Release 2021-B
3) dSPACE Release 2022-A
4) dSPACE Release 2022-B
5) For 32-bit target architecture only.
6) For 32- and 64-bit target architectures.

Compatibility between EIC files and RTMaps

RTMaps supports EIC files created with the ECU Interface Manager 2.10 and later.

Migrating to ECU Interface Manager 2022-B

Automatic migration of projects

You can reuse projects in the ECU Interface Manager 2022-B 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 2022-B, they are migrated automatically.

Note

In the ECU Interface Manager 2022-B, 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 2022-B 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 22.2

General enhancements

Usually, the Firmware Manager can start a required reboot after a firmware update. For some dSPACE platforms, this behavior was not supported. The user was informed to manually reboot the hardware.

Now, the automatic reboot when updated to firmware version 22.2 is, under some conditions, available for the following platforms:

- MicroAutoBox III

  The previously loaded DS1403 firmware version must be equal to or greater than 5.0, i.e., the installed Firmware Archives setup version must be 3.2 (dSPACE Release 2020-B) or higher.
- SCALEXIO AutoBox/LabBox

  The previously loaded DS6001 firmware version must be equal to or greater than 5.0, i.e., the installed Firmware Archives setup version must be 3.2 (dSPACE Release 2020-B) or higher.
- SCALEXIO Real-Time PC

  The DS2502 IOCNET Link Board must have a CPLD version equal to or greater than 4.0 and an FPGA version equal to or greater than 5.1. The previously loaded firmware version must be equal to or greater than 5.1, i.e., the installed Firmware Archives setup version must be 3.3 (dSPACE Release 2021-A) or higher.

FPGA Programming Blockset

Where to go from here

Information in this section

New Features of the FPGA Programming Blockset 2022-B

Extended Xilinx® support

The 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 2022.1[2)]	▪ MATLAB R2021a ▪ MATLAB R2021b	Windows operating system that is supported by the RCP and HIL software of the current Release. Exception: Only the FPGA Programming Blockset-Processor Interface supports Windows 11. Windows 11 cannot be used to model or handcode FPGA applications due to missing Xilinx support. For a list of supported operating systems, refer to Operating System on page 236. The listed Windows Server operating systems are not officially supported by Xilinx, but tested by dSPACE.

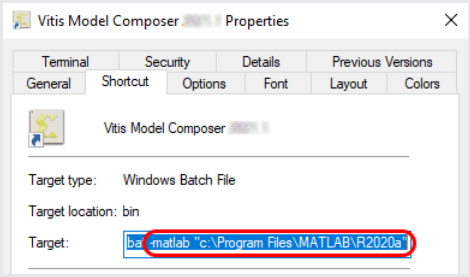
1) The **Processor Interface** sublibrary of the FPGA Programming Blockset also supports MATLAB R2022a and R2022b.

2) In general, Vivado ML Enterprise edition and the Xilinx® Vitis™ Model Composer are needed. The Vivado ML Standard edition with Xilinx Vitis Model Composer can also be used for the DS2655 (7K160) and DS6601 FPGA base boards. The Xilinx Vitis Model Composer is required only for modeling FPGA applications with the FPGA Programming Blockset.

As of Vivado 2021.1, the Xilinx System Generator for DSP is part of the Vitis Model Composer. This is the unified Xilinx add-on for MATLAB Simulink. For licensing issues related to this change, refer to https://xilinx.com/support/answers/76039.html.

Tip

To use the Vitis Model Composer with a specific MATLAB version, the `-matlab "<matlabdir>"` parameter must be specified. You can add this parameter to the Vitis Model Composer icon on the desktop, for example.



Enhancements to the MicroAutoBox III/SCALEXIO frameworks

Preprocessing and postprocessing of processor signals The SCALEXIO and the MicroAutoBox III blocksets now support scaling subsystems as a part of the FPGA model. Scaling subsystems lets you specify the preprocessing and postprocessing of the signals between the FPGA and the real-time processor. Scaling subsystems are executed on the real-time processor even though they are part of the FPGA model.

Scaling subsystems can be used for the following points:

- Scaling the signals.
- Implementing complex algorithms to process the signals. For example, an FPGA input signal can be calculated from several processor output signals.
- Grouping of signals for more clarity.

For more information, refer to Basics on Preprocessing and Postprocessing the Processor Communication (FPGA Programming Blockset Guide 🕮).

New container file format for the build result files The FPGA Programming Blockset now supports the FPGAC file format for the build results. FPGAC files let you manage FPGA applications in repositories. If you add or update an FPGAC file to a search path of ConfigurationDesk, ConfigurationDesk adds the FPGA custom function block type to the Functions browser and updates instantiated function blocks if you open the ConfigurationDesk application or you reload the custom functions.

For more information, refer to How to Add FPGA Custom Function Block Types Provided by an FPGAC File (ConfigurationDesk I/O Function Implementation Guide 🕮).

New script function for MicroAutoBox III/SCALEXIO The script interface is enhanced with a script function to update an existing FPGAC file, for example, to implement a changed scaling subsystem without building a new FPGA application.

For more information, refer to How to Update an FPGAC File with Modified Scaling Subsystems (FPGA Programming Blockset Guide 🕮).

Enhancements to the SCALEXIO frameworks

64-bit fixed-point support for processor communication The data types without a binary position (Fix_64_0 and UFix_64_0) for the **Register64** and **Buffer64** access types now support the full 64-bit fixed-point resolution.

All other 64-bit fixed-point data types are still converted to double and their fixed-point resolution is restricted to 53 bits. This also applies to the **Bus** access types.

Note

The default data type of generated processor interface blocks is always *double*. To use the full resolution of a 64-bit fixed-point data type, you have to set the data type of the generated blocks to *int64* or *uint64*. Refer to How to Transfer 64-Bit Fixed-Point Data with Full Resolution (SCALEXIO) (FPGA Programming Blockset Guide 🕮).

General enhancements to the FPGA Programming Blockset

Backup of the build results Before you build an FPGA application, you can now enable and configure the automatic backup of the FPGA application, models, and reports that are generated and used during the build process. A link to the backup folder is provided in the MATLAB Command Window after the build process finished.

For configuring the automatic backup, refer to Backing Up the Build Results (FPGA Programming Blockset Guide 🕮).

New reports about the model and build results The framework now provides model and build reports that are generated at the end of an FPGA build.

The model report contains information about the FPGA channels, software and tools used. The build report contains information about the FPGA utilization and the build duration.

Links to the reports are provided in the MATLAB Command Window after the build process finished. The reports are located in the build folder or in the backup folder.

Word length calculator dialog The word length calculator now lets you calculate an optimal fixed-point data type for a user-defined floating-point value. Refer to Word Length Calculator (FPGA Programming Blockset - FPGA Interface Reference 🕮).

Enhancement to the FPGA Build Monitor

The FPGA Build Monitor 2022-B now provides the warning messages, if any were issued during the build process.

For more information, refer to FPGA Build Monitor 2022-B Reference (FPGA Programming Blockset Guide 🕮).

Migrating to the FPGA Programming Blockset 2022-B

Migrating from previous FPGA Programming Blocksets

If you implemented an FPGA application with the FPGA Programming Blockset Version 1.1 and later and want to use it with the FPGA Programming Blockset 2022-B, the framework automatically updates to the current framework version.

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

You can also use a script to migrate processor interface blocks, for example, to migrate RTI processor models without the FPGA model. For more information, refer to MigrateToModelPortBlocks (FPGA Programming Software Script Interface Reference 🕮).

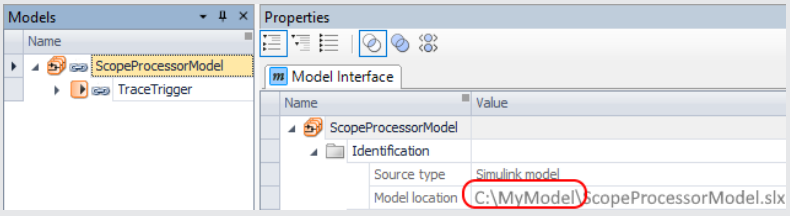
Changed process to generate FPGA scope rasters If you use an FPGA custom function built with the FPGA Programming Blockset 3.13 (Release 2022-A) in ConfigurationDesk, you have to generate the FPGA scope raster that is used by ControlDesk.

As of FPGA Programming Blockset 2022-B, an FPGA custom function block with FPGA scope functions automatically generates the FPGA scope raster for ControlDesk.

If you use an existing ConfigurationDesk project and update the FPGA custom function of Release 2022-A with a newer FPGA custom function, the existing FPGA scope raster (`<application>_usr.trc`) leads to an error.

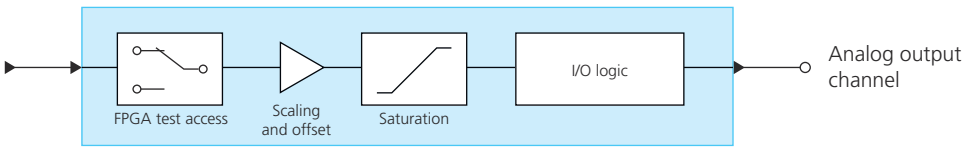
Note

Before you update the existing FPGA custom function in ConfigurationDesk with a new FPGAC file, delete the `<application>_usr.trc` file in the model folder. ConfigurationDesk shows the location of the model.



Changed interface to modify analog output signals FPGA scaling lets you modify analog output signals. As of Release 2022-B, the interface scales the signal values that are replaced with FPGA test access.

The following illustration shows the analog output interface. For more information on FPGA test access and scaling, refer to Basics on FPGA Test Access and Scaling (FPGA Programming Blockset Guide 🕮).



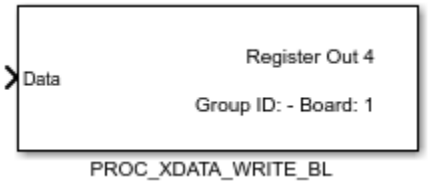
FPGA_IO_WRITE_BL

Up to Release 2022-A, the interface scales signal values first and then you can replace the scaled values with FPGA test access.

Bus In blocks replace Buffer64 blocks As of RTI FPGA Programming Blockset 3.12, the **Bus In/Bus Out** blocks for the MicroAutoBox III/SCALEXIO frameworks replace the transfer mode of the **Buffer64 In/Buffer64 Out** blocks. The update process automatically replaces the **Buffer64 In/Buffer64 Out** blocks that use the bus transfer mode with **Bus In/Bus Out** blocks.

Display of migrated processor interfaces With the FPGA Programming Blockset 3.4 ... 3.8, you modeled the processor interface of a SCALEXIO system/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
	

Related topics

Basics

Migrating and Updating Existing FPGA Models (FPGA Programming Blockset Guide 🕮)

MicroAutoBox III Firmware

New Features of the MicroAutoBox III Firmware 22.2

DS1403 Processor Board	The DS1403 Processor Board now supports the use of a real-time-application with an activated data logging configuration. It is no longer necessary to use ControlDesk to load the data logging configuration and the related real-time application (RTA) file to the flash memory of the MicroAutoBox III. As a consequence, the ControlDesk PC no longer needs to be connected to the real-time hardware to load a data logger configuration to it. For more information, refer to Data Logging (ControlDesk Measurement and Recording 🕮).
DS1521 Bus Board	The DS1521 Bus Board now supports full duplex RS485 mode.

Model Compare

Where to go from here

Information in this section

New Features of Model Compare 2022-B

Improved Difference Overview

Model Compare 2022-B provides an improved Difference Overview dialog, which gives you a quick overview of changed model parts. Subsystems, which contain child elements with differences, are now displayed in bold letters and can be identified more easily. Beside SLX files the Difference Overview dialog now supports OPC (Open Packaging Conventions) MDL files, which are new with MATLAB R2021b.

Related documentation

- Difference Overview (Model Compare Reference 🕮)

Improved workflow integration

Model Compare 2022-B provides an improved workflow integration:

- Dialogs are suppressed in batch mode.
- CMD proxy scripts for version control systems are provided.
- Libraries are automatically unlocked during Comfort Copy.
- Mask parameters are supported in Comfort Copy.
- Added/deleted elements of subsystems are, by default, listed in the difference report.
- With the textconv driver, git diff and blame on SLX and OPC MDL files give you quick and readable results. The `ModelCompare.VCSTextConv.cmd` is located in the installation folder of Model Compare. Please contact dSPACE Support for detailed information.

Related documentation

- The ModelCompare [Options] API Command (Model Compare Reference 🕮)
- Migration to Model Compare 2022-B on page 107

Improved filtering

With Model Compare 2022-B, filtering is improved for Block Type and Property filters. Wildcards such as * and ? let you now set your filters more easily and flexibly.

Related documentation

- Block Type Filter Page (Model Compare Reference 🕮)
- Property Filter Page (Model Compare Reference 🕮)

Improved support of large bus blocks

With Model Compare 2022-B, changes in large bus blocks, e.g., a Bus Creator block, are easier to identify in comparison results and difference reports.

Related documentation

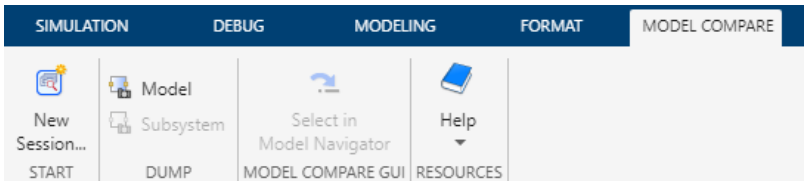
- Display of bus signals (Model Compare Reference 🕮)

New tutorial videos

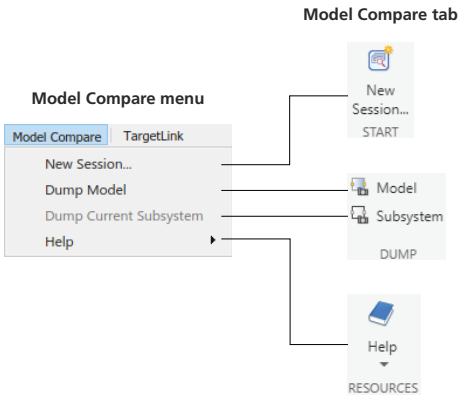
Model Compare provides several tutorial videos to support you in your daily work. Refer to the latest tutorial video Model Compare Three-Way Merge (4:17) ▶ in dSPACE Help.

New Model Compare tab on the Simulink Toolstrip

With Model Compare 2022-B, the new Model Compare tab on the Simulink Toolstrip is introduced.



Up to MATLAB R2021b, the following commands are available in the Model Compare menu. As of MATLAB R2022a, the commands are available in the new Model Compare tab on the Simulink Toolstrip.



Model Compare Menu (Up to MATLAB R2021b)	Model Compare Tab (As of MATLAB R2022a)
New Session	START - New Session
Dump Model	DUMP - Model
Dump Current Subsystem	DUMP - Subsystem
Help - dSPACE Help	RESOURCES - Help - Help
Help - Useful Links - dSPACE Website	RESOURCES - Help - dSPACE Website
Help - Useful Links - TargetLink Support Center	RESOURCES - Help - dSPACE Support Request
-	MODEL COMPARE GUI - Select in Model Navigator
-	RESOURCES - Help - New Features and Migration
-	RESOURCES - Help - Using dSPACE Help

Migration to Model Compare 2022-B

Changes concerning difference reports

With Model Compare 2022-B, the displayed content of the difference report changes slightly. Elements of added or deleted subsystems are now displayed per default. This helps you to review changes in the difference report that you can automatically generate during a pull request, for example. Refer to Change is the Only Constant. Continuously Reviewing Binary Resources via Pull Requests.

To restore the old behavior, select the **Ignore content changes of subsystems without a corresponding system** checkbox on the **Basic Filters** page of the **Comparison Settings** dialog. Refer to Basic Filters Page (Model Compare Reference 🕮).

Note

It is recommended to use only XML dump files that were created with this version of Model Compare. Otherwise, merge operations and some of the new features of Model Compare will not be available.

Model Container Utility

About Model Container Utility

Introduction

The Model Container Utility lets you modify existing model implementation containers and package containers (PKC files) without having to generate them again.

Complementation of the dSPACE SIL testing tool chain

The Model Container Utility complements the dSPACE SIL testing tool chain. It is useful especially in workflows where model containers are exchanged between OEMs and suppliers:

- The utility lets you *protect intellectual property rights* by letting you exclude IP-relevant files from an existing model implementation container or package.
- The utility lets you *compile* an existing model implementation container or package container *in advance*. You can also compile only parts of each, i.e., packages.
- The command line interface of the Model Container Utility lets you easily compile containers or packages *for many users as part of a continuous build scenario*.

Supported container types and simulation platforms

- The Model Container Utility currently supports the following container types:
  - V-ECU implementation container (VECU) files
  - Package container (PKC) files
- The Model Container Utility currently supports container compilation for the VEOS simulation platform.

For more information, refer to Basics on the Model Container Utility (Model Container Utility Manual 🕮).

Model and Sensor Interface Blockset

Where to go from here

Information in this section

New Features of Model and Sensor Interface Blockset 2022-B

Model and Sensor Interface Blockset

As of dSPACE Release 2020-B, the Model and Sensor Interface Blockset (MSI Blockset) must be used for simulation with sensors in a software-in-the-loop (SIL) or in a hardware-in-the-loop (HIL) test environment.

In Sensor Simulation, the blockset is required to calculate simulation data and transmit it to the MotionDesk PC for visualization and to the 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. The unit splits the sensor raw data coming from the SensorSim application and inserts it into the relevant sensor hardware, for example, into imaging sensor interfaces of camera sensors. You can add blocks to the Simulink model to receive feedback on the connected

sensors from the Environment Sensor Interface Unit. You can also configure sensor failures for specific 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.

For detailed information on the features of the Model and Sensor Interface Blockset, refer to Features of the Model and Sensor Interface Blockset (Model and Sensor Interface Blockset Manual 🕮).

New features

The following new features for the Model and Sensor Interface Blockset 2022-B are available:

- **TCP Server connection for VEOS**: An Ethernet TCP Server block is provided in the **Utilities** subsystem to connect your Sensor Simulation systems and applications to a VEOS simulation platform for software-in-the-loop simulations.
- **Simulink simulation**: Simulink simulation in normal and accelerated mode is supported.
- **ASM Traffic demo support for AURELION**: The Model and Sensor Interface Blockset Animation Interface is included in the ASM Traffic demo for Simulink, VEOS, and SCALEXIO that can be used in ModelDesk projects to support animation and sensor simulation with AURELION.

  For more information on the new features of the ASM Traffic Demo Model including the requirements for a SCALEXIO hardware-specific configuration, refer to Changes in the ASM Traffic Demo Model on page 41.

Related topics

Basics

Adapting Simulink Models (Model and Sensor Interface Blockset Manual 🕮)
Features of the Model and Sensor Interface Blockset (Model and Sensor Interface Blockset Manual 🕮)
New Features of MotionDesk 2022-B 119

Migrating to Model and Sensor Interface Blockset 2022-B

MicroAutobox III

The MicroAutobox III platform is not supported by the Model and Sensor Interface Blockset.

For more information on using MicroAutobox III with the blockset, contact dSPACE Support.

Migrating from previous Releases

Migration from previous Releases as of dSPACE Release 2020-B Models created with previous Releases of the Model and Sensor Interface Blockset as of version 1.0 in dSPACE Release 2020-B are migrated automatically when you

open the model. Messages are displayed in the MATLAB Command Window and in the migration log to confirm successful migration or notify you of failures.

Migration from the Model and Sensor Interface Blockset Solution Models created with the Model and Sensor Interface Blockset Solution that was available prior to dSPACE Release 2020-B must be manually migrated. When you open a model that uses blocks from the blockset solution, a message is displayed with instructions on how to migrate the model to use the blocks of the latest Release of the Model and Sensor Interface Blockset.

ModelDesk

Where to go from here

Information in this section

New Features of ModelDesk 2022-B

Road Generator

Traffic objects Traffic objects can be placed on road elements or junctions using absolute x, y, and z coordinates.

Migration to ModelDesk 2022-B

Project migration

As of ModelDesk 2022-B, you can migrate only projects created with ModelDesk version 5.0 (dSPACE Release 2018-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 in the layouts used for plotting are still connected. If you replace the migrated ASM model with an ASM model created with Release 2020-B, you must connect the signals again. ASM models created with Release 2020-B use `ASMSignalInterface` blocks that can be used in different parts of the model.

ASM project

Creating new ModelDesk projects on the basis of ASM projects is no longer supported for DS1006 target platforms. However, DS1006 platforms are still

supported by ModelDesk. You can still parameterize real-time applications running on DS1006 platforms.

Model Interface Package for Simulink

Where to go from here

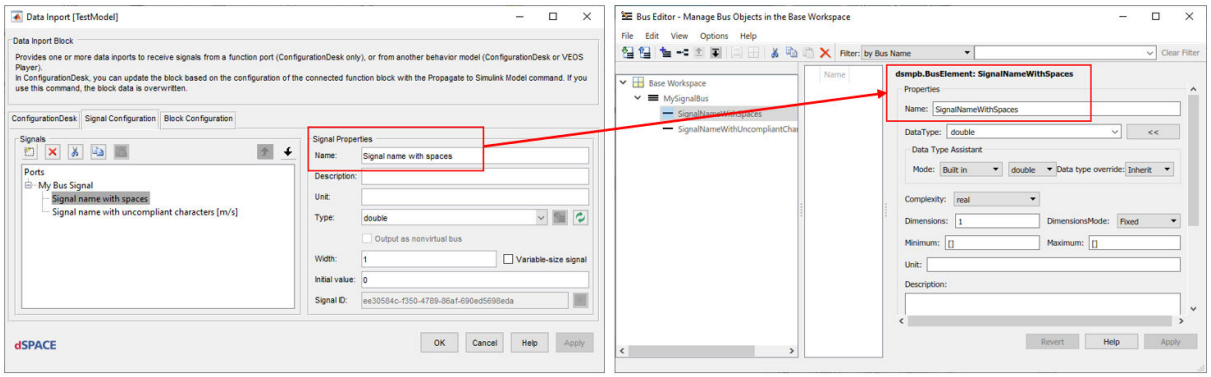
Information in this section

New Features of the Model Interface Package for Simulink 2022-B

Support of signal names with incompliant characters for Simulink.Bus objects

When creating Simulink.Bus objects, the Model Interface Package for Simulink supports signal names that are not valid C identifiers or MATLAB identifiers, such as signal names with spaces or brackets, or signal names with more than 63 characters. In the model port block dialogs and in model port blocks in ConfigurationDesk, the signal names remain unchanged. This allows you to perform propagate operations and automatic model port mapping for bus configurations with structured signals whose names contain incompliant characters.

If you specify such signal names in the model port block dialog, the Model Interface Package for Simulink provides dsmpb.BusElement objects instead of Simulink.BusElement objects with names that match the naming rules for MATLAB variables. Refer to the following illustration:



Note

Signal names that have more than 63 characters are truncated after the 63rd character.

MATLAB compatibility

The Model Interface Package for Simulink 2022-B is compatible with the following MATLAB® Releases:

- MATLAB R2022b
- MATLAB R2022a
- MATLAB R2021b
- MATLAB R2021a

Migrating to the Model Interface Package for Simulink 2022-B

Migration

No migration steps are necessary for the Model Interface Package for Simulink 2022-B.

MotionDesk

Where to go from here

Information in this section

New Features of MotionDesk 2022-B

MotionDesk

There are no new features in MotionDesk 2022-B.

Migrating to MotionDesk 2022-B

Discontinuation of sensor simulation with radar, lidar, and laser sensors

As of MotionDesk Release 2022-B, sensor simulation based on MotionDesk with radar, lidar, and laser sensors is no longer supported.

AURELION dSPACE AURELION is a new product and the successor of MotionDesk and Sensor Simulation for ADAS/AS simulations with camera, radar, and lidar sensors to validate driving functions and sensor output.

For more information and to prepare for your migration to AURELION, refer to https://www.dspace.com/en/pub/home/support/kb/faqs/faq433.cfm or contact dSPACE Support.

Planned discontinuations

MotionDesk and Sensor Simulation

- Sensor Simulation based on MotionDesk

  Sensor Simulation based on MotionDesk for all sensor types will be discontinued after Release 2022-B. For more information, refer to

https://www.dspace.com/en/pub/home/support/pli/elas/elassw/elasensim.cfm.

- MotionDesk

  MotionDesk will be discontinued after Release 2023-B. For more information, refer to https://www.dspace.com/go/elamd.

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.

In advanced lighting mode, the static objects used for domes are not suitable for building the virtual world. You have to use the endless sky.

Migration of custom object VRML files

If you want to import 3-D custom objects in VRML2 format to MotionDesk, you must first convert the VRML2 files to COLLADA (*.dae) format using MotionDesk up-to Release 22-A. You can then import and use the COLLADA files in Release 22-B and later.

You convert the files in the **Import Objects Dialog** of the **Library Manager Tool**.

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 2022-B

Datastreaming	Only for SCALEXIO Processing Unit, DS6001 Processor Board, and MicroAutoBox III: You can specify the size of the buffer that is used for datastreaming between the host PC and the real-time system. This allows to stream more data if the default buffer size leads to an out-of-memory error.
Improved compatibility of MicroAutoBox III	For the MicroAutoBox III platform, the compatibility between host PC and real-time platform is extended. You can manage the MicroAutoBox III platform with firmware version as of 6.0 (Real-Time Testing version 5.2 included, released with Release 2022-A) from a host PC on which the current Real-Time Testing version is installed.
Documentation	The new Real-Time Testing Tutorial guides you through your first steps with Real-Time Testing. Refer to Real-Time Testing Tutorial 🕮.

Migrating to Real-Time Testing 2022-B

Incompatible BCG files

BCG files generated on a Windows PC cannot be used on a Linux PC and vice versa. Generate the BCG files under the operating system that you want to use to manage real-time tests.

The BCG files generated with Real-Time Testing 4.0 or earlier cannot be used for Real-Time Testing 2022-B. 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.

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 has the following new feature:

- Support of MATLAB® R2022b.

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

Simulink.ConfigSet objects that were saved in MAT files using dSPACE Release 2013-B or older cannot be migrated to Release 2022-B.

RTI Bypass Blockset

Migrating to RTI Bypass Blockset 2022-B

Discontinued support of on-target bypassing on virtual ECUs

As of Version 2022-B, the RTI Bypass Blockset no longer supports on-target bypassing on virtual ECUs (V-ECUs) on the following dSPACE hardware:

- SCALEXIO systems
- MicroAutoBox III

Additional migration steps in some cases

To migrate to the RTI Bypass Blockset 2022-B from versions earlier than the RTI Bypass Blockset 3.18 and reuse existing models, you might have to carry out additional migration steps. For more migration information, refer to History of Migration Steps (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 2022-B, 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 that was built with RTI Bypass Blockset 2.5 or earlier and you open it with RTI Bypass Blockset 2022-B, 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 2022-B 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 2022-B (.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 3.18*

  If a Simulink model was built with RTI Bypass Blockset 2.6 up to RTI Bypass Blockset 3.18 and you open it with RTI Bypass Blockset 2022-B, 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 3.18, 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

Migrating to RTI CAN MultiMessage Blockset 2022-B

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 (RTI CAN MultiMessage Blockset Reference 🕮) 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 2022-B from versions earlier than the RTI CAN MultiMessage Blockset 5.8 and reuse existing models, you might have to carry out additional migration steps. For more migration information, refer to History of Migration Steps (RTI CAN MultiMessage Blockset Reference 🕮).

RTI LIN MultiMessage Blockset

Migrating to RTI LIN MultiMessage Blockset 2022-B

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 (RTI LIN MultiMessage Blockset Reference 🕮) block.

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

RTI Synchronized Time Base Manager Blockset

New Features of the RTI Synchronized Time Base Manager Blockset 2022-B

STBM_SYNCHRONOUS_TRIGGER block

The new STBM_SYNCHRONOUS_TRIGGER block lets you generate triggers based on the time of the dSPACE ECU time base manager, which can be used to configure events. The block also provides GTS synchronization status information of the underlying schedule table.

Refer to STBM_SYNCHRONOUS_TRIGGER (RTI Synchronized Time Base Manager Blockset Reference 🕮).

SCALEXIO Firmware

Where to go from here

Information in this section

New Features of the SCALEXIO Firmware 22.2

New supported hardware

The SCALEXIO firmware supports the following new hardware:

- DS6330M4 Automotive Ethernet Module

  The DS6330M4 module provides support for automotive Ethernet with a transfer rate of 100 and 1000 Mbit/s and integrated MACsec (media access control security) functionality. Although the hardware supports MACsec, a software patch must be installed to use this feature.

  The DS6330M4 module has two ports for connecting an Ethernet network with the automotive Ethernet standard. You can use up to two modules on the DS6333-CS, DS6333-PE, and DS6335-CS Ethernet board.

RS485 full duplex

The UARTs of the DS2671 Bus Board and DS6321 UART Board support RS485 full duplex.

Data logging

The DS6001 Processor Board now supports the use of a real-time-application with an activated data logging configuration. It is no longer necessary to use ControlDesk to load the data logging configuration and the related real-time application (RTA) file to the flash memory of the DS6001. As a consequence, the host PC no longer needs to be connected to the real-time hardware to load a data logger configuration to it.

For more information, refer to Data Logging (ControlDesk Measurement and Recording 🕮).

Related topics

Basics

Ethernet Board Configuration Page (SCALEXIO Hardware Installation and Configuration 🕮)
Ethernet Board Configuration Page (SCALEXIO Hardware Installation and Configuration 🕮)

Migrating to SCALEXIO Firmware 22.2

Migrating to the 64-bit Linux-based operating system

With dSPACE Release 2022-A, the default SCALEXIO firmware is a 64-bit Linux-based distribution. The following items built for dSPACE Release 2021-B and earlier are no longer compatible with the SCALEXIO system and must be (re-)built from source code based on dSPACE Release 2022-A or later:

- Real-time applications
- Binary libraries contained in model containers (i.e., SIC, BSC, FMU, and CTLGZ files)
- SIC and BSC 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 and adapt Linux real-time and non-real-time applications to the new hypervisor, especially regarding interrupts, shared memory, and the assignment of hardware resources.

Discontinuations in SCALEXIO Firmware

Processing Unit

The SCALEXIO Processing Unit with the Intel® Core™ i7-860 mainboard (Real-Time PC Version 1.0) is not supported by the SCALEXIO firmware.

Sensor Simulation

Where to go from here

Information in this section

New Features of Sensor Simulation 2022-B

SensorSim application

There are no new features in the SensorSim application for Sensor Simulation 2022-B.

Related topics

Basics

MotionDesk Sensor Simulation Control
Sensor Simulation Manual
Sensor Simulation Overview

Migrating to Sensor Simulation 2022-B

Discontinuation of sensor simulation with radar, lidar, and laser sensors

As of Sensor Simulation Release 2022-B, sensor simulation based on MotionDesk with radar, lidar, and laser sensors is no longer supported.

AURELION dSPACE AURELION is a new product and the successor of MotionDesk and Sensor Simulation for ADAS/AS simulations with camera, radar, and lidar sensors to validate driving functions and sensor output.

For more information and to prepare for your migration to AURELION, refer to https://www.dspace.com/en/pub/home/support/kb/faqs/faq433.cfm or contact dSPACE Support.

Planned discontinuations

MotionDesk and Sensor Simulation

- Sensor Simulation based on MotionDesk

  Sensor Simulation based on MotionDesk for all sensor types will be discontinued after Release 2022-B. For more information, refer to https://www.dspace.com/en/pub/home/support/pli/elas/elassw/elasensim.cfm.

- MotionDesk

  MotionDesk will be discontinued after Release 2023-B. For more information, refer to https://www.dspace.com/go/elamd.

SYNECT

Where to go from here

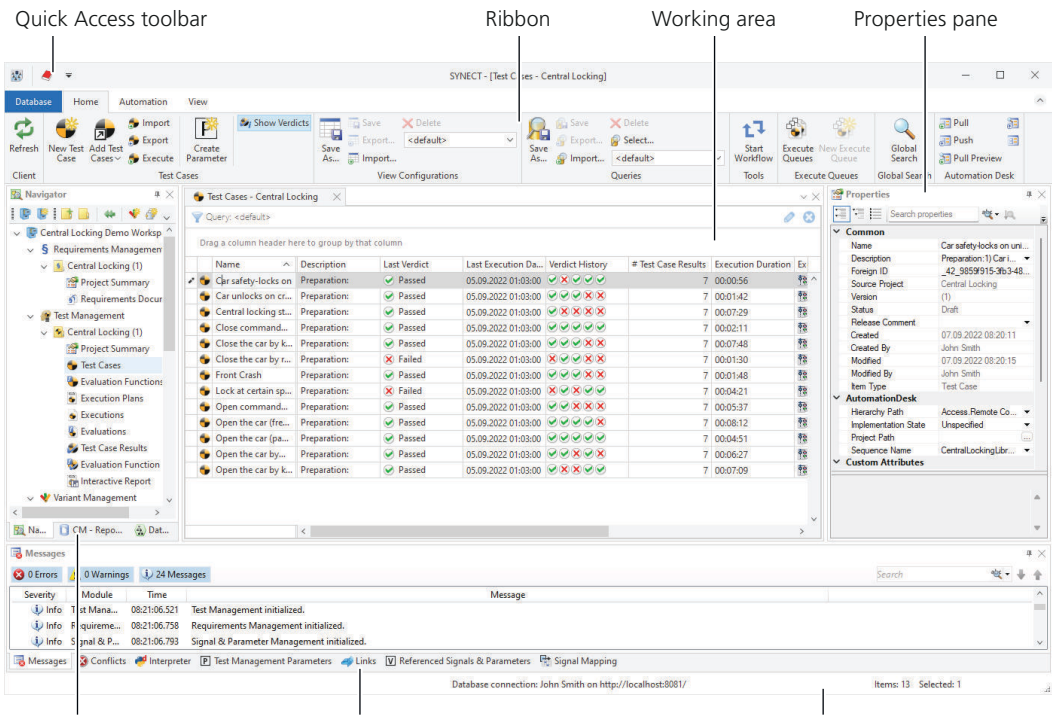
Information in this section

New Features of SYNECT 2022-B

New Features

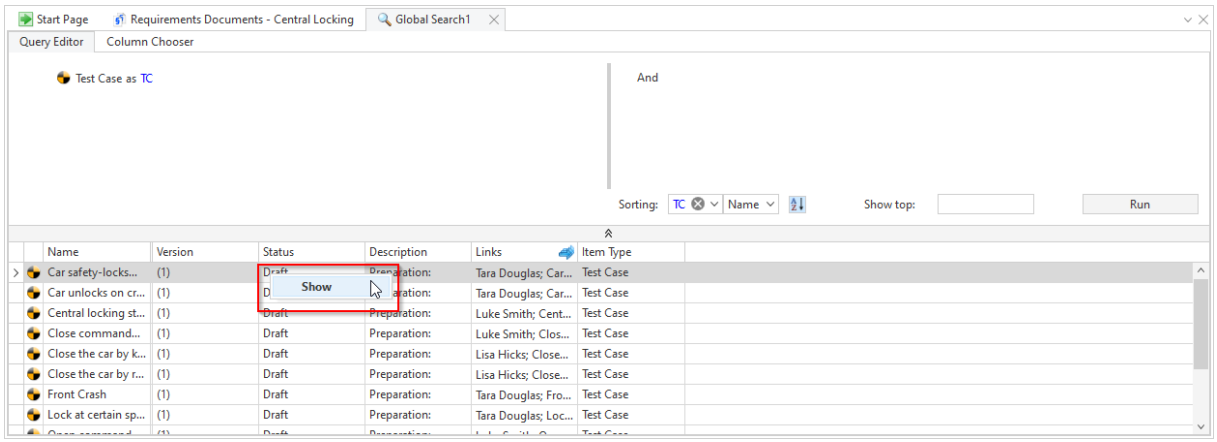
Modernized user interface framework

SYNECT now provides a modernized user interface framework with an improved handling of pages and panes. Refer to the followig illustration.



Improved global search

You can now navigate to items found by search queries using the context menu of result items. Refer to the following illustration:



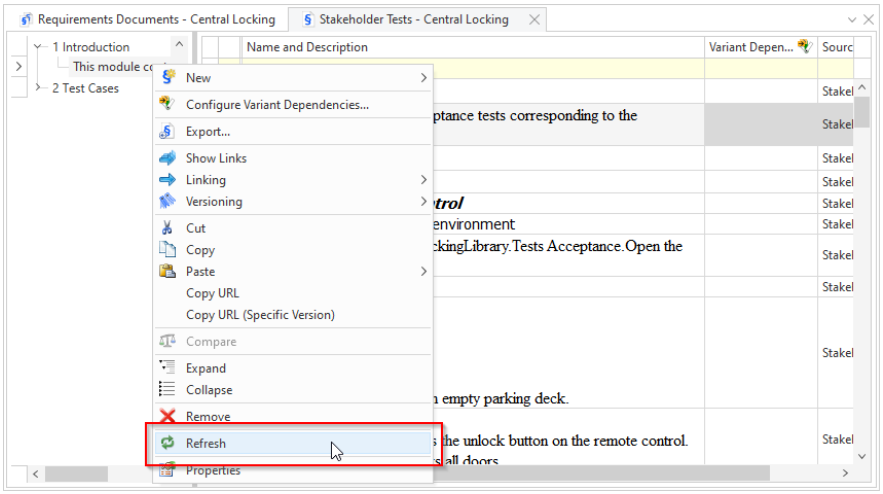
SYNECT opens the item in a data grid, such as the Test Cases data grid.

Improved refreshing of items

You can now refresh specific items. This improves the performance of data grids.

The context menu of items in specific data grids, such as the Requirements Management Documents data grid provide the Refresh command.

Refer to the following illustration that shows refreshing the contents of a requirement.



SYNECT client API The `Application` interface of the SYNECT client API provides the `RefreshItems` method that lets you refresh a list of items specified by their ID.

Refer to Application / ISnApplication <<Interface>> (SYNECT Client API Reference 🕮).

SYNECT server API support for item URLs

You can now generate URLs for SYNECT items of specific item types using the server API. The URLs let you navigate to the respective items in the SYNECT client. You can use this to implement extensions for reports and custom add-ons that link SYNECT items.

The `ItemBase` class implements the `FindSynectUrls` method. The method is available for each database item that is inherited from the **ItemBase** class, such as projects, test cases, executions, models, etc.. The method returns a list of URLs. Refer to ItemBase (SYNECT Server API Reference 🕮).

Extending context menus

SYNECT provides client events for extending the context menu of items, which are available from panes in the SYNECT user interface.

You can now extend the context menu of items on the **Links** pane, which displays the outgoing and incoming links of a selected item. You can add custom commands to the context menu of the displayed link items.

Refer to Extending Context Menus of the SYNECT Client (SYNECT Guide 🕮).

Improved SYNECT client API

The SYNECT client API lets you automate client operations, such as opening user interface panes, selecting items, and importing or exporting files.

You can now open a compare view that displays the differences of two items using the SYNECT client API.

Refer to CompareView / ISnCompareView <<Interface>> (SYNECT Client API Reference 🕮).

Improved import of requirements management documents

You can now import parts of requirements management documents.
You can control the import behavior of plug-ins with the new `DeleteUnspecifiedItems` ECXML property.

If enabled, the plug-ins import complete requirements documents. Requirements contents that are not available in imported documents are removed. This is the well-known behavior of previous SYNECT versions, e.g., useful for importing DOORS documents.

If disabled, the import plug-ins do not change the requirements document structure. Available requirements contents are updated, new requriements contents are added. The new behavior lets you import lists of requirements contents, e.g., useful for importing items from Azure DevOps or issues from Atlassian Jira

Migrating to SYNECT 2022-B

Where to go from here

Information in this section

Migrating from SYNECT 2.13

Migrating SYNECT client automation scripts

The SYNECT client API of the Internal Python interpreter was changed with the modernized user interface framework. You have to adapt automation scripts to the new API. For reference information on the Internal Python interpreter API, refer to Python (SYNECT Client API Reference 🕮).

Removed automation interfaces and enumerations The following automation interfaces and enumerations were removed:

- `FoldingType` enumeration
- `IPiBookmarks`
- `IPiFolding`
- `IPiFoldmarks`
- `IPiLineWrapping`
- `IPiRightEdge`
- `IPiTokenStyle`
- `IPiTokenStyles` collection
- `IPiZoom`
- `LineWrappingMode` enumeration
- `RightEdgeMode` enumeration
- `VisualWrappingCue` enumeration

Removed properties and methods Properties and methods of the following interfaces were removed:

Interface	Removed Property/Method
`PythonThreads / IPiPythonThreads <<Interface>>`	Properties: ▪ `IDs` ▪ `Count` ▪ `MainThreadID` Methods: ▪ `Stop`
`InterpreterProperties / IPiInterpreterProperties <<Interface>>`	Properties: ▪ `BackgroundColor` ▪ `ErrorTextColor` ▪ `Font` ▪ `HighlightBraceMatching` ▪ `Keywords` ▪ `OutputTextColor` ▪ `TextColor` ▪ `TokenStyles`
`EditorProperties / IPiEditorProperties <<Interface>>`	Properties: ▪ `BackgroundColor` ▪ `BookmarksColor` ▪ `Folding` ▪ `FoldingMargin` ▪ `Font` ▪ `HighlightBraceMatching` ▪ `HScrollVisible` ▪ `Keywords` ▪ `LineNumberMargin` ▪ `LineWrapping` ▪ `MarkerMargin` ▪ `RightEdge` ▪ `ShowIndentationGuides` ▪ `TextColor` ▪ `TokenStyles` ▪ `VScrollVisible`
`EditorControl / IPiEditorControl <<Interface>>`	Properties: ▪ `Bookmarks` ▪ `CodePage` ▪ `Foldmarks` ▪ `Zoom`

Migrating Databases

Introduction

To use the data from previous SYNECT versions with SYNECT 2022-B, you have to migrate the SYNECT database.

To migrate databases from SYNECT Versions 2.0 - 2.13 to SYNECT 2022-B, use the **Database Migrator** of SYNECT 2022-B.

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 🕮).

Data Model Changes from SYNECT 2.13 to SYNECT 2022-B

Introduction

The data model did not change from SYNECT 2.13 to SYNECT 2022-B.

SystemDesk

Where to go from here

Information in this section

New Features of SystemDesk 2022-B

Where to go from here

Information in this section

New General Features

Classic Platform support by SystemDesk 2022-B

AUTOSAR release for modeling SystemDesk lets you model Classic Platform software and system architectures with a data model according to the AUTOSAR R21-11 Release. However, SystemDesk also lets you exchange data of other AUTOSAR releases.

Data exchange support SystemDesk supports AUTOSAR R21-11, R20-11, R19-11, 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

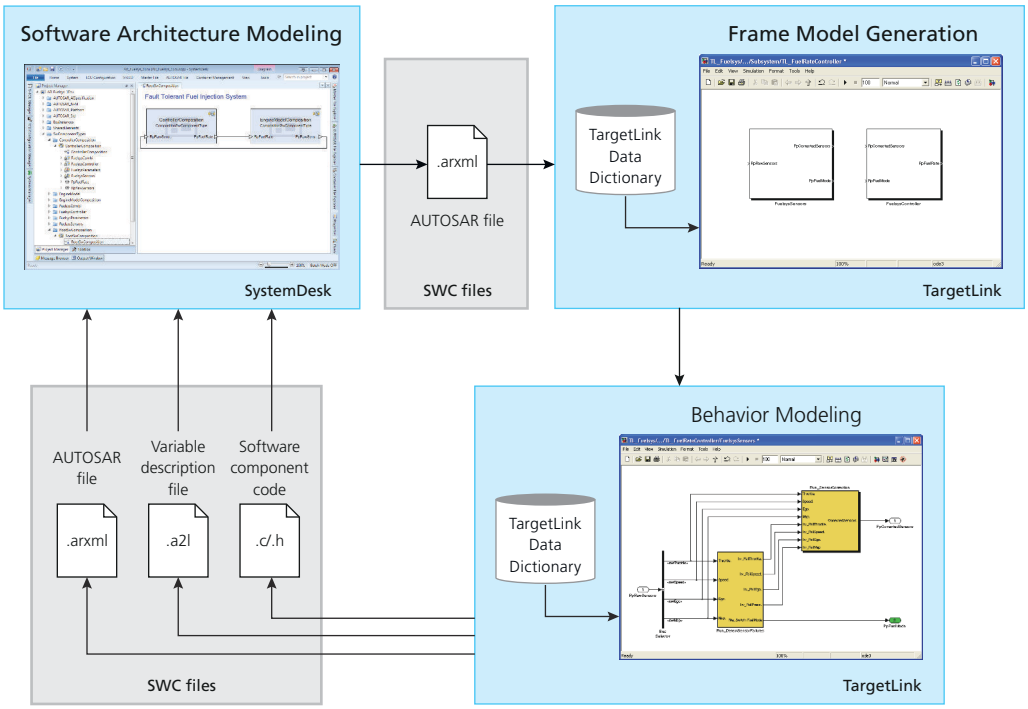
SystemDesk now supports AUTOSAR R21-11 for developing Adaptive Platform software. For exchanging data, AUTOSAR R20-11, R19-11, and R19-03 are supported.

Best practices for exchanging software components

For ECU software development according to AUTOSAR, SystemDesk acts as a *system-level design tool*. For *behavior modeling*, dSPACE provides the TargetLink AUTOSAR Module. If you follow this use case, you have to transfer relevant parts of the AUTOSAR model from SystemDesk to TargetLink.

dSPACE supports SIL testing of software components in the context of closed-loop simulations of V-ECUs and environment models. If you follow this use case, you additionally have to reference code files in the AUTOSAR model and specify implementation-specific aspects herein. This enables you to generate model-based V-ECUs with SystemDesk.

The illustration below shows the interoperability of SystemDesk and TargetLink:



Since the discontinuation of the *Container Manager*, a dSPACE tool for exchanging software components between TargetLink and SystemDesk, we recommend that you follow best practices, which include the assignment of AUTOSAR elements to packages and AUTOSAR files, specific import/export-related settings in SystemDesk and the TargetLink Data Dictionary, and putting exchanged AUTOSAR files under version control. We also recommend the use of a compare and merge tool for AUTOSAR files.

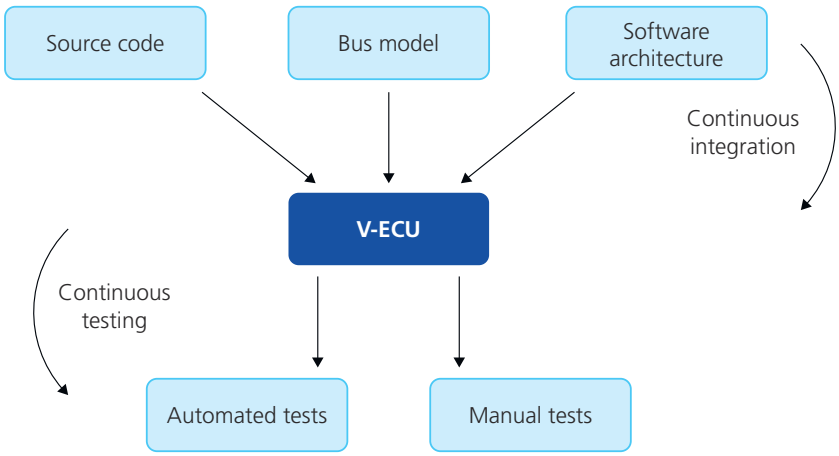
For details, refer to the best practices in the user documentation of your dSPACE tool:

dSPACE Tool	Documentation
SystemDesk	Best Practices for Exchanging Software Components (SystemDesk Manual 🕮)
TargetLink	Best Practices for Exchanging Software Components (TargetLink Interoperation and Exchange Guide 🕮)

Using SystemDesk in build pipelines

You can generate the implementation of V-ECUs with SystemDesk in build pipelines for continuous integration and continuous testing.

Refer to the following illustration that shows the characteristics of the concept:



Refer to Using SystemDesk in Build Pipelines (SystemDesk Manual 🕮).

New V-ECU Generation Features

Compatibility of V-ECUs generated with SystemDesk 2022-B

SystemDesk lets you export V-ECUs to be built and simulated in the following formats:

- *V-ECU implementation containers*: Version 3.2

  V-ECU implementation containers are a dSPACE-specific format that supports all features of the dSPACE SIL solution. You can build and simulate version 3.2 V-ECUs with dSPACE VEOS 2022-B.

  For detailed compatibility information, refer to Compatibility of VEOS 2022-B on page 229.
- *V-ECU FMUs*: FMI 2.0/3.0

  You can export V-ECU FMUs with SystemDesk to support OEM-supplier scenarios where not all partners use the dSPACE SIL solution. You can build and simulate V-ECU FMUs with FMI-compliant simulators.

  For detailed and up-to-date compatibility information on dSPACE FMI support, refer to http://www.dspace.com/go/FMI-Compatibility.

Improved support for V-ECU FMUs

With dSPACE Release 2022-B the following features have been added for V-ECU FMUs:

- Support of the FMI 3.0 standard.
- Support of XCP access to V-ECU FMUs.

  You can now integrate an XCP service in V-ECU FMUs that you can access from meassurement and calibration tools, such as dSPACE ControlDesk. This lets you access V-ECU variables.

- Support for execution on Linux.

  You can now build V-ECU FMUs for Linux systems, 32-bit and 64-bit. This lets you execute V-ECU FMUs on FMI-compliant simulators using a Linux execution environment.
- Support for restarting V-ECU FMUs.

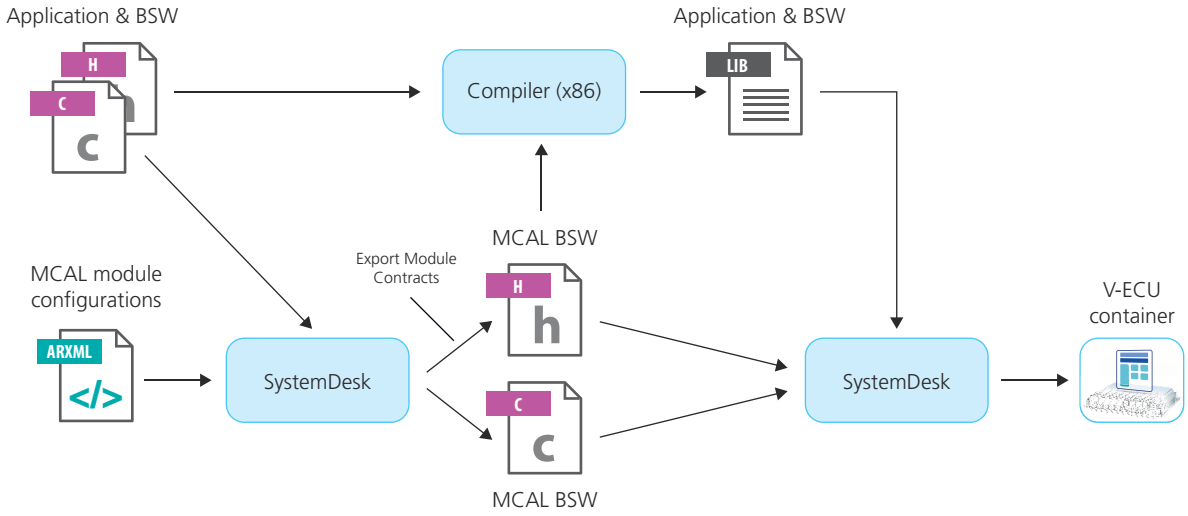
Refer to Configuring ECUs for FMU Export (SystemDesk Manual 🕮).

Exporting contract header files

Header files of dSPACE MCAL modules are required, when you develop or build basic software for V-ECUs. If you want to include basic software as binaries, the header files are required during the build process.

This applies to scenarios where referencing all the souce code files in SystemDesk is not an option, e.g., due to large file numbers.

Refer to the following illustration.



You can now export header files of generated basic software in the SystemDesk V-ECU Manager. This lets you easily include header files in your development IDE.

You can proceed as follows to create binary files for V-ECUs:

1. Prepare a V-ECU with the required dSPACE basic software and the basic software you develop.
2. Export the dSPACE header files using SystemDesk.

   You can select the basic software modules for which to export header files. SystemDesk can analyze dependencies between basic software modules and lets you export additional required contract header files.
3. Include the header files in your development IDE.
4. Build the binary files that you want to share in an OEM supplier workflow.

Refer to Basics on Exporting Module Contracts (SystemDesk Manual 🕮).

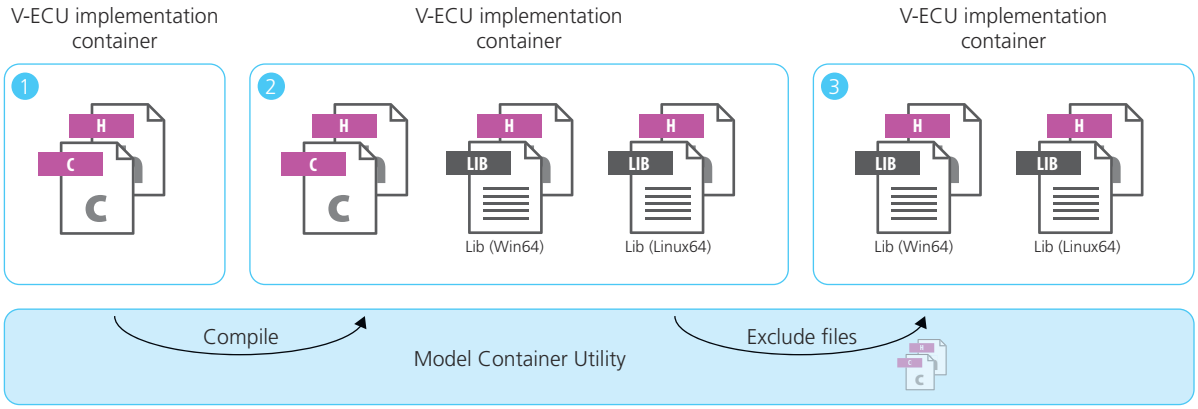
Using the Model Container Utility

VEOS provides the Model Container Utility to modify existing V-ECU implementation containers (VECU files) and package containers (PKC files) without having to generate them again.

The Model Container Utility supports use cases, such as the following:

- Compiling source code of container packages in advance for saving build times in multi-user or continuous integration scenarios.
- Excluding source code files of built container packages for ensuring IP protection.

Refer to the following illustration:



For details refer to Model Container Utility Manual 🕮.

Migrating to SystemDesk 2022-B

Where to go from here

Information in this section

Discontinuations as of dSPACE Release 2023-B

Discontinued partial AUTOSAR Import and Export commands

The following partial AUTOSAR Import and Export commands for software components will no longer be available in SystemDesk 2023-B:

- Import – AUTOSAR, Update Component Ports
- Import – AUTOSAR, Update Internal Behavior
- Export – AUTOSAR, Ports Only
- Export – AUTOSAR, Internal Behavior Only

SystemDesk provides support for splittables that you can use instead. You have to assign AUTOSAR elements to master files and split software components and internal behaviors to different files as required.

We recommend that you migrate your workflows already with SystemDesk 2022-B.

Discontinuation of views on ECU configurations

As of dSPACE Release 2023-B, the ECU Configuration Manager will no longer support different views.

You will no longer be able to switch to the AUTOSAR View or Dependency View. However, all SystemDesk features for configuring ECUs will still be available in the default Flat List View.

Migrating to SystemDesk 2022-B

Automatic migration of projects

SystemDesk 2022-B automatically migrates SystemDesk 5.5, and 5.6 SDP project files when opened.

Note

You are recommended to install the most recent patch for SystemDesk 5.5 or 5.6. Then save the SDP project files you want to migrate before opening them in SystemDesk 2022-B.

Migrating scripts for automating SystemDesk

The SystemDesk API was changed as of SystemDesk 2022-B. Some interfaces were added compared to SystemDesk 5.6 and certain interfaces were changed.

For more information, refer to *API Changes from SystemDesk 5.6 to SystemDesk 2022-B* in the *SystemDesk API Reference*, which is available only in dSPACE Help.

TargetLink

New versioning scheme

Note

The versioning used for dSPACE products until now is changed to a calender-based versioning scheme. Refer to *New versioning scheme* in General Enhancements and Changes on page 13.

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 2022-B

Where to go from here

Information in this section

Modeling in Simulink or Stateflow

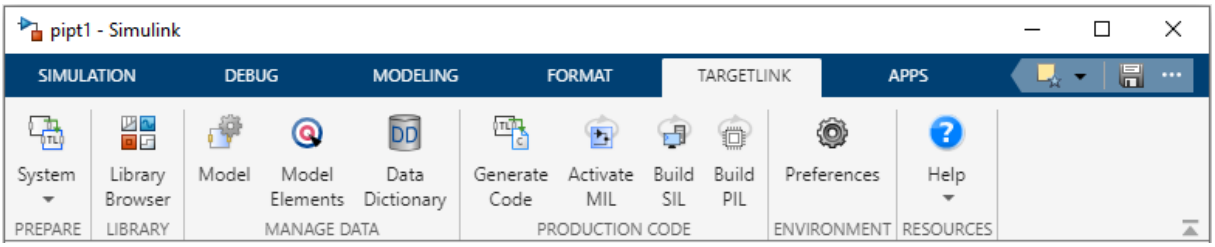
Where to go from here

Information in this section

TargetLink Tab on the Simulink Toolstrip

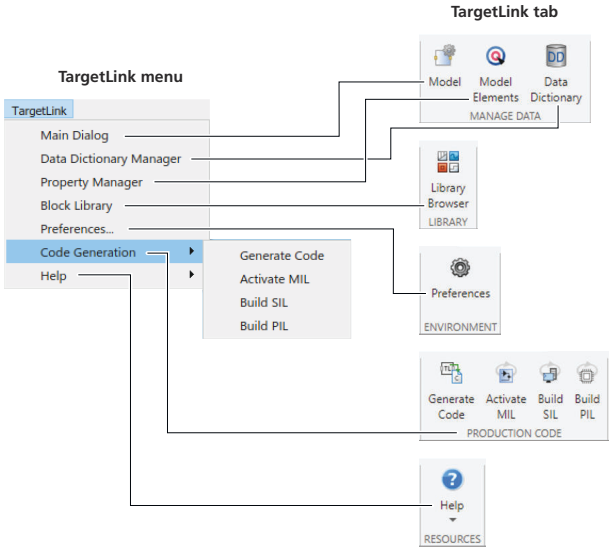
New TargetLink tab on the Simulink Toolstrip

TargetLink 2022-B introduces the new TargetLink tab on the Simulink® Toolstrip. The TargetLink tab replaces the TargetLink menu as of MATLAB® version R2022a.



Overview of available commands

Up to MATLAB R2021b, the following commands are available in the TargetLink menu. As of MATLAB R2022a, the commands are available in the new TargetLink tab on the Simulink Toolstrip.



TargetLink menu (Up to MATLAB R2021b)	TargetLink tab (As of MATLAB R2022a)
Block Library	LIBRARY - Library Browser
Code Generation - Activate MIL	PRODUCTION CODE - Activate MIL
Code Generation - Build PIL	PRODUCTION CODE - Build PIL
Code Generation - Build SIL	PRODUCTION CODE - Build SIL
Code Generation - Generate Code	PRODUCTION CODE - Generate Code
Data Dictionary Manager	MANAGE DATA - Data Dictionary
Help - About TargetLink	RESOURCES - Help - About TargetLink
Help - Demos	RESOURCES - Help - Demos
Help - dSPACE Help	RESOURCES - Help - Help
Help - New Features and Migration	RESOURCES - Help - New Features and Migration
Help - Useful links	RESOURCES - Help - Online
Help - Useful links - dSPACE Website	RESOURCES - Help - dSPACE Website
Help - Useful links - TargetLink Support Center	RESOURCES - Help - Visit the TargetLink Support Center
Help - Using dSPACE Help	RESOURCES - Help - Using dSPACE Help

TargetLink menu (Up to MATLAB R2021b)	TargetLink tab (As of MATLAB R2022a)
Main Dialog	MANAGE DATA - Model
Make System/Library TargetLink-Compliant	PREPARE - System - Make System TargetLink-Compliant
Preferences…	Environment - Preferences
Property Manager	MANAGE DATA - Model Elements
-	PREPARE - System - Prepare System
-	PREPARE - System - Synchronize Data Types

Improved Bus Handling

Mapping between Simulink.Bus objects and DD Typedef objects

With TargetLink 2022-B it is now possible to map bus objects and DD Typedef objects by the DD SimulinkBusObject property. The DD Typedef objects are then called mapped struct types that are relevant in the following contexts:

- During model preparation
- During frame model generation
- During code generation

If specified at interfaces, TargetLink can automatically use these mapped struct types down the signal line at bus-capable blocks whose Inherit properties checkbox is selected. This is especially useful for using mapped struct types at Bus Creator blocks. Refer to Improved Array-of-Struct Modeling on page 157.

Related documentation

- Set bus objects at interfaces (TargetLink Preparation and Simulation Guide 🕮)
- Mapping bus objects and DD Typedef objects (TargetLink Preparation and Simulation Guide 🕮)
- Basics on AUTOSAR Development Approaches and Frame Model Generation in TargetLink (TargetLink Classic AUTOSAR Modeling Guide 🕮)
- How To Create Mapped Struct Types for Bus Signals or Array-of-Bus Signals (TargetLink Preparation and Simulation Guide 🕮)
- TargetLink demo model BUS_CC

Related topics

Basics

Basics on AUTOSAR Development Approaches and Frame Model Generation in TargetLink (TargetLink Classic AUTOSAR Modeling Guide 🕮)
BUS_CC (TargetLink Demo Models 🕮)

Improved Array-of-Struct Modeling

Support of Bus Creator blocks to create bus-with-array-of-bus-as-payload signals

With TargetLink 2022-B, you can now use a Bus Creator block to connect a plain-array-of-bus or a bus-with-array-of-bus-as-payload signal and other signals to an bus-with-array-of-bus-as-payload signal.

Related documentation

- Basics on Array-of-Bus Signals and Array-of-Struct Variables (TargetLink Preparation and Simulation Guide 🕮)
- Overview of Modeling Patterns for Accessing Arrays of Buses (TargetLink Preparation and Simulation Guide 🕮)
- Example of Modeling the Dynamic Access to Bus-with-Array-of-Bus-as-Payload and Plain-Array-of-Bus Signals via Bus Port Blocks (TargetLink Preparation and Simulation Guide 🕮)

Using DD Typedef objects to implement array-of-struct variables

Beside referencing array-of-struct variables in TargetLink blocks directly, TargetLink 2022-B now supports the specification of array-of-struct variables by referencing an array-of-struct type using an DD Typedef object.

Related documentation

- How To Create Mapped Struct Types for Bus Signals or Array-of-Bus Signals (TargetLink Preparation and Simulation Guide 🕮)

Array-of-struct variables as reference parameters

With TargetLink 2022-B, it is possible to use array-of-struct variables as reference parameters in functions.

Related documentation

- Basics on Function Interfaces (TargetLink Customization and Optimization Guide 🕮)

Array-of-struct variables at the interface of incremental code generation units

TargetLink 2022-B supports array-of-struct variables at the interface of incremental code generation units.

Further new features regarding array-of-struct variables or array-of-bus signals

- Improved Array-of-Struct Support (Classic AUTOSAR) on page 163.

Improved Support for Temporal Logic in Stateflow

Support for temporalCount operator

TargetLink now supports the temporal logic operator `temporalCount`. However, the following keywords are not supported:

- `sec`
- `msec`
- `usec`

Related documentation

- Stateflow (Limitations) (TargetLink Limitation Reference 🕮)

Interpolation Using Prelookup block

Support of table data with up to five dimensions

With TargetLink 2022-B, the Interpolation Using Prelookup block supports table data with up to five dimensions.

Related documentation

- Supported Dimensionalities for Look-Up Tables (TargetLink Preparation and Simulation Guide 🕮)
- How to Specify The Values of Look-Up Tables (TargetLink Preparation and Simulation Guide 🕮)
- Example of Modeling With Prelookup and Interpolation Using Prelookup Blocks for a 3-D Table (TargetLink Preparation and Simulation Guide 🕮)

Related topics

References

Interpolation Using Prelookup Block (TargetLink Model Element Reference 🕮)

MATLAB Code

Enhanced MATLAB Code Functions and Operations

Supported MATLAB code functions and operations

TargetLink 2022-B added support for the following MATLAB® code functions and operations:

- `bitxor`
- `norm`

Related documentation

- Supported MATLAB Code Function Statements and Function Operators (TargetLink Code Generation Reference for MATLAB® Code in Simulink® Models 🕮)

Adaptive AUTOSAR

Where to go from here

Information in this section

Support of INOUT Operation Arguments in Adaptive AUTOSAR

Support of ARGINOUT operation arguments

Beside operation arguments of the ARGIN and ARGINOUT kind, TargetLink 2022-B now supports operation arguments of the ARGINOUT kind in methods.

Related documentation

- How to Model a Method Call (TargetLink Adaptive AUTOSAR Modeling Guide 🕮)
- How to Model a Method Implementation (TargetLink Adaptive AUTOSAR Modeling Guide 🕮)
- How to Create Communication Subjects for Adaptive AUTOSAR (TargetLink Adaptive AUTOSAR Modeling Guide 🕮)

Changes Concerning Adaptive AUTOSAR Data Dictionary Import and Export

Support of DD ArTrigger objects in import and export

TargetLink 2022-B supports DD ArTrigger objects in the DD AUTOSAR import and export.

Related documentation

- How to Create Communication Subjects for Adaptive AUTOSAR (TargetLink Adaptive AUTOSAR Modeling Guide 🕮)

Importing and Exporting AdminData of InternalBehavior

TargetLink 2022-B provides the new DD AdminDataOfInternalBehavior object as a data container to store AdminData of InternalBehavior objects, which have no representation in the Data Dictionary, during ARXML import. The stored AdminData can be exported again.

Array-of-Struct Support (Adaptive AUTOSAR)

Array-of-struct support in Adaptive AUTOSAR

As of TargetLink 2022-B, the use of array-of-structs with the following Adaptive AUTOSAR communication subjects is supported:

- Fields
- Events
- Methods

Adaptive AUTOSAR Data Types

Native support of ara::core::Array and std::array data types

TargetLink 2022-B supports code generation for `ara::core::Array` and `std::array` data types. These data types can be used in the same way as Classic AUTOSAR array data types.

Prior to TargetLink 2022-B, `ara::core::Array` data types were only used in the ARA adapter code.

Related documentation

- Introduction to Data Types in Adaptive AUTOSAR (TargetLink Adaptive AUTOSAR Modeling Guide 🕮)

Support for C++ namespaces

TargetLink 2022-B supports code generation for data types that are defined in C++ namespaces.

Prior to TargetLink 2022-B, C++ namespaces were only used in the ARA adapter code.

Related documentation

- Introduction to Data Types in Adaptive AUTOSAR (TargetLink Adaptive AUTOSAR Modeling Guide 🕮)

Classic AUTOSAR

Where to go from here

Information in this section

Supported Classic AUTOSAR Releases

Supported Classic AUTOSAR Releases

The following Classic AUTOSAR Releases are supported:

Classic AUTOSAR Release	Revision
R21-11	21-11[1)]
R20-11	20-11
R19-11	19-11
4.4	4.4.0
4.3	4.3.1 4.3.0
4.2	4.2.2 4.2.1
4.1	4.1.3 4.1.2 4.1.1
4.0	4.0.3 4.0.2

1) New in TargetLink 2022-B

Support of INOUT Operation Arguments in Classic AUTOSAR

Support of ARGINOUT operation arguments

Beside operation arguments of the ARGIN and ARGOUT kind, TargetLink 2022-B now supports operation arguments of the ARGINOUT kind for the communication Kind `Operation`.

Related documentation

- Basics on Client-Server Communication (TargetLink Classic AUTOSAR Modeling Guide 🕮)
- Basics on Combining a Synchronous Operation Call with a Server Operation's Implementation (TargetLink Classic AUTOSAR Modeling Guide 🕮)
- How to Create Communication Subjects for Classic AUTOSAR Communication (TargetLink Classic AUTOSAR Modeling Guide 🕮)

Changes Concerning Classic AUTOSAR Data Dictionary Import and Export

Importing and Exporting AdminData of InternalBehavior

TargetLink 2022-B provides the new DD AdminDataOfInternalBehavior object as a data container to store AdminData of InternalBehavior objects, which have no representation in the Data Dictionary, during ARXML import. The stored AdminData can be exported again.

Changes according to AUTOSAR Version R21-11

Avoiding SWC name conflicts during code generation

To avoid name conflicts, which occur using SWCs with identical names on one ECU, TargetLink 2022-B gives you the possibility to set the DD SymbolicName property. If you set this property, TargetLink uses the property value as name for the SWC in the generated code instead of the DD SoftwareComponent object name.

Furthermore TargetLink offers you the new name macro `$(SwComponentSymbolicName)`. If the AUTOSAR version is R21-11 or highter, it is replaced by the value of the DD SymbolicName property during code generation. If this property is not set or the AUTOSAR version is R20-11 or lower, the name macro is replaced by the name of the DD SoftwareComponent object.

Related documentation

- How to Create Software Components (TargetLink Classic AUTOSAR Modeling Guide 🕮)
- Details on the Expansion of Name Macros (TargetLink Customization and Optimization Guide 🕮)

Providing custom-defined header files for user-specified data types

According to AUTOSAR, with TargetLink 2022-B and AUTOSAR version R21-11 it is possible to provide the definitions of user-specified data types in a separate header file. This is specified by the DD **TypeEmitter** property of a DD **Typedef** object.

Related documentation

- How to Provide Custom-Defined Header Files for User-Specified Classic AUTOSAR Data Types (TargetLink Classic AUTOSAR Modeling Guide 🕮)

Improved Array-of-Struct Support (Classic AUTOSAR)

Generating and updating frame models

TargetLink 2022-B lets you generate and update frame models from Classic AUTOSAR data containing array-of-structs. TargetLink generates port or data store blocks. The following communication kinds are supported:

- InterRunnable
- Sender-Receiver
- NvData
- Operation

For faster modeling of complex array-of-bus signals, TargetLink can generate adapter blocks.

Related documentation

- Details on Generating and Updating Frame Models with Array-of-Bus Signals in Classic AUTOSAR (TargetLink Classic AUTOSAR Modeling Guide 🕮)

More communication kinds supported at port blocks

As of TargetLink 2022-B, you can use port-block-based modeling styles to generate RTE API function for Classic AUTOSAR data that contains array-of-structs. In addition to the **Operation** communication kind, which is supported since TargetLink 5.2, the following communication kinds are now supported:

- InterRunnable
- Sender-Receiver
- NvData

Related documentation

- Basics on Working with Arrays of Structs When Modeling Classic AUTOSAR in TargetLink (TargetLink Classic AUTOSAR Modeling Guide 🕮)

More communication kinds supported at data store blocks

As of TargetLink 2022-B, you can model the following via data store blocks in conjunction with Classic AUTOSAR data that contains array-of-structs:

- Explicit communication
- Implicit communication: added support for `Rte_IWrite` and `Rte_IrvIWrite`

Related documentation

- Basics on Working with Arrays of Structs When Modeling Classic AUTOSAR in TargetLink (TargetLink Classic AUTOSAR Modeling Guide 🕮)

Code Generation Core Functionality

Where to go from here

Information in this section

Improved Code Efficiency for Array-of-Struct

With TargetLink 2022-B, the efficiency of the code generated for array-of-struct variables was increased. Structs in general are now optimized in the same way as non-struct variables.

This pertains especially to structures that are generated from bus signals: here, struct assignments take precedence over other optimizations.

The precedence of struct assignments might result in code that is less efficient. TargetLink provides the following Code Generator options to mitigate for this:

- StructAssignmentDecompositionAuxVarMemoryUsageThreshold
- StructAssignmentDecompositionStatementMultiplicationThreshold

Complementary changes Together with the above improvements, the following optimizations were improved:

- Rescheduling of state update statements
- Scope reduction of structures

Migration issue Depending on your specific modeling situation and use cases, obtaining production code as in previous TargetLink versions might be possible by applying a temporary workaround. Contact dSPACE product support for TargetLink.

Related documentation

- AllowStructAssignments (TargetLink Model Element Reference 🕮)
- ExtendedLifeTimeOptimization (TargetLink Model Element Reference 🕮)
- StructAssignmentDecompositionAuxVarMemoryUsageThreshold (TargetLink Model Element Reference 🕮)
- StructAssignmentDecompositionStatementMultiplicationThreshold (TargetLink Model Element Reference 🕮)

Code Classification for Modules

Mark modules as simulation code

With TargetLink 2022-B, you can mark a DD Module object as simulation code via the CodeClassification property. If the property is set to `SimulationCode`, the resulting code file will be placed in the simulation code folder, e.g., `TLSim`.

Related documentation

- Description of ModuleInfo
  (Refer to TargetLink Data Dictionary Reference)

Model-in-the-Loop Simulation (MIL)

Improve Performance by Disabling the Overflow Detection

Disable the overflow detection to improve performance in MIL simulation mode

To improve the performance in MIL simulation mode, TargetLink 2022-B lets you now disable the overflow detection. This means that when starting the simulation, the quite time-consuming reading of the TargetLink block data only has to take place for TargetLink blocks with activated signal logging.

Especially if you want to log only a few TargetLink blocks in a large model, switching off the overflow detection is a good option to improve performance. You can deactivate the overflow detection for your model via the `tl_set(<model>, 'logopt.detectoverflows', 'off')` command.

Related documentation

- Basics on Overflow Detection (TargetLink Preparation and Simulation Guide 🕮)
- Various Migration Aspects on page 189

Target Simulation (PIL)

Changes in the Target Simulation Modules With 2022-B

New and discontinued compiler versions

The following table shows changes in the compiler versions that are supported by TargetLink 2022-B:

Microcontroller Family	Compiler	New	Discontinued
Freescale MPC5700VLE	GreenHill	2022	2021
Renesas RH850	GreenHill	2022	2021

For a complete list of evaluation boards and compilers that are 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

Usability Improvements

Data Dictionary and Property Manager

TargetLink 2022-B provides several usability improvements in the Data Dictionary Manager and Property Manager.

In the Property Manager, for example, the column selection using the Column Chooser is optimized.

Related documentation

- Show Column Chooser (Property View) (TargetLink Tool and Utility Reference 🕮)
- Column Chooser Dialog (Property View) (TargetLink Tool and Utility Reference 🕮)

Code Generator Options

New Code Generator Options

The following new Code Generator options are available with TargetLink 2022-B:

- ReduceLifetimeOfConstantVariables (TargetLink Model Element Reference 🕮)

  Lets you reduce struct variables that hold an initial value and are not modified to automatic storage duration.
- StructAssignmentDecompositionAuxVarMemoryUsageThreshold

  Lets you specify a value expressing the percentage of the original struct variable's memory usage that remains at most if the variable is removed and the necessary components are replaced by auxiliary variables.
- StructAssignmentDecompositionStatementMultiplicationThreshold

  Lets you define a number that indicates that in cases where less components of the struct are actually used, TargetLink generates component assignment statements instead of struct assignments.
- UseBusObjectMappingInsteadOfInheritedProperties (TargetLink Model Element Reference 🕮)

  Lets you use the mapped struct type instead of inheriting properties from predecessor blocks to create a variable.

Related documentation

- Improved Code Efficiency for Array-of-Struct on page 164
- StructAssignmentDecompositionAuxVarMemoryUsageThreshold (TargetLink Model Element Reference 🕮)
- StructAssignmentDecompositionStatementMultiplicationThreshold (TargetLink Model Element Reference 🕮)
- ReduceLifetimeOfConstantVariables (TargetLink Model Element Reference 🕮)
- UseBusObjectMappingInsteadOfInheritedProperties (TargetLink Model Element Reference 🕮)

Migration aspects of Code Generator options

For more information, refer to Migration Aspects Regarding Code Generator Options on page 180.

API Functions and Hook Scripts

Where to go from here

Information in this section

New API Functions

API Function	Purpose
`tlExportFixedPointLibrary`	Exports TargetLink fixed point library as package container or ZIP file to create V-ECU containers or FMUs with SystemDesk, for example.

Related topics

References

tlExportFixedPointLibrary (TargetLink API Reference 🕮)

New Hook Scripts

Hook Script	Description
tl_post_add_arrayofbusesadapterblocks_hook	Lets you enter custom commands that are executed after array of buses adapter blocks are added to the model. It is called for each **BusInport/BusOutport** or **DataStoreRead/DataStoreWrite** block that routes an array-of-bus signal. Refer to Details on Generating and Updating Frame Models with Array-of-Bus Signals in Classic AUTOSAR (TargetLink Classic AUTOSAR Modeling Guide 🕮).
tl_pre_add_arrayofbusesadapterblocks_hook	Lets you enter custom commands that are executed before array of buses adapter blocks are added to the model. It is called for each **BusInport/BusOutport** or **DataStoreRead/DataStoreWrite** block that routes an array-of-bus signal.

Hook Script	Description
	Refer to Details on Generating and Updating Frame Models with Array-of-Bus Signals in Classic AUTOSAR (TargetLink Classic AUTOSAR Modeling Guide 🕮).

Related topics

Basics

Details on Generating and Updating Frame Models with Array-of-Bus Signals in Classic AUTOSAR (TargetLink Classic AUTOSAR Modeling Guide 🕮)

References

tl_post_add_arrayofbusesadapterblocks_hook (TargetLink File Reference 🕮)
tl_pre_add_arrayofbusesadapterblocks_hook (TargetLink File Reference 🕮)

Other

TargetLink Demos and Examples

Modified demo models

Changed Demo Model	Description
AAR_COMMUNICATION	The `AAR_COMMUNICATION` demo model was adapted to show array implementation data types as described by Adaptive AUTOSAR.
BUS_CC	The `BUS_CC` demo model was adapted to show the benefit of using bus objects and mapped struct types.
BUS_STRUCT	The `BUS_STRUCT` demo model was adapted to show the following benefits: ▪ Using mapped struct types. ▪ Generating reference parameters from array-of-bus signals. ▪ Assembling array-of-bus signals via BusCreator blocks.
STRUCT_INTERFACES	The `STRUCT_INTERFACES` demo model was adapted to show the benefit of using pointer-to-struct-transport variables to propagate pointer arguments down the function-call hierarchy.
AUTOSAR_FUELSYS	The `AUTOSAR_FUELSYS` demo model was adapted to reflect the removal of container exchange.
AR_MEMORY_MAPPING	The `AR_MEMORY_MAPPING` demo model was adapted to reflect the removal of container exchange.

Removed demo models	The `AR_ARRAY_OF_STRUCT_DATA` demo model was removed from the product as you can now use array-of-bus signals to generate parameters of RTE API functions.

Migrating to TargetLink 2022-B and TargetLink Data Dictionary 2022-B

Where to go from here

Information in this section

General Migration Information

Where to go from here

Information in this section

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 2022-B 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 2022-B comprises the steps 4.0 to 4.1 to 4.2 to 4.3 to 4.4 to 5.0 to 5.1 to 5.2 to 2022-B.

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

If you open a DD file that was last saved with an earlier TargetLink version, it must be upgraded. The upgrade is executed automatically. The **Subsystems** area and the **<Application>** area are no longer compatible and thus removed. To change the upgrade behavior, refer to tl_pref (TargetLink API Reference 🕮).

If you save the upgraded DD file, the original DD file is automatically saved to a backup DD file in the same directory as the DD file before it is overwritten with the upgraded version. To disable the automatic backup, refer to **Data Dictionary Upgrade** in Topic Navigator (TargetLink Tool and Utility Reference 🕮).

To upgrade DD files with included DD files, refer to How to Upgrade and Save a Data Dictionary with Included DD Files on page 174.

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 TargetLink 2022-B:

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 2022-B because the automatic upgrade does not save a library in the newer TargetLink version.

Only the current TargetLink version 2022-B is available Use TargetLink 2022-B and the `tlUpgrade` API command to make the library TargetLink-compliant. Refer to How to Manually Upgrade Libraries and Models Using the API on page 175. If you follow the instructions, the library is saved in TargetLink 2022-B. Therefore, it cannot be used with TargetLink versions earlier than TargetLink 2022-B.

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 downward compatibility

You cannot use models, libraries, or 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 using 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 Using the API 175

References

tlUpgrade (TargetLink API Reference 🕮)

How to Upgrade and Save a Data Dictionary with Included DD Files

Precondition

In the main DD file to be loaded, the AutoLoad property of the DD DDIncludeFiles objects is set to on.

Method

To upgrade and save a Data Dictionary with included DD files

1. Open the Data Dictionary Manager and the main DD file using File - Open.
   The main DD file and all included DD files are automatically upgraded to the latest data model revision.
2. Set the AutoSave property of all desired `/Config/DDIncludeFiles/<DDIncludeFile>` objects to On to save the included DD files together with the main DD file.
3. Save the main DD file using File - Save.

Result

You upgraded and saved 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. Backups of the original DD files are also saved to the same directory as the DD file.

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 Using the API

Objective

To prepare a central upgrade of libraries and models in a tool chain 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 DD project file has been opened and automatically upgraded, for example, using `dsdd_manage_project('Open','<name>.dd')` or `dsdd('Open', '<name.dd>')`.

Method

To manually upgrade libraries and models using 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 libraries whose blocks and subsystems 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 and 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

TargetLink 4.3 introduced DD CodeDecorationSet and CodeDecoration objects.

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	DeclarationStatements or SectionName properties are set.	1. Create a DD CodeDecorationSet object. 2. Create 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. Reference the CodeDecorationSet object at the original object.
FunctionClass		
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 is set to `ALL_TRUE`. ▪ Settings.VariableClass references a DD VariableClass object whose DeclarationStatements or SectionName property is 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 DD VariableClassTemplate object with the same value at the Filter.VariableClassSpec property, add a DD CodeDecoration object to the CodeDecorationSet object. 4. Specify the DD CodeDecoration object as required. 5. Reference the DD CodeDecorationSet object at the VariableClass object created in step 1. 6. Reference the DD VariableClass object created in step 1 by 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 using 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 DD CodeDecoration objects can influence the generated production code 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 🕮).

Changes in Rounding Behavior for Exactly Representable Floating-Point Numbers (Microsoft)

Changes in Rounding Behavior for Exactly Representable Floating-Point Numbers (Microsoft)

Introduction

Microsoft changed the rounding behavior of the `printf` family of functions. Starting in Windows 10 version 2004 (build 19041), these functions now print exactly representable floating-point numbers as described in IEEE 754-2019.

Impact on code generation with TargetLink

This change made by Microsoft impacts code generation with TargetLink. Depending on the Windows version, the rounding behavior can differ between machines. Note, that this is independent of TargetLink.

Windows 10 < 2004 (build 19041)	Windows 10 ≥ 2004 (build 19041)
Exactly representable floating-point numbers ending in 5 would always round up.	Banker's rounding as described in IEEE 754-2019.

Migrating from TargetLink 5.2 to 2022-B

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 using the **All options...** button on this page.

Related documentation

- OptimizedBoolType (TargetLink Model Element Reference 🕮)

Recommended compatibility settings

Specify the following settings for new TargetLink 2022-B Code Generator options to ensure downward compatibility:

Code Generator Option	Setting
ReduceLifetimeOfConstantVariables (TargetLink Model Element Reference 🕮)	`off`
UseBusObjectMappingInsteadOfInheritedProperties (TargetLink Model Element Reference 🕮)	`off`

The effect of the above mentioned Code Generator options is often minor compared to the other changes regarding bus structs. Refer to Migration Aspects Regarding Optimization on page 189.

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 have been used since TargetLink 4.1.

If a model-based option value is identical to the old default value, it is automatically changed to the new default value during the upgrade. If a DD-based option value is identical to 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 is identical to 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 $TargetLink_A$ ⇒ $TargetLink_B$ ⇒ $TargetLink_C$ and upgrading $TargetLink_A$ ⇒ $TargetLink_C$ can result in 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 is identical to the default.

New Code Generator options

For more information on new Code Generator options, refer to New Code Generator Options on page 167.

AUTOSAR

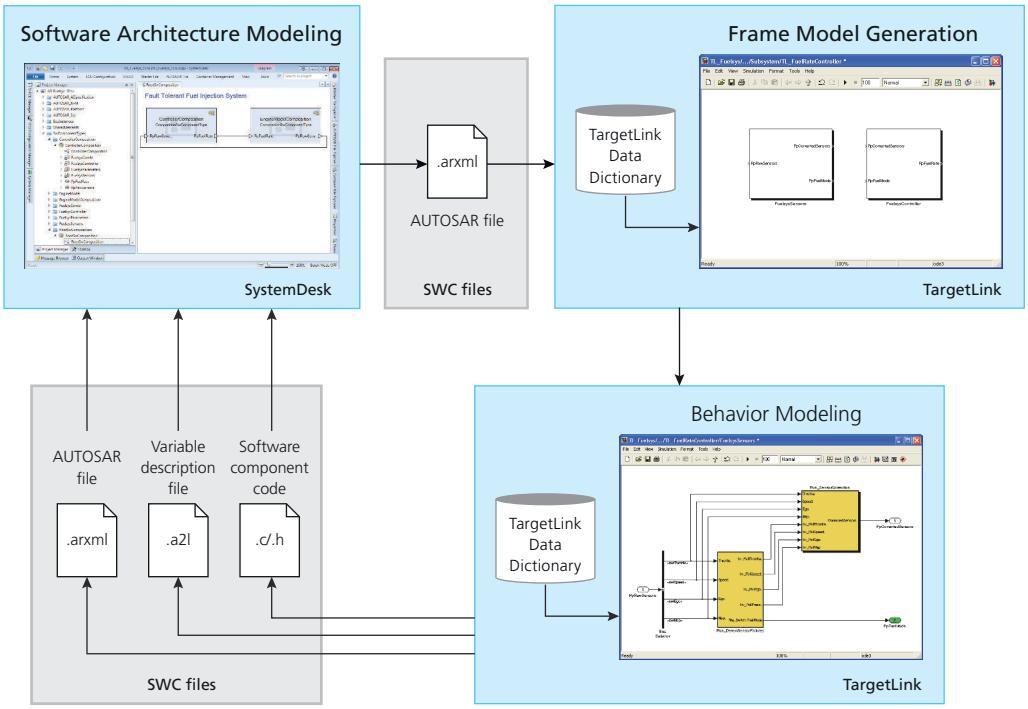
Migration Aspects Regarding AUTOSAR

Best practices for exchanging software components

For ECU software development according to AUTOSAR, SystemDesk acts as a *system-level design tool*. For *behavior modeling*, dSPACE provides the TargetLink AUTOSAR Module. If you follow this use case, you have to transfer relevant parts of the AUTOSAR model from SystemDesk to TargetLink.

dSPACE supports SIL testing of software components in the context of closed-loop simulations of V-ECUs and environment models. If you follow this use case, you additionally have to reference code files in the AUTOSAR model and specify implementation-specific aspects herein. This enables you to generate model-based V-ECUs with SystemDesk.

The illustration below shows the interoperability of SystemDesk and TargetLink:



Since the discontinuation of the *Container Manager*, a dSPACE tool for exchanging software components between TargetLink and SystemDesk, we recommend that you follow best practices, which include the assignment of

AUTOSAR elements to packages and AUTOSAR files, specific import/export-related settings in SystemDesk and the TargetLink Data Dictionary, and putting exchanged AUTOSAR files under version control. We also recommend the use of a compare and merge tool for AUTOSAR files.

For details, refer to the best practices in the user documentation of your dSPACE tool:

dSPACE Tool	Documentation
SystemDesk	Best Practices for Exchanging Software Components (SystemDesk Manual 🕮)
TargetLink	Best Practices for Exchanging Software Components (TargetLink Interoperation and Exchange Guide 🕮)

Exporting files for AUTOSAR

With TargetLink 2022-B, software-component-related files, such as ARXML files, can be generated and exported with the **TargetLink File Export Utility**.

Related documentation

- How to Export Generated Files from the TargetLink Environment (TargetLink Interoperation and Exchange Guide 🕮)
- TargetLink File Export Utility (TargetLink Tool and Utility Reference 🕮)

Changes in AUTOSAR frame model generation

As of TargetLink 2022-B, Simulink.Bus objects are created for **Bus Inport** and **Bus Outport** blocks that do not pass array-of-bus signals but reference structured DD **Typedef** objects. This behavior can be disabled via the **ForceBusObjectDatatypeUsage** property.

Related documentation

- ForceBusObjectDatatypeUsage
  (Refer to TargetLink Data Dictionary Reference)
- Basics on AUTOSAR Development Approaches and Frame Model Generation in TargetLink (TargetLink Classic AUTOSAR Modeling Guide 🕮)
- Basics on Simulink® Bus Objects Used in TargetLink (TargetLink Preparation and Simulation Guide 🕮)

Changes in modeling Adaptive AUTOSAR-related data types

As of TargetLink 2022-B, the modeling of Adaptive AUTOSAR-related data types in the **TargetLink Data Dictionary** has been changed. During the **Data Dictionary** upgrade, existing DD workspaces are automatically upgraded to the changed modeling. After the upgrade process, previously used DD **Module** objects for Adaptive AUTOSAR type definitions may be left as unused.

Related documentation

- Introduction to Data Types in Adaptive AUTOSAR (TargetLink Adaptive AUTOSAR Modeling Guide 🕮)
- Details on Defining User-Specified Data Types in Adaptive AUTOSAR (TargetLink Adaptive AUTOSAR Modeling Guide 🕮)

$(Component) macro replacement

As of TargetLink 2022-B, the `$(Component)` macro is replaced with `$(SwComponentSymbolicName)` in the following situations:

- DD upgrade

  The macro is replaced for the AccessFunction.FunctionName, Variable.NameTemplate, Runnable.NameTemplate, and CodeDecoration.Settings.DeclarationStatements properties.
- DD command `CreateActivationReasonCodeElements`

  The command replaces the macro for the NameTemplate and ModulRef properties of the created DD Variable objects.
- DD command `CreateVariableObjects`

  The command creates the DD Variable objects whose NameTemplate property is specified as `Rte_$(SwComponentSymbolicName)_$(Group)_$D` instead of `Rte_$(Component)_$(Group)_$D`.
- It is replaced for variables, macros, and functions whose name is derived from an affected variable, macro, or function. Especially the local access-function-buffer variables for vectors and macros are affected.

Related documentation

- CreateActivationReasonCodeElements (Runnable)

  (Refer to TargetLink Data Dictionary Reference)
- CreateVariableObjects (NvReceiverPort, NvSenderReceiverPort, ...)

  (Refer to TargetLink Data Dictionary Reference)

API Functions and Hook Scripts

Where to go from here

Information in this section

Changes in TargetLink and TargetLink Data Dictionary API Functions

ds_error_display

Changed property behaviors

Property	Description
`ClearMessage`	Clears displayed messages from TargetLink's message system. The property now defaults to `off`.

Related documentation ds_error_display (TargetLink API Reference 🕮)

dsdd_validate

New output parameters

Output parameter	Description
`bError`	Error flag that is true in case of an error. Otherwise the error flag is false.

Related documentation dsdd_validate (TargetLink API Reference 🕮)

tl_compile_host

Removed properties The `RebuildAll` property has been removed.

Related documentation tl_compile_host (TargetLink API Reference 🕮)

tl_compile_target

Removed properties The `RebuildAll` property has been removed.

Related documentation tl_compile_target (TargetLink API Reference 🕮)

tl_export_files

New properties

Property	Description
`CopyAutosarFiles`	Copies AUTOSAR-related files to a specified folder.

Related documentation tl_export_files (TargetLink API Reference 🕮)

tl_generate_swc_model

New properties

Property	Description
`AddArrayOfBusesAdapterBlocks`	Specifies whether to add array-of-buses adapter blocks and connect them to BusInport or DataStoreRead blocks and BusOutports or DataStoreWrite blocks that pass array-of-bus signals. Default: `off`
`BusObjectsNamePrefix`	Specifies the name prefix of Simulink.Bus objects created for unmapped structured DD Typedef objects and for non-DD Typedef objects. Default: `tlSwcGen_`
`ForceBusObjectDatatypeUsage`	Specifies whether Simulink.Bus objects should be created for BusInport or BusOutport blocks

Property	Description
	that do not pass array-of-bus signals Default: on
`UpdateBusObjectWithMappedDDTypedef`	Specifies whether Simulink.Bus objects should be updated with mapped DD Typedef objects if they are incompatible with each other. Default: on

Related documentation tl_generate_swc_model (TargetLink API Reference 🕮)

tl_pref

Renamed preferences The `AutoBackupDdFile` preference is renamed to `SaveBackupOnDDUpgrade`.

Related documentation tl_pref (TargetLink API Reference 🕮)

tl_sim

Provides an automatic correction of Simulink© output port parameters of TargetLink blocks that affect signal logging during MIL simulation. Refer to Basics on Logging Signals in MIL Simulation Mode (TargetLink Preparation and Simulation Guide 🕮).

Related documentation tl_sim (TargetLink API Reference 🕮)

tlSimulinkBusObject

Changed property behaviors

Property	Description
`DDVariableName`	Specifies the name or path of the DD Variable object.
`CreateTypedef`	Specifies if a DD Typedef object is created from the Simulink.Bus object and referenced by the DD Variable object. The property now defaults to on.

Related documentation tlSimulinkBusObject('CreateDDVariable', busObject, propertyName, propertyValue, ...) (TargetLink API Reference 🕮)

Related topics

Basics

Basics on Logging Signals in MIL Simulation Mode (TargetLink Preparation and Simulation Guide 🕮)

References

ds_error_display (TargetLink API Reference 🕮)
dsdd_validate (TargetLink API Reference 🕮)
tl_compile_host (TargetLink API Reference 🕮)
tl_compile_target (TargetLink API Reference 🕮)
tl_export_files (TargetLink API Reference 🕮)
tl_generate_swc_model (TargetLink API Reference 🕮)
tl_pref (TargetLink API Reference 🕮)
tl_sim (TargetLink API Reference 🕮)
tlSimulinkBusObject('CreateDDVariable', busObject, propertyName, propertyValue, ...) (TargetLink API Reference 🕮)

Changes in Hook Scripts

tl_post_load_hook

The predefined variable `system` was renamed to `mdlName`.

Related documentation tl_post_load_hook (TargetLink File Reference 🕮)

Related topics

References

tl_post_load_hook (TargetLink File Reference 🕮)

Migration Aspects Regarding Look-Up Table Blocks

Prelookup Block and Interpolation Using Prelookup Block

Change in block properties

With TargetLink 2022-B, the block properties of the Prelookup and the Interpolation Using Prelookup blocks have changed:

Block Property	TargetLink ≤ 2022-B	TargetLink 2022-B
Use last breakpoint for input at or above upper limit checkbox of the Prelookup block	Not evaluated by TargetLink.	Synchronized with Simulink block property and used in code generation.
Valid index input may reach last index checkbox of the Interpolation Using Prelookup block		

Potential problems

- Message `E21581` is displayed.
- Message `E21583` is displayed.
- Unexpected code change, refer to Different implementation of 1-D and 2-D Prelookup blocks on page 216.

Solution Perform the following steps:

- Configure the **Prelookup** block as described in MISRA C:2012 compliance (TargetLink Model Element Reference 🕮).
- Configure the **Interpolation Using Prelookup** block as described in MISRA C:2012 compliance (TargetLink Model Element Reference 🕮).

Related code change Refer to Different implementation of 1-D and 2-D Prelookup blocks on page 216.

Changes in DD Look-up table objects

The DD Look-up table objects for **Prelookup** blocks and **Interpolation Using Prelookup** blocks have been changed to reflect the changes in the corresponding block properties:

Property of DD Look-Up Table Object	TargetLink ≤ 2022-B	TargetLink 2022-B
Use last breakpoint for input at or above upper limit of IndexSearch Block object	–	Automatically set to `off` during migration.
Valid index input may reach last index property of DD Interpolation Block object	–	

Potential problem

- Unexpected code changes, refer to Different implementation of 1-D and 2-D Prelookup blocks on page 216.
- Message `E21583` is displayed.

Solution Adapt the DD Look-up table object to your requirements.

Related topics

References

IndexSearch Block Object Dialog (Prelookup) (TargetLink Data Dictionary Manager Reference 🕮)
Interpolation Block Object Dialog (TargetLink Data Dictionary Manager Reference 🕮)
Interpolation Using Prelookup Block (TargetLink Model Element Reference 🕮)
Prelookup Block (TargetLink Model Element Reference 🕮)

Optimization

Migration Aspects Regarding Optimization

Struct assignments

The improvements of the code efficiency for array-of-struct variables might lead to less efficient code in the rescheduling of state update statements and the scope reduction of variables. Refer to Improved Code Efficiency for Array-of-Struct on page 164.

Depending on your specific situation and use cases, obtaining production code as in previous TargetLink versions might be possible by applying a temporary workaround. Contact dSPACE product support for TargetLink.

Other Migration Aspects

Various Migration Aspects

Mapping Simulink.Bus objects to DD Typedef objects

TargetLink 2022-B introduces the SimulinkBusObject property for DD Typedef objects. The property is used to show the correspondence between a DD Typedef object and a structurally identical Simulink.Bus object. It is highly recommended to update the mappings in the Data Dictionary before opening any TargetLink model that is associated with the Data Dictionary.

Related documentation

- Basics on Simulink® Bus Objects Used in TargetLink (TargetLink Preparation and Simulation Guide 🕮)
- SimulinkBusObject
  (Refer to TargetLink Data Dictionary Reference)

Pointer-to-struct variables at function interfaces

With TargetLink 2022-B you have more options to explicitly control formal parameters of pointer-to-struct types at function interfaces using the new DD PassStructToComponentReferences property.

In prior versions, TargetLink automatically generated pointer-to-struct-transport parameters at the function interface if the following conditions applied:

- A model element that contributed to the function interface referenced a structured DD Variable object or one of its components.
- The structured DD Variable object referenced a DD VariableClass object whose Scope property was set to `ref_param`.

These formal parameters have special semantics regarding the actual parameters and their use across several functions. Conversely, at bus port blocks they behaved differently from block variables specified via a structured DD **Typedef** object reference and a reference to a DD **VariableClass** object with its **Scope** property is set to `ref_param`.

When upgrading an DD file to TargetLink 2022-B, the DD **PassStructToComponentReferences** property is added to all DD **Variable** objects.

The **PassStructToComponentReferences** property is set to **on** if all of the following conditions apply:

- The DD **Variable** object is structured but not a struct component.
- The DD **Variable** object is not an array-of-struct variable and does not contain array-of-struct components.
- The **Scope** property of the referenced DD **VariableClass** object is set to `ref_param`.

The resulting variables are called pointer-to-struct-transport variables. You can reference their components at model elements.

By default, new DD **Variable** objects have their **PassStructToComponentReferences** property not set. That is equivalent to `off`. Referencing a DD **VariableClass** object with its **Scope** property is set to `ref_param`, they are interpreted as bus structs, i.e., representing a complex value with struct type rather than a collection of individual values collected in a structured variable. You are not allowed to reference components of this structured variable at model elements. But the resulting code is structurally identical to reference parameters of other types for bus port or port blocks if referenced at a bus port block.

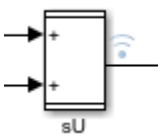
Related documentation

- Defining the Function Interface (TargetLink Customization and Optimization Guide 🕮)
- Working With Pointers to Structures (TargetLink Preparation and Simulation Guide 🕮)
- STRUCT_INTERFACES (TargetLink Demo Models 🕮)

Direct mapping of the TargetLink logging property to the Simulink data logging

As of TargetLink 2022-B, the **Data to log** (logdata.loggingmode) property is mapped to the DataLogging parameter of the Simulink port or the Stateflow data object.

Thus, as soon as logging is activated for a TargetLink block, data logging of the corresponding Simulink output port or Stateflow object is also activated. This results in a viewer icon being displayed at the output of the respective block.



If you want to make these viewer icons invisible in your model, you can enter one the following commands in the MATLAB Command Window:

- `set_param(<model>, 'ShowViewerIcons', 'off')`
- `set_param(<model>, 'ShowTestPointIcons', 'off')`

Impact on comparing models This changed property mapping can have an impact on downstream analyses, such as model comparisons with Model Compare.

For example, up to TargetLink 5.2, changing the global logging option only resulted in a single TargetLink model property (logopt.globalloggingmode) being changed.

As of TargetLink 2022-B, changing the global logging option additionally changes the Simulink parameters of very many TargetLink block ports. This leads to correspondingly numerous Model Compare differences between an otherwise identical model with, e.g., global logging option **Log according to block data** or **Log signal histories**.

Related documentation

- Basics on Logging Signals in MIL Simulation Mode (TargetLink Preparation and Simulation Guide 🕮)
- Basics on Logging Global Variables By Address (TargetLink Preparation and Simulation Guide 🕮)

New log macros

Up to TargetLink 5.2, if Simulink logging was enabled in a TargetLink model for output ports of TargetLink blocks, this was ignored.

As of TargetLink 2022-B, if the **Global logging option** is set to **Log according to block data**, the **Data to log** property is also set to **Signal history** for these TargetLink blocks. This in turn leads to log macros being generated during code generation.

Related documentation

- Impact of Log Macros on the Generated Code (TargetLink Preparation and Simulation Guide 🕮)

MIL/SIL comparison for logging by address

Up to TargetLink 5.2, for variables for which logging by address (in SIL/PIL mode) was specified in the Data Dictionary, their corresponding block signal was also automatically logged in MIL mode.

As of TargetLink 2022-B, this no longer works.

In order to perform a MIL/SIL comparison for a model with logging by address, the following conditions must apply:

- The **Clean code** checkbox on the **Code Generation** page of the TargetLink Main Dialog is selected.

  Accordingly, there is no code instrumentation and therefore there are no log macros.
- The following applies to the TargetLink blocks to be logged:
  - Signal logging (local or global) is activated.

    Due to the set **Clean code** option, this only affects the MIL simulation.

- After code generation, logging by address is specified in the Data Dictionary for the variables to be logged.

  Due to the changed TargetLink behavior, this only affects the SIL and PIL simulation.

Related documentation

- Basics on Logging Signals in MIL Simulation Mode (TargetLink Preparation and Simulation Guide 🕮)
- Basics on Logging Global Variables By Address (TargetLink Preparation and Simulation Guide 🕮)

No overflow detection with set model parameter ConditionallyExecuteInputs

With TargetLink 2022-B the overflow detection and the min/max logging (each without signal logging) only work if the parameter `ConditionallyExecuteInputs` is not set.

For this reason, `tlUpgrade` sets the `ConditionallyExecuteInputs` parameter to `off` and during a model upgrade a corresponding message is displayed.

As a consequence of disabling the `ConditionallyExecuteInputs` parameter, calculations that were previously excluded from optimization are now performed and the execution time may increase during MIL simulation. This only affects the pure Simulink parts of the model, since in previous TargetLink versions the conditional branch execution optimization of affected TargetLink blocks was prevented locally.

If the `ConditionallyExecuteInputs` model parameter is set to `on`, it will result in a warning that the overflow detection and the min/max logging will not work. You can avoid this warning by disabling overflow detection and not setting the global logging option to Log min/max-values.

Either overflow detection or conditional input branch execution - depending on your use case The default settings are optimized for using the overflow detection and working in MIL simulation mode.

For focusing on code generation and working in SIL or PIL simulation mode you should switch off the overflow detection and set the `ConditionallyExecuteInputs` model parameter to `on`. If MIL logging is no longer needed, the performance can be improved this way.

Related documentation

- Overflow Detection (Limitations) (TargetLink Limitation Reference 🕮)
- Basics on Overflow Detection (TargetLink Preparation and Simulation Guide 🕮)

Local logging does not correspond to global logging setting

As of TargetLink 2022-B, the Simulink logging parameters are set, when the global logging option is changed.

If you set the block-specific TargetLink logging properties by the `tl_set` command, the block dialog or the Property Manager, TargetLink ensures that the local properties correspond to the defined global logging option.

However, in the following cases this consistency is not guaranteed:

- New TargetLink blocks are added to the model.
- Simulink blocks are enhanced.
- The Simulink logging parameters are changed using Simulink means (e.g., `set_param`).

To avoid such inconsistencies at simulation start, TargetLink automatically performs a correction of such inappropriate Simulink logging parameters when starting a simulation using the `tl_sim` command or the **Start simulation** button () in the **TargetLink Main Dialog** or in a plot window. Additionally, the logging of signals that lead into inports of Merge blocks is also deactivated.

This automatic correction can be deactivated by the `tl_set(<model>, 'logopt.applygloballogging', 'off')` command.

Related documentation

- ApplyGlobalLogging (TargetLink Model Element Reference 🕮)
- Basics on Logging Signals in MIL Simulation Mode (TargetLink Preparation and Simulation Guide 🕮)

Performance improvement in MIL simulation mode

To improve the performance in MIL simulation mode, TargetLink 2022-B lets you deactivate the overflow detection. This means that when starting the simulation, the quite time-consuming reading of the TargetLink block data only has to take place for TargetLink blocks with activated signal logging.

Especially if you want to log only a few TargetLink blocks in a large model, switching off the overflow detection is a good option to improve performance. You can deactivate the overflow detection for your model using the `tl_set(<model>, 'logopt.detectoverflows', 'off')` command.

Disabling affects the global logging modes **Log according to block data** and **Do not log anything**. For performance reasons, in the global logging mode **Log according to block data** the block-specific logging is ignored if it is set to **Min/Max values**.

Related documentation

- Basics on Overflow Detection (TargetLink Preparation and Simulation Guide 🕮)

Simulink error with Merge block inport logging

When a signal is fed into the inport of a merge block with logging enabled, the simulation may abort with a Simulink error in the following cases:

- The simulation is started by the Simulink `sim` command or by clicking the **Run** button on the model toolbar.
- The ApplyGlobalLogging model property is set to `'off'`.

  In this case, the error may appear even if the simulation is started by the `tl_sim` command or the **Start simulation** button () in the **TargetLink Main Dialog** or in a plot window.

To avoid this error, deactivate the signal logging at the relevant blocks.

Related documentation

- Basics on the Merge Block (TargetLink Preparation and Simulation Guide 🕮)
- Basics on Logging Signals in MIL Simulation Mode (TargetLink Preparation and Simulation Guide 🕮)
- Basics on Logging Using Log Macros (TargetLink Preparation and Simulation Guide 🕮)
- ApplyGlobalLogging (TargetLink Model Element Reference 🕮)

Errors while upgrading models containing linked library blocks

During the upgrade process of TargetLink 2022-B, MATLAB® may report errors on modifying locked (read-only) libraries. Those could be related to incorrect modeling as follows:

- A TargetLink block is directly linked to a library.
- A TargetLink block within a library is linked to a block in the same library.

To solve those errors, upgrade the library manually via the `tlUpgrade` (TargetLink API Reference 🕮) API command:

```
tlUpgrade('Model', '<MyLibrary>', 'CheckModel', 'FixIssues');
```

Related documentation

- How to Manually Upgrade Libraries and Models Using the API on page 175

Message handling in custom look-up functions

With TargetLink 2022-B, the custom look-up functions API `tlscript` collects all messages in the message stack. This ensures that no error is undetected.

To suppress messages, you have to use the `ds_error_none` API function as follows:

```
tlsscript('Set', ...)

if ds_error_check
  ds_error_none
end
```

Related documentation

- Basics on Using Custom Look-Up Functions (TargetLink Preparation and Simulation Guide 🕮)
- Variable Initialization and Setting Attributes (TargetLink API Reference 🕮)
- ds_error_none (TargetLink API Reference 🕮)

New PassStructToComponentReferences property

With TargetLink 2022-B, the PassStructToComponentReferences property was added for DD Variable objects. Via the PassStructToComponentReferences property, you can specify whether leaf struct components of pointer-to-struct-transport variables are passed by reference via the pointer-to-struct-transport mechanism.

During DD upgrade, the property is automatically activated for DD Variable objects with a DD VariableClass object whose Scope property is set to `ref_param`. This ensures that the pointer-to-struct mechanism of TargetLink 5.2 is used.

If you are using scripts that specify the **Scope** property to `ref_param` or reference another DD **VariableClass** object whose **Scope** property is set to `ref_param`, the **PassStructToComponentReferences** property remains unchanged and must be explicitly activated to use the TargetLink 5.2 pointer-to-struct mechanism.

Related documentation

- PassStructToComponentReferences
  (Refer to TargetLink Data Dictionary Reference)
- Basics on the Pointer-to-Struct-Transport Mechanism (TargetLink Preparation and Simulation Guide 🕮)

Access function templates at struct components

With TargetLink 2022-B, the Code Generator prefers struct assignments to copy structs. If the struct components reference a DD **VariableClass** object that references a DD **AccessFunctionTemplate** object or inherit an AccessFuntionTemplate reference from a parent struct variable, the encapsulation by access function calls may not happen.

To ensure encapsulation, you can model component accesses explicitly or deactivate the **AllowStructAssignments** Code Generator option.

Related documentation

- AllowStructAssignments (TargetLink Model Element Reference 🕮)

Code Changes Between TargetLink 5.2 and TargetLink 2022-B

Where to go from here

Information in this section

Struct-Related

Changes in variable names

In the context of structures, different variables from those in TargetLink 5.2 might be removed by optimization. This can result in changes of variable names.

Additionally, the names of auxiliary variables that are generated for structured formal parameters that are qualified as `const` have changed. Consider the following example::

TargetLink ≤ 5.2	TargetLink 2022-B
`DE_Record_Aux = *p_DE_Record;` `foo(DE_Record_Aux);`	`In = *p_DE_Record;` `foo(In);`

Reason Increased consistency

Related code changes Naming of auxiliary variables on page 215

Access functions

Due to the precedence of struct copy via struct assignment, struct components that are encapsulated via access functions may effectively lose this encapsulation, as the respective components are evaluated or modified via parent struct accesses. Encapsulating component-wise accesses for bus structs can be achieved by disassembling the bus, inserting blocks that explicitly read or write the respective value, and reassembling the bus. Note that access functions for struct components can be specified either directly via the components' respective variable class or via access function template propagation via the PropagateStructToComponents setting of a parent struct's AccessFunctionTemplate object.

For access functions that apply to structs and whose CreateLocalValueCopy property is set to anything but `avoid`, code might change in the following contexts:

- The structure is the output of a block within a subsystem.
- The structure is an intermediate variable in a function.

In these contexts, the code might contain auxiliary variables that have a structured type in order to minimize the number of calls to access functions.

Reason Increased consistency

Actual parameters

Code might change if one of the following blocks is used to name an actual parameter of a function:

- Bus Inport block
- Bus Outport block
- Signal Conversion block

Optimization might move assignments that are using the actual parameter variable into conditionally executed control flows. If the function call cannot be moved into the conditionally executed control flow branch and if elimination as an intermediate variable is prohibited, an additional variable and struct assignment results.

TargetLink ≤ 5.2	TargetLink 2022-B

`struct Foo W_E;`
 	
`ExternFunction(pInput1, &W_E);`
 	
`if (Input2 > 0) {`
 	
`... Read from W_E ...`
	`struct Foo Out;`
	
	`ExternFunction(pInput1, &Out);`
	
	`if (Input2 > 0) {`
	`struct Foo W_E;`
	
	`W_E = Out;`
	`... Read from W_E ...`

Reason Increased consistency

Migration issue Set the Optimization property of the variable classes that are associated with the struct and its components consistently to either full or no optimization:

- Either clear both the MOVABLE and the ERASABLE checkboxes in order to prevent moving the subsequent assignment into conditionally executed control flow branches,
- Or select both the MOVABLE and the ERASABLE checkboxes in order to allow elimination of the actual parameter.

Replace struct assignment by component assignment

TargetLink might now replace a struct assignment by individual assignments to struct components. This takes place when a sufficient number of struct components are unused so that the resulting number of assignments remains small. Consider the following example:

TargetLink ≤ 5.2	TargetLink 2022-B

`S = T`
 	
`... = S.c`
	`S.c = T.c`
	
	`... = S.c`

You can influence this behavior via the StructAssignmentDecompositionStatementMultiplicationThreshold (TargetLink Model Element Reference 🕮) Code Generator option.

Additionally, the replacement of a struct assignment by assignments of auxiliary variables to struct components is possible, if this results in a sufficiently large improvement of memory usage, because most of the components are not used. Consider the following example:

TargetLink ≤ 5.2	TargetLink 2022-B

`S = T`
 	
`... = S.c`
	`Aux_S_c = T.c`
	
	`... = Aux_S_c`

You can influence this behavior via the StructAssignmentDecompositionAuxVarMemoryUsageThreshold (TargetLink Model Element Reference 🕮) Code Generator option.

Reason Code efficiency

Structured reference parameters

TargetLink 2022-B now consistently distinguishes between the following:

- Passing additional structured reference parameters to a function.
- Passing components via pointer-to-struct-transport parameters. This can include referencing the root struct variable as a whole, see below.

Because TargetLink now treats array-of-struct variables like other structures and, because TargetLink does not support array-of-struct variables as pointer-to-struct-transport variables, you might observe the following code change:

TargetLink ≤ 5.2	TargetLink 2022-B
Additional array-of-struct function parameters could be erroneously propagated down the function-call hierarchy.	Additional array-of-struct function parameters only appear in the functions they were selected for at the corresponding model element. They are no longer propagated.

Tip

The `STRUCT_INTERFACES` demo model shows the different ways to work with structured function parameters.

Note that you can reference pointer-to-struct-transport variables at bus port blocks just like ordinary struct variables with `ref_param` scope. The former aims for propagation through the model with exactly one actual variable for all calls, the latter leads to one actual variable per resulting call.

Reason Increased consistency

Related topics

References

Bus Inport Block (TargetLink Model Element Reference 🕮)
Bus Outport Block (TargetLink Model Element Reference 🕮)
Signal Conversion Block (TargetLink Model Element Reference 🕮)

Bus-Struct-Related

Changes with structures resulting from buses

The optimization was improved for block variables of bus port blocks and bus-capable blocks when DD Typedef objects are used for explicit specification of the struct. TargetLink now creates struct assignments for struct variable copying with high priority and based on these assignments, consistently treats scalar struct variables like scalar numerical variables with regard to optimization order and performed optimizations whenever possible, and nonscalar struct variables similar to numerical vectors and matrices.

TargetLink now more often generates struct assignments than assignments to components and eliminates unnecessary intermediate variables of struct type more reliably.

As a side effect of this improvement, the generated code might be less optimized in the following contexts:

- Transitions between different structured types, especially when using the `IMPLICIT_STRUCT` type or an empty struct typedef with fixed name.
- Assembly of structured variables from non-structured variables.

  In rare cases, additional structured variables might remain in the generated code, especially with interspersed code from other model elements or with conditional control flows.
- Creation of structured variables from structure constants, e.g., for the block variables of Constant blocks or Stateflow initialization variables.

  Initialized structured variables with static or automatic storage duration might remain in the optimized code, allowing for faster copying. The initial value is not propagated.
  - To get code as with prior versions, do not use initialized variables but rather values or force component-wise struct copying by using a different struct type.

Reason Increased consistency

Fewer component assignments in the context of buses

With TargetLink 2022-B, struct assignments take precedence before component assignments. In the context of buses, this generally results in a reduced number of auxiliary variables. In modeling situations where only partial bus signals are selected, auxiliary variables might remain in the optimized code, because individual structured variables might be removed, leaving the structured type untouched.

Reason

- Increased consistency
- Readability

Related code changes Replace struct assignment by component assignment on page 197

Signal line splits

As a side effect of the precedence of struct assignments, the code resulting from a signal line split might not be optimized as well as in prior versions.

Reason Side effect

To be fixed in a future TargetLink version.

Related code changes Changes with structures resulting from buses on page 198

Stateflow charts and bus signals

For bus signals that are input into a Stateflow Chart object where the SFInput object has createinputvariable set to `true`, the code might not be optimized as well as in prior versions.

Structured input variables of Stateflow Chart objects can be removed by optimization if their initialization can be done via a struct assignment. For example:

```
Main() {
    S = T;
    Chart();
}
Chart()
{
    a = S.a;
    b = S.b;
}
```

can be optimized to:

```
Main() {
    Chart();
}

Chart() {
    a = T.a;
    b = T.b;
}
```

If S can be assigned only component-wise, because of change in struct type or transition from plain variables to struct components, no optimization takes place. For example:

```
S.a = a1;
S.b = b1;
```

Migration issue To facilitate optimization, you can do one of the following:

- Use the same structured type for S and T.
- Do not use structured types.
- Set createinputvariable to `false`.

Reason Side effect

No inheritance of variable classes of struct components

In prior TargetLink versions, a bug existed concerning the determination of variable classes for block variables of bus-capable blocks. This bug occurred when a block B, that had its Inherit properties checkbox selected, inherited the structured type from another block A . This has been fixed with TargetLink 2022-B as shown in the following table:

TargetLink ≤ 5.2	TargetLink 2022-B
Inheriting the variable class of struct components from block A to block B.	Variable class of struct components not inherited from block A but determined by TargetLink at block B: TargetLink now sets the Optimization properties of the variable classes of the struct components consistently with the Optimization properties of the variable class specified for the root of the structure.

Reason

- Bug fix
- Increased consistency

Migration issue Two cases might be observed:

1. The struct components of block A are not ERASABLE, *and* the root of the structure at block B *is* ERASABLE:

   TargetLink will now remove the structured variable of block B.

2. The struct components of block `A` are `ERASABLE` *and* the root of the structure at block `B` *is not* `ERASABLE`:

   TargetLink no longer removes the structured variable of block `B`.

To explicitly specify the variable class for the components of the block variable of block `B`, you can either use a DD Variable object or a DD Typedef object and the Component.class property.

Array-of-struct reference parameters at function interfaces

TargetLink now supports array-of-struct variables as reference parameters. Accordingly, the code generator now uses the internal default templates `SLFcnInputStruct` and `SLFcnOutputStruct` instead of circumventing them. The underlying variable classes of these templates by default have `ref_param` scope.

This results in the following code change in the context of atomic subsystems that are not function-call-triggered:

TargetLink ≤ 5.2	TargetLink 2022-B
TargetLink implemented the subsystem's unenhanced ports passing array-of-bus signals as variables with `global` scope.	TargetLink implements the subsystem's unenhanced ports passing array-of-bus signals as variables with `ref_param` scope.

Migration issue If you want to generate code as in TargetLink 5.2, do the following:

- Enhance the Simulink port blocks to TargetLink bus port blocks.
- If you specify the bus via a Simulink BusObject and do not yet have a mapping, create a mapped struct type via the `tlSimulinkBusObject('CreateDDTypedef', busObject, propertyName, propertyValue, ...)` API function and reference it at the bus port blocks.
- Reference a DD VariableClass whose Scope property is set to `global` at the bus port blocks.

Reason Increased consistency

Outports of scaling-invariant subsystems

Code has changed for scaling-invariant subsystems if all of the following conditions apply:

- A bus signal leaves the scaling-invariant subsystem via a scaling-invariant Bus Outport block.
- The bus signal is implemented as a struct with `global` scope.
- TargetLink creates intermediate variables in the unoptimized code at the left-hand side of assignments after the call of the scaling-invariant function.

TargetLink ≤ 5.2	TargetLink 2022-B

Each leaf struct components was assigned to a non-structured intermediate variable:
`/* call of function: TL_Root/Caller/ScalingInvariant */`
`Sa3_ScalingInvariant();`
 	
`/* update(s) for outport TL_Root/Caller/ScalingInvariant/OutPort */`
`a = Sa3_OutPort.a;`
`b = Sa3_OutPort.b;`
 	
`/* BusOutport: TL_Root/Caller/OutPort */`
`Sa2_OutPort.a = a;`
`Sa2_OutPort.b = b;`
	The intermediate variable is a struct that has the same type as the global struct generated for the bus signal:
	`/* call of function: TL_Root/Caller/ScalingInvariant */`
	`Sa3_ScalingInvariant();`
	
	`/* update(s) for outport TL_Root/Caller/ScalingInvariant/OutPort */`
	`Sa3_OutPort_a = Sa3_OutPort;`
	
	`/* BusOutport: TL_Root/Caller/OutPort */`
	`Sa2_OutPort = Sa3_OutPort_a;`

Reason Increased consistency

Related topics

References

Bus Outport Block (TargetLink Model Element Reference 🕮)

Code Order and Loop Structure

Changed position of state and Data Store Memory updates

With TargetLink 2022-B, struct assignments take precedence over other optimizations. The algorithm behind the ExtendedLifeTimeOptimization Code Generator option has been reworked to support this change:

- Other optimization algorithms such as elimination of intermediate variables or moving code into conditionally executed control-flow branches are given precedence over rescheduling state update or data store memory update statements.
- The initial order of the update statements is preserved if reordering serves no immediate purpose. This leads to greater code stability across TargetLink versions.
- Value assignments to Merge block output variables and output variables of conditionally executed subsystems are no longer considered as state updates.

This can result in code differences that are not related to structs. The following might be observed for update statements:

- Consistently ordered from lowest to highest index.
- Interchanged pairwise
- Changed position further down in the code as other accesses to the state or data store memory have been moved downwards or this kind of variable is no longer considered relevant.

- Changed position further up in the code as other control flow optimizations such as loop merging or combining control flow branches allow for additional freedom of movement.
- As a result of the above: Placed in a different loop.

Also refer to Differently merged loops on page 203.

Reason

- Increased consistency
- Code stability with future TargetLink versions

Migration issue Refer to Migration Aspects Regarding Optimization on page 189.

Differently merged loops

With TargetLink 2022-B, struct assignments take precedence over other optimizations and over encapsulation via access functions for struct components. In combination with the changes in the algorithm behind the ExtendedLifeTimeOptimization Code Generator option, loop merging may behave differently:

- More loop merging
- Statements jumping between loops
- Less loop merging because of read/write access functions:
  - The loops are separated by calls of access functions.
  - More conservative data flow assumptions for access functions accessing one element depending on the loop iteration, i.e., loops with reading and loops with writing access function calls are no longer merged.

Reason Increased consistency

Migration issue Refer to Migration Aspects Regarding Optimization on page 189.

Changed loop code for trigger patterns

The LoopUnrollThreshold is now used in the context of triggered subsystems. This can result in the following changes:

- Loops might be unrolled.
- Loops for state-update code are merged more often.
- The data type of the loop variable changed to the default integer type, S32 in most cases.

Reason Increased consistency

Scope-Reduction-Related

Scope reduction of structured variables

Regarding scope reduction, structured variables are now treated in the same way as numerical variables. Structured variables are now further reduced in scope which can result in code changes.

Reason Increased consistency

Related code changes

- Scope reduction in the context of initialization on page 204
- Changed position of variable definitions on page 204

Scope reduction in the context of initialization

The scope of variables (V) can change, if they are used in the initialization of other variables (X):

- TargetLink can now reduce the scope of V to the scope of X.
- Due to improvements in the optimization sequence, these scope reductions take place more frequently than in prior versions.

Reason Code efficiency

Changed position of variable definitions

With TargetLink 2022-B, the scope reduction of structures was improved. This can result in the following code changes:

1. The positions of variable definitions might change, if both of the following conditions are fulfilled:
   - The (structured) variable or one of its components is used in the definition of a structured variable V.
   - The scope of V is reduced during optimization.
2. This can yield further changes in the position of other variable definitions.

Reason

1. Code efficiency
2. Facilitates compilability

Reduction of storage duration of unchanged structs

TargetLink now reduces struct variables to automatic storage duration if all of the following conditions apply:

- They hold an initial value.
- They are not modified.

You can control this behavior via the ReduceLifetimeOfConstantVariables Code Generator option.

Reason Code efficiency

Improved Code Efficiency

Comparisons with identical operands

TargetLink's optimization of comparisons with identical operands was improved. The following expressions are now optimized:

Expression	Replacement
`a < a`	`0`
`a > a`	
`a != a`	
`a <= a`	`1`
`a >= a`	
`a == a`	

Reason Code efficiency

Variable vector width variables in Restart functions

The initialization of variable vector width variables with homogeneous initial values was improved. Superfluous auxiliary variables for initialization are no longer generated:

TargetLink ≤ 5.2

```
#if SR_CIRCULAR_BUFFER_SIZE == 200
int16_t Temp__a[SR_CIRCULAR_BUFFER_SIZE] =
{
/* [0..9] */ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
/* [10..19] */ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
...
};
#elif SR_CIRCULAR_BUFFER_SIZE == 300
int16_t Temp__a[SR_CIRCULAR_BUFFER_SIZE] =
{
/* [0..9] */ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
/* [10..19] */ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
...
};

...

for (Aux_S32 = 0; Aux_S32 < SR_CIRCULAR_BUFFER_SIZE;
Aux_S32++)

{
algSensorHistoryBuff.SensorHistory1_as16[Aux_S32] =
Temp__a[Aux_S32];
}
```

TargetLink 2022-B

```
for (Aux_S32 = 0; Aux_S32 < SR_CIRCULAR_BUFFER_SIZE;
Aux_S32++)

{
algSensorHistoryBuff.SensorHistory1_as16[Aux_S32] = 0;
}
```

Reason Code efficiency

Improved propagation of initial values for matrix accesses

Propagation of initial values for matrix access was improved in situations where the following conditions apply:

- One of the matrix dimensions is unrolled.
- All initial values are the same.

TargetLink ≤ 5.2	TargetLink 2022-B

`for (Aux_S32 = 0; Aux_S32 < 5; Aux_S32++)`
`{`
`  /* Sink: Subsystem/Enabled Subsystem/Sink2 */`
`  Sa2_Sink2[Aux_S32][0] = Struct_OpArg.Matrix[Aux_S32][0];`
`  Sa2_Sink2[Aux_S32][1] = Struct_OpArg.Matrix[Aux_S32][1];`
`  Sa2_Sink2[Aux_S32][2] = Struct_OpArg.Matrix[Aux_S32][2];`
`  Sa2_Sink2[Aux_S32][3] = Struct_OpArg.Matrix[Aux_S32][3];`
`}`
	`for (Aux_S32 = 0; Aux_S32 < 5; Aux_S32++)`
	`{`
	`  /* Sink: Subsystem/Enabled Subsystem/Sink2 */`
	`  Sa2_Sink2[Aux_S32][0] = 42;`
	`  Sa2_Sink2[Aux_S32][1] = 42;`
	`  Sa2_Sink2[Aux_S32][2] = 42;`
	`  Sa2_Sink2[Aux_S32][3] = 42;`
	`}`

Reason Code efficiency

Removal of superfluous cast of function call

For additions, subtractions, or multiplications TargetLink now removes superfluous casts from function calls.

TargetLink ≤ 5.2	TargetLink 2022-B
`U16Var = (UInt16)((UInt16)U16ReturningFcn() + 1);`	`U16Var = (UInt16)(U16ReturningFcn() + 1);`

Reason Code efficiency

Conditional compilation for struct variable encapsulation

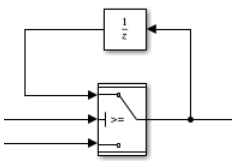
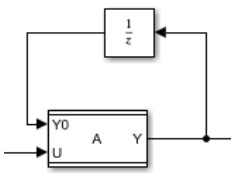
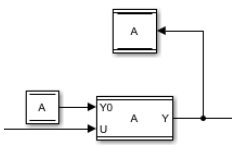
The code for struct variables whose components point to variables has changed. To decrease RAM consumption, TargetLink now encapsulates these structures in `#if`/`#ifdef` directives if all of the following conditions apply:

- All uses of the variables are in the `#if` range
- All of the variable components are in the `#if` range

Reason Code efficiency

Signal line splits

The optimization of the following modeling patterns was improved:

Modeling Style	Modeling Example
Switch block followed by a signal line split that is fed back into a Delay block or Unit Delay block.	
Assignment block followed by a signal line split that is fed back into a Delay block or Unit Delay block.	
Assignment block or Switch block is followed by a signal line split that is fed back into a Data Store Write block and the corresponding Data Store Read block drives the Assignment block or Switch block.	

Consider the following example for an Assignment block:

TargetLink ≤ 5.2	TargetLink 2022-B

`for (i = 0; i < 10; i++) {`
 	
`  Assign[i] = X[i];`
 	
`}`
 	
`Assign[index] = U;`
 	
`... Use Assign[]`
 	
`for (k = 0; k < 10; k++) {`
 	
`  X[k] = Assign[k];`
`  ...`
	`X[index] = U;`
	
	`... Use X[]`

Reason Code efficiency

Related code changes Utilization of known alias relationships on page 213

Function inlining

The inlining of functions has changed, if the following conditions apply:

- They use user-specified variables with `local` scope and `static` storage duration.
- They are called several times from exactly one caller.

TargetLink ≤ 5.2	TargetLink 2022-B
Not inlined	Inlined

This can result in a different order of further optimizations and usually results in better code. Additionally, message `W15504` might be displayed more or less often.

Migration issue Generally, we recommend that you specify a variable whose DD VariableClass object is specified as follows:

- The Scope property is set to `global`.
- The Optimization property is set to `SCOPE_REDUCIBLE`.
- The ScopeReducedClass property references one of the following:
  - A DD VariableClass object whose Scope property is set to `local` and whose Storage property is set to `static`.
  - A scope-reduction chain, that contains a DD VariableClass object whose Scope property is set to `local` and whose Storage property is set to `static`.

If the static local variable is required, you can prohibit inlining by referencing an appropriate function class at the parent function.

Reason Code efficiency

Related topics

References

Assignment Block (TargetLink Model Element Reference 🕮)
Data Store Read Block (TargetLink Model Element Reference 🕮)
Data Store Write Block (TargetLink Model Element Reference 🕮)
Delay Block (TargetLink Model Element Reference 🕮)
Switch Block (TargetLink Model Element Reference 🕮)
Unit Delay Block (TargetLink Model Element Reference 🕮)

Adaptive AUTOSAR-Related

Limited optimization of simulation code

Simulation code that is generated from the following model elements is not fully optimized:

- Method Call subsystem

Only the following optimizations are performed:

- Generation of struct assignments
- Scope reduction of variables
- Replacement of several variables by a common auxiliary variable

C++ data types in behavior and simulation code

If TargetLink generates code for a C++ module, type aliases are used instead of type definitions.

TargetLink ≤ 5.2	TargetLink 2022-B
`typedef int MyType[3];`	`using MyType = std::array<int, 3>;`

As of TargetLink 2022-B, TargetLink supports type definitions in C++ namespaces.

```
namespace my {
   namespace types {
      using MyInt16_WithNS = std::int16_t;
      using MyArrV3 = std::array<MyInt16_WithNS, 3>;
   }
}

struct MyStrType {
   my::types::MyInt16_WithNS a;
   std::int16_t b[2][3];
   my::types::MyArrV3 c;
};
```

Reason Matches user expectations

C++ array data types

As of the native support for `ara::core::Array`, `ara::core:Array` data types are used at Adaptive AUTOSAR function interfaces instead of C arrays.

For `std::array` and `ara::core::Array` data types that are used as function parameters, TargetLink generates references.

```
/**********************************************************************************\
 ***  FUNCTION:
 ***      Sa2_FcnVarsWithTypedef
 ***
 ***  DESCRIPTION:
 ***
 ***
 ***  PARAMETERS:
 ***      Type               Name               Description
 ***      ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
 ***      MyInt16            Sa2_InPort
 ***      MyArrayV_3 &       Sa2_InPort1
 ***      MyArrayM_2x3 &     Sa2_InPort2
 ***      MyInt16 *          Sa2_OutPort
 ***      MyArrayV_3 &       Sa2_OutPort1
 ***      MyArrayM_2x3 &     Sa2_OutPort2
 ***
 ***  RETURNS:
 ***      void
 ***
 ***  SETTINGS:
 ***
\**********************************************************************************/
void Sa2_FcnVarsWithTypedef(MyInt16 Sa2_InPort, MyArrayV_3 & Sa2_InPort1,
  MyArrayM_2x3 & Sa2_InPort2, MyInt16 * Sa2_OutPort, MyArrayV_3 & Sa2_OutPort1,
  MyArrayM_2x3 & Sa2_OutPort2)
```

The changed function signatures cause omission of the copy operations from C to C++ data types in the following adapter code functions:

- Events: `Receive_` and `Send_`.
- Fields: `Get_`, `Set_`, and `Update_`.
- Methods: `Call_`, `Method_`.

Since the C++ data types may belong to a namespace, the corresponding namespace is generated with the data type.

The Code Generator introduces auxiliary variables in the following situations:

- The formal parameter of a function is a C array and the actual parameter is a C++ array.
- The formal parameter of a function is a C++ array and the actual parameter is a C array.

TargetLink ≤ 5.2	TargetLink 2022-B

`AdaptiveAutosarFunction_For_MethodBehavior_2(..., STopFunctionSystemID1_InPort1, ...);`	`ds::VectorIdt_W3 Aux__a; // ara::core::Array type`
	`Aux__a[0] = STopFunctionSystemID1_InPort1[0];`
	`Aux__a[1] = STopFunctionSystemID1_InPort1[1];`
	`Aux__a[2] = STopFunctionSystemID1_InPort1[2];`
	
	`AdaptiveAutosarFunction_For_MethodBehavior_2(..., Aux__a, ...);`

Reason Code efficiency

No anonymous structure type definitions

For structure types in C++ files, TargetLink generates the structure types in a fixed format, regardless of whether the CreateTypedef property is set to on or off. The structure type always has a name.

TargetLink ≤ 5.2

```
struct MyNtdStrType_Typedefs_tag {
   MyInt16 aa;
   MyArrayV_3 ab;
};

typedef struct MyStrType_Typedefs_tag {
   MyInt16 aa;
   MyArrayV_3 ab;
} MyStrType_Typedefs;
```

TargetLink 2022-B

```
struct MyNtdStrType_Typedefs {
   MyInt16 aa;
   MyArrayV_3 ab;
};

struct MyStrType_Typedefs {
   MyInt16 aa;
   MyArrayV_3 ab;
};
```

This also affects uses of structure types that are defined in C++ files. For structure types whose CreateTypedef property is set to off, the type name is used as name.

TargetLink ≤ 5.2

```
extern struct MyNtdStrType_Typedefs_tag
MyNtdStrVar_Init;
```

TargetLink 2022-B

```
extern MyNtdStrType_Typedefs
MyNtdStrVar_Init;
```

Reason Matches user expectations

Array data type definitions

Prior TargetLink 2022-B, the base types were used in the type definitions for multi-dimensional array types. As of TargetLink 2022-B, the types that are specified via the DD Typedef object are used. This effects dependencies between type definitions and can lead to a different sorting of the type definitions.

TargetLink ≤ 5.2

```
typedef int16_t MyArrayM_2x3[2][3];
typedef int16_t MyArrayV_3[3];
typedef int16_t MyInt16;

typedef struct MyStrType_NoTypedefs_a_tag {
   int16_t aa;
   int16_t ab[3];
} MyStrType_NoTypedefs_a;

typedef struct MyStrType_Typedefs_a_tag {
   MyInt16 aa;
   MyArrayV_3 ab;
} MyStrType_Typedefs_a;

typedef struct MyStrType_NoTypedefs_tag {
   MyStrType_NoTypedefs_a a;
   int16_t b[2][3];
} MyStrType_NoTypedefs;

typedef struct MyStrType_Typedefs_tag {
   MyStrType_Typedefs_a a;
   MyArrayM_2x3 b;
} MyStrType_Typedefs;
```

TargetLink 2022-B

```
using MyInt16 = std::int16_t;

struct MyStrType_NoTypedefs_a {
   int16_t aa;
   int16_t ab[3];
};
using MyArrayV_3 =
ara::core::Array<MyInt16, 3>;

struct MyStrType_NoTypedefs {
   MyStrType_NoTypedefs_a a;
   int16_t b[2][3];
};
using MyArrayM_2x3 =
ara::core::Array<MyArrayV_3, 2>;

struct MyStrType_Typedefs_a {
   MyInt16 aa;
   MyArrayV_3 ab;
};

struct MyStrType_Typedefs {
   MyStrType_Typedefs_a a;
   MyArrayM_2x3 b;
};
```

In addition, in C array data type definitions the user type is used instead of the base type. This means that for matrix C array data types that use a vector data type whose **CreateTypedef** property is set to **on**, the vector data type is used instead of the base type of the scalar chain member to define the matrix data type.

TargetLink ≤ 5.2	TargetLink 2022-B

`typedef int16_t MyInt16;`
`typedef int16_t MyArrayV_3[3];`
`typedef int16_t MyArrayM_2x3[2][3];`
	`using MyInt16 = int16_t;`
	`using MyArrayV_3 = MyInt16[3];`
	`using MyArrayM_2x3 = MyArrayV_3[2];`

Reason Matches user expectations

Classic-AUTOSAR-Related

Changes with RTE API functions

With TargetLink 2022-B, struct assignments take precedence over other optimizations. As a side effect, this can result in code differences that are not related to structs. The following might be observed regarding RTE API functions:

- RTE API functions are placed closer to the code location where their return is required. This is especially visible for RTE API functions that have a pointer return.

Reason

- Increased consistency
- Code stability with future TargetLink versions

Migration issue Refer to Migration Aspects Regarding Optimization on page 189.

Limited optimization of simulation code

Simulation code that is generated from the following model elements is not fully optimized:

- Synchronous operation call subsystem
- Asynchronous operation call subsystem
- Operation result provider subsystem

Only the following optimizations are performed:

- Generation of struct assignments
- Scope reduction of variables
- Replacement of several variables by a common auxiliary variable

Improved optimization of scalar actual parameters

The optimization was improved in contexts, where all of the following conditions apply:

- **v** is a scalar variable.
- The address of **v** is used as an actual parameter in a function call.

In prior versions, such actual parameters could be replaced only by the addresses of other scalar variables.

With TargetLink 2022-B, such actual parameters can now also be replaced by pointers or the addresses of vector elements or matrix elements.

Consider the following examples:

TargetLink ≤ 5.2

```
Int16 DE_Scalar_a;
Rte_Read_ReceiverPort_DE_Scalar(&DE_Scalar_a); // Write access
SSecond1_Second((sint32) (-(DE_Scalar_a * 2)));
Sa2_Assignment[2] = DE_Scalar_a;
```

```
Aux_S32 = *Arg_Scalar;
Rte_Call_new_ClientPort_Scalar_ARGINOUT(instance, &Aux_S32);
*Arg_Scalar = Aux_S32;
```

TargetLink 2022-B

```
Rte_Read_ReceiverPort_DE_Scalar(&(Sa2_Assignment[2]));
SSecond1_Second((sint32) (-(Sa2_Assignment[2]* 2)));?
```

```
Rte_Call_new_ClientPort_Scalar_ARGINOUT(instance, Arg_Scalar);
```

Reason Code efficiency

Server runnables

The code generated for server runnables could previously contain unneeded pointers to matrices. This is now fixed:

TargetLink ≤ 5.2

```
FUNC(void, Swc_ForOpCall_CODE) RunnableForOpCall(P2CONST(MyScalar, AUTOMATIC, RTE_APPL_DATA)
  Matrix_In)
{
   /* SLLocalConst: Default storage class for local const variables | Width: N.A. */
   const sint32 (*p_Matrix_In)[3];

   p_Matrix_In = (const sint32 (*)[3]) Matrix_In;
}
```

TargetLink 2022-B

```
FUNC(void, Swc_ForOpCall_CODE) RunnableForOpCall(P2CONST(MyScalar, AUTOMATIC, RTE_APPL_DATA)
  Matrix_In)
{
}
```

Reason Bug fix

Name macro $ (SwComponentSymbolicName) instead of $ (Component)

With TargetLink 2022-B, the name macro `$(SwComponentSymbolicName)` is the default name macro when generating RTE API calls.

The name macro `$(Component)` is replaced by `$(SwComponentSymbolicName)` in situations as described in $(Component) macro replacement on page 184.

Depending on the AUTOSAR version, the name macro `$(SwComponentSymbolicName)` expands as follows:

AUTOSAR Version	Expands To
< 21-11	Expands to the name of the DD SoftwareComponent object (similar to `$(Component)`).
>= 21-11	▪ Expands to the value of the DD SymbolicName property of the DD SoftwareComponent object. ▪ If the DD SymbolicName property is not set, it expands to the name of the DD SoftwareComponent object (similar to `$(Component)`).

Example The name of the DD SoftwareComponent object is `Controller`.

The AUTOSAR version is < 21-11 or the AUTOSAR version is >= 21-11 but the SymbolicName property is not set:

```
Rte_Controller_ProvidedSignal_upi;
Rte_ActivationVector_Controller_PosController = activation;
```

The AUTOSAR version is >= 21-11 and the SymbolicName property is set to ControllerSymbol:

```
Rte_ControllerSymbol_ProvidedSignal_upi;
Rte_ActivationVector_ControllerSymbol_PosController = activation;
```

Reason AUTOSAR compliance

Utilization of known alias relationships

When modeling communication as described by Classic AUTOSAR, different pointers to the same buffer can be involved. This can implicitly result in contexts as described in Signal line splits on page 206.

In such contexts, the optimization is improved, when all of the following conditions apply:

- The removal of an intermediate variable results in the propagation of an access into a loop.
- Due to an alias, where pointers are potentially pointing to the same variable, the generated code actually is an `a=a` expression.

Consider the following example:

TargetLink 5.2

```
for (Aux_S32 = 0; Aux_S32 < 5; Aux_S32++) {
    Sa4_Assignment1[Aux_S32] = p_DataElement3_5[Aux_S32];
}

Sa4_Assignment1[3] = DataElement_In1_3;

for (Aux_S32 = 0; Aux_S32 < 5; Aux_S32++) {
    p_DataElement3_5_a[Aux_S32] = Sa4_Assignment1[Aux_S32];
}
```

TargetLink 2022-B

```
for (Aux_S32 = 0; Aux_S32 < 5; Aux_S32++)
{
    p_DataElement3_5_a[Aux_S32] = p_DataElement3_5[Aux_S32];
}
p_DataElement3_5_a[3] = DataElement_In1_3;
```

Here, `Sa4_Assignment1` will be removed, even though `p_DataElement3_5` and `p_DataElement3_5_a` alias each other.

This results in the following assignment:

```
p_DataElement3_5_a[Aux_S32] = p_DataElement3_5[Aux_S32];
```

This assignment will not be removed because the pointer variables are not the same. TargetLink makes no assumptions about them always pointing to the same variable to avoid potential problems, e.g., buffer code that does not comply with AUTOSAR.

Reason Code efficiency

Related code changes Signal line splits on page 206

Simulink-Function-Subsystem-Related

Implementation of unenhanced Inport blocks with preceding Argin block

The implementation of Simulink Inport blocks has changed in situations where all of the following conditions apply:

- The Simulink Inport block is connected to an preceding ArgIn block of a Simulink Function subsystem.
- The variable class of the ArgIn block is `default`.

TargetLink ≤ 5.2	TargetLink 2022-B
TargetLink implemented the variable of the Simulink Inport block independently of the block variable of the ArgInblock. The variable was derived from the `SlFunctionInput` or `SlFunctionInputStruct` internal default templates.	If TargetLink implements the block variable of the ArgIn block as a global variable, it attempts to use the same variable as the global interface variable of the Simulink Inport blocks.

Reason Matches user expectations

Removal of unused local struct

The code has changed for ArgIn blocks of Simulink function subsystems if all of the following conditions apply:

- The ArgIn block references a pointer-to-struct-transport variable.
- The NameTemplate property of the pointer-to-struct-transport variable does not specify the name of the actual parameter.
- The Simulink Function subsystem is called by a Function Caller block at least twice.

TargetLink ≤ 5.2	TargetLink 2022-B
TargetLink erroneously creates a struct variable with local scope that is updated before the function call of the Simulink Function subsystem but this struct variable is never read.	TargetLink no longer generates a struct variable with local scope but updates the pointer-to-struct-transport variable that is used as actual parameter in each function call.

Reason Bug fix

Related topics

References

ArgIn Block (TargetLink Model Element Reference 🕮)

Other

Bitwise expressions

Stateflow/MATLAB code TargetLink now adds additional casts in bitwise expressions from Stateflow or MATLAB code, if a variable in the expression is replaced by a numeric constant during optimization. In these cases, the numeric constant is cast to the data type of the replaced variable. Consider `Int16Var & 3 > 0` as an example:

TargetLink ≤ 5.2	TargetLink 2022-B
`5 & 3 > 0`	`((Int16) 5) & 3 > 0`

This ensures that the type the bitwise operation is calculated in does not change.

While this cast might be removed by further optimizations, it can have side effects. For example, that the sign changes in the calculation of a folded operation.

Non-Stateflow/MATLAB code The following changes might be observed:

- Operations that are an operand of a bitwise operation are not folded if they are created in the `CConvention` mode of Stateflow, because in this case the Simulink types and the TargetLink types differ.
- Integer casts of floating-point constants are not replaced by an integer constant, if the cast is the operand of a bitwise operation.

Reason Bug fix

#include directives

The following changes can take place regarding `#include` directives:

- Unnecessary `#include` directives are removed.
- `#include` directives are moved from a header file (H) to the corresponding source file (C).

This is especially visible for includes of the following header files: `tl_basetypes`, `udt_<SubsystemID>.h`, and `tl_defines_<SubsystemID>.h`.

As a benefit, this improvement can yield to the removal of include cycles that were accidentally specified by the user.

Reason Reduction of compile dependencies

Naming of auxiliary variables

Due to improvements of the Code Generator, the unoptimized code might contain a smaller number of auxiliary variables. This can result in changes of the names of the auxiliary variables that remain in the unoptimized code. Consider the following example:

TargetLink ≤ 5.2	TargetLink 2022-B
`Aux_, Aux__b, Aux__e, Aux__f`	`Aux_, Aux__a, Aux__b, Aux__c`

Reason Side effect

Additional pointer-to-struct-transport parameters

For pointer-to-struct-transport parameters, the following change might be observed under rare circumstances:

TargetLink ≤ 5.2	TargetLink 2022-B
Actual parameters have global scope	Actual parameters are function local. As a consequence, more function calls have pointer-to-struct-transport parameters. This means that the propagation of the pointer-to-struct-transport variable starts in a function higher in the function call hierarchy.

Reason Increased consistency

Related code changes Scope-Reduction-Related on page 203

Additional log macros

With TargetLink 2022-B, the TargetLink block property `logdata.loggingmode` is mapped to the `DataLogging` parameter of Simulink `port` or Stateflow `data` objects.

This can result in additional log macros in the generated production code. Refer to Direct mapping of the TargetLink logging property to the Simulink data logging on page 190.

Different implementation of 1-D and 2-D Prelookup blocks

Simulink and TargetLink block properties are now synchronized The production code that is generated for 1-D and 2-D Prelookup blocks might change if all of the following conditions apply:

Property	Value
ImplementLookUpTableFunctionsWithPointerArithmetics Code Generator option	on
Use last breakpoint for input at or above upper limit checkbox of the Prelookup block	`off`[1)]

1) Can also result from the new default values of a DD Look-up table object that is referenced at the Prelookup block.

The implementation of the look-up table function changed as follows:

TargetLink ≤ 2022-B	TargetLink 2022-B
`TabIdxS`	`TabIdx2S`

Migration issue To revert to the old implementation of the look-up table function, set the Use last breakpoint for input at or above upper limit checkbox of the Prelookup block to on. Also refer to Prelookup Block and Interpolation Using Prelookup Block on page 187.

Reason

- Matches user expectations
- Increased consistency

New table function implementation to fix critically false code The production code that is generated for 1-D and 2-D Prelookup blocks might change if all of the following conditions apply:

Property	Value
ImplementLookUpTableFunctionsWithPointerArithmetics Code Generator option	`off`
Use last breakpoint for input at or above upper limit checkbox of the Prelookup block	on

The implementation of the look-up table function changed as follows:

TargetLink ≤ 2022-B	TargetLink 2022-B
`TabIdx2S` The last breakpoint of the table was never reached, which resulted in critically false code.	`TabIdx3S` The last breakpoint of the table is reached.

Reason Bug fix

Additional auxiliary variable for calls of FIR-filter functions

For reasons of stability, calls of FIR-filter functions now contain an additional auxiliary variable as actual parameter, if the following conditions apply:

- The array that holds the coefficients has the `const` qualifier.
- The array is passed to the FIR-filter function.

This happens, because the formal parameter of the FIR-filter function currently is not qualified as `const`.

TargetLink ≤ 5.2

```
Int32 Sa1_FIR_Filter1_coeff_Aux[5];
 /* FIR Filter: Subsystem/FIR Filter1 */
 *Sa1_FIR_Filter1
= (Int32) (AF__I32TCFIR32_SAT_I32I32(Sa1_FIR_Filter1_coeff_Aux
,
  (UInt32 *) 3, Sa1_FIR_Filter1_coeff_Aux, &p_DelayLine2) >> 1);
```

TargetLink 2022-B

```
Int32 Sa1_FIR_Filter1_coeff_Aux[5];
for (Aux_ = 0; Aux_ < 5; Aux_++)   {
      Sa1_FIR_Filter1_coeff_Aux[Aux_] =
Sa1_FIR_Filter1_coeff[Aux_];
}

 /* FIR Filter: Subsystem/FIR Filter1 */
 *Sa1_FIR_Filter1
= (Int32) (AF__I32TCFIR32_SAT_I32I32(TlAuxVar, (UInt32
*) 3,
  Sa1_FIR_Filter1_coeff_Aux, &p_DelayLine2) >> 1);
```

Reason Code stability

To be changed in a future TargetLink version.

Automatic RDI destination mapping

The automatic mapping of replaceable data items (RDI) was changed to guard against cyclic includes. This can result in different headers.

Reason Code stability

Related topics

References

ImplementLookUpTableFunctionsWithPointerArithmetics (TargetLink Model Element Reference 🕮)
Prelookup Block (TargetLink Model Element Reference 🕮)

Discontinuations as of TargetLink 2022-B

Where to go from here

Information in this section

Discontinued TargetLink Features

Discontinuation of SWC container exchange

The support of exchanging SWC containers between TargetLink and SystemDesk is discontinued with TargetLink 2022-B. However, you can still exchange software components using ARXML files. With dSPACE AUTOSAR Compare, you can compare and merge these files. Refer to Best Practices for Exchanging Software Components (TargetLink Interoperation and Exchange Guide 🕮).

The discontinued software parts include:

- In SystemDesk, the **Container File Explorer** and the commands for preparing and exchanging SWC containers.
- In TargetLink, the Data Dictionary commands for exchanging SWC containers.
- The Container Manager, which is a stand-alone tool for managing the contents of SWC containers. It is no longer available from TargetLink or SystemDesk.

Obsolete Limitations

No support of array-of-struct variables at the interface of incremental code generation units	TargetLink does not support array-of-struct variables at the interface of incremental code generation units.
Array-of-struct datatypes as parameters of RTE-API functions	Aside from RTE-API functions that belong to client-server communication, TargetLink does not support array-of-struct datatypes as function parameters of RTE-API functions.
CombineControlFlowStatements may interfere with formation of struct assignments	The CombineControlFlowStatements Code Generator option can interfere with the formation of struct assignments from leaf struct component assignments. This occurs if the Code Generator option merges loops with certain conditions, such as read or write accesses that cannot be rearranged into continuous struct assignments. In these cases, the effect of the Code Generator option on code size and run time can be reduced, non-existent, or negative.
The use of temporalCount in Stateflow	The use of the `temporalCount` function which returns the number of events occurred or the number of seconds elapsed since the activation of the state is not supported.
No data logging for referenced subsystems in library blocks	TargetLink does not support signal logging for instances of referenced subsystems if the instance is used as part of a Library block.
Logging restriction	No Stateflow Local object is logged during MIL simulation if the simulation was started by either clicking the Simulink Run button or by using the Simulink `sim` command, instead of the TargetLink Start simulation button or `tl_sim` command.
Logging of signals in Enabled, If Action, or Switch Action subsystems	You cannot log a signal history in MIL simulation mode if the signal meets the following conditions: ▪ The signal resides in either the Enabled, If Action, or Switch Action subsystem. ▪ The plot interval is different from Inf. Instead, the minimum and maximum values of the signal are logged.

Signal logging in referenced subsystems with Fast Restart	If a model contains an instance of a referenced subsystem and the Fast Restart mode is active, TargetLink does not support the following for signals in the model : ▪ Signal logging ▪ Min/Max logging ▪ Overflow detection
Missing SVG images	If you use Internet Explorer, the generated HTML documentation does not display images of SVG format if the following two conditions are fulfilled at the same time: ▪ The generated documentation is stored on a network drive ▪ The Internet Explorer's compatibility view is enabled. Refer to the Tools - Compatibility View settings - Display Intranet sites in Compatibility View option.
No support of referencing array-of-struct types at blocks	To use an array-of-struct variable at blocks, you must reference the DD Variable object. Referencing the DD Typedef object is not supported.
Restrictions for the Scope property of DD Variable objects	TargetLink does not support the value `ref_param` of the Scope property for the VariableClass object that is referenced at an array-of-struct variable.
Simulink's Run button	If you use Simulink's Run button on the model toolbar to start a simulation in MIL mode, signals in the referenced models are not logged. To ensure that signals inside referenced models are logged, use the `tl_sim` API function, or the Start simulation button of one of the following TargetLink windows: ▪ TargetLink Main Dialog ▪ TargetLink Plot Overview Window ▪ TargetLink Detailed Plot Window ▪ TargetLink Signal Comparison Window
Interpolation Using Prelookup block	TargetLink always requires the Valid index input may reach last index Simulink block property to be enabled (refer to the Main Simulink block dialog page). Otherwise, MIL and SIL/PIL simulations differ. No error message is displayed if this property is disabled.
Prelookup block	TargetLink always requires the Use last breakpoint for input at or above upper limitValid index input may reach last index Simulink block property to be enabled (refer to the Main Simulink block dialog page). Otherwise, MIL and SIL/PIL simulations differ. No error message is displayed if this property is disabled.

Related topics

References

TargetLink Main Dialog Block (TargetLink Model Element Reference 🕮)

Obsolete API Functions

Obsolete TargetLink MATLAB API Functions

Function	Status	Replacement
`tl_export_container`	Error[1)]	-

1) The function was removed from TargetLink.

Obsolete Hook Scripts

List of obsolete hook scripts

Hook Script	Description
`tl_pre_containerexport_hook`	The hook script was removed from TargetLink.

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.

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 in a future TargetLink version.

SWC ReceiverPort and SWC SenderPort blocks (TargetLink AUTOSAR Blockset)

Support of `SWC ReceiverPort` and `SWC SenderPort` blocks will be discontinued in a future TargetLink version. Instead, `InPort` and `OutPort` blocks can be used.

Related topics

References

InPort Block (TargetLink Model Element Reference 🕮)
OutPort Block (TargetLink Model Element Reference 🕮)
SWC ReceiverPort Block (TargetLink Model Element Reference 🕮)
SWC SenderPort Block (TargetLink Model Element Reference 🕮)

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 2022-B

Support of the FMI 3.0 release

VEOS 2022-B now also supports FMUs implemented according to the FMI 3.0 release.

With regard to FMI 3.0, VEOS supports the same functionalities as for FMI 2.0. Furthermore, it supports the following concepts and features introduced with FMI 3.0:

- Support for all newly-introduced data types except for the *clock* data type.
  This includes the *binary* data type for handling nonnumeric data, such as sensor data.
- Support for array variables.

For more information on the FMI 3.0 release, refer to https://fmi-standard.org/news/2022/05/10/fmi-3.0-release.html.

For more information on the VEOS support for the FMI 3.0 release, refer to Basics on Integrating Functional Mock-Up Units (FMUs) (VEOS Integrating the Simulation System 🕮).

Model Container Utility support

VEOS is supported by the new *Model Container Utility*, which complements the dSPACE SIL testing tool chain.

For more information on the Model Container Utility, refer to Basics on the Model Container Utility (Model Container Utility Manual 🕮).

Co-simulation with VEOS

dSPACE provides the VeosCoSim interface for implementing co-simulations with VEOS. Co-simulation with VEOS is implemented in a client-server architecture, where the VEOS Simulator is the co-simulation master, i.e., it provides the overall simulation time for all co-simulation participants.

The VeosCoSim interface is not part of the VEOS installation. To obtain the interface and the related user documentation, refer to https://www.dspace.com/go/VEOS-Utilities.

SDK for integrating C code and libraries into a V-ECU (V-ECU SDK)

dSPACE provides the new V-ECU SDK, which brings SIL testing to your series development process. The SDK provides command line tools that enable you to easily integrate C code and libraries written with your favorite IDE into a V-ECU. The V-ECU can then be simulated and debugged with VEOS.

This lets you develop, prototype, and test your ECU software in open-loop and closed-loop scenarios with environment models or other V-ECUs.

The V-ECU SDK utility is not part of the VEOS installation. To obtain the utility, refer to https://www.dspace.com/go/VEOS-Utilities.

Building V-ECUs and SICs on Linux (VEOS Build Console)

You can now use the VEOS Build Console *on Linux* to build classic and adaptive V-ECU implementation container (VECU) and Simulink implementation container (SIC) files for VEOS.

For more information, refer to VEOS Build Console Command Reference (VEOS VEOS User Interfaces Reference 🕮).

ISO 26262 certification in the context of the SIMPHERA workflow

SIMPHERA uses VEOS for code-based offline simulation in cloud infrastructures. dSPACE now has achieved certification of SIMPHERA in accordance with ISO 26262.

ISO 26262 and ASIL ISO 26262 is an international standard for safety-relevant electrics/electronics (E/E) systems in motor vehicles. It is applied wherever electronic functions and software that have an impact on functional safety are used in the vehicle.

The automotive safety integration level (ASIL) specifies the level of risk reduction that is required to prevent hazards. The ASIL determines the steps you have to take during the development of safety-critical E/E systems according to ISO 26262, such as the use of qualified tools.

Qualification of SIMPHERA SIMPHERA is qualified for the use in ASIL projects up to ASIL D. The qualification of SIMPHERA is certified by TÜV SÜD. As a result, vehicle manufacturers and suppliers can exclude SIMPHERA from the

ISO 26262 qualification of their entire processes and can concentrate on proving the functional safety of their own process chains.

SIMPHERA and VEOS SIMPHERA uses:

- The VEOS System Builder to compile simulation applications including virtual ECUs and environment models.
- The SIMPHERA execution framework initiates the VEOS Simulator to run code-based offline simulations in a cloud infrastructure.

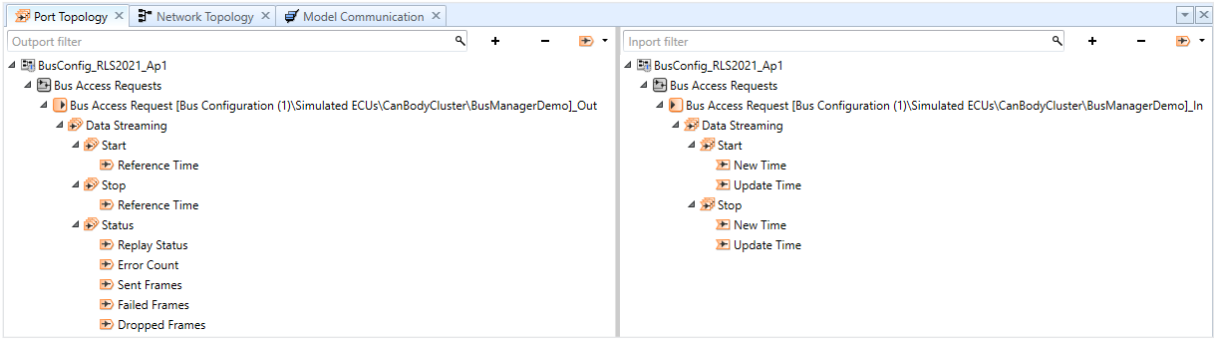
Status signals for CAN/Ethernet data replay

For each bus access request for which data replay is enabled, the Port Topology pane provides a structure of incoming and outgoing signals.

As of VEOS 2022-B, the following outgoing status signals are also available:

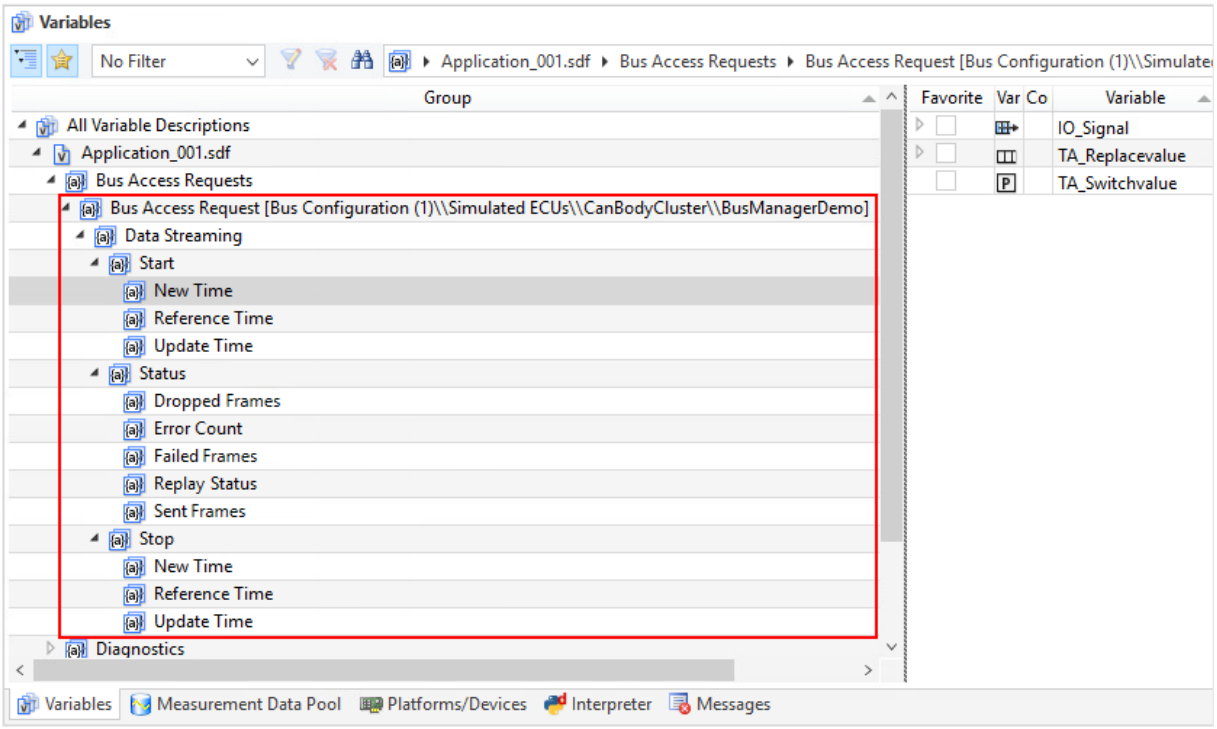
- Replay Status
- Error Count
- Sent Frames
- Failed Frames
- Dropped Frames

The following illustration shows an example structure:



Variables that correspond to the outgoing status signals are also generated into the SDF variable description file that you can use in ControlDesk.

The following illustration shows the ControlDesk **Variables** pane with the related variables.



For more information, refer to Replaying Bus Data with VEOS (VEOS Manual 🕮).

Specifying MSVC-related installation paths

The **Simulation Target Manager** now lets you specify the compiler and Windows SDK installation path also for simulation targets based on the MSVC compiler.

This is helpful when VEOS cannot detect the compiler installation, for example, when the MSVC compiler is installed on a network drive.

For more information, refer to Simulation Target Manager (VEOS VEOS User Interfaces Reference 🕮).

Support for MSVC compiler Version 14.2x

VEOS now supports Version 14.2x of the Microsoft Visual C/C++ Compiler for the HostPC32 and HostPC64 simulation targets. Version 14.2x is provided by Microsoft Visual Studio 2019.

Support for Version 14.1x provided by Microsoft Visual Studio 2017 is continued.

For more information, refer to Simulation Target Manager (VEOS VEOS User Interfaces Reference 🕮).

Compatibility of VEOS 2022-B

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 Integrating the Simulation System 🕮).

Note

When you build model containers *in separate steps*, e.g., when you compile an existing model container in advance with the dSPACE Model Container Utility, dSPACE recommends you to use *compatible versions of the compiler and linker*. This ensures binary compatibility of the build result.
For more information, refer to Ensuring Binary Compatibility When Building Model Containers (VEOS Manual 🕮).

Supported operating systems

Windows For information on the Windows operating systems supported by VEOS, refer to Operating System on page 236.

Linux dSPACE recommends using Ubuntu Linux 20.04 LTS. If you want to use a different Linux distribution, contact dSPACE Support.

Virtualizing VEOS For information on virtualizing VEOS for cloud/cluster environments, refer to Virtualization of VEOS (VEOS Manual 🕮).

BSC file compatibility

VEOS 2022-B is compatible with bus simulation container (BSC) files created with the Bus Manager of dSPACE Release 2022-B (BSC version 1.11).

- If a BSC file was generated *with an SIC file*, the target architecture selected for SIC file generation specifies the simulation targets available to the VEOS build process.
  Refer to Basics on Simulink Implementation Containers (Model Interface Package for Simulink - Modeling Guide 🕮).
- If a BSC file was generated *without an SIC file*, you can select any simulation target supported by the Simulation Target Manager.

FMU file compatibility

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

- FMI 2.0
- FMI 3.0
  For more information on the FMI 3.0 release, refer to https://fmi-standard.org/news/2022/05/10/fmi-3.0-release.html.

For detailed and up-to-date compatibility information on dSPACE FMI support, refer to:

http://www.dspace.com/go/FMI-Compatibility.

OSA file compatibility

The following table shows the compatibility between VEOS 2022-B and offline simulation application (OSA) files:

OSA Files Created with Products Of ...	OSA Version
dSPACE Release 2022-B	22.2
dSPACE Release 2022-A	5.4[1)]
dSPACE Release 2021-B	5.3[1)]
dSPACE Release 2021-A	5.2[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, you can edit port and network connections, and enable/disable experimentation services. It is recommended that you rebuild the binary OSA files from existing model implementation container files (BSC, FMU, SIC, VECU) when you migrate from one VEOS version to another.

SIC file compatibility

The following table shows the compatibility between VEOS 2022-B and Simulink implementation container (SIC) files:

SIC Files Created With ...	SIC Version[1)]
dSPACE Release 2022-B: ▪ Model Interface Package for Simulink 2022-B ▪ TargetLink 2022-B	1.11
dSPACE Release 2022-A: ▪ Model Interface Package for Simulink 22.1 ▪ TargetLink 5.2	1.10[2)]
dSPACE Release 2021-B: ▪ Model Interface Package for Simulink 4.6 ▪ TargetLink 5.2	1.10[2)]
dSPACE Release 2021-A: ▪ Model Interface Package for Simulink 4.5 ▪ TargetLink 5.1	1.10[2)]

1) The target architecture selected for SIC file generation specifies the simulation targets available to the VEOS build process. Refer to Basics on Simulink Implementation Containers (Model Interface Package for Simulink - Modeling Guide 🕮).

2) 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 2022-B (dSPACE Release 2022-B). For more information, refer to Migrating ASM Models (VEOS New Features and Migration 🕮).

SMC file compatibility

The following table shows the compatibility between VEOS 2022-B and system model container (SMC) files:

SMC Files Created With ...	SMC Version
dSPACE Release 2022-B: ▪ VEOS 2022-B	1.2
dSPACE Release 2022-A: ▪ SYNECT 2.13 ▪ VEOS 5.4	1.2
dSPACE Release 2021-B: ▪ SYNECT 2.12 ▪ VEOS 5.3	1.2
dSPACE Release 2021-A: ▪ SYNECT 2.11 ▪ VEOS 5.2	1.2

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 2022-B imports neither the unsupported container nor the connections to the application process based on the unsupported container.

VECU file compatibility

The following table shows the compatibility between VEOS 2022-B and V-ECU implementation container (VECU) files:

V-ECU Implementations Created With...	V-ECU Implementation Version
dSPACE Release 2022-B: ▪ SystemDesk 2022-B ▪ TargetLink 2022-B	3.2
dSPACE Release 2022-A and 2021-B: ▪ SystemDesk 5.6 ▪ TargetLink 5.2	3.1
dSPACE Release 2021-A and 2020-B: ▪ SystemDesk 5.5 ▪ TargetLink 5.1	3.0

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
... of VEOS 2022-B	Real-Time Testing Version 2022-B
... of VEOS 5.4	Real-Time Testing Version 5.2
... of VEOS 5.3	Real-Time Testing Version 5.1
... of VEOS 5.2	Real-Time Testing Version 5.0

VEOS Simulator	RTT Version
... of VEOS 5.1	Real-Time Testing Version 4.4
... of VEOS 5.0	Real-Time Testing Version 4.3

ControlDesk 2022-B automatically uses the VEOS Simulator of VEOS 2022-B. You can therefore use RTT in connection with VEOS and ControlDesk if Real-Time Testing Version 2022-B is active on the PC.

AUTOSAR Adaptive Platform compatibility

For the simulation of AUTOSAR adaptive V-ECUs, VEOS 2022-B is compatible with the following release of the AUTOSAR Adaptive Platform:

- R20-11

Related topics

Basics

Ensuring Binary Compatibility When Building Model Containers (VEOS Manual 🕮)
Hypervisor Configuration for Adaptive V-ECU Simulation on Windows (VEOS Working with Adaptive V-ECUs 🕮)

Migrating to VEOS 2022-B

Introduction

To migrate from VEOS 5.4 to VEOS 2022-B, you might have to perform certain migration steps.

Note

To migrate to VEOS 2022-B from versions earlier than 5.4, 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 Basics on Integrating Simulink Implementation Containers (SICs) (VEOS Integrating the Simulation System 🕮).

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 🕮).

Discontinuations as of VEOS 2023-A

Simulating with VEOS 4.0 ... 4.4

When VEOS 2023-A or later is installed, performing offline simulations on the same PC will no longer be possible with the following VEOS versions:

- VEOS 4.0 from dSPACE Release 2017-A
- VEOS 4.1 from dSPACE Release 2017-B
- VEOS 4.2 from dSPACE Release 2018-A
- VEOS 4.3 from dSPACE Release 2018-B
- VEOS 4.4 from dSPACE Release 2019-A

Building and configuring an offline simulation application with these versions, however, is not affected and remains possible.

Compatibility Information

Where to go from here

Information in this section

Supported MATLAB Releases

MATLAB®/Simulink®

Various dSPACE products require a MATLAB installation on the host PC.

Tip

For system requirements of MathWorks® software, refer to https://www.mathworks.com/support/sysreq.html.

MATLAB Release...	...Is Supported by dSPACE Release 2022-B					
	RCP and HIL 2022-B [1), 2)]	AutomationDesk 2022-B[3)]	TargetLink 2022-B	Model Compare 2022-B	dSPACE Python Extensions 2022-B [4)]	XIL API .NET MAPort 2022-B
R2022b	✓ [5)]	✓	✓	✓	✓	✓
R2022a	✓ [5)]	✓	✓	✓	✓	✓
R2021b	✓	✓	✓	✓	✓	✓
R2021a	✓	✓	✓	✓	✓	✓

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 is not supported by Automotive Simulation Models (ASM).

3) The AutomationDesk MATLAB Access Library requires MATLAB.

4) matlablib2 of dSPACE Python Extensions requires MATLAB.

5) R2022a and R2022b are not supported by the FPGA Programming Blockset – FPGA Interface.

For up-to-date information on additional MATLAB releases that can be used in combination with dSPACE software, refer to https://www.dspace.com/go/MATLABCompatibility.

Operating System

Windows operating system on host PC

The dSPACE products of dSPACE Release 2022-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 and Windows 10 S editions are not supported.
  - Long-Term Servicing Channel: LTSC 2019
  - Long-Term Servicing Channel: LTSC 2021
  - General Availability Channel (replaces the previous Semi-Annual Channel): 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 21H2 version of the General Availability Channel for testing.

- The following editions, channels, and servicing options of Windows 11:
  - Windows 11 Professional, Education, and Enterprise (64-bit versions)

    The Windows 11 Home edition is not supported.

    Note

    Valid for the FPGA Programming Blockset: Only the FPGA Programming Blockset - Processor Interface supports Windows 11. Windows 11 cannot be used to model or handcode FPGA applications due to missing Xilinx support.

  - General Availability Channel: 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 21H2 version of the General Availability Channel for testing.
- Windows Server LTSC 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.

Support of Windows Docker You can also run some dSPACE products in a Windows Docker container. For more details, contact dSPACE Support (https://www.dspace.com/go/supportrequest).

Linux operating system on host PC

The dSPACE products of dSPACE Release 2022-B with Linux compatibility support the following operating system:

- Ubuntu 20.04 LTS with the General Availability Kernel in the Desktop, Server, and Cloud version each based on the AMD 64-bit processor architecture.
- Ubuntu 18.04 LTS is no longer supported.

Some limitations apply when you use dSPACE software in conjunction with features of Linux. Refer to Limitations for Using Linux Features on page 246.

Support of Linux Docker You can also run the dSPACE products with Linux compatibility in a Linux Docker container. For more details, contact dSPACE Support (https://www.dspace.com/go/supportrequest).

Using MicroAutoBox Embedded PC as host PC

ControlDesk can also be installed on:

- 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 2022-B.
- Windows Server 2016

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. Any PC with CodeMeter Runtime software can be used as a license server.

Valid for servers without dSPACE software You can use operating systems from Microsoft Windows on a license server in combination with protected dSPACE software.

Note

- If you want to use Ubuntu Linux as operating system for the license server, contact dSPACE Support (www.dspace.com/go/supportrequest).
- Other operating systems 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>\dSPACE MotionDesk 2022-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:
  - For Windows: 111 (TCP and UDP), 3702 (UDP), 7214 (TCP and TCP6), 7215 (TCP and UDP), 7216 (TCP), 8090 (TCP), 9923 (UDP), 49152 ... 65535 (TCP, TCP6 and UDP)
  - For Linux: 111 (TCP and UDP), 7215 (TCP and UDP), 7216 (TCP), 8090 (TCP), 9923 (UDP), 32768 ... 60999 (TCP, TCP6 and UDP)
- MotionDesk requires the following open TCP/IP network port: 15000 (UDP)

- ConfigurationDesk used together with one or more connected MATLAB installations requires up to 10 open TCP/IP network ports in the range 49196 ... 49205 (TCP):
  - Whenever a connected MATLAB is started, the first available port from the specified range is taken.
  - When ConfigurationDesk is started, it also uses the first free port from the specified range.
  - If ConfigurationDesk is not running or no MATLAB instance is running, then no ports from the specified range are used.
- 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 by the following prefixes: `HelpAbsLayer<xxx>`, `HelpInstaller<xxx>`.

Using dSPACE Software on Virtual Machines (VMs)

Introduction

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.

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	✓	—	—
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	✓	—	—
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.	—

Product	Full Support	Support with Known Limitations	No Support
		▪ 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.	—
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 Hypervisor Configuration for Adaptive V-ECU Simulation on Windows (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 using a physical PC if high performance is required by an application.

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 the 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 licenses, 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 from the Windows Start menu (**Start – Control Panel – Appearance and Personalization – Display – Adjust ClearType text**).

Using the 'Revert to snapshot' feature

NOTICE

Using the 'Revert to snapshot' feature causes licenses to become invalid.

If you use the 'Revert to snapshot' feature in a VM, all software-based CmContainers on your host PC (dSPACE Activation Container and/or dSPACE Borrow Container) become invalid and the contained licenses are lost.

- Do not use the 'Revert to snapshot' feature for VMs that contain software-based CmContainers with activated licenses.
- Store the license information on CmDongles. There, the CmContainers do not become invalid after use the 'Revert to snapshot' feature.

Moving the virtual machine to a host PC with a different hardware configuration

NOTICE

Moving the virtual machine to a PC with a different hardware configuration causes licenses to become invalid.

If you move your virtual machine to a host PC with a different hardware configuration, all software-based CmContainers on your host PC (dSPACE Activation Container and/or dSPACE Borrow Container) become invalid and the contained licenses are lost. This happens, for example, if the CPU type of the physical PC changes.

- Do not move a virtual machine that contains software-based CmContainers with activated licenses to a host PC with a different hardware configuration.
- Store the license information on CmDongles. There, the CmContainers do not become invalid after the virtual machine is moved.
- However, if moving the virtual machine is absolutely necessary, contact dSPACE Support (www.dspace.com/go/supportrequest) beforehand to find solutions that can avoid major downtimes.

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 2022-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 hereafter.

  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 https://www.dspace.com/go/CompPatch.
- RCP and HIL software products (of Release 2022-B) cannot be used in combination with RCP and HIL software products from earlier dSPACE Releases.

Recommendation for working with a SCALEXIO system or with MicroAutoBox III The products for working with a SCALEXIO system or

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 https://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 that you close the dSPACE software manually before shutting down the PC.

User Account Control

It is recommended that you 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 high-DPI monitors

The following dSPACE software products have limitations for working with high-DPI monitors:

- SYNECT: If you use high-DPI 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 high-DPI monitors.

High-contrast mode not supported

dSPACE software was not developed for or tested against the Windows high-contrast mode. You are recommended to disable the high-contrast mode when working with dSPACE software. Otherwise, the visualization of the user interfaces might be impaired.

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://docs.microsoft.com/en-us/windows/security/threat-protection/fips-140-validation.

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 https://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.
Limitation for using single-user licenses	If you use a PC with a Windows Server operating system, for example, Windows Server 2019, and you want to work with dSPACE software on this PC using Microsoft Remote Desktop Connection, you have to use a floating network license to activate license-protected software. Single-user licenses are not supported in this case.

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: ▪ Ubuntu 20.04 LTS: https://ubuntu.com/blog/fips-certification-ubuntu-20-04-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.

S

T

V

W