ConfigurationDesk

User Interface Reference

For ConfigurationDesk 6.7

Release 2021-A – May 2021

How to Contact dSPACE

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

How to Contact dSPACE Support

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

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

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

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

Software Updates and Patches

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

Important Notice

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

Contents

Hardware Management 63

External Device Management 105

Function Block Management 143

Model Interface Management 165

Signal Chain Configuration 205

Working View Management 255

Executable Application and Task Modeling 277

Resolving Conflicts 319

Build and Download Management 321

Bus Manager Handling 343

About This Reference

Content

ConfigurationDesk is a graphical user interface that allows you to implement real-time applications intuitively and effectively. This reference provides information on ribbon commands, context menu commands, dialogs and windows of ConfigurationDesk. It also contains descriptions for the file formats relevant when working with ConfigurationDesk.

Required knowledge

Knowledge in handling the host PC and the Microsoft Windows operating system is presupposed.

Symbols

dSPACE user documentation uses the following symbols:

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

Naming conventions

dSPACE user documentation uses the following naming conventions:

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

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

Special folders

Some software products use the following special folders:

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

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

or

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

Documents folder A standard folder for user-specific documents.

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

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

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

Accessing dSPACE Help and PDF Files

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

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

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

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

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

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

Safety Precautions

Where to go from here

Information in this section

General Warning

Danger potential

Using dSPACE software can be dangerous. You must observe the following safety instructions and the relevant instructions in the user documentation.

⚠ **WARNING**

Improper or negligent use can result in serious personal injury and/or property damage.
Using the ConfigurationDesk software can have a direct effect on dSPACE systems and technical (electrical, hydraulic, mechanical) systems connected to them.

- **Only persons who are qualified to use dSPACE software, and who have been informed of the above dangers and possible consequences, are permitted to use it.**
- All applications where malfunctions or misoperation involve the danger of injury or death must be examined for potential hazards by the user, who must if necessary take additional measures for protection (for example, an emergency off switch).

Liability

It is your responsibility to adhere to instructions and warnings. Any unskilled operation or other improper use of this product in violation of the respective safety instructions, warnings, or other instructions contained in the user documentation constitutes contributory negligence, which may lead to a limitation of liability by dSPACE GmbH, its representatives, agents and regional dSPACE companies, to the point of total exclusion, as the case may be. Any exclusion or limitation of liability according to other applicable regulations, individual agreements, and applicable general terms and conditions remain unaffected.

Data loss during operating system shutdown

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

Safety Precautions for Configuration and Mapping

Objective

To avoid risk of injury and/or damage to the hardware and to achieve safe and trouble-free operation, you have to observe the following rules.

Assigning device ports to device pins

You assign the device ports to device pins of your external device connector with the device pin assignment. The device pin assignment is required to calculate the external wiring information.

NOTICE

If the device pin assignment is incorrect the wiring information that is calculated (for the external harness) will not match to the electrical requirements of the external devices (for example your ECU).

Risk of damage to the connected devices.

Ensure that the device pin assignment matches the pin configuration of the device.

Mapping device ports to signal ports

The basis for the mapping rules between device ports and signal ports is the configuration of the device ports. The setting of the port type (inport, outport, bidirectional port, reference port, undefined port) plays a major role.

NOTICE

If the device port configuration is incorrect, and/or the device port mapping is incorrect, the wiring information that is calculated (for the external harness) will not match to the electrical requirements of the external devices (for example your ECU).

Risk of damage to the connected devices and/or the dSPACE real-time hardware.

- Configure the device ports and particularly the port type so that they correspond as closely as possible to the characteristics of your device. Use the undefined port type only as an exception.
- Check the mapping lines. Ensure, that the mapping will not result in damage to the connected device because the electrical characteristics of the mapped ports do not match.

Simulating failures for multiple signals simultaneously

Using the Failure Simulation module of your experiment software, you can simulate failures for multiple signals simultaneously. For example, you can simulate a situation where a short to GND for one signal and an open circuit for a different signal happen at the same time.

As a precondition in ConfigurationDesk, the Activation by FRU relay property must be set to Allowed for all signals which you want to use in a multiple failure scenario.

In multiple failure scenarios, the SCALEXIO system uses the electro-mechanical relays of the failure routing unit (FRU relays) to activate the failure directly by switching the relays under load instead of just switching the signal to the failplane.

NOTICE

Risk of increased wear and permanent damage

When you use the “Activation by FRU relay” feature, there is a risk of increased wear and permanent damage to the relays of the dSPACE failure simulation hardware. For details, refer to Safety Precautions for Simulating Electrical Errors with a SCALEXIO System (SCALEXIO – Hardware and Software Overview 🕮).

Load compare check during download

If you download a real-time application to your hardware, ConfigurationDesk checks if the channel's internal load description which was used during the build process matches the internal load description stored on the hardware. If there is a deviation, then the application is possibly not compatible with the hardware system. ConfigurationDesk generates a warning message and you can abort the download.

You cannot disable this check.

NOTICE

Ignoring the warning message can cause material damage.

Risk of damage to the connected external device.

- Cancel the download of the real-time application if you are not sure about the consequences.
- Disable the affected channels of the internal loads if you do not need them.
- Ensure that any rejection of the affected internal loads cannot result in damage to the connected external device.

  If the two internal load descriptions conflict, load rejection of the affected internal load will always be enabled.

Project and Application Management

Objective

ConfigurationDesk's project and application management enables you to structure your project tasks in ConfigurationDesk.

Where to go from here

Information in this section

Information in other sections

Managing ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide)

Activate (Application)

Purpose

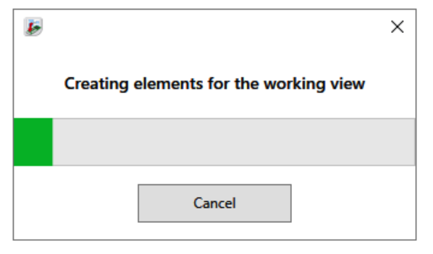
To activate a ConfigurationDesk application.

Result

The selected ConfigurationDesk application is activated and its latest file data is loaded.

Tip

If a large working view with many signal chain elements is open in the activated application, a progress bar is displayed while the working view is prepared.



Creating elements for the working view

Click **Cancel** to stop the opening process. The working view is not opened. You can open it later from the **Working View Manager**.

If dSPACE real-time hardware is displayed in the **Platform Manager**, different scenarios can occur regarding the state of your application. For details, refer to Basics on Connecting a ConfigurationDesk Application to a Hardware System (ConfigurationDesk Real-Time Implementation Guide).

Description

Only one application can be active at a time within a ConfigurationDesk project. All the other applications are inactive. To work with an application, it must be active.

Related topics

Basics

Handling ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 📖)

HowTos

How to Activate a ConfigurationDesk Application (ConfigurationDesk Real-Time Implementation Guide 📖)

Add Folder

Purpose

To add a folder to the selected project or application.

Description

ConfigurationDesk opens the Create New Folder dialog.

Create New Folder dialog

Name of the new folder Lets you enter a name for the new folder.

Location of the new folder Lets you specify where the new folder is stored.

OK Creates the new folder and opens the Configure Folder Settings dialog which lets you specify project-specific folder settings. Refer to Configure Folders / Configure (Folder) on page 33.

Related topics

Basics

Handling ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 📖)

Backup Project

Purpose

To save the loaded project and all applications in a ZIP archive.

Result

Opens a standard Save As dialog that lets you provide the name and location for the ZIP archive.

Description

ZIP archives contain only files that are stored within the project tree. The files are archived with relative paths. They can thus be extracted anywhere.

To load a project from backup, you can use the Open Project and Application from Backup command.

> **Note**
>
> Folders and their contents (documents and further subfolders) are included in a ZIP archive only if they are displayed in the **Project Manager**. To include subfolders, you have to enable the **Show subfolders** option in the **Configure Folder Settings** dialog. To include documents, you have to specify a comma-separated list with file name extensions in the **Documents to be shown in this folder** edit field. Refer to Configure Folders / Configure (Folder) on page 33.

Related topics

Basics

Handling ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 📖)

HowTos

How to Back up and Transfer a Project (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Configure Folders / Configure (Folder).. 33
Open Project and Application from Backup.. 52

Calculate / Calculate External Wiring

Purpose

To calculate the wiring information for the external cable harness.

Result

ConfigurationDesk calculates the wiring information. Afterwards you can export the wiring information to an XLSX or ECHX file. The XLSX file contains information necessary for building the external cable harness. The ECHX file can be used to import the wiring information to a different ConfigurationDesk project.

Description

It is useful to calculate the wiring information for the external cable harness after you have performed the following steps:

- Assigning device pins to device ports.
- Adding devices to the signal chain.
- Mapping device blocks to function blocks.
- Assigning hardware resources to function blocks.

ConfigurationDesk calculates the connection between device pins (for example of your ECU) and the I/O connector pins of the dSPACE hardware. Bridges are also determined, for example, in the case of channel multiplication caused by current enhancement.

Using an existing cable harness You can use the representation of an existing cable harness in your ConfigurationDesk application to avoid building a new cable harness.

> **Note**
>
> During creation of the signal chain, wiring conflicts can occur, which are displayed in ConfigurationDesk. Recalculating the wiring information can solve these conflicts. However, if this is done, the external cable harness changes. If you want to use an existing cable harness, you must not use this command. You have to solve wiring conflicts at other positions in the signal chain.

Related topics

Basics

Calculating an External Cable Harness (ConfigurationDesk Real-Time Implementation Guide)

Clear (Topology)

Purpose

To delete all elements from topologies or the calculated external cable harness.

Result

ConfigurationDesk deletes all elements from the topology or calculated external cable harness. If elements are used in the active ConfigurationDesk application, they become unresolved. This is shown as follows: . To resolve elements again, you can use the following commands:

- Make Resolved on page 114 (for device topology elements).
- New – Missing xxxx (Rack/Unit/Board) on page 70 (for hardware topology elements).

- Calculate / Calculate External Wiring on page 30 (for the external cable harness).

Related topics

Basics

Basics on Components of ConfigurationDesk Applications (ConfigurationDesk Real-Time Implementation Guide)

Clear Build Results

Purpose

To remove existing build result files from the active ConfigurationDesk application.

Result

You are prompted to confirm the removal of the build results. After clicking **OK**, the build result files are removed from the **Build Results** folder of the currently active application in the **Project Manager** and in your file system.

Description

Only files that are actual results of a ConfigurationDesk build process are removed. For details on build result files, refer to Build Result Files of the Build Process (ConfigurationDesk Real-Time Implementation Guide).

Related topics

Basics

Build Result Files of the Build Process (ConfigurationDesk Real-Time Implementation Guide)
Building Real-Time Applications (ConfigurationDesk Real-Time Implementation Guide)

Clear Model Communication Description File

Purpose

To delete the model communication description file (MCD file) reference from the model topology of your active ConfigurationDesk application.

Result

The reference to the MCD file is deleted. Now you can use the models in your ConfigurationDesk application regardless of any changes to the model communication modeled in the behavior model. If you want to reuse the information in the MCD file you have to re-add the MCD file to the model topology via the Import / Replace Topology (Model Topology) on page 46 command.

Related topics

Basics

Setting Up Model Communication (ConfigurationDesk Real-Time Implementation Guide)

Close Project / Close

Purpose

To close the currently loaded project and all applications belonging to it.

Description

If the current project was changed since the last save operation, you are asked whether you want to save it.

Related topics

Basics

Handling ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide)

Configure Folders / Configure (Folder)

Purpose

To configure project-specific folder settings, for example, which folders and which documents to display in each folder in the Project Manager.

Result

Opens the Configure Folder Settings dialog.

Description

The Configure Folder Settings dialog enables you to specify settings for a specific folder in the current project.

Configure Folder Settings dialog

Lets you specify project-specific folder settings.

A tree with the folders of the project or application is displayed on the left side. The selected folder is highlighted. You can change the selection in the tree.

Folders listed belong to The project or application name.

Documents to be shown in this folder Lets you specify a comma-separated list with file name extensions, for example: .docx,.txt,.pdf.

Only documents with one of the specified extensions are displayed in the Project Manager, others are hidden.

Show sub folders Lets you specify whether subfolders and their contents (documents and further subfolders) of a specific folder are shown in the Project Manager. The documents to be shown in a subfolder are taken from the parent folder where you enable the Show sub folders option.

Note

> Folders and the files they contain are included in a project backup (via ConfigurationDesk's Backup Project on page 29 command) only if the folders are displayed and the files are shown in the Project Manager.

Related topics

Basics

Handling ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 🕮)

Create Application

Purpose

To create a new ConfigurationDesk application in the current project.

Result

A new ConfigurationDesk application with the defined Application name is added to the currently open project and activated.

Application

Application name Enter the desired application name. The name you enter must not contain a dot or whitespace as the first or last character. By default, the application is named ’Application_nnn’, where ’nnn’ stands for a number that will be increased if the name already exists as file name in the directory of the project.

Models

Application process options Lets you specify the assignment of the selected model implementations to the created application processes. You can select one of the following options:

- Create one preconfigured application process for each model (default).

  If you select this option, ConfigurationDesk creates a separate application process for each model implementation that is to be added to the active ConfigurationDesk application.
- Create one application process for all models and optimize configuration.

  If you select this option, ConfigurationDesk creates one application process and assigns all the selected model implementations to it. ConfigurationDesk optimizes the configuration of the new application process by grouping runnable functions in tasks, and specifying the task priorities. The default name of the new application process is **ApplicationProcess_<n>**. n is an increasing number.
- Create no application process.

  If you select this option, ConfigurationDesk adds the selected model implementations to the ConfigurationDesk application without creating application processes.

If the executable application does not contain a processing unit application, the preconfigured application process is created in a new processing unit application. If the executable application contains exactly one processing unit application, the preconfigured application process is created in the existing processing unit application.

Note

For the **Create one preconfigured application process for each model** option and the **Create one application process for all models and optimize configuration** option the following applies:

- For Simulink models, these options are available only if you select the **Analyze model (including task information)** checkbox. If the model analysis fails, the preconfigured application process is not created.
- These option are not available if the executable application contains more than one processing unit application.

Add model Lets you select a model implementation or a model topology that you want to add to the active ConfigurationDesk application. If you already opened behavior models in Simulink, you can select them directly from the **Models open in Simulink** list. You can also click **Add model from file**. If you do, a standard **Open** dialog opens for you to select the file type and the file you want to import. You can select one of the following file types:

- Simulink model (`*.slx`;`*.mdl`)

  Tip

  - It can be useful to save the referenced Simulink model file directly to a project folder that is configured to display MDL/SLX files. Refer to Add Folder on page 29. You can use the folder to save data belonging to your project, such as model files necessary for implementing your real-time application.
  - You find a **Models** folder with an appropriate configuration in the project folder by default.

- Simulink implementation container (`*.sic`)
- Bus simulation container (`*.bsc`)
- V-ECU implementation container (`*.vecu`)
- Functional Mock-up Unit (`*.fmu`)
- (Not available in the **Add Model** dialog) Model communication description file (`*.mcd`)
- (Not available in the **Add Model** dialog) Model topology file (`*.mtfx`)

List of selected model(s) Lists the selected model(s) that you want to add to the ConfigurationDesk application. If you click **Create**, all the selected models are added to the ConfigurationDesk application at once.

The following options are visible if you select one or more models that you want to add to the ConfigurationDesk application. You can specify them separately for each selected model.

Analyze model (including task information) (Only available for MDL, SLX, MTFX, and MCD files) Lets you analyze the model interface of your behavior model. If you select this checkbox, a complete analysis of the selected model is performed, and the changes are made available in ConfigurationDesk. For more information, refer to Analyze Simulink Model (Including Task Information) on page 171.

Note

If the model analysis fails, no behavior model is added to the ConfigurationDesk application. A dialog displays error messages and refers to the MATLAB Command Window which provides more information on the errors that have occurred in the Simulink behavior models. While you correct the errors in the behavior models, the **Add Model** dialog remains open and keeps its configuration. Then, you can add the behavior models to the ConfigurationDesk application again without having to select them again.

Model initialization command (Only available for Simulink models and MCD files) Lets you enter a MATLAB command that is executed before the behavior model is opened by ConfigurationDesk, for example, to prepare data in the MATLAB Base Workspace that is referenced by the behavior model.

ConfigurationDesk lets you specify locations of scripts, data, etc. via relative paths in the **Model initialization command** edit field. This enables you to transfer the ConfigurationDesk project to another path. Refer to How to Back up and Transfer a Project (ConfigurationDesk Real-Time Implementation Guide 🕮). For this purpose, ConfigurationDesk provides the following path macros:

Path Macro Name	Description
`%ProjectRoot%`	Expands to the root directory of the current project.
`%ApplicationRoot%`	Expands to the root directory of the currently active ConfigurationDesk application.
`%ModelRoot%`	Expands to the directory the selected Simulink model is stored in.

Hardware / Import Hardware Topology dialog

Add hardware Lets you select a hardware platform or file to be used for creating a hardware topology. Registered platforms are available in the **Registered Hardware** list. A **Default Hardware** based on a platform or a file is also available if you saved one via Save as Default on page 70.

Add hardware from file Lets you specify that ConfigurationDesk imports topology information from a hardware topology file (HTFX file). An **Add Hardware File** dialog opens for you to select a HTFX file.

Usually, the hardware topology file was created for the same system and using ConfigurationDesk. However, it is also possible to import a hardware topology file that was created for another system.

Tip

- ConfigurationDesk offers a number of predefined hardware topologies that you can add to your ConfigurationDesk application. The topologies are stored in the following folder, which is opened by default:

  `<Documents folder>\PredefinedHardware`
- You can click the down arrow next to the added hardware to make the **Save as Default** command available. Using the command makes the hardware the default hardware for future hardware topology imports.

Register platforms Opens the Register Platforms on page 78 dialog where you can scan for and register a dSPACE real-time hardware platform. Afterwards, the platform is available in the **Registered Hardware** list.

Related topics

Basics

Managing ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 🕮)

Create ControlDesk Experiment

Purpose

To create a ControlDesk experiment from the currently active ConfigurationDesk application based on the build results.

Note

Currently this feature supports only ConfigurationDesk applications that contain a single application process, i.e., the SDF file references one variable description file. If the SDF file references multiple variable description files, e.g., in a multicore or multi-processing-unit application, the created project will be incomplete. For these applications we recommend that you still create the project and experiment in ControlDesk.

Result

Note

The following preconditions must be fulfilled for the command to work:

- The dSPACE experiment software ControlDesk must be installed on the same system.
- A ConfigurationDesk project must be open and one of its applications must be active.
- The active ConfigurationDesk application must contain build results.
- If you want the platform name in the ControlDesk experiment to match your registered platform, make sure that the platform is the only registered platform in ConfigurationDesk and that the application state is Matching platform connected.

A new ControlDesk project named after the ConfigurationDesk project is created in the ControlDesk project root directory and opened in ControlDesk.

An experiment named after the ConfigurationDesk application is created and activated in the ControlDesk project.

The experiment contains a layout with a Numeric Input instrument connected to the Task Turnaround Time variable of the Periodic Task 1 in the variable description file.

Note

- If a matching platform is registered and connected to the ConfigurationDesk application, the platform in the ControlDesk experiment is named after it. Otherwise, a default name is used.
- If a ControlDesk project with the name of the ConfigurationDesk project already exists, it is opened and the experiment is added to it.
- If an existing ControlDesk project with the same name already contains an experiment with the name of the ConfigurationDesk application, it is not overwritten. However, missing information such as the new name of a matching platform is added.

Related topics

HowTos

How to Create a ControlDesk Experiment from a ConfigurationDesk Application (ConfigurationDesk Real-Time Implementation Guide)

Create New Project and Application

Purpose

To create a new ConfigurationDesk project and application.

Result

- A new ConfigurationDesk project with the defined **Project name** is created in the selected root folder.
- A ConfigurationDesk application with the defined **Application name** is added to it.
- The new project opens. The new ConfigurationDesk application is active.

Project

Root directory Select a project root folder from the list.

...	Opens the **Project** page of the **ConfigurationDesk Options** dialog. You can specify further project root folders on this page. Refer to Project Page on page 485.

Note

To avoid errors, adhere to the following rules:

- Do not specify a ConfigurationDesk project folder as a project root folder.
- Do not specify a project root folder in a project root folder.

Project name Enter the desired project name in the edit field. The name you enter must not contain a dot or whitespace as the first or last character. By default, the project is named 'Project_nnn', where 'nnn' stands for a number that will be increased if the name already exists as a file name in the project root folder.

Application

You can immediately add a ConfigurationDesk application and, if desired, some components of it to the new project. Refer to Create Application on page 34.

> **Tip**
>
> To create a project without an application, clear the Application name field.

Related topics

Basics

Managing ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 🕮)

Create Shortcut

Purpose

To create a desktop shortcut to a ConfigurationDesk application.

Result

A shortcut to the ConfigurationDesk application is created on your desktop.

Description

You can create a desktop shortcut for each ConfigurationDesk application. This allows you to open ConfigurationDesk and quickly load a specific application.

Related topics

Basics

Managing ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 🕮)

Explore Folder

Purpose

To open the folder of the selected item in the File Explorer.

Result

An Explorer Window opens showing the folder of the selected item.

Related topics

Basics

Managing ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 🕮)

Export (Application) / Export Application

Purpose

To export a ConfigurationDesk application as a dSPACE archive.

Result

ConfigurationDesk opens a standard **Save As** dialog to save the selected ConfigurationDesk application and all the files belonging to it as one unit in a dSPACE archive. The archive has the file name extension DSA.

Related topics

HowTos

How to Export and Import a ConfigurationDesk Application (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Import Application.. 49

Export / Export Topology / Export Hardware – Topology / Export Devices

Purpose

To export topologies or the external cable harness for reuse in other ConfigurationDesk applications.

Result

ConfigurationDesk opens a standard **Export <Component>** dialog to export the data to a ConfigurationDesk-specific XML file format.

Description

The following formats are used for the respective components:

Component	File Format
Device Topology	DTFX
Hardware Topology	HTFX
Model Topology	MTFX
External Cable Harness	ECHX

Related topics

HowTos

How to Export Data of a Specific Application Component (ConfigurationDesk Real-Time Implementation Guide 🕮)

Export Configuration

Purpose

To export the configuration data of the currently active ConfigurationDesk application to an Excel™ file (XLSX file).

Result

ConfigurationDesk opens an Export Configuration dialog to export the data to an Excel™ file.

Export Configuration dialog

The Export Configuration dialog is a standard Save As dialog that lets you save the application data to an XLSX file.

Description

For a description of the content of the exported XLSX file, refer to Exporting Data for Documentation Purposes (ConfigurationDesk Real-Time Implementation Guide 🕮).

Related topics

Basics

Exporting Data for Documentation Purposes (ConfigurationDesk Real-Time Implementation Guide 🕮)

Export to XLSX

Purpose

To export the ConfigurationDesk application component (device topology or external cable harness) to an Excel™ file (XLSX file). This lets you use the exported data with other tools and documents. A device topology can also be imported via an Excel™ file (XLSX file).

Result

ConfigurationDesk opens an Export Device Topology or Export External Cable Harness dialog to export the data to an Excel™ file.

Description

Device topology ConfigurationDesk exports only device topology elements with device ports.

For more information on exporting a device topology, refer to How to Export a Device Topology to a Microsoft Excel Sheet (ConfigurationDesk Real-Time Implementation Guide).

External Cable Harness For more information on exporting the external cable harness, refer to How to Export Wiring Information for an External Cable Harness (ConfigurationDesk Real-Time Implementation Guide).

Related topics

HowTos

How to Export a Device Topology to a Microsoft Excel Sheet (ConfigurationDesk Real-Time Implementation Guide)
How to Export Wiring Information for an External Cable Harness (ConfigurationDesk Real-Time Implementation Guide)

Import (External Cable Harness)

Purpose

To import external cable harness information to the active ConfigurationDesk application.

Result

ConfigurationDesk opens an Open dialog for you to select an ECHX file.

Description

ConfigurationDesk imports the external cable harness information to the active ConfigurationDesk application.

> **Note**
>
> The imported cable harness information will be lost when you recalculate the external cable harness.

Related topics

Basics

Calculating an External Cable Harness (ConfigurationDesk Real-Time Implementation Guide)

HowTos

How to Import an External Cable Harness (ConfigurationDesk Real-Time Implementation Guide)

Import / Import Devices – Replace

Purpose

To import a device topology (stored in a DTFX or XLSX file) to the active ConfigurationDesk application.

Result

ConfigurationDesk opens an Open dialog for you to select a DTFX or XLSX file.

Description

After confirming the dialog by clicking OK, ConfigurationDesk imports the device topology information to the active ConfigurationDesk application. The device topology is also displayed hierarchically in the **External Device Browser**.

> **Note**
>
> Existing device topology elements are removed from the device topology. If they were used in the signal chain, they become unresolved, unless there are matching elements in the imported topology.

Related topics

HowTos

How to Import a Device Topology (ConfigurationDesk Real-Time Implementation Guide 📖)

Import / Import Hardware – Replace / Import Topology – Replace

Purpose

To import a hardware topology to the active ConfigurationDesk application.

Description

ConfigurationDesk opens an Import Hardware Topology dialog.

Hardware / Import Hardware Topology dialog

Add hardware Lets you select a hardware platform or file to be used for creating a hardware topology. Registered platforms are available in the Registered Hardware list. A Default Hardware based on a platform or a file is also available if you saved one via Save as Default on page 70.

Add hardware from file Lets you specify that ConfigurationDesk imports topology information from a hardware topology file (HTFX file). An Add Hardware File dialog opens for you to select a HTFX file.

Usually, the hardware topology file was created for the same system and using ConfigurationDesk. However, it is also possible to import a hardware topology file that was created for another system.

Tip

- ConfigurationDesk offers a number of predefined hardware topologies that you can add to your ConfigurationDesk application. The topologies are stored in the following folder, which is opened by default:

  ```
  <Documents folder>\PredefinedHardware
  ```
- You can click the down arrow next to the added hardware to make the Save as Default command available. Using the command makes the hardware the default hardware for future hardware topology imports.

Register platforms Opens the Register Platforms on page 78 dialog where you can scan for and register a dSPACE real-time hardware platform. Afterwards, the platform is available in the Registered Hardware list.

Result

ConfigurationDesk imports the hardware topology information to the active ConfigurationDesk application. The hardware topology is also displayed in a hierarchical structure in the Hardware Resource Browser.

When you import a hardware topology to the active application, ConfigurationDesk automatically performs an identity check: It checks whether the hardware topology of any hardware system displayed in the **Platform Manager** is identical to the hardware topology of your active application. For details, refer to Basics on Connecting a ConfigurationDesk Application to a Hardware System (ConfigurationDesk Real-Time Implementation Guide 🕮).

Note

Existing hardware topology elements that do not match the imported topology are removed. If removed topology elements were used for hardware resource assignments, they are still displayed in the topology and marked as unresolved.

Related topics

HowTos

How to Import a Hardware Topology (ConfigurationDesk Real-Time Implementation Guide 🕮)

Import / Replace Topology (Model Topology)

Purpose

To import a model topology to the active ConfigurationDesk application.

Description

ConfigurationDesk opens an **Import Model Topology** dialog.

Add Model/Import Model Topology dialog

Application process options Lets you specify the assignment of the selected model implementations to the created application processes. You can select one of the following options:

- Create one preconfigured application process for each model (default).

  If you select this option, ConfigurationDesk creates a separate application process for each model implementation that is to be added to the active ConfigurationDesk application.
- Create one application process for all models and optimize configuration.

  If you select this option, ConfigurationDesk creates one application process and assigns all the selected model implementations to it. ConfigurationDesk optimizes the configuration of the new application process by grouping runnable functions in tasks, and specifying the task priorities. The default name of the new application process is ApplicationProcess_<n>. n is an increasing number.

- Create no application process.

  If you select this option, ConfigurationDesk adds the selected model implementations to the ConfigurationDesk application without creating application processes.

If the executable application does not contain a processing unit application, the preconfigured application process is created in a new processing unit application. If the executable application contains exactly one processing unit application, the preconfigured application process is created in the existing processing unit application.

> **Note**
>
> For the **Create one preconfigured application process for each model** option and the **Create one application process for all models and optimize configuration** option the following applies:
>
> - For Simulink models, these options are available only if you select the **Analyze model (including task information)** checkbox. If the model analysis fails, the preconfigured application process is not created.
> - These option are not available if the executable application contains more than one processing unit application.

Add model Lets you select a model implementation or a model topology that you want to add to the active ConfigurationDesk application. If you already opened behavior models in Simulink, you can select them directly from the **Models open in Simulink** list. You can also click **Add model from file**. If you do, a standard **Open** dialog opens for you to select the file type and the file you want to import. You can select one of the following file types:

- Simulink model (`*.slx`;`*.mdl`)

  > **Tip**
  >
  > - It can be useful to save the referenced Simulink model file directly to a project folder that is configured to display MDL/SLX files. Refer to Add Folder on page 29. You can use the folder to save data belonging to your project, such as model files necessary for implementing your real-time application.
  > - You find a **Models** folder with an appropriate configuration in the project folder by default.

- Simulink implementation container (`*.sic`)
- Bus simulation container (`*.bsc`)
- V-ECU implementation container (`*.vecu`)
- Functional Mock-up Unit (`*.fmu`)
- (Not available in the **Add Model** dialog) Model communication description file (`*.mcd`)
- (Not available in the **Add Model** dialog) Model topology file (`*.mtfx`)

List of selected model(s) Lists the selected model(s) that you want to add to the ConfigurationDesk application. If you click OK, all the selected models are added to the ConfigurationDesk application at once.

The following options are visible if you select one or more models that you want to add to the ConfigurationDesk application. You can specify them separately for each selected model.

Analyze model (including task information) (Only available for MDL, SLX, MTFX, and MCD files) Lets you analyze the model interface of your behavior model. If you select this checkbox, a complete analysis of the selected model is performed, and the changes are made available in ConfigurationDesk. For more information, refer to Analyze Simulink Model (Including Task Information) on page 171.

> **Note**
>
> If the model analysis fails, no behavior model is added to the ConfigurationDesk application. A dialog displays error messages and refers to the MATLAB Command Window which provides more information on the errors that have occurred in the Simulink behavior models. While you correct the errors in the behavior models, the **Add Model** dialog remains open and keeps its configuration. Then, you can add the behavior models to the ConfigurationDesk application again without having to select them again.

Model initialization command (Only available for Simulink models and MCD files) Lets you enter a MATLAB command that is executed before the behavior model is opened by ConfigurationDesk, for example, to prepare data in the MATLAB Base Workspace that is referenced by the behavior model.

ConfigurationDesk lets you specify locations of scripts, data, etc. via relative paths in the **Model initialization command** edit field. This enables you to transfer the ConfigurationDesk project to another path. Refer to How to Back up and Transfer a Project (ConfigurationDesk Real-Time Implementation Guide). For this purpose, ConfigurationDesk provides the following path macros:

Path Macro Name	Description
`%ProjectRoot%`	Expands to the root directory of the current project.
`%ApplicationRoot%`	Expands to the root directory of the currently active ConfigurationDesk application.
`%ModelRoot%`	Expands to the directory the selected Simulink model is stored in.

Result

ConfigurationDesk imports the model topology information to the active ConfigurationDesk application. The model topology is also displayed in a hierarchical structure in the **Model Browser**.

Note

Existing model topology elements are removed from the model topology.

Related topics

HowTos

How to Import a Model Topology (ConfigurationDesk Real-Time Implementation Guide 🕮)

Import Application

Purpose

To import a ConfigurationDesk application saved as a dSPACE archive.

Result

A standard Open dialog is displayed to import the selected ConfigurationDesk application. It is extracted below the selected project path. The imported files are added to the project.

Related topics

HowTos

How to Export and Import a ConfigurationDesk Application (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Export (Application) / Export Application... 41

Manage Projects

Purpose

To manage projects.

Result

Opens the Manage Projects dialog for you to manage existing projects.

Manage Projects dialog

To select a project to be managed.

Root directory Lets you select the project root folder.

📂	Opens a **Browse for Folder** dialog to select a project folder. The selected folder is added to the list of root folders if necessary.

Projects Lets you browse in the list of projects available in the selected project root folder.

You can only manage projects that are closed.

Delete Lets you remove the currently selected project from the file system, including all files in the project folders. You have to click **Yes** in the confirmation prompt before the project is removed.

Manage Opens the **Project Page** of the **ConfigurationDesk Options** dialog, where you can add or remove root folders. For more information, refer to Project Page on page 485.

Close Lets you close the dialog.

Related topics

Basics

Managing ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 🕮)

New (Backstage View)

Purpose

To provide access to commands that allow you to define a new project and/or application.

Description

You have access to the following commands:

- If no project is open: Create New Project and Application on page 39
- If a project is open: Create Application on page 34

New Application

Purpose

To open the **New** ribbon group on the **File** ribbon, where you can add a new ConfigurationDesk application to the current project.

Result

The **New** ribbon group on the **File** ribbon opens. Here, you can access the Create Application command.

Related topics

HowTos

How to Add a ConfigurationDesk Application to a Project (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Create Application.. 34

New Project + Application

Purpose

To open the **New** ribbon group on the **File** ribbon, where you can create a new ConfigurationDesk project.

Result

The **New** ribbon group on the **File** ribbon opens. Here, you can access the Create New Project and Application command if no ConfigurationDesk project is currently open.

Related topics

Basics

Create New Project and Application.. 39

HowTos

How to Create a Project (ConfigurationDesk Real-Time Implementation Guide 🕮)

Open (Backstage View)

Purpose

To provide access to commands that allow you, for example, to open a project.

Description

You have access to the following commands:

- Open Project/Open Project and Application on page 53
- Open Project and Application from Backup on page 52
- Import Application on page 49
- Open File or Project on page 470
- Manage Projects on page 49

Open (External Document)

Purpose

To open an external document in the associated application.

Description

External documents are, for example, PDF or XLSX files. This command opens the selected document in the corresponding program, for example, in Adobe Acrobat Reader™ or Microsoft Excel™.

Open Project and Application from Backup

Purpose

To open a project and application(s) from a backup file.

Result

Opens a dialog that lets you load a project and application(s) from a backup file.

If dSPACE real-time hardware is displayed in the **Platform Manager**, different scenarios can occur regarding the state of your application. For details, refer to Basics on Connecting a ConfigurationDesk Application to a Hardware System (ConfigurationDesk Real-Time Implementation Guide).

Open dialog

Look in Lets you specify the directory containing the backup file to be opened.

File name Lets you specify the name of the backup file.

Files of type Lets you specify the file type of the backup file.

The project will be extracted to this root directory Lets you specify the root folder to which the project and application(s) are extracted. The project will be loaded from this root folder.

Overwrite existing external files Allows ConfigurationDesk to overwrite files outside the project without a confirmation prompt.

> **Note**
>
> If there are references to files outside the project, ConfigurationDesk always asks you to confirm overwriting, even if **Overwrite existing files** is activated. This prevents you from unintentionally overwriting a newer version of a file with an older one and from overwriting files that may be used in other projects.

Related topics

HowTos

How to Back up and Transfer a Project (ConfigurationDesk Real-Time Implementation Guide)

Open Project/Open Project and Application

Purpose

To open a project.

Result

Opens the **Select a Project** dialog that allows you to select an existing project to open.

You can also select an application of the project that you want to be active after opening the project.

> **Tip**
>
> On the **File** ribbon, you can select a project and application from a list of recently opened projects and applications in the **Recently Used** ribbon group.

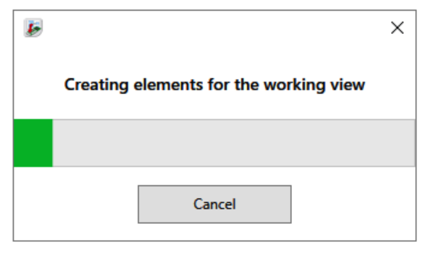
If dSPACE real-time hardware is displayed in the **Platform Manager**, different scenarios can occur regarding the state of your application. For details, refer to Basics on Connecting a ConfigurationDesk Application to a Hardware System (ConfigurationDesk Real-Time Implementation Guide).

Description

If another project is already open, it is closed. Then the project you selected is opened. A previously active application of a project is activated. If you selected an application, it is activated.

Tip

If a large working view with many signal chain elements is open in the activated application, a progress bar is displayed while the working view is prepared.



Creating elements for the working view

Click Cancel to stop the opening process. The working view is not opened. You can open it later from the Working View Manager.

Select a Project dialog

To select a project to be opened.

Root directory Lets you select the project root directory.

📂	Opens a Browse for Folder dialog to select a project folder. The selected folder is added to the list of root folders if necessary.

Projects Lets you browse in the list of projects and applications available in the selected project root folder.

Delete Lets you remove the currently selected project from the file system including all files in the project folders. You have to click Yes in the confirmation prompt before the project is removed.

Manage Opens the Project page of the ConfigurationDesk Options dialog where you can add or remove root directories. For more information refer to Project Page on page 485.

OK Opens the selected project.

Cancel Closes the Select a Project dialog.

Related topics

Basics

Managing ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 📖)

HowTos

How to Open a Project (ConfigurationDesk Real-Time Implementation Guide 📖)

Project Manager

Purpose

To display the **Project Manager** that lets you manage a project and the applications it contains.

Description

For more information on the **Project Manager**, refer to Handling ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 📖).

Related topics

Basics

Handling ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 📖)

Recently Used (Projects and Applications)

Purpose

To open one of the most recent projects and applications that were open in ConfigurationDesk.

Result

If another project ⍰ is already open, it is closed. Then the project and application you selected is opened.

Recent Projects and Applications

List of projects and applications Lets you select one of the most recent projects and applications that were open in ConfigurationDesk.

Open (Available from the context menu of list items) Opens the selected project and application. You can also simply open a project and application by left-clicking it in the list.

Remove from List (Available from the context menu of list items) To remove the selected project and application from the list of recently opened projects and applications.

Clear Recent Projects + Applications List (Available from the context menu of list items) To clear the list of recently opened projects and applications.

Reset Sort Direction You can click the column headers to sort the projects in ascending or descending order according to a column. To remove this sorting you can reset the sort direction to its default, which is according to the time the projects were last opened, starting with the most recently opened project.

Visible Columns - Path/Opened/Modified (Available from the context menu of the column header of the list of projects) Lets you specify whether to display:

- The project path
- The point in time when the project was last opened
- The point in time when the project was last modified

Size of recent projects list (Available at the bottom of the ribbon) Lets you specify the maximum number of list entries. You can specify a value in the range 4 ... 20.

Recent Files

List of files Lets you select one of the most recent files that were open in ConfigurationDesk, such as Python scripts. The list contains files that are not part of a project/application.

Size of recent files list Lets you specify the maximum number of list entries. You can specify a value in the range 4 ... 16.

Refresh (Application)

Purpose

To renew the folder structure of the active ConfigurationDesk application shown in the Project Manager.

Result

The folder structure of the active ConfigurationDesk application and the included files are re-read and the view of the structure shown in the **Project Manager** is refreshed. Newly added documents are shown.

Note

If you want to refresh the folder structure of the project and not of a specific application, you have to use the **Refresh (Project)** on page 57 context menu command of the project.

Related topics

Basics

Managing ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide)

Refresh (Project)

Purpose

To renew only the folder structure of a project (with the exception of application folder structures) shown in the Project Manager.

Result

The folder structure and included files of a project are re-read and the view of the structure shown in the **Project Manager** is refreshed. Newly added documents are shown.

> **Note**
>
> The folder structures of the ConfigurationDesk applications in a project are not refreshed. If you want to refresh these, you have to use the **Refresh (Application)** on page 56 context menu command of the active application.

Related topics

Basics

Managing ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide)

Reload Model Communication Description File

Purpose

To reload the model communication description file which is referenced in the model topology of your active ConfigurationDesk application.

Result

The MCD file is reloaded and the Model Topology and the Model Communication Package table are updated. Keep in mind that changes in the behavior model (related to the model separation) take effect only when a new MCD file is generated with the Model Separation Setup block in MATLAB/Simulink.

Related topics

Basics

Setting Up Model Communication (ConfigurationDesk Real-Time Implementation Guide)

Remove from Project / Remove

Purpose

To remove an inactive ConfigurationDesk application or a selected file or folder from the currently loaded project.

Result

Files or folders that are removed from the project tree are removed from the project irretrievably.

> **Tip**
>
> You cannot remove a whole project this way. You can delete a whole project via the **Manage Projects** dialog (refer to Manage Projects on page 49).

Related topics

Basics

Managing ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 🕮)

Rename

Purpose

To rename the selected item.

Result

A **Rename** dialog opens for you to rename the selected item. The name you enter for a Project or a ConfigurationDesk application can contain letters, digits, underscores, minus, parentheses, and dots. The name must not contain a dot or whitespace as the first or last character.

Description

If you select a project or an active application, ConfigurationDesk closes it before it is renamed. Afterwards, it is reloaded.

Rename dialog

To specify another name for the selected item.

Enter a new name for Lets you enter a new name for the selected item. If you rename an application, the name must be unique for each project and must not cause conflicts with existing folders.

Related topics

Basics

Managing ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 🕮)

Save As (Application)

Purpose

To save the selected ConfigurationDesk application and add it to the project with a new name.

Result

A **Save As** dialog opens that lets you save the selected ConfigurationDesk application under a new name.

Save As dialog

To specify another name for the selected ConfigurationDesk application.

Enter a new name for Lets you enter another application name. The last saved version of the ConfigurationDesk application is saved with the old name. The current, i.e., still unsaved version of the ConfigurationDesk application is saved with the new name and activated.

The application name must be unique within a project and must not cause conflicts with existing folders. The name you enter can contain letters, digits, underscores, minus, parentheses, and dots. The name must not contain a dot or whitespace as the first or last character.

Description

If you save an active ConfigurationDesk application under a new name, the new ConfigurationDesk application is activated.

When you save a ConfigurationDesk application, the expand/collapse states of hierarchies and blocks are preserved.

Related topics

Basics

Managing ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 🕮)

Save As (Backstage View)

Purpose

To provide access to commands that allow you, for example, to save a project under a new name.

Description

You have access to the following commands:

- Save As (Project) / Save Project As on page 60
- Backup Project on page 29
- Export (Application) / Export Application on page 41

Save As (Project) / Save Project As

Purpose

To save the currently opened project under a new name.

Result

A **Save Project As** dialog opens for you to save the project under a new name.

Description

The project must first be closed and reloaded. The new project is opened after you enter a new name and select a project root folder in the **Save Project As** dialog.

Save Project As dialog

Project name Lets you enter a project name. You must change or replace the original name.

Project root Lets you select a project root folder. You can click the 📂 icon to open the **Project Page** of the **ConfigurationDesk Options** dialog, where you can add additional project root folders. Refer to Project Page on page 485.

Related topics

Basics

Managing ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 📖)

Save Project / Save

Purpose

To save the loaded project and/or the active ConfigurationDesk application.

Result

The loaded project and/or the active ConfigurationDesk application are saved.

When you save a ConfigurationDesk application, the expand/collapse states of hierarchies and blocks are preserved.

Related topics

Basics

Managing ConfigurationDesk Projects and Applications (ConfigurationDesk Real-Time Implementation Guide 🕮)

Hardware Management

Objective

ConfigurationDesk lets you efficiently manage platform connections and create hardware topologies. Hardware topologies represent real-time hardware in ConfigurationDesk.

Hardware properties provide information on the system and the boards of a specific hardware topology.

Where to go from here

Information in this section

Information in other sections

Managing Real-Time Hardware (ConfigurationDesk Real-Time Implementation Guide 🕮)

Hardware Resource Browser

Objective

To be independent from connected real-time hardware or stored hardware topologies, you can create and extend a hardware topology in ConfigurationDesk.

Where to go from here

Information in this section

Switch to Network View...72
To display a network-based view of the hardware topology in the **Hardware Resource Browser**.

Clear Default

Purpose

To remove the default hardware for new ConfigurationDesk applications.

Result

The default hardware for hardware topologies ↗ of new ConfigurationDesk applications ↗ is removed from the **Replace Hardware Topology** dialog and the **New** backstage view.

Note

- After removing the default hardware, new ConfigurationDesk applications again have an empty hardware topology by default.
- The internally saved default hardware is deleted and no longer available.

Related topics

References

Import / Import Hardware – Replace / Import Topology – Replace...45
New (Backstage View)..50
Save as Default...70

Delete Assignment (Channel)

Purpose

To delete the assignment of selected channels to function blocks.

Result

The hardware resource assignment of the selected channels is deleted. The released channels can now be assigned to other function blocks.

Related topics

Basics

Assigning Hardware Resources to Function Blocks (ConfigurationDesk Real-Time Implementation Guide 🕮)

Export Hardware

Purpose

To access different commands that let you export hardware topology elements.

Result

Several commands for exporting hardware topology elements are available in a submenu.

Description

The following commands are available in the submenu:

- Export / Export Topology / Export Hardware – Topology / Export Devices on page 41
- Export Hardware – Topology Part on page 66

Export Hardware – Topology Part

Purpose

To export a selected system or rack and all its subelements from the hardware topology for reuse in other ConfigurationDesk applications.

Result

ConfigurationDesk opens the Export Hardware Topology Part dialog to export the data to an HTFX file. HTFX is the ConfigurationDesk-specific XML file format for hardware topologies.

Description

All hardware topology elements of the selected rack or system in the Hardware Resource Browser are exported to the HTFX file.

Related topics

HowTos

How to Export a Hardware Topology Part (ConfigurationDesk Real-Time Implementation Guide 🕮)

Hardware Resource Browser

Purpose

To display the hardware components of the hardware topology that is contained in your active ConfigurationDesk application.

Description

The **Hardware Resource Browser** displays all the hardware components of the hardware topology that is contained in your active ConfigurationDesk application in a hierarchical structure. Each hardware component provides properties via the **Properties Browser**, which lets you view hardware details and configure settings if a property is configurable.

Symbols

Hardware components of the SCALEXIO system or MicroAutoBox III system whose topology is stored in your active application are displayed with one of the following symbols:

Symbol	Description
▤	Indicates a SCALEXIO rack.
▢	Indicates a MicroAutoBox III system.
▭	Indicates a box or unit.
⧉	Indicates an angle unit set.
◁	Indicates an angle unit.
⧉	Indicates an Ethernet controller set.
⊞	Indicates an Ethernet controller.
▪	Indicates a processing unit.
🟩	Indicates a signal measurement board, a signal generation board, a processor board, a bus board, a bus module, or an I/O module.
ETH	Indicates an Ethernet board.
✣	Indicates an IOCNET link board.
⋰	Indicates a channel set (based on a specific channel type).
╲	Indicates a channel.
∞	Indicates that a channel set or a channel is used in the application.
!	Indicates that a channel set or a channel is missing because it has been deleted from the hardware topology.

Import Hardware

Purpose

To access different commands that let you import a hardware topology.

Result

Several commands for importing a hardware topology are available in a submenu.

Description

The following commands are available in the submenu:

- Import Hardware – Merge on page 68
- Import / Import Hardware – Replace / Import Topology – Replace on page 45

Import Hardware – Merge

Purpose

To merge a hardware topology, which is stored in an HTFX file, with the hardware topology of the active ConfigurationDesk application.

Result

ConfigurationDesk opens the Import Hardware Topology dialog for you to select an HTFX file.

Description

After you select an HTFX file and confirm the dialog by clicking Open, ConfigurationDesk merges the added hardware topology with the hardware topology of the active ConfigurationDesk application.

The added hardware topology elements are displayed in the **Hardware Resource Browser**.

Imported elements that already existed in the ConfigurationDesk application are updated.

Related topics

HowTos

How to Merge Hardware Topologies (ConfigurationDesk Real-Time Implementation Guide 🕮)

New – <Hardware Topology Elements>/Multiple <Hardware Topology Elements>

Purpose

To add different elements to the hardware topology.

Result

Depending on which area or element you right-clicked in the Hardware Resource Browser, different elements to add to the hardware topology are available in a submenu.

Description

You can create and extend hardware topologies to use ConfigurationDesk independently of real-time hardware or stored hardware topologies. You need real-time hardware only for downloading the real-time application.

ConfigurationDesk lets you only create a hardware topology (with components inside) which complies with the rules for building real hardware systems. To get an overview of the selectable hardware, refer to:

- SCALEXIO: Hardware for a SCALEXIO System (SCALEXIO Hardware Installation and Configuration 🕮)
- MicroAutoBox III: Hardware Components of the MicroAutoBox III and their Features (MicroAutoBox III Hardware Installation and Configuration 🕮)

You can add a single instance of a hardware topology element or use the according Multiple <hardware topology elements> command to open the Create Multiple <hardware topology elements> dialog and add multiple instances of the same element.

Create Multiple <hardware topology elements> dialog

Name pattern For SCALEXIO racks, MicroAutoBox III, units, or battery simulation controllers, you can define the name pattern. You can use the backslash \ as placeholder to add an incremented number to the name of each new SCALEXIO rack, MicroAutoBox III, unit, or battery simulation controller. This avoids duplicate names.

Note

The names of units must be unique.

Number of Instances Lets you define the number of elements to add to the hardware topology.

Related topics

HowTos

How to Create Hardware Topologies from Scratch (ConfigurationDesk Real-Time Implementation Guide 📖)
How to Extend Existing Hardware Topologies (ConfigurationDesk Real-Time Implementation Guide 📖)

New – Missing xxxx (Rack/Unit/Board)

Purpose

To restore a missing (unresolved) hardware topology element of the active ConfigurationDesk application.

Result

If you add the missing hardware resources to the hardware topology again, the original hardware resource assignment is restored. The ❗ symbol next to the missing element(s) disappears.

> **Note**
>
> Only missing (unresolved) subelements with hardware resources assigned to them are restored. Subelements that do not contain any assigned hardware resources, such as a processing unit without assigned angle units, are not restored.

Description

If you remove hardware topology elements containing channels (or angle units) that are assigned to function blocks, the elements and the channels are marked as missing (unresolved) in the hardware topology. ConfigurationDesk considers the elements and all their channels affected by hardware resource assignment to be missing hardware resources. Missing hardware resources are displayed with a warning symbol in the **Hardware Resource Browser**.

To restore a hardware topology using the **New – Missing xxxx** command, you must start at the highest missing element in the hierarchy (e.g., a missing SCALEXIO rack) before restoring the units or boards it contains.

Save as Default

Purpose

To save hardware as the default hardware for hardware topologies of new ConfigurationDesk applications.

Result

The hardware is saved as the default hardware that is used to preconfigure the hardware topology of new ConfigurationDesk applications.

Description

The default hardware is automatically added to the hardware topology when you create a new ConfigurationDesk application.

You can make hardware the default hardware for new ConfigurationDesk applications at several occasions:

- Whenever you add hardware while creating a new ConfigurationDesk application.
- Whenever you import a hardware topology to a ConfigurationDesk application.
- Anytime on the basis of the current hardware topology of an active ConfigurationDesk application.

This is possible for hardware added via HTFX file or via registered platforms.

> **Note**
>
> The default hardware is saved internally. Any later changes to the source HTFX file or platform are not updated in the default hardware.

Related topics

References

Clear Default..65
Import / Import Hardware – Replace / Import Topology – Replace...45
New (Backstage View)..50

Select Assigned Function

Purpose

To select and show function blocks to which specific hardware channels are assigned.

Result

ConfigurationDesk selects and shows you the assigned function blocks in the Signal Chain Browser in the Signal Chain view set.

Related topics

HowTos

How to Select and Show Assigned Elements (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Select Assigned Hardware Channel... 161

Switch to Assembly View

Purpose

To display a mechanical-based view of the hardware topology in the Hardware Resource Browser.

Result

The **Hardware Resource Browser** displays the physical assembly of the hardware topology of your active ConfigurationDesk application. The browser displays the hardware topology in a hierarchical structure, for example, with the SCALEXIO rack at the top. The SCALEXIO rack node serves as a container, for example, for I/O units.

Related topics

References

Switch to Network View.. 72

Switch to Network View

Purpose

To display a network-based view of the hardware topology in the Hardware Resource Browser.

Result

The **Hardware Resource Browser** displays the logical network (I/O network) of the hardware topology of your active ConfigurationDesk application, with the SCALEXIO processing units at the top.

If the hardware topology contains I/O units that are not connected to a processing unit, these I/O units are identified and displayed below the Unconnected Elements node. You can establish the required connections via the **Uplink** property of the affected units or via the **Downlink** property of the respective parent units.

Related topics

HowTos

How to Establish a Network Connection in the Hardware Topology (SCALEXIO) (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Switch to Assembly View.. 72

Platform Connection Management

Objective

ConfigurationDesk makes it easy to connect to platforms via Ethernet connection.

Where to go from here

Information in this section

Clear System

Purpose

To remove all the registered platforms from the Platform Manager and the Manage Recent Platform Configuration dialog.

Result

All the platforms are removed from the Platform Manager and the Manage Recent Platform Configuration dialog.

Recovering platforms

Platform information is stored in two files:

- `RecentHardware.xml`
- `RecentHardware_Backup.xml`

Both files are located in:

```
%CommonProgramFiles%\dSPACE\PlatformManagement
```

The `RecentHardware.xml` file is updated each time registered hardware is changed in ConfigurationDesk. Thus, it does not contain platform information after all the platforms were removed via Clear System on the Platforms ribbon or via Remove All in the Manage Recent Platform Configuration dialog.

The `RecentHardware_Backup.xml` file is updated only when a platform is registered via the **Register Platforms** dialog. Thus, it still contains registered platforms after all the platforms were removed. You can use the file to restore registered platforms by importing it via the **Manage Recent Platform Configuration** dialog.

Manage Platforms / Manage Recent Platform Configuration

Purpose

To display and manage the platforms that were registered in your system.

Result

ConfigurationDesk opens the Manage Recent Platform Configuration dialog, which lets you manage your recent platform configurations. You can remove elements from the recent platform configurations and hide registered platforms in the **Platform Manager**. You can import configurations for registered platforms from an XML file or export the recent hardware configuration to an XML file.

Description

When you register a SCALEXIO system or MicroAutoBox III system, ConfigurationDesk stores the registration data in the recent platform configuration.

Manage Recent Platform Configuration dialog

To manage the registered platforms and import/export the configuration of registered hardware.

Recent Platform Configuration Lists the platforms that were registered in your system and whose registration data is stored in the recent platform configuration, and displays several information on the registered platforms. The platform list also provides an **Active** switch for each platform. Via the switch you can specify whether the platform is hidden and not displayed in the **Platform Manager**. If the switch is set to 0, the platform is hidden. If the switch is set to 1, the platform is listed and displayed in the **Platform Manager** (if the platform is available).

Commands The following commands are available via buttons and from the menus or context menus:

Command	Access	Description
Activate	▪ Context menu of an inactive platform ▪ Shortcut key: **`Alt+A`**	Lets you activate the selected inactive platforms. An active platform is displayed in the **Platform Manager**.
Collapse	Context menu of a platform	Lets you collapse the member items of the platform selected in the platform list.

Command	Access	Description
Deactivate	▪ Context menu of an active platform ▪ Shortcut key: `Alt+D`	Lets you deactivate the selected platforms. An inactive platform is hidden. It is not displayed in the Platform Manager.
Expand	Context menu of a platform	Lets you expand the collapsed elements of the platform selected in the platform list.
Export	▪ Button ▪ File menu ▪ Shortcut key: `Alt+E`	Lets you select the XML file to which you want to export the recent platform configuration.
Group by Active State	View menu	Lets you group the platforms according to their Active state.
Group by Platform Type	View menu	Lets you group the platforms according to their platform type.
Import	▪ Button ▪ File menu ▪ Shortcut key: `Alt+I`	Lets you select the XML file containing the platform configuration you want to import. The currently active platform configuration is replaced by the content of the imported XML file. **Note** You are recommended to import only recent platform configurations that you previously exported from ConfigurationDesk.
Refresh	▪ View menu ▪ Context menu of a platform ▪ Shortcut key: `F5`	Lets you refresh the visualization of the recent platform configuration in the dialog.
Remove	▪ Button ▪ Edit menu ▪ Context menu of a platform ▪ Shortcut key: `Del`	Lets you remove the currently selected platform from the recent hardware configuration. The platform is no longer available as an assignable registered platform and is no longer displayed in the Platform Manager.
Remove All	▪ Button ▪ Edit menu ▪ Shortcut key: `Shift+Del`	Lets you remove all listed platforms from the recent hardware configuration. The platforms are no longer available as assignable

Command	Access	Description
		registered platforms and are no longer displayed in the Platform Manager. **Tip** Platform information is stored in two files: ▪ `RecentHardware.xml` ▪ `RecentHardware_Backup.xml` Both files are located in: `%CommonProgramFiles%\dSPACE\PlatformManagement` The `RecentHardware.xml` file is updated each time registered hardware is changed in ConfigurationDesk. Thus, it does not contain platform information after all the platforms were removed via Clear System on the Platforms ribbon or via Remove All in the Manage Recent Platform Configuration dialog. The `RecentHardware_Backup.xml` file is updated only when a platform is registered via the Register Platforms dialog. Thus, it still contains registered platforms after all the platforms were removed. You can use the file to restore registered platforms by importing it via the Manage Recent Platform Configuration dialog.
Select All	▪ Context menu of a platform ▪ Shortcut key: **`Ctrl+A`**	Lets you select all the items in the platform list.
Sort Alphabetically	View menu	Lets you sort the platform list alphabetically in ascending order by platform names.

Related topics

Basics

Basics on Registering Real-Time Hardware (ConfigurationDesk Real-Time Implementation Guide 🕮)

HowTos

How to Register dSPACE Real-Time Hardware (ConfigurationDesk Real-Time Implementation Guide 🕮)

Refresh Interface Connections

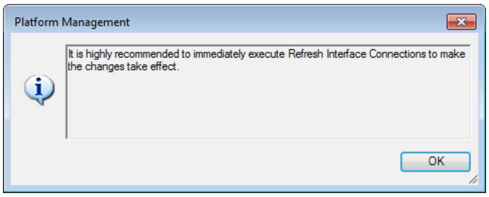
Purpose

To solve problems with the display of SCALEXIO or MicroAutoBox III systems in the Platform Manager, for example, after they have been switched off and on.

Description

ConfigurationDesk refreshes the connections by resetting the device drivers of the platforms and reinitializing the **Platform Manager** with the information from the **Manage Recent Platform Configuration** dialog. The device drivers are reset only if at least one platform is registered.

In some cases, for example, after importing platform information via the **Manage Recent Platform Configuration** dialog, ConfigurationDesk will prompt you to refresh interface connections so that recent changes are reflected in the **Platform Manager**.



Platform Management

It is highly recommended to immediately execute Refresh Interface Connections to make the changes take effect.

Register Platforms

Purpose

To register dSPACE real-time hardware.

Result

ConfigurationDesk now recognizes the registered platforms.

Description

In ConfigurationDesk, the registered hardware systems are treated as platforms which are displayed and which can be accessed via the **Platform Manager**. To register platforms, you must enter their IP addresses, or alias names or board names, or MAC addresses in the **Register Platforms** dialog. As an alternative, you can scan for available processing units.

Real-time hardware can be registered independently of ConfigurationDesk projects and applications.

Tip

For multi-PU applications, you can combine several processing units to a multi-processing-unit system (multi-PU system).

After you register a new platform by clicking **Register**, ConfigurationDesk starts to search the local network for the entered settings (board names or MAC addresses). If they are found in the same network as your host PC, the corresponding platforms are automatically read out and displayed in the dialog and in the Platform Manager. You have successfully registered your hardware in ConfigurationDesk.

Note

> Using the MAC address or board name to find and register the hardware is supported only if the host PC and hardware are part of the same subnetwork. If your hardware system is installed in a different subnetwork connected to your host PC's network via a router or gateway, you must use the IP address or network alias for registering. Otherwise ConfigurationDesk is unable to find the hardware system.

Platform information is stored in two files:

- `RecentHardware.xml`
- `RecentHardware_Backup.xml`

Both files are located in:

```
%CommonProgramFiles%\dSPACE\PlatformManagement
```

The `RecentHardware.xml` file is updated each time registered hardware is changed in ConfigurationDesk. Thus, it does not contain platform information after all the platforms were removed via **Clear System** on the **Platforms** ribbon or via **Remove All** in the **Manage Recent Platform Configuration** dialog.

The `RecentHardware_Backup.xml` file is updated only when a platform is registered via the **Register Platforms** dialog. Thus, it still contains registered platforms after all the platforms were removed. You can use the file to restore registered platforms by importing it via the **Manage Recent Platform Configuration** dialog.

Register Platforms dialog

The **Register Platforms** dialog serves to specify the register settings for a SCALEXIO or MicroAutoBox III system and to get information on the platforms registered so far.

Platforms Displays the platform type which you can register in ConfigurationDesk.

Register settings Lets you view and specify the register settings for the platform:

- **Platform type**: Displays the type of the selected platform.
- **Platform name**: Lets you specify a user-defined platform name. The name must fulfill the following conditions:
  - Allowed characters: A … Z, a … z, 0 … 9, _
  - Number of characters: max. 64
  - The name must be unique. This means no other registered platform (activated and deactivated) must use this name.

  After registration, the new name is displayed in ConfigurationDesk's **Platform Manager** as well as in the Platform Managers of other dSPACE software products which support SCALEXIO or MicroAutoBox III hardware and which are installed on your host PC. Thus, you can use this name to identify the platform in different dSPACE products.

- **Connection parameter**: Lets you specify a connection parameter. You can select one of the following parameters:
  - **IP address**: Lets you specify the network client for registration.
  - **MAC address**: Lets you specify the MAC address of the processing hardware.
  - **Board name**: Lets you specify the board name of the processing hardware.
  - **Alias name**: Lets you specify the alias name of the connection that is used for registration. For details on using alias names, refer to Using an Alias Name for the SCALEXIO System (SCALEXIO Hardware Installation and Configuration 🕮) or Using an Alias Name for a MicroAutoBox III (MicroAutoBox III Hardware Installation and Configuration 🕮).
- **Scan for available processing units** (SCALEXIO): Lets you open the **Scan Local Network for Processing Units** dialog. Refer to Scan Local Network for Processing Units/Platforms dialog on page 81.
- **Scan for available platforms** (MicroAutoBox III): Lets you open the **Scan Local Network for Platforms** dialog. Refer to Scan Local Network for Processing Units/Platforms dialog on page 81.
- **Processing units** (SCALEXIO): Lists all processing units entries belonging to the multi-processing-unit system you want to register.
- **Platform** (MicroAutoBox III): Shows the platform you want to register.
- 🗋 (SCALEXIO): Lets you add a new processing unit entry to the **Processing units** list. Depending on the specified **Connection parameter**, you can select an IP address, or specify a MAC address, a board name, or an alias name of a processing unit you want to register.
- ☒ (SCALEXIO): Lets you delete a processing unit entry from the **Processing units** list.

Register Lets you complete the registration. The registered platform is displayed together with the platform properties in the **Registered platforms** list. The registered platform is also displayed in the **Platform Manager**.

Registered platforms list Displays all the registered platforms with the following information: platform name, platform type, network client, serial number/identifier, MAC address, port address and processor name.

You can customize the display in the **Registered platforms** list using the following commands available from the context menu of column headers:

- **Best Fit**: Lets you optimize the width of the selected column.
- **Best Fit (all columns)**: Lets you optimize the widths of all columns according to the width of the editor or browser.
- **Column Chooser**: Lets you open a dialog for customizing the columns of the platforms list. To add a column to the list, drag it from the opened dialog to the list header. To remove a column from the list, drag its header to the dialog. The changes are automatically saved and available the next time you open the **Register Platforms** dialog.
- **Sort Ascending**: Lets you sort the list alphabetically in ascending order according to the selected column.
- **Sort Descending**: Lets you sort the list alphabetically in descending order according to the selected column.

Scan Local Network for Processing Units/Platforms dialog

Type Lets you select the filter item type you want to filter the results list by. If you select 'None', no filter is applied.

Value Lets you enter a filter string.

Match whole word Lets you specify to search only for a matching pattern substring.

(Re)scan Lets you start the scan process. ConfigurationDesk scans the subnetwork your host PC is connected to for connected platform hardware and lists all the platform hardware found in the network.

List of available processing units/platforms Displays all the processing units or platforms found in the network during the scan process according to the specified filter. The results list contains the IP address, MAC address, board name, system name and identifier for each processing unit or platform found.

List of selected processing units/Selected platform Displays all the processing units or the platform selected for registration. When you click Apply, the listed platform hardware is assigned to the platform you want to register, and the connection parameter value of each list item is transferred to the Register Platforms dialog.

The following buttons are available to move elements from one list to the other list:

⬇	Moves the selected element(s) from the Available processor boards/processing units/platforms list to the Selected processor boards/processing units/platform list.
⬆	Moves the selected element(s) from the Selected processor boards/processing units/platform list to the Available processor boards/processing units/platforms list.

Related topics

Basics

Basics on Registering Real-Time Hardware (ConfigurationDesk Real-Time Implementation Guide 🕮)

HowTos

How to Register dSPACE Real-Time Hardware (ConfigurationDesk Real-Time Implementation Guide 🕮)

Platform Manager

Objective

The Platform Manager allows you to display registered hardware in different views.

Where to go from here

Information in this section

Assembly View

Purpose

To display a mechanical-based view of the real-time hardware in the Platform Manager.

Result

The Platform Manager displays the physical assembly of the real-time hardware. The hardware is displayed in a hierarchical structure, for example, with the SCALEXIO rack at the top. The SCALEXIO rack node serves as a container, for example, for I/O units.

Related topics

References

Network View.. 99

Collapse (Platform Manager)

Purpose

To collapse the subcomponents of the selected hardware component in the Platform Manager.

Result

The subcomponents of the selected hardware component are hidden in the Platform Manager.

Description

The Platform Manager displays all the hardware components of the connected hardware. If the hardware consists of many components, it may be useful to reduce complexity by hiding subcomponents.

Related topics

References

Expand (Platform Manager).. 98

Create Support Info

Purpose

To generate an XML file containing textual information on platforms that are currently registered in the Platform Manager.

Result

To help dSPACE Support analyze a problem, ConfigurationDesk generates the `SupportInfo.xml` file. The file contains relevant information on all the platforms that are currently registered in the **Platform Manager**. The file is saved automatically and the path is displayed in a message.

The `SupportInfo.xml` file is overwritten each time you call the Create Support Info command.

Ethernet Switch Configuration

Purpose

To specify and manage the Ethernet switch and physical layers configuration of:

- ETH/AETH ports of a MicroAutoBox III
- DS6333-PE Automotive Ethernet Boards, DS6333-CS Automotive Ethernet Boards, or DS6335-CS Ethernet Boards that are installed in a SCALEXIO system

Result

ConfigurationDesk opens the Ethernet Switch Configuration dialog which lets you specify and manage the Ethernet switch configuration. The Ethernet switch configuration comprises the configuration of the board's physical layers (PHYs) and the Ethernet switch. The Ethernet switch configuration is stored on the related SCALEXIO processing hardware or MicroAutoBox III.

Ethernet Switch Configuration dialog

To select a default Ethernet switch configuration, and save or delete a user-specific Ethernet switch configuration.

> **Tip**
>
> An asterisk is displayed as a change indicator next to fields that you have edited but not yet saved.

Board type Displays the board type of the related Ethernet board.

Serial number Displays the serial number of the board that provides the selected Ethernet switch.

Slot (Available only for SCALEXIO) Displays the slot to which the board is installed:

- (DS6001) Displays the board's PCIe slot number.

  > **Note**
  >
  > The board's I/O slot number differs from the PCIe slot number that is used to generate the configuration's default name `SwitchX_Y`. (In a SCALEXIO LabBox, the I/O slot numbers 3 … 7 correspond to the PCIe slot numbers 1 … 5.)

- (SCALEXIO Processing Unit) Displays the slot number for the related Ethernet board. This number can change if you remove or install another Ethernet board.

Layer (Available only for MicroAutoBox III) Displays the layer of the MicroAutoBox III to which the board is installed.

The processor board is always installed to layer 0.

Processing unit Displays the name and the IP address of the related processing hardware.

Configuration type (Available only for SCALEXIO) Lets you select whether to specify the default switch configuration as a slot-specific or board-specific configuration.

- A slot-specific switch configuration is intended for use with a specific slot. All Ethernet boards that are operated at this slot and that have no board-specific switch configuration work with this slot-specific configuration.

  A slot-specific switch configuration is stored with the naming scheme SwitchX_Y. X is a number that is currently assigned to the board's slot, Y indicates the index of the switch on the board (currently, Y is always 1).
- A board-specific switch configuration is intended for use with a specific board, i.e., the configuration is stored on the hardware together with the serial number of the board. If the board is plugged into another slot, this type of switch configuration is also valid at this new slot.

  A board-specific switch configuration must be saved with a free name, which must differ from the SwitchX_Y naming scheme.

If both switch configuration types exist for a board, the board-specific configuration has the higher priority.

Switch name Lets you specify a new name for the Ethernet switch with a case-sensitive string. The maximum number of characters is 63.

Relevant for SCALEXIO only:

- If you selected the slot-specific switch configuration, you cannot specify a name for the switch. The default name `SwitchX_Y` is always used.
- If you selected the board-specific switch configuration, you can specify an individual name for the switch. The name must differ from the naming scheme `SwitchX_Y`. If you enter a name according to the naming scheme `SwitchX_Y`, it cannot be saved as a board-specific configuration.

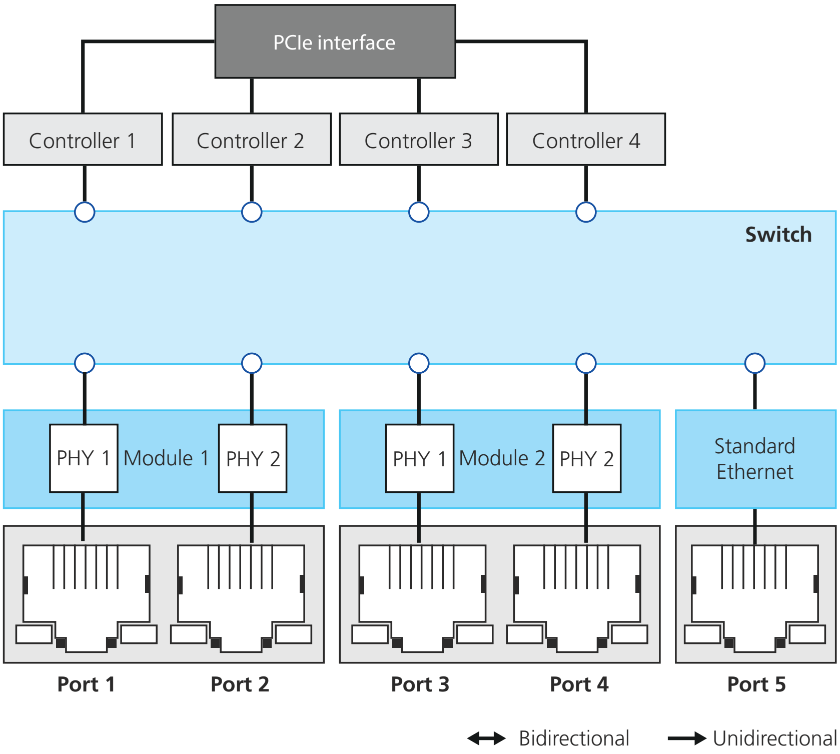
Configuration mode Lets you select one of the predefined settings for the Ethernet switch. The configuration modes define how the Ethernet ports and the internal controllers are connected to each other and how the data streams can be used internally.

- For SCALEXIO, currently six predefined configurations are offered for selection. Refer to Preconfigured Ethernet switch configurations for SCALEXIO on page 86.
- For MicroAutoBox III, currently eight predefined configurations are offered for selection. Refer to Preconfigured Ethernet switch configurations for MicroAutoBox III on page 90.

Preconfigured Ethernet switch configurations for SCALEXIO The following preprogrammed configuration modes are defined for SCALEXIO. The information is also provided as tooltips. Move the mouse pointer over a configuration mode to open the tooltip with the respective information.

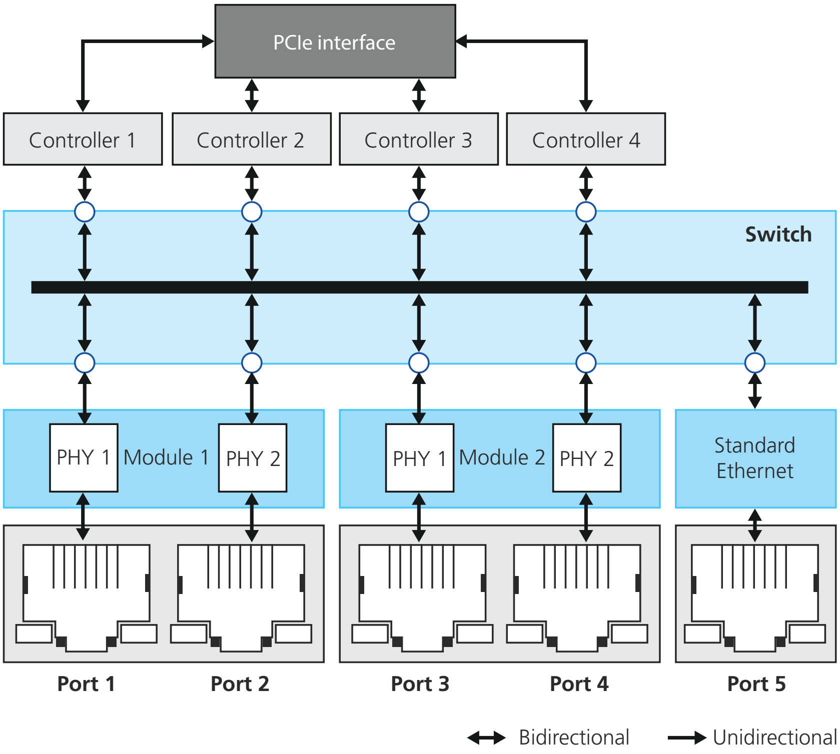
- Configuration 1:

  This configuration disables all Ethernet controllers and ports. This configuration is stored on the hardware and is always used during the initialization phase.



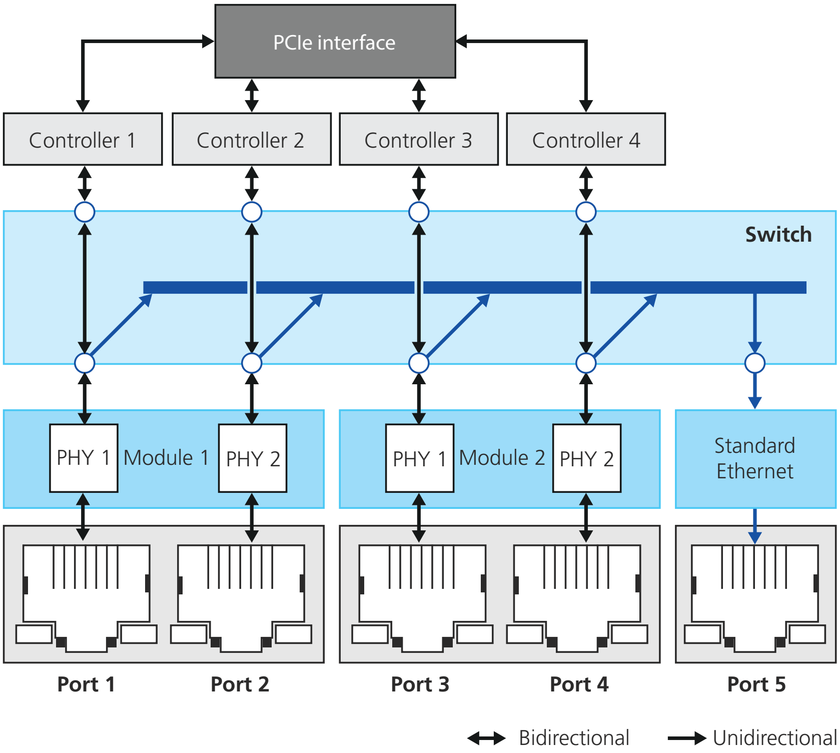
- Configuration 2:

  This configuration sets the internal Ethernet switch to the unmanaged mode. All Ethernet ports and controllers are switched to each other.



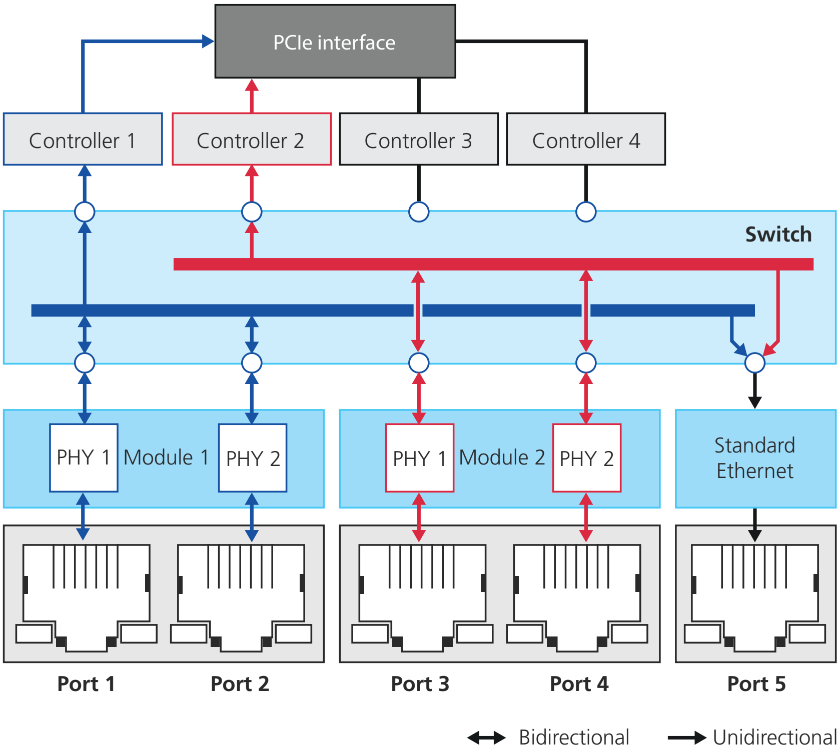
- Configuration 3:

  This configuration connects each Ethernet port to a specific Ethernet controller except port 5. Port 5 lets you monitor the I/O Ethernet communication of the other ports. All Ethernet ports are unswitched.



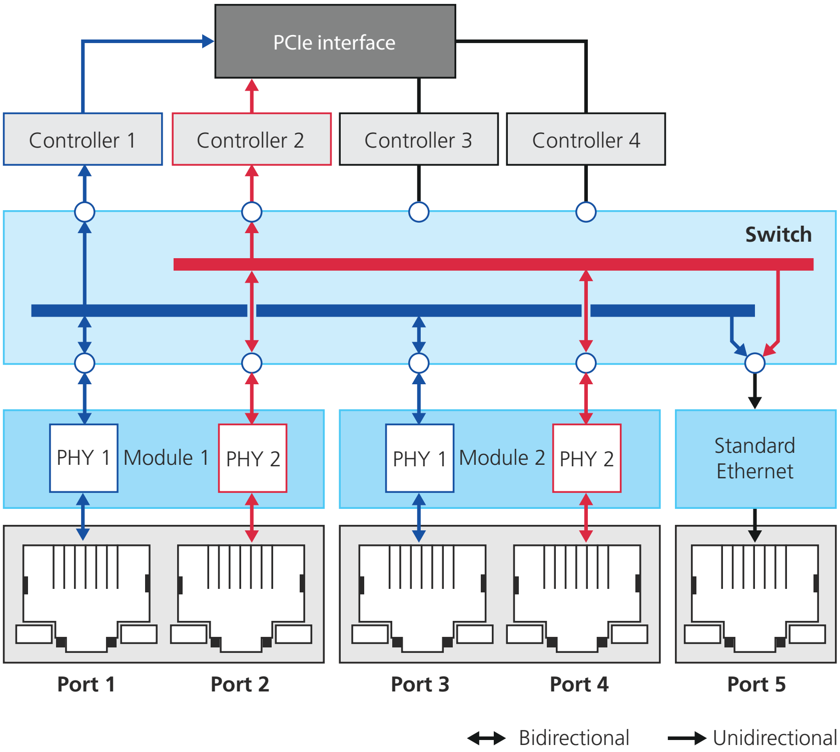
- Configuration 4:

  This configuration provides one pass-through monitoring channel for the ports of the Ethernet module 1 and one for the ports of the Ethernet module 2.



- Configuration 5:

  This configuration provides two channels for converting automotive Ethernet to standard Ethernet and vice versa if different Ethernet modules are installed. The Ethernet controllers 1 and 2 and Port 5 let you monitor the I/O Ethernet communication.



- Configuration 6:

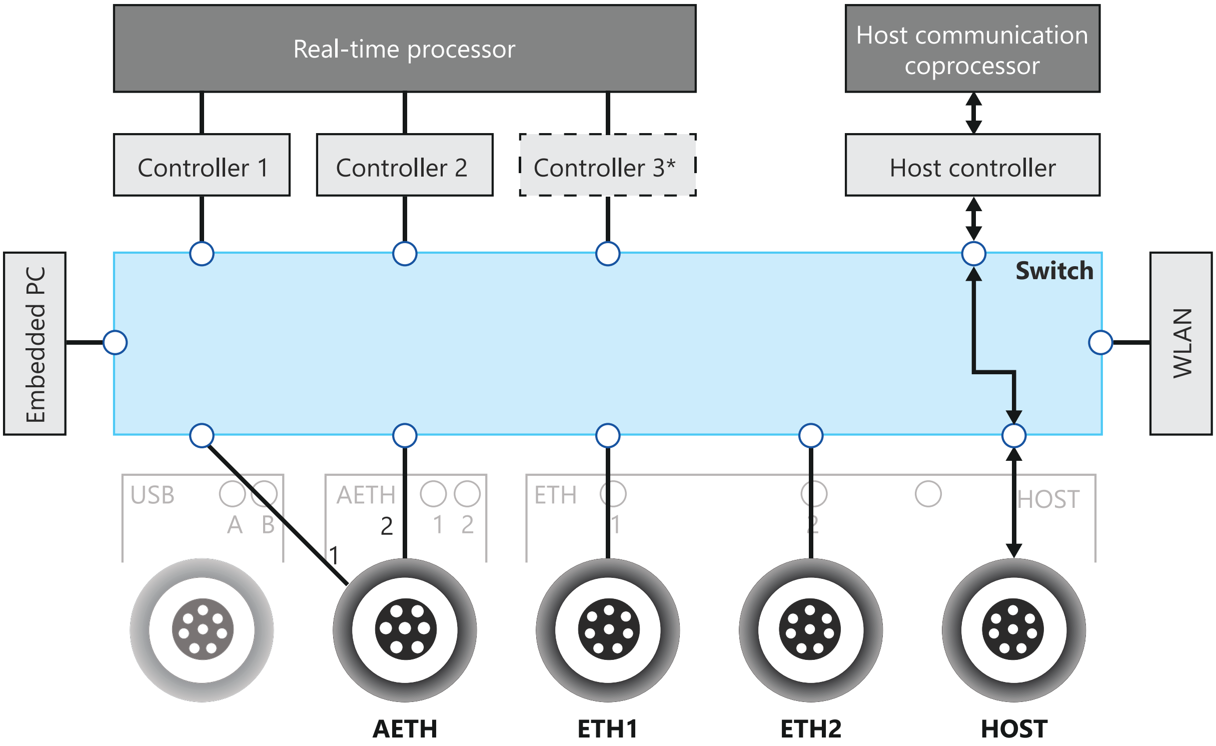
  This configuration uses the permanently saved Ethernet switch and PHY configuration of a real-time application.

  Refer to Real-Time-Application-Specific Configuration on page 97.

Preconfigured Ethernet switch configurations for MicroAutoBox III The following preprogrammed configuration modes are defined for the DS1403 Processor Board. The following information is also provided as tooltips. Move the mouse pointer over a configuration mode to open the tooltip with the respective information.

- Configuration 1:

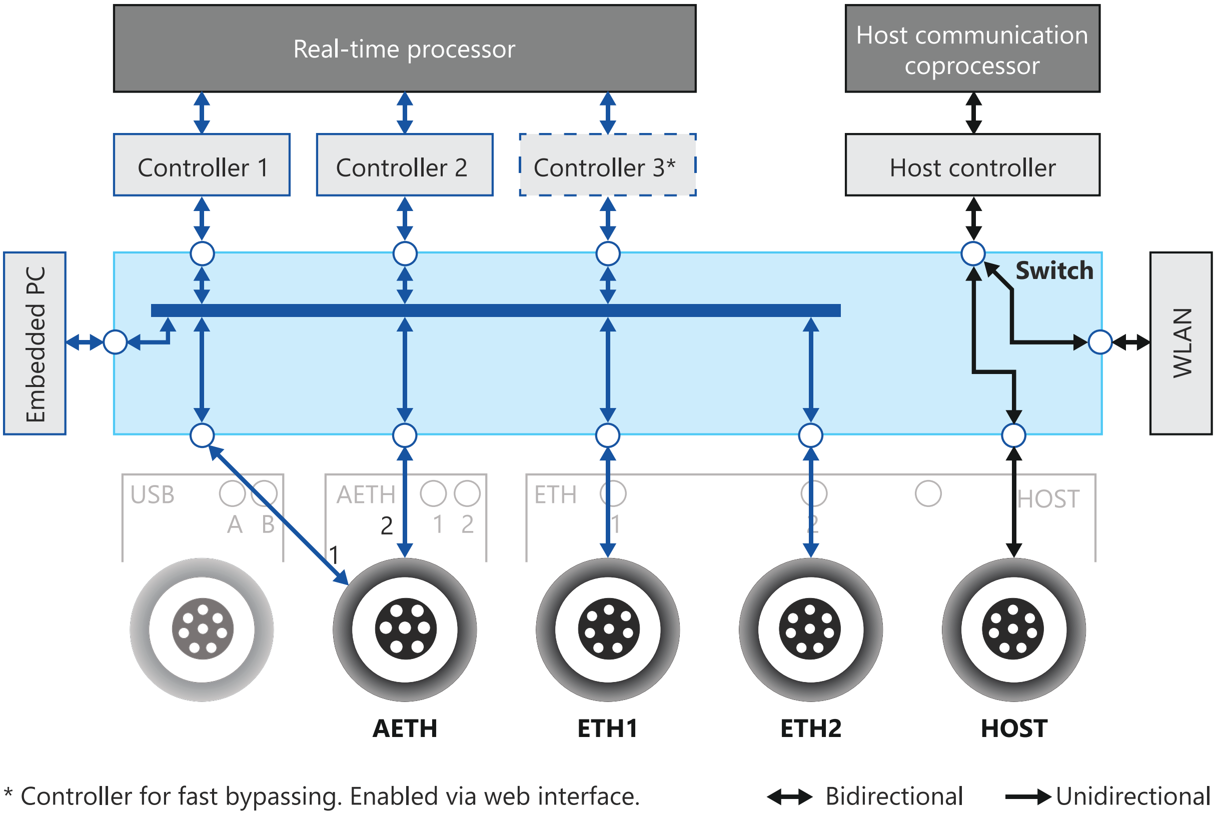
  This configuration enables only the host communication via the HOST port. All other switch ports are disabled. This configuration is stored on the hardware and is always used during the initialization phase. Using this default configuration saves initialization time for preloading another configuration.



* Controller for fast bypassing. Enabled via web interface.

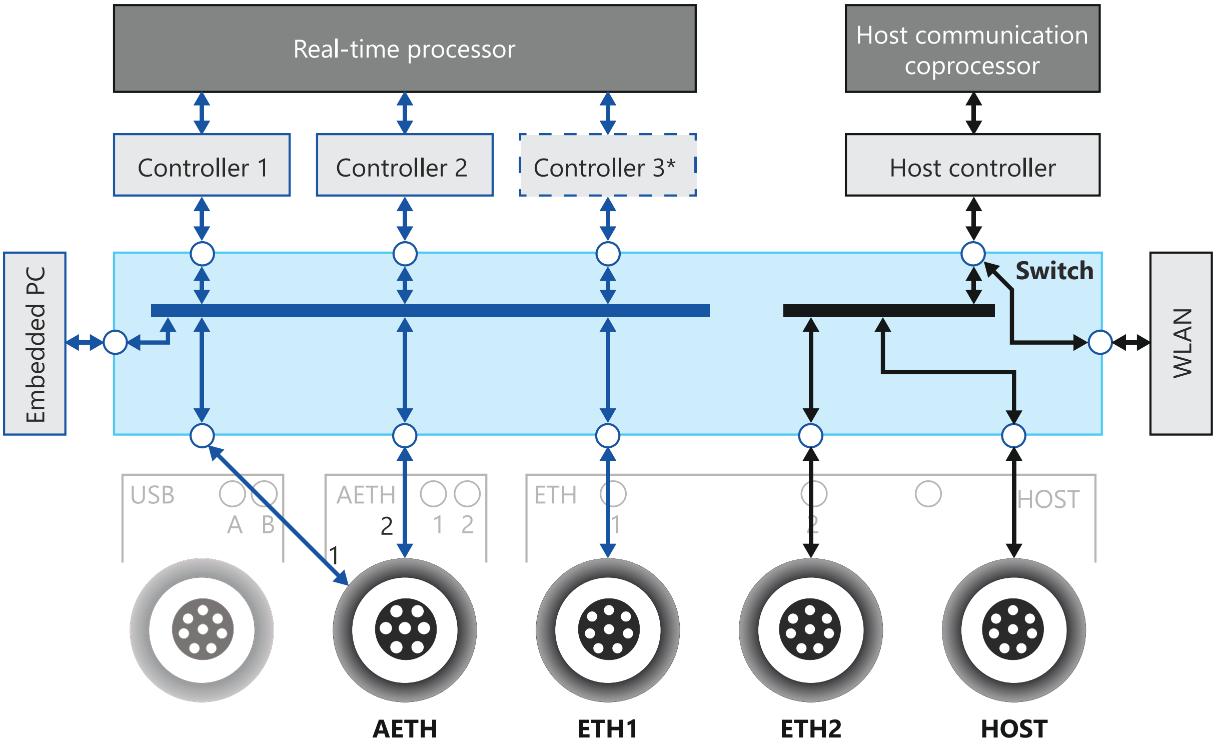
- Configuration 2:

  This configuration separates the host communication and the I/O Ethernet communication. The ports for I/O Ethernet communication are switched to each other.



* Controller for fast bypassing. Enabled via web interface.

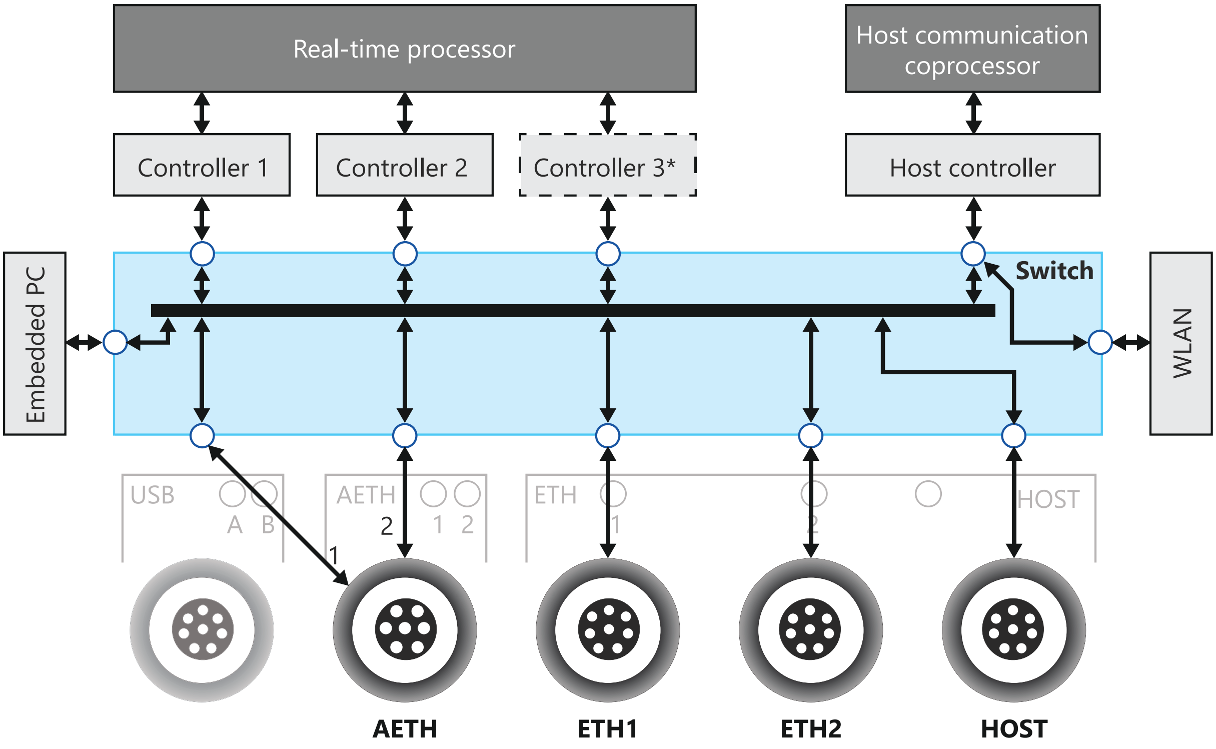
- Configuration 3:
  This configuration sets the ETH2 port as an additional port for the wired host communication. The host communication and the I/O Ethernet communication are separated. The ports for wired host communication are switched to each other, just as the ports for I/O Ethernet communication.



* Controller for fast bypassing. Enabled via web interface. ↔ Bidirectional → Unidirectional

- Configuration 4:

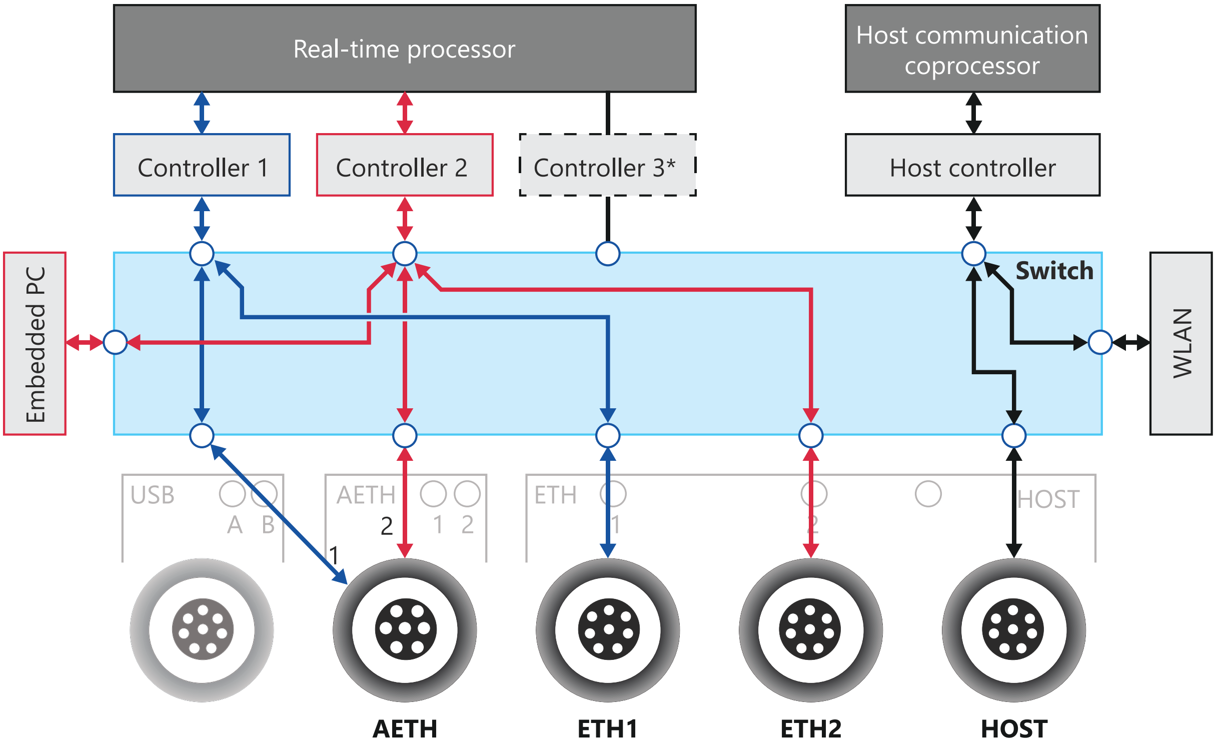
  This configuration sets the internal Ethernet switch to the unmanaged mode. All switch ports are switched to each other, except WLAN. The wired host communication is not separated from the I/O Ethernet communication.



* Controller for fast bypassing. Enabled via web interface.

- Configuration 5:

  This configuration connects each switch port for I/O Ethernet communication to a specific Ethernet controller. The switch ports are unswitched. It is recommended to use only one port per I/O Ethernet controller, either the AETH port, the ETH port, or the switch port for a built-in Embedded PC.



* Controller for fast bypassing. Enabled via web interface.

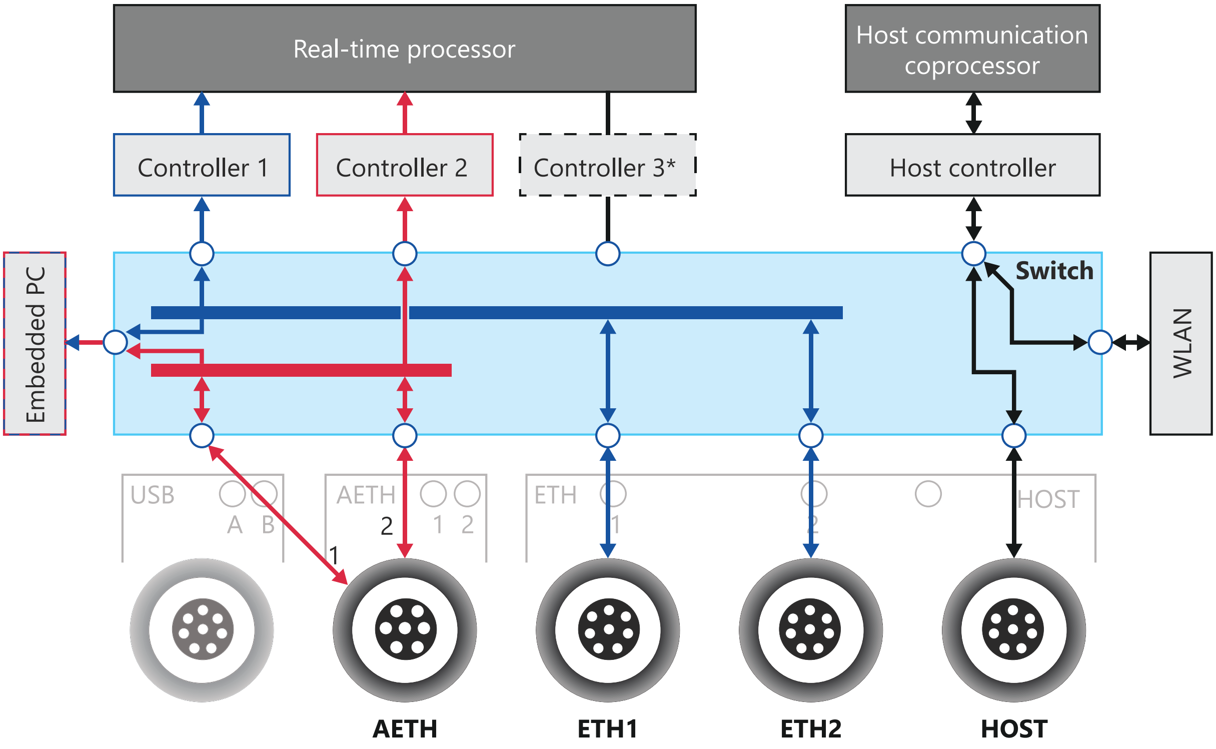
↔ Bidirectional → Unidirectional

- Configuration 6:

  This configuration provides one pass-through monitoring channel for standard I/O Ethernet and one for automotive I/O Ethernet. The I/O Ethernet controllers and a built-in Embedded PC can monitor the I/O Ethernet communication.

  A high data throughput at the Ethernet ports can lead to data loss on the unidirectional monitoring channel. However, the data flow through the monitored ports is not affected.

  For example: The monitoring of the transmit and receive data lines in full duplex mode via an unidirectional monitoring channel can exceed the maximum data rate of 1 GBit/s.



* Controller for fast bypassing. Enabled via web interface.

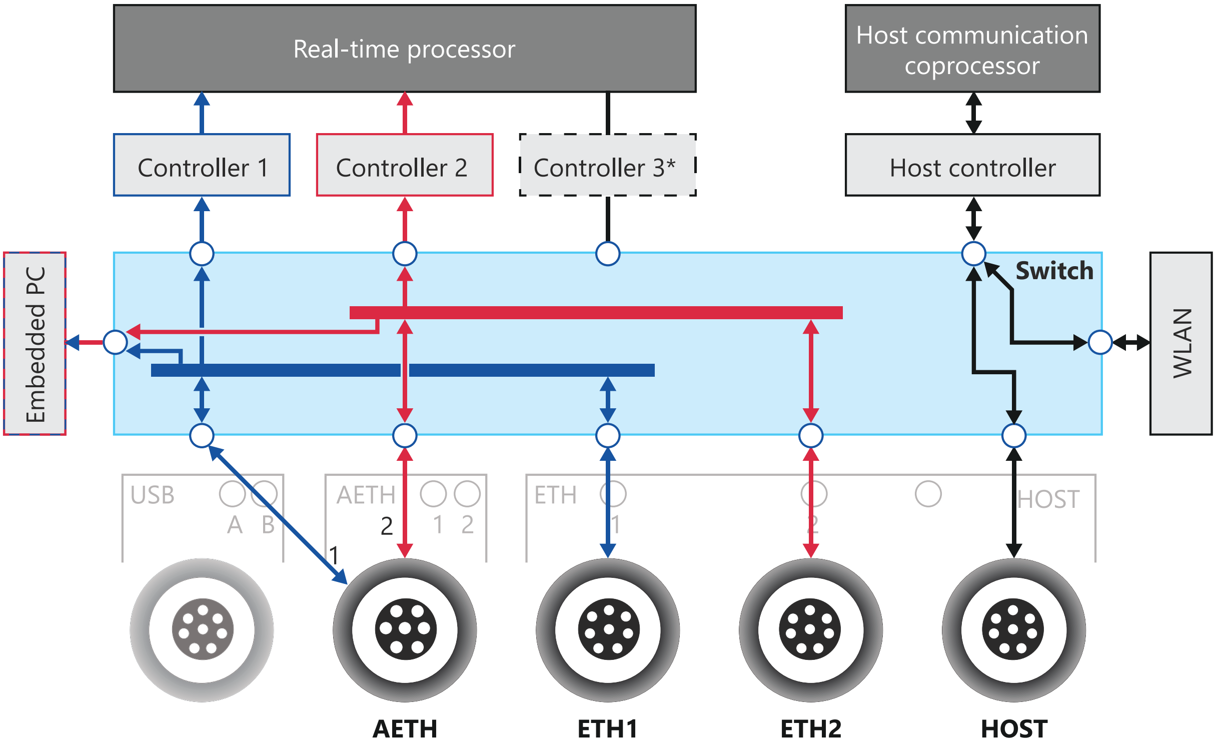
↔ Bidirectional → Unidirectional

- Configuration 7:

  This configuration provides two channels for converting automotive Ethernet to standard Ethernet and vice versa. The I/O Ethernet controllers and a built-in Embedded PC can monitor the I/O Ethernet communication.

  A high data throughput at the Ethernet ports can lead to data loss on the unidirectional monitoring channel. However, the data flow through the monitored ports is not affected.

  For example: The monitoring of the transmit and receive data lines in full duplex mode via an unidirectional monitoring channel can exceed the maximum data rate of 1 GBit/s.



* Controller for fast bypassing. Enabled via web interface. ↔ Bidirectional → Unidirectional

- Configuration 8:

  This configuration uses the permanently saved Ethernet switch and PHY configuration of a real-time application.

  Refer to Real-Time-Application-Specific Configuration on page 97.

Real-Time-Application-Specific Configuration Lets you save an Ethernet switch and PHY configuration of a real-time application. When you click Save, ConfigurationDesk permanently saves the most recently used Ethernet switch and PHY configuration provided by a real-time application. The Switch name is used as the configuration name. To delete a permanently saved user-specific configuration, click the Delete button.

The most recently used Ethernet switch and PHY configuration can be saved only if the real-time application is unloaded and the SCALEXIO or MicroAutoBox III system is not restarted.

Note

Saving or deleting a user-specific configuration is done immediately after clicking Save or Delete. This differs from other changes that are made in the dialog, which are only effective and written to the hardware when you click OK or Apply.

A stored user-specific switch and PHY configuration can be activated by selecting the appropriate configuration mode (SCALEXIO: configuration mode 6, MicroAutoBox III: configuration mode 8) and restarting the hardware.

Physical layer configuration properties The lower part of the dialog contains the PHY configurations of the single Ethernet ports that are connected to the Ethernet switch.

Module name (Available only for SCALEXIO) Displays information on the module type for the selected port.

Power Lets you switch on or off the PHY of the selected port.

Link speed Lets you specify the data rate and duplex mode for the selected port.

If the port is set to Autonegotiation, the port automatically detects the data rate and the duplex mode. The autonegotiation mode is standard for Gigabit Ethernet.

Mode (Available only for Automotive Ethernet ports) Lets you specify the selected port as master or slave port.

One PHY of an automotive Ethernet connection must be the master, the other must be the slave to establish a link between the connected ports.

- Master

  The PHY of the Automotive Ethernet Module/MicroAutoBox III starts the training process to establish a link between the AETH port and the connected automotive Ethernet device.
- Slave

  The connected automotive Ethernet device starts the required link training process.

Related topics

HowTos

How to Configure I/O Ethernet Ports (ConfigurationDesk Real-Time Implementation Guide 🕮)
How to Configure Internal Ethernet Switches (ConfigurationDesk Real-Time Implementation Guide 🕮)

Expand (Platform Manager)

Purpose

To expand the collapsed subcomponents of the selected hardware component in the Platform Manager.

Result

The hidden subcomponents of the selected hardware component are now displayed in the Platform Manager.

Description

If you want to view collapsed subcomponents, you can expand them.

Related topics

References

Collapse (Platform Manager)... 83

Network View

Purpose

To display a network-based view of the real-time hardware in the Platform Manager.

Result

The Platform Manager displays the logical network (I/O network) of the real-time hardware, with the SCALEXIO processing units at the top.

Related topics

HowTos

How to Establish a Network Connection in the Hardware Topology (SCALEXIO) (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Assembly View... 83

Platform Manager

Purpose

To display the hardware components of all hardware systems connected to your host PC and accessible via ConfigurationDesk.

Description

The Platform Manager displays all the hardware components which can be accessed via ConfigurationDesk. The hardware components are arranged in a hierarchical tree structure. Each node in the tree displays the name of the hardware component it represents. If you select a hardware component in the Platform Manager, you can open a context menu with component-specific commands.

The Platform Manager allows you to:

- Download, start, stop, and unload real-time applications.
- Check the state of a real-time application. Its state is visualized in the Platform Manager by symbols below the processing unit node.
- Change the system name.

Symbols

The Platform Manager gives access to all the hardware components and subcomponents of connected SCALEXIO systems or MicroAutoBox III systems which can be accessed via ConfigurationDesk. It displays each component together with a symbol giving information on the component type.

Symbol	Meaning
🖥	Indicates the host PC.
▤	Indicates a SCALEXIO rack or system.
▢	Indicates a MicroAutoBox III system.
▣	Indicates real-time PC.
▭	Indicates a box or unit.
⧉	Indicates an angle unit group.
◧	Indicates an angle unit.
⧉	Indicates an Ethernet controller set.
⊞	Indicates an Ethernet controller.
📄	Indicates the real-time application.
▶	Indicates a running real-time application.
🟥	Indicates a stopped real-time application.
!🟥	Indicates a terminated real-time application.
?🟥	Indicates an undefined status for the real-time application. This icon is displayed if you register a SCALEXIO platform consisting of processing units with application processes already running on them, but not all the application processes belonging to the parent application are part of your registered SCALEXIO platform.
🟩	Indicates a signal measurement board, a signal generation board, a processor board, a bus board, an I/O module, or a bus module.
ETH	Indicates an Ethernet board.
✣	Indicates an IOCNET link board.
≡	Indicates a channel group.
⁄	Indicates a channel.

Symbol	Meaning
!	Indicates a firmware warning. There are firmware inconsistencies. For example, the firmware version of a SCALEXIO Processing Unit differs from the firmware version of your current RCP and HIL software installation, or hardware components belonging to the SCALEXIO platform contain different firmware versions. Associated warning messages are shown in the Message Viewer.

Refresh Platform Configuration

Purpose

To refresh the hardware configuration.

Result

The platform configurations are refreshed. The view of the structure shown in the Platform Manager is updated.

Show Connected Clients

Purpose

To display the clients to which the selected SCALEXIO platform or SCALEXIO Processing Unit is connected.

Connected Client Overview dialog

For each processing unit, the Connected Client Overview dialog displays all the client processes that access the unit.

Host Name Displays the host name of the client that is connected to the selected processing unit.

IP Address Displays the IP address of the client that is connected to the selected processing unit.

User Name Displays the user name of the client that is connected to the selected processing unit.

Connection Time Displays the time when the client connection to the selected processing unit was established.

Process Name Displays the name of the client process that accesses the selected processing unit.

Refresh Lets you update the display of the client processes.

Update Firmware

Purpose

To open the Update Firmware Wizard for updating or repairing the firmware of the selected SCALEXIO or MicroAutoBox III system.

Result

The Update Firmware Wizard opens for updating or repairing firmware components.

Description

ConfigurationDesk checks whether the installation contains appropriate firmware which is later than the firmware that is currently installed on the hardware. If a later version is available, the firmware should be updated.

Firmware version deviations for a SCALEXIO system or MicroAutoBox III system are indicated in the **Platform Manager**. The affected hardware components are marked by the ⚠ symbol. You can find associated warning messages in the Message Viewer.

For important notes on the firmware versions and compatibility, and notes on the update process, refer to Basics on Updating SCALEXIO or MicroAutoBox III Firmware (ConfigurationDesk Real-Time Implementation Guide).

> **Note**
>
> The Update Firmware Wizard supports SCALEXIO systems as of dSPACE Release 2015-B. If you want to update the firmware version on a SCALEXIO system to an earlier version, you have to use ConfigurationDesk from an earlier dSPACE Release.

The Update Firmware Wizard provides the following dialogs to configure and start a firmware update.

Select Mode dialog

Lets you select the firmware update mode.

- Firmware update mode

  The firmware handling process is configured for updating all firmware components of the real-time hardware to a later version.
- Firmware repair mode

  The firmware handling process is configured for repairing the selected firmware components of the real-time hardware by reloading the same firmware versions.

Next > Opens the next dialog.

OK Closes the Update Firmware Wizard without starting a firmware update.

Select Firmware Archive dialog

To select the firmware archive to be loaded.

Currently selected archive Displays the selected firmware archive. If the firmware archives are installed on the default installation path, either the latest firmware archive for the registered and selected real-time hardware is displayed, or the firmware archive that you specified manually via the Browse for archive button.

Archive selection Lets you select the firmware archive to be used for firmware update.

- Select an archive from installation

  By default, the latest firmware archive is selected that corresponds to the active dSPACE Release installation.
- Select an archive from file system

  Lets you select a firmware archive in another version or from another path.

< Back Opens the previous dialog to change the update mode.

Next > Opens the next dialog.

OK Closes the Update Firmware Wizard without starting a firmware update.

Select Firmware Components dialog

To start the prepared firmware update.

Name Displays the names of the selected platform, its hardware components, and the related firmware components.

Current FW Displays the firmware versions currently loaded to the listed hardware components.

Available FW Displays the firmware versions available in the specified firmware archive.

Update

- In firmware update mode, the firmware components to be updated are marked. You cannot modify the selection.
- In firmware repair mode, you have to select at least one firmware component to enable the repair process.

Status Displays the status of the firmware update process. If the firmware component does not provide progress information, only the states 50% and 100% are displayed. If the update process finished successfully, the status is set to OK.

Update/Repair

- In firmware update mode, the Update button is enabled to start a firmware update process if later firmware versions are available in the firmware archive than those currently loaded to the hardware.

- In firmware repair mode, the **Repair** button is enabled to start a firmware repair process, if the firmware versions of the loaded firmware components and the versions of the specified firmware archive are identical, and if you have selected at least one firmware component in the **Update** column to be repaired.

< Back Opens the previous dialog to change the selected firmware archive.

OK Closes the **Update Firmware Wizard** without starting a firmware update.

External Device Management

Objective

Before you can use external devices in ConfigurationDesk, you need a representation of them in ConfigurationDesk. External device topologies represent the external devices which are connected to the dSPACE hardware, for example ECUs or external loads. The topologies are the basis for using external devices in a ConfigurationDesk application.

You can display and configure the properties of external device topology elements in the **Properties Browser**.

Where to go from here

Information in this section

Information in other sections

Specifying the External Device Interface (ConfigurationDesk Real-Time Implementation Guide 🕮)

External Device Topology

Objective

You can use a device topology in your application to add elements from it to the application.

The **External Device Browser** displays the device topology of the application and lets you add, delete and manage device topology elements.

Where to go from here

Information in this section

Copy (Device Connector)

Purpose

To copy the selected device pins and device connectors (and the containing pins) to the Clipboard.

Result

The selected device pins and device connectors are copied to the Clipboard. You can now paste them to a different position. You can also copy & paste device pins from the Pins and External Wiring table to the External Device Connectors table.

Tip

You can select multiple elements by pressing `Ctrl` while selecting.

Related topics

HowTos

How to Group Device Pins (ConfigurationDesk Real-Time Implementation Guide 📖)

References

External Device Connectors Table 113
Paste Before (Device Connector) 122
Paste Into (Device Connector) 124
Paste Multiple Before (Device Connector) 125
Paste Multiple Into (Device Connector) 127
Pins and External Wiring Table 129

Copy (Device Topology)

Purpose

To copy the selected device topology elements and their subelements to the Clipboard.

Result

The selected device topology elements and their subelements are copied to the Clipboard.

Tip

You can select multiple elements by pressing `Ctrl` while selecting.

Description

You can now do one of the following:

- Paste the elements to the device topology in the **External Device Browser** or the **External Device Configuration** table once or multiple times. You can also paste them into another element of the device topology, for example, a port into another port group.
- Paste the elements to a working view once or multiple times. This automatically adds them to the signal chain and the device topology.
- Add the elements to a working view. This only works for elements that are not already part of the current working view. Device topology elements that were not used in the signal chain before adding them to a working view are added to the signal chain.

Related topics

HowTos

How to Copy and Paste Device Topology Elements (ConfigurationDesk Real-Time Implementation Guide)

References

Add to Working View	257
Cut (Device Topology)	111
Paste Before (Device Topology)	123
Paste Into (Device Topology)	124
Paste Multiple Before (Device Topology)	126
Paste Multiple Into (Device Topology)	128

Create Custom Property

Purpose

To add custom external device properties to document specific characteristics of device topology elements or to define sort keys for tables.

Result

A Custom Property Name dialog opens.

Description

Via the Custom Property Name dialog, you can add custom external device properties to document specific characteristics of device topology elements or to define sort keys for the table. For instructions, refer to How to Add Custom Device Properties (ConfigurationDesk Real-Time Implementation Guide).

Related topics

HowTos

How to Add Custom Device Properties (ConfigurationDesk Real-Time Implementation Guide)

References

Delete Custom Property	112

Cut (Device Connector)

Purpose

To cut the selected device pins and device connectors (and the containing pins) to the Clipboard.

Result

The selected device pins and device connectors are cut to the Clipboard. You can now paste them to a different position. You can also cut & paste device pins from the **Pins and External Wiring** table to the **External Device Connectors** table and vice versa.

> **Tip**
>
> You can select multiple elements by pressing `Ctrl` while selecting.

Related topics

HowTos

How to Group Device Pins (ConfigurationDesk Real-Time Implementation Guide)

References

External Device Connectors Table........ 113
Paste Before (Device Connector)........ 122
Paste Into (Device Connector)........ 124
Pins and External Wiring Table........ 129

Cut (Device Topology)

Purpose

To cut the selected device topology elements and their subelements to the Clipboard.

Result

The selected elements and their subelements are cut to the Clipboard. In the **External Device Browser** or in a table, you can now paste the elements into the device topology.

> **Tip**
>
> You can select multiple elements by pressing `Ctrl` while selecting.

Related topics

Basics

Creating and Extending Device Topologies (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Copy (Device Topology).. 109
Paste Before (Device Topology)... 123
Paste Into (Device Topology)... 124

Delete Custom Property

Purpose

To delete a custom property.

Result

A Remove Elements dialog opens.

Description

Via the Remove Elements dialog, you can remove custom external device ⍰ properties that you previously added. For instructions, refer to How to Delete Custom Properties (ConfigurationDesk Real-Time Implementation Guide 🕮).

Related topics

HowTos

How to Delete Custom Properties (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Create Custom Property.. 110

External Device Browser

Purpose

To display and manage the device topology of your active ConfigurationDesk application.

Description

For more information on the External Device Browser, refer to Basics on Device Topologies (ConfigurationDesk Real-Time Implementation Guide 🕮).

Related topics

Basics

Basics on Device Topologies (ConfigurationDesk Real-Time Implementation Guide 🕮)

External Device Configuration Table

Purpose

To access and configure the most important properties of device topology elements via a table.

Description

For more information, refer to Using Tables to Access and Configure Elements (ConfigurationDesk Real-Time Implementation Guide 🕮).

Related topics

Basics

Specifying the External Device Interface (ConfigurationDesk Real-Time Implementation Guide 🕮)
Using Tables to Access and Configure Elements (ConfigurationDesk Real-Time Implementation Guide 🕮)

External Device Connectors Table

Purpose

To specify the representation of the physical connectors of your external device including the device pin assignment.

Description

For more information on using the External Device Connectors table, refer to How to Group Device Pins (ConfigurationDesk Real-Time Implementation Guide 🕮).

For general information on using tables, refer to Using Tables to Access and Configure Elements (ConfigurationDesk Real-Time Implementation Guide 🕮).

Related topics

HowTos

How to Group Device Pins (ConfigurationDesk Real-Time Implementation Guide 📖)

References

New – Connector	115
New – Multiple Connectors	117
New – Multiple Pins	118
New – Pin	121

Import Devices

Purpose

To access different commands that let you import a device topology.

Result

Several commands for importing a device topology are available in a submenu.

Description

The following commands are available in the submenu:

- Merge / Import Devices – Merge on page 115
- Import / Import Devices – Replace on page 44

Make Resolved

Purpose

To restore an unresolved device topology element of the active ConfigurationDesk application.

Result

The unresolved device topology elements are restored in the device topology. The ❗ symbol next to the unresolved elements disappears.

Description

If you remove device topology elements that are part of device blocks, i.e., they are used in the signal chain of the active ConfigurationDesk application, the elements are marked as unresolved in the device topology. Unresolved device topology elements are displayed with a warning symbol in the **External Device Browser**.

Resolving a subelement of a device topology (device port or device port group) automatically resolves its higher level elements (device and port group) if they are unresolved.

Related topics

Basics

Basics on Device Topologies (ConfigurationDesk Real-Time Implementation Guide)

Merge / Import Devices – Merge

Purpose

To merge a device topology, which is stored in a DTFX or XLSX file, with the device topology of the active ConfigurationDesk application.

Result

ConfigurationDesk opens an Open dialog for you to select a DTFX or XLSX file.

Description

After confirming the dialog by clicking OK, ConfigurationDesk merges the added device topology with the device topology of the active ConfigurationDesk application.

The added device topology elements are displayed in the **External Device Browser** at the bottom of the tree.

Imported elements that already existed in the ConfigurationDesk application are updated.

Related topics

HowTos

How to Merge Device Topologies (ConfigurationDesk Real-Time Implementation Guide)

New – Connector

Purpose

To add a new device connector to the ECU connector representation.

Result

You added a new device connector to the ECU connector representation.

Description

An external device such as an ECU might have several connectors or even connectors subdivided into connector sections. To group pins in connectors that represent the structure of the real device connector, use the Device connector and Device pin elements in the External Device Connectors table.

> **Note**
>
> The names of device connectors and device pins have to be unique in the devices and device connectors.

Related topics

HowTos

How to Group Device Pins (ConfigurationDesk Real-Time Implementation Guide)

References

External Device Connectors Table.. 113
New – Multiple Connectors.. 117
New – Multiple Pins.. 118
New – Pin.. 121

New – Device

Purpose

To add a new external device to the device topology.

Result

You added a new external device with the default configuration to the device topology.

Description

The default name of the device is Device block followed by an index in parentheses.

Related topics

Basics

Creating and Extending Device Topologies (ConfigurationDesk Real-Time Implementation Guide 🕮)

HowTos

How to Import a Device Topology (ConfigurationDesk Real-Time Implementation Guide 🕮)

New – Multiple Connectors

Purpose

To add multiple new device connectors to the ECU connector representation at once.

Result

ConfigurationDesk opens a **Create Multiple Connectors** dialog.

Create Multiple Connectors/Pins dialog

Name pattern Lets you define the name pattern for the new device connectors ⧉ or device pins ⧉. You can use the backslash \ to add an incremented number to the name of each new connector or pin. This avoids duplicate names.

> **Note**
>
> The names of device connectors and device pins have to be unique in the devices and device connectors.

Number of Instances Lets you define the number of device connectors or device pins to add to the **External Device Connectors** table.

Related topics

HowTos

How to Group Device Pins (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

External Device Connectors Table.......... 113
New – Connector.......... 115
New – Multiple Pins.......... 118
New – Pin.......... 121

New – Multiple Devices

Purpose

To add multiple new devices to the device topology at once.

Result

ConfigurationDesk opens a Create Multiple Devices dialog.

Create Multiple Devices/Port Groups/Ports dialog

Name pattern Lets you define the name pattern for the new devices, device port groups, or device ports. You can use the backslash \ as placeholder to add an incremented number to the name of each new device. This avoids duplicate names.

Tip

- The names of devices have to be unique.
- The names of port groups or ports have to be unique within a port group or at the root level of a device.

Number of instances Lets you define the number of devices, port groups, or ports to add to the device topology.

Related topics

HowTos

How to Create and Extend Device Topologies via External Device Browser (ConfigurationDesk Real-Time Implementation Guide)

New – Multiple Pins

Purpose

To add multiple new device pins to the ECU connector representation at once.

Result

ConfigurationDesk opens a Create Multiple Pins dialog.

Create Multiple Connectors/Pins dialog

Name pattern Lets you define the name pattern for the new device connectors or device pins. You can use the backslash \ to add an incremented number to the name of each new connector or pin. This avoids duplicate names.

Note

The names of device connectors and device pins have to be unique in the devices and device connectors.

Number of Instances Lets you define the number of device connectors or device pins to add to the External Device Connectors table.

Related topics

HowTos

How to Group Device Pins (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

New – Multiple Port Groups

Purpose

To add multiple new device port groups to the device topology at once.

Result

ConfigurationDesk opens a Create Multiple Port Groups dialog.

Create Multiple Devices/Port Groups/Ports dialog

Name pattern Lets you define the name pattern for the new devices, device port groups, or device ports. You can use the backslash \ as placeholder to add an incremented number to the name of each new device. This avoids duplicate names.

Tip

- The names of devices have to be unique.
- The names of port groups or ports have to be unique within a port group or at the root level of a device.

Number of instances Lets you define the number of devices, port groups, or ports to add to the device topology.

Related topics

HowTos

How to Create and Extend Device Topologies via External Device Browser (ConfigurationDesk Real-Time Implementation Guide 🕮)

New – Multiple Ports

Purpose

To add multiple new device ports to the device topology at once.

Result

ConfigurationDesk opens a Create Multiple Ports dialog.

Description

The new ports are configured with the default configuration.

Create Multiple Devices/Port Groups/Ports dialog

Name pattern Lets you define the name pattern for the new devices, device port groups, or device ports. You can use the backslash \ as placeholder to add an incremented number to the name of each new device. This avoids duplicate names.

Tip

- The names of devices have to be unique.
- The names of port groups or ports have to be unique within a port group or at the root level of a device.

Number of instances Lets you define the number of devices, port groups, or ports to add to the device topology.

Related topics

HowTos

How to Create and Extend Device Topologies via External Device Browser (ConfigurationDesk Real-Time Implementation Guide 🕮)

New – Pin

Purpose

To add a new device pin to the ECU connector representation.

Result

You added a new device pin to the External Device Connectors table.

Description

An external device such as an ECU might have several connectors or even connectors subdivided into connector sections. To group pins in connectors that represent the structure of the real device connector, use the Device connector ⍰ and Device pin ⍰ elements in the External Device Connectors table.

> **Note**
>
> The names of device connectors and device pins have to be unique in the devices and device connectors.

Related topics

HowTos

How to Group Device Pins (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

External Device Connectors Table..113
New – Connector..115
New – Multiple Connectors..117
New – Multiple Pins..118

New – Port

Purpose

To add a new device port to the device topology.

Result

You added a new device port ⍰ with the default configuration to the device topology ⍰.

Description

The default name is Port followed by an index in parentheses.

Related topics

HowTos

How to Create and Extend Device Topologies via External Device Browser (ConfigurationDesk Real-Time Implementation Guide)

New – Port Group

Purpose

To add a new device port group to the device topology.

Result

You added a new device port group to the device topology.

Description

The default name is Port Group followed by an index in parentheses.

Related topics

HowTos

How to Create and Extend Device Topologies via External Device Browser (ConfigurationDesk Real-Time Implementation Guide)

Paste Before (Device Connector)

Purpose

To paste copied or cut pins and/or device connectors (and the containing pins) before the selected element.

Result

The device pins and/or device connectors are pasted before the selected element. If the elements were copied (not cut), a suffix is added to identify the copy.

Note that copied elements can only be pasted to suitable positions. In the External Device Connectors table, for example, you cannot paste a device connector before a device.

Related topics

HowTos

How to Group Device Pins (ConfigurationDesk Real-Time Implementation Guide)

References

Copy (Device Connector) 108
Cut (Device Connector) 111
External Device Connectors Table 113
Pins and External Wiring Table 129

Paste Before (Device Topology)

Purpose

To paste one or more device topology elements (copied or cut to the Clipboard) before the currently selected device topology element.

Result

You pasted the device topology elements from the Clipboard before the currently selected device topology element. If the elements were copied (not cut), a suffix is added to identify the copy.

Description

Device topology elements must be uniquely addressed in the device topology by port group addresses. If the address of a pasted element is not unique, a conflict is generated. For details on port group addresses, refer to Basics on Device Topologies (ConfigurationDesk Real-Time Implementation Guide).

The configuration of device ports is also pasted, except for **Pins** and **Referenced port(s)** (the **Pins** property is only pasted if you copied a complete device). You have to configure these properties manually. For details on the **Pins** and **Referenced port(s)** properties, refer to Port Properties on page 135.

Related topics

HowTos

How to Copy and Paste Device Topology Elements (ConfigurationDesk Real-Time Implementation Guide)

References

Copy (Device Topology) 109
Cut (Device Topology) 111

Paste Into (Device Connector)

Purpose

To paste copied or cut pins and/or device connectors (and the containing pins) into the selected device or device connector.

Result

The device pins and/or device connectors are pasted into the selected device or device connector. If the elements were copied (not cut), a suffix is added to identify the copy.

You can also paste copied or cut pins from the **Pins and External Wiring** table into the **External Device Connectors** table.

Related topics

HowTos

How to Group Device Pins (ConfigurationDesk Real-Time Implementation Guide)

References

Copy (Device Connector)........ 108
Cut (Device Connector)........ 111
External Device Connectors Table........ 113
Pins and External Wiring Table........ 129

Paste Into (Device Topology)

Purpose

To paste device topology elements (copied or cut to the Clipboard) into the selected device topology element as subelements. You can also paste a copied device into an empty area of the device topology.

Result

You pasted external device topology elements into the device topology. The elements become subelements of the currently selected element. If the elements were copied (not cut), a suffix is added to identify the copy.

Description

Device topology elements must be uniquely addressed in the device topology by port group addresses. If the address of a pasted element is not unique, a conflict is generated. For details on port group addresses, refer to Basics on Device Topologies (ConfigurationDesk Real-Time Implementation Guide).

The configuration of device ports is also pasted, except for **Pins** and **Referenced port(s)** (the **Pins** property is only pasted if you copied a complete device). You have to configure these properties manually. For details on the Pins and Referenced port(s) properties, refer to Port Properties on page 135.

Related topics

HowTos

How to Copy and Paste Device Topology Elements (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Paste Multiple Before (Device Connector)

Purpose

To paste multiple instances of copied pins and/or device connectors (and the containing pins) before the selected element.

Result

ConfigurationDesk opens a Paste Multiple dialog.

Description

The device pins and/or device connectors are pasted before the selected element. Note that copied elements can only be pasted to suitable positions. In the External Device Connectors table, for example, you cannot paste device connectors before a device.

Paste Multiple dialog

Name pattern Lets you define the name pattern for the pasted elements. You can use the backslash \ to add an incremented number to the name of each pasted element. This avoids duplicate names.

Use original names The original names of the copied elements are used for all pasted instances. A suffix and an incremented number are added to identify the copies and to avoid duplicate names.

Number of instances Lets you define the number of pasted elements to add.

Related topics

HowTos

How to Group Device Pins (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Paste Multiple Before (Device Topology)

Purpose

To paste multiple instances of device topology elements (copied to the Clipboard) before the currently selected device topology element.

Result

ConfigurationDesk opens a Paste Multiple dialog.

Description

Device topology elements must be uniquely addressed in the device topology by port group addresses. If the address of a pasted element is not unique, a conflict is generated. For details on port group addresses, refer to Basics on Device Topologies (ConfigurationDesk Real-Time Implementation Guide 🕮).

The configuration of device ports is also pasted, except for Pins and Referenced port(s) (the Pins property is only pasted if you copied a complete device). You have to configure these properties manually. For details on the Pins and Referenced port(s) properties, refer to Port Properties on page 135.

Paste multiple dialog

Name pattern Lets you define the name pattern for the pasted elements. You can use the backslash \ to add an incremented number to the name of each pasted element. This avoids duplicate names.

Use original names The original names of the copied elements are used for all pasted instances. A suffix and an incremented number are added to identify the copies and to avoid duplicate names.

Number of instances Lets you define the number of pasted elements to add to the device topology.

Related topics

HowTos

How to Copy and Paste Device Topology Elements (ConfigurationDesk Real-Time Implementation Guide)

References

Copy (Device Topology).. 109

Paste Multiple Into (Device Connector)

Purpose

To paste multiple instances of copied pins and/or device connectors (and the containing pins) into the selected device or device connector.

Result

ConfigurationDesk opens a Paste Multiple dialog.

Description

The device pins and/or device connectors are pasted into the device or device connector. You can also paste copied pins from the Pins and External Wiring table into the External Device Connectors table.

Paste Multiple dialog

Name pattern Lets you define the name pattern for the pasted elements. You can use the backslash \ to add an incremented number to the name of each pasted element. This avoids duplicate names.

Use original names The original names of the copied elements are used for all pasted instances. A suffix and an incremented number are added to identify the copies and to avoid duplicate names.

Number of instances Lets you define the number of pasted elements to add.

Related topics

HowTos

How to Group Device Pins (ConfigurationDesk Real-Time Implementation Guide)

References

Copy (Device Connector).. 108
External Device Connectors Table.. 113
Pins and External Wiring Table.. 129

Paste Multiple Into (Device Topology)

Purpose

To paste multiple instances of device topology elements (copied to the Clipboard) into the currently selected device topology element as subelements. You can also paste multiple copied devices into an empty area of the device topology.

Result

ConfigurationDesk opens a Paste Multiple dialog.

Description

Device topology elements must be uniquely addressed in the device topology by port group addresses. If the address of a pasted element is not unique, a conflict is generated. For details on port group addresses, refer to Basics on Device Topologies (ConfigurationDesk Real-Time Implementation Guide 🕮).

The configuration of device ports is also pasted, except for Pins and Referenced port(s) (the Pins property is only pasted if you copied a complete device). You have to configure these properties manually. For details on the Pins and Referenced port(s) properties, refer to Port Properties on page 135.

Paste multiple dialog

Name pattern Lets you define the name pattern for the pasted elements. You can use the backslash \ to add an incremented number to the name of each pasted element. This avoids duplicate names.

Use original names The original names of the copied elements are used for all pasted instances. A suffix and an incremented number are added to identify the copies and to avoid duplicate names.

Number of instances Lets you define the number of pasted elements to add to the device topology.

Related topics

HowTos

How to Copy and Paste Device Topology Elements (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Copy (Device Topology)..109

Pins and External Wiring Table

Purpose

To access the external wiring information.

Description

For more information on using the Pins and External Wiring table, refer to How to View the Wiring Information (ConfigurationDesk Real-Time Implementation Guide 🕮).

For general information on using tables, refer to Using Tables to Access and Configure Elements (ConfigurationDesk Real-Time Implementation Guide 🕮).

Related topics

Basics

Calculating an External Cable Harness (ConfigurationDesk Real-Time Implementation Guide 🕮)

Select Associated Device Port

Purpose

To select and show device ports to which specific pins are assigned.

Result

ConfigurationDesk selects and shows you the assigned device ports in the External Device Browser.

Related topics

Basics

Basics on Device Pin Assignment (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Select Associated Pins.. 129

Select Associated Pins

Purpose

To select and show pins that are assigned to specific device ports in the Pins and External Wiring table.

Result

ConfigurationDesk selects and shows you the assigned pins in the Pins and External Wiring table.

Related topics

Basics

Basics on Device Pin Assignment (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Select Associated Device Port... 129

Select Pins in Device Connectors Table

Purpose

To select and show pins that are assigned to specific device ports in the External Device Connectors table.

Result

ConfigurationDesk selects and shows you the assigned pins in the External Device Connectors table.

Related topics

Basics

Basics on Device Pin Assignment (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Select Associated Device Port... 129

External Device Properties

Objective

The Properties Browser and the External Device Configuration table let you display and configure elements of the external device topology.

Where to go from here

Information in this section

Device Properties

Objective

Let you configure and display properties belonging to the selected device.

Name

Lets you change the name of the device to fit your own requirements.

Value / Range	▪ All characters are allowed, including * ? | < > : / " . These are the exceptions: ▪ The semicolon ; and backslash \ are not allowed. ▪ Number of characters: no limit. ▪ The default name is **Device**.
Value description	The name is used and displayed in the following elements: ▪ **External Device Browser** ▪ Device block in working views ▪ Tables ▪ **Properties Browser** ▪ Information messages, error messages and conflicts ▪ Exported ConfigurationDesk files If you use a device name that is already used within the device topology, ConfigurationDesk generates a conflict which is displayed in the **Conflicts Viewer**. Duplicate name conflicts within the device topology can lead to signals having the same identifier in the FIU configuration data.

	This will cause an error in the Failure Simulation module of the experiment software. To avoid this, resolve all duplicate name conflicts before executing the build process.
Dependencies	—

Type

Lets you select the device type.

Value / Range	▪ ECU ▪ Load
Value description	▪ ECU is recommended for all external devices except external loads. ▪ Load is recommended for external loads.
Dependencies	▪ The Load rejection port property is not available for ECU devices. ▪ The Load rejection port property is available only for Load devices.

Background color

Lets you change the background color for the selected device block.

Value / Range	▪ Light blue ▪ White ▪ Red ▪ Green ▪ Blue ▪ Cyan ▪ Magenta ▪ Yellow ▪ Gray ▪ Orange ▪ Dark green ▪ Beige
Value description	—
Dependencies	—

Description

Lets you enter a string-based description to document your device.

Value / Range	▪ All characters are allowed, including * ? | < > : / \ " . ▪ Number of characters: no limit.
Value description	—
Dependencies	—

Custom properties

Lets you display user properties you added via Create Custom Property in the External Device Browser.

Value / Range	▪ All characters are allowed, including * ? | < > : / \ " . ▪ Number of characters: no limit.
Value description	You can use custom properties to document specific characteristics of device topology elements or to define sort keys for tables. For instructions on adding custom properties, refer to How to Add Custom Device Properties (ConfigurationDesk Real-Time Implementation Guide). For instructions on deleting custom properties, refer to How to Delete Custom Properties (ConfigurationDesk Real-Time Implementation Guide).
Dependencies	Entered custom properties are displayed and configurable in the Properties Browser.

Related topics

Basics

Basics on Configuring External Devices (ConfigurationDesk Real-Time Implementation Guide)

HowTos

How to Add Custom Device Properties (ConfigurationDesk Real-Time Implementation Guide)
How to Delete Custom Properties (ConfigurationDesk Real-Time Implementation Guide)

Port Group Properties

Objective

Let you configure and display properties belonging to the selected port group. Port groups let you group device ports by functionality and/or the location in the external device connector.

Name

Lets you change the name of the port group to fit your own requirements.

Value / Range	▪ All characters, including * ? | < > : / ", are allowed, with two exceptions: The semicolon ; and the backslash \ are not allowed. ▪ Number of characters: no limit. ▪ The default name is Port Group.

Value description	The name is used and displayed in the following elements: - **External Device Browser** - Device block in working views - Tables - **Properties Browser** - Information messages, error messages and conflicts - Exported ConfigurationDesk files You can rename the port group and nest port groups in port groups. The port group name is a part of the port group address. It must be unique within a device. Duplicate name conflicts within the device topology can lead to signals having the same identifier in the FIU configuration data. This will cause an error in the Failure Simulation module of the experiment software. To avoid this, resolve all duplicate name conflicts before executing the build process. For details on port group addresses, refer to Basics on Device Topologies (ConfigurationDesk Real-Time Implementation Guide).
Dependencies	—

Description

Lets you enter a string-based description to document the selected port group.

Value / Range	- All characters are allowed, including * ? | < > : / \" . - Number of characters: no limit.
Value description	—
Dependencies	—

Custom properties

Lets you display user properties you added via Create Custom Property in the External Device Browser.

Value / Range	- All characters are allowed, including * ? | < > : / \ " . - Number of characters: no limit.
Value description	You can use custom properties to document specific characteristics of device topology elements or to define sort keys for tables. For instructions on adding custom properties, refer to How to Add Custom Device Properties (ConfigurationDesk Real-Time Implementation Guide). For instructions on deleting custom properties, refer to How to Delete Custom Properties (ConfigurationDesk Real-Time Implementation Guide).
Dependencies	Entered custom properties are displayed and configurable in the Properties Browser.

Related topics

Basics

Basics on Configuring External Devices (ConfigurationDesk Real-Time Implementation Guide 📖)

Port Properties

Objective

Let you configure and display properties belonging to the selected device port.

Name

Lets you change the name of the port to your own requirements.

Value / Range	▪ All characters, including * ? | < > : / ", are allowed, with two exceptions: The semicolon ; and backslash \ are not allowed. ▪ Number of characters: no limit. ▪ The default name is Port.
Value description	The name is used and displayed in the following elements: ▪ **External Device Browser** ▪ Device block in working views ▪ Tables ▪ **Properties Browser** ▪ Information messages, error messages and conflicts ▪ Exported ConfigurationDesk files Device topologies are hierarchical, and ports are addressed via the device name and the port group. ConfigurationDesk checks the new name and generates a conflict if it does not uniquely identify the port. The conflict is displayed in the **Conflicts Viewer**. Duplicate name conflicts within the device topology can lead to signals having the same identifier in the FIU configuration data. This will cause an error in the Failure Simulation module of the experiment software. To avoid this, resolve all duplicate name conflicts before executing the build process.
Dependencies	—

Port type

Lets you select the port type. Port types define the signal direction of a port and are the basis for device port mapping.

Value / Range	▪ In ▪ Out ▪ Bidirectional

	▪ Reference ▪ Unspecified
Value description	▪ In: The data direction or current flow goes from the real-time platform to the external device port. ▪ Out: The data direction or current flow goes from the external device port to the real-time platform. ▪ Bidirectional: This port is independent of a data direction or current flow. Bidirectional ports are used to implement bus communication, etc. ▪ Reference: A specific port which represents the reference signal of any other ports. For example: With differential signals, this is the inverted signal. With single-ended signals, this is the ground signal (GND). ▪ Unspecified: The device port specifics are undefined. It is not recommended to use this port type because the system cannot assist you in device port mapping.
Dependencies	▪ In ConfigurationDesk, the port type is the main aid to mapping. ▪ If you extend the signal chain via context menu, the port type also determines which function blocks you can select.

Pins

Lets you assign device pins to the selected device port to specify the pins of the external device. You can also assign several pins to one device port. However, one external device pin cannot be assigned to more than one device port.

Value / Range	▪ All characters are allowed, including * ? | < > : / " . This is the exception: ▪ The characters ; and \ are not allowed. ▪ Number of characters: no limit.
Value description	Device pin addresses are organized according to the following scheme (relative address): <pin group name (layer 1)>\ ... \<pin group name (layer n)> \<pin name> . For example, **Engine Chamber\Temp1** stands for pin group **Engine Chamber** and pin **Temp1**. This helps you to group your device pins by their location in the connector. For details, refer to Basics on Configuring External Devices (ConfigurationDesk Real-Time Implementation Guide 🕮).
Dependencies	—

For instructions on device pin assignment, refer to How to Assign Device Ports to External Device Pins (ConfigurationDesk Real-Time Implementation Guide 🕮).

Electrical characteristics

Lets you enter a string-based description to document the electrical characteristics of the selected port.

Value / Range	- All characters are allowed, including * ? | < > : / \" . - Number of characters: no limit.
Value description	—
Dependencies	—

Physical attributes

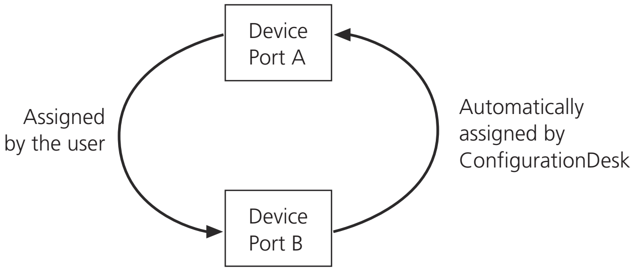
Lets you select the physical attributes of the port.

Value / Range	- Voltage - Current - Resistance - Unspecified - Bus
Value description	- Voltage: The port handles mainly voltage signals, e.g. a switch input signal. - Current: The port handles mainly current signals, e.g. a current sink for a throttle. - Resistance: The port handles mainly resistance signals, e.g. an input for a resistance sensor. - Unspecified: The physical attribute is undefined. - Bus: The port handles mainly bus signals. ConfigurationDesk uses physical attributes together with port types to help you add I/O functions via device ports.
Dependencies	If you extend the signal chain via context menu, the physical attributes narrow down the number of selectable function blocks.

Referenced port(s)

Displays and lets you assign device ports which belong to the same signal.

Value / Range	All other device ports from the same device are available for selection.
Value description	The selected port(s) are assigned to the currently selected port as reference port(s).

	ConfigurationDesk automatically assigns the ports in both directions as shown below.  Device Port A Assigned by the user Automatically assigned by ConfigurationDesk Device Port B For instructions, refer to How to Assign Reference Ports (ConfigurationDesk Real-Time Implementation Guide 📖).
Dependencies	You can only assign device ports of the same device.

Failure simulation

Lets you enable failure simulation for the selected port. This gives you access to the failure classes, which you can set to allowed or not allowed.

Value / Range	▪ Disabled ▪ Enabled
Value description	If set to **Enabled**, failure simulation is enabled for the selected port, and the failure classes are available.
Dependencies	—

Open circuit

Lets you allow the failure class to be simulated via the experiment software.

Value / Range	▪ Not allowed ▪ Allowed
Value description	If set to **Allowed**, a disconnection between the external device and the real-time hardware can be simulated.
Dependencies	The property is configurable only if the **Failure simulation** property is set to **Enabled**.

Short to GND

Lets you allow the failure class to be simulated via the experiment software.

Value / Range	▪ Not allowed ▪ Allowed
Value description	If set to **Allowed**, a short circuit to the ground potential of the real-time hardware can be simulated.
Dependencies	The property is configurable only if the **Failure simulation** property is set to **Enabled**.

Short to VBAT

Lets you allow the failure class to be simulated via the experiment software.

Value / Range	▪ Not allowed ▪ Allowed
Value description	If set to Allowed, a short circuit to the battery potential of the real-time hardware can be simulated.
Dependencies	The property is configurable only if the Failure simulation property is set to Enabled.

Short to signal generation channel

Lets you allow the failure class to be simulated via the experiment software.

Value / Range	▪ Not allowed ▪ Allowed
Value description	If set to Allowed, a short circuit between the device port and a signal generation channel can be simulated.
Dependencies	The property is configurable only if the Failure simulation property is set to Enabled.

Short to signal measurement channel

Lets you allow the failure class to be simulated via the experiment software.

Value / Range	▪ Not allowed ▪ Allowed
Value description	If set to Allowed, a short circuit between the device port and a signal measurement channel can be simulated.
Dependencies	The property is configurable only if the Failure simulation property is set to Enabled.

Short to bus channel

Lets you allow the failure class to be simulated via the experiment software.

Value / Range	▪ Not allowed ▪ Allowed
Value description	If set to Allowed, a short circuit between the device port and a bus channel can be simulated.
Dependencies	The property is configurable only if the Failure simulation property is set to Enabled.

Load rejection

Lets you protect sensitive loads against damage by automatically disconnecting them when a failure such as a short circuit is simulated.

Value / Range	- Enforced - Not enforced
Value description	If set to **Enforced**, load rejection is enforced for the device port. For details, refer to Basics on Load Rejection (ConfigurationDesk Real-Time Implementation Guide 🕮).
Dependencies	This property is available only if device type is set to **Load**.

Load description

Lets you enter a string-based description for the expected load (internal and/or external) connected to the external device port.

Value / Range	- All characters are allowed, including * ? | < > : / \ " . - Number of characters: no limit.
Value description	For example, you could describe the resistance value (for example, **100 Ohm**) or use inventory numbers.
Dependencies	—

Description

Lets you enter a string-based description to document the selected port.

Value / Range	- All characters are allowed, including * ? | < > : / \ " . - Number of characters: no limit.
Value description	—
Dependencies	—

Custom properties

Lets you display user properties you added via **Create Custom Property** in the **External Device Browser**.

Value / Range	- All characters are allowed, including * ? | < > : / \ " . - Number of characters: no limit.
Value description	You can use custom properties to document specific characteristics of device topology elements or to define sort keys for tables. For instructions on adding custom properties, refer to How to Add Custom Device Properties (ConfigurationDesk Real-Time Implementation Guide 🕮). For instructions on deleting custom properties, refer to How to Delete Custom Properties (ConfigurationDesk Real-Time Implementation Guide 🕮).
Dependencies	Entered custom properties are displayed and configurable in the **Properties Browser**.

Related topics

Basics

Basics on Configuring External Devices (ConfigurationDesk Real-Time Implementation Guide 🕮)
Handling Loads (ConfigurationDesk Real-Time Implementation Guide 🕮)
Specifying Failure Simulation in ConfigurationDesk (ConfigurationDesk Real-Time Implementation Guide 🕮)

HowTos

How to Add Custom Device Properties (ConfigurationDesk Real-Time Implementation Guide 🕮)
How to Add Function Blocks to the Signal Chain via Device Ports (ConfigurationDesk Real-Time Implementation Guide 🕮)
How to Assign Device Ports to External Device Pins (ConfigurationDesk Real-Time Implementation Guide 🕮)

Function Block Management

Objective

Function blocks are used to add I/O functionality to the signal chain.

Where to go from here

Information in this section

Information in other sections

Implementing I/O Functionality (ConfigurationDesk Real-Time Implementation Guide 📖)

Assign

Purpose

To execute a default command for hardware resource assignment or to access a submenu with different hardware resource assignment commands.

Description

The following commands are available:

- Assign First Fit on page 146

Tip

This command is the default command that is executed if you click the main icon part of the split button. It is also available in the submenu.

- Assign Channel Set – <Channel Set> on page 146
- Delete Assignment (Channel Set) on page 152

Assign Channel Set – <Channel Set>

Purpose

To assign a specific channel set to function blocks.

Result

ConfigurationDesk assigns a channel set to the function block and channels to all generated channel requests. The assigned channel set is marked with a checkmark to the left.

Description

The list of channel sets that is offered only contains sets which match specific basic requirements and do not cause a conflict. The order of the list depends on the order of the hardware topology.

Related topics

References

Assign First Fit..146

Assign Channels

Purpose

To assign channels of a channel set to a function block.

Result

ConfigurationDesk assigns the next possible channel(s) of the hardware resource to the function block.

Assign First Fit

Purpose

To assign the first suitable channel set and channels to the function block.

Result

ConfigurationDesk assigns the first suitable channel set to the function block and channels to all generated channel requests.

Related topics

Basics

Resolving Conflicts (ConfigurationDesk Real-Time Implementation Guide 🕮)

Clear Channels

Purpose

To delete only the assignment of channels to the selected function block(s).

Result

ConfigurationDesk resets the channels. The channel set assignment remains unchanged.

Configure Block-Commutated Sector Signals

Purpose

To open a dialog for the configuration of sector and stationary signals.

Result

ConfigurationDesk opens the Sector and Stationary Signals Configuration dialog.

Sector and Stationary Signals Configuration dialog

For more information, refer to Configuring Sector and Stationary Signals (Block-Commutated PWM Out) (ConfigurationDesk I/O Function Implementation Guide 🕮).

Related topics

Basics

Configuring Sector and Stationary Signals (Block-Commutated PWM Out) (ConfigurationDesk I/O Function Implementation Guide 🕮)

Configure VLAN

Purpose

To access different commands to open dialogs for VLAN configuration.

Result

Different commands to open dialogs for VLAN configuration are available in a submenu.

Description

The commands available in the submenu are:

- Configure VLAN – Port-based VLAN on page 148 (only available after hardware resources were assigned)
- Configure VLAN – Protocol-based VLAN on page 148

Configure VLAN – Port-based VLAN

Purpose

To open a dialog for the configuration of the Ethernet traffic routing based on the defined interconnection of the Ethernet switch ports.

Result

ConfigurationDesk opens the **Port-Based VLAN Configuration** dialog.

Port-Based VLAN Configuration dialog

For more information, refer to Configuring the Basic Functionality (Ethernet Switch) (ConfigurationDesk I/O Function Implementation Guide 🕮).

Related topics

Basics

Configuring the Basic Functionality (Ethernet Switch) (ConfigurationDesk I/O Function Implementation Guide 🕮)

Configure VLAN – Protocol-based VLAN

Purpose

To open a dialog for the configuration of the Ethernet traffic routing based on the VLAN IDs of the Ethernet frames.

Result

ConfigurationDesk opens the **Protocol-Based VLAN Configuration** dialog.

Protocol-Based VLAN Configuration dialog

For more information, refer to Configuring VLANs Based on the Protocol (Ethernet Switch) (ConfigurationDesk I/O Function Implementation Guide 🕮).

Related topics

Basics

Configuring VLANs Based on the Protocol (Ethernet Switch) (ConfigurationDesk I/O Function Implementation Guide 🕮)

Copy (Function Blocks)

Purpose

To copy selected function blocks to the Clipboard.

Result

The selected function blocks are copied to the Clipboard.

Tip

You can select multiple function blocks by pressing `Ctrl` while selecting.

Description

You can now do one of the following:

- Paste the function blocks to a working view once or multiple times. This automatically creates new instances of the according function block types and adds them to the signal chain.
- Add the function blocks to a working view. This only works if they are not already part of the current working view.

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Add to Working View	257
Paste Into (Signal Chain)	241
Paste Multiple Into (Signal Chain)	243

Create Custom Function Code

Purpose

To create template source code (CPP) and header (H) files for a ConfigurationDesk custom function block type.

Result

First a **Save a Custom Function Block Header File** dialog opens for you to save the custom function header file template. Then a **Save a Custom Function Block Cpp File** dialog opens for you to save the custom function source code file template.

Description

The template files are always created for the selected custom function block type.

For more information on custom function blocks, refer to ConfigurationDesk Custom I/O Function Implementation Guide 🕮.

Save a Custom Function Block <xxx> File dialogs

You can select the file name and location for the header file and the source code file.

> **Note**
>
> The default location is the custom function directory, which already contains XML and code files for custom functions. If you overwrite previously modified files in this directory, your changes will be lost.

Related topics

Basics

ConfigurationDesk Custom I/O Function Implementation Guide

References

Create Custom Function Type Definition

Purpose

To create a template header file (H file) containing the type definitions for a ConfigurationDesk custom function block type.

Result

A **Save a Custom Function Block Type Definition File** opens for you to save the custom function type definition file template.

Description

The template file is always created for the selected custom function block type.

For more information on custom function blocks, refer to ConfigurationDesk Custom I/O Function Implementation Guide.

Save a Custom Function Block Type Definition File

You can select the file name and location for the type definition file.

> **Note**
>
> The default location is the custom function directory, which already contains XML and code files for custom functions. If you overwrite previously modified files in this directory, your changes will be lost.

> **Tip**
>
> The custom function type definition file is automatically created during a build process.

Related topics

Basics

ConfigurationDesk Custom I/O Function Implementation Guide

References

Create Custom Function Code..149

Create Updated Custom Function Type XML

Purpose

To update a custom function XML file to the current XML schema version.

Result

The **Generate a Custom Function Type XML File** dialog opens.

Description

With ConfigurationDesk version 5.3, a new XML schema was implemented for custom function blocks (refer to `<InstallationFolder>\ConfigurationDesk\Implementation\UserFiles\CustomFunctionTypeSchema_V3_1.xsd`). ConfigurationDesk lets you update custom function files from the last schema version (introduced with ConfigurationDesk version 4.3) to the new schema via the **Create Updated Custom Function Type XML** command.

Note

You cannot downgrade an updated custom function file to the previous schema version. You are recommended to back up the existing files before the update.

Generate a Custom Function Type XML File dialog

You can select the file name and location for the updated custom function XML file.

Delete Assignment (Channel Set)

Purpose

To delete the channel set assignment of the selected function block(s).

Result

ConfigurationDesk deletes the channel set assignment and the entered channels of the function block(s).

Filter for Supported Functions / Filter for Functions Supported by Hardware Topology

Purpose

To display only function block types in the function library hierarchy that are supported by the hardware topology of the active ConfigurationDesk application.

Tip

This filter is active by default.

Result

This filter causes only function block types and their higher-level elements in the function library hierarchy that support the current hardware topology to be displayed. In context menus, the active filter is indicated by a checkmark to the left. It is also highlighted on the **Home** ribbon.

Description

The filter affects the **Function Browser** or each table separately.

Related topics

Basics

Using Display Filters (ConfigurationDesk Real-Time Implementation Guide)

Filter for Used Functions

Purpose

To display only function blocks used in the signal chain (in the table window with their subelements) and their higher-level elements (e.g., the function block types) in the function library hierarchy.

Result

If you set this filter, only function blocks used in the signal chain and their higher-level elements in the function library hierarchy are displayed. In context menus, the active filter is indicated by a checkmark to the left. It is also highlighted on the **Home** ribbon.

Description

The filter affects the **Function Browser** or each table separately.

Related topics

Basics

Using Display Filters (ConfigurationDesk Real-Time Implementation Guide)

Function Browser

Purpose

To display the function library and to instantiate function blocks from function block types.

Description

The **Function Browser** displays the function library in a hierarchical tree structure. Function block types are grouped in function classes. Instantiated function blocks are added below the corresponding function block type.

> **Note**
>
> The function library allows access to the I/O functionality provided by the software installation on your host PC. Thus, the availability of function block types depends only on the software installation and not on the configuration of your ConfigurationDesk application.

The **Function Browser** allows you to:

- Display function block types.
- Instantiate a function block from a function block type.
- Access further commands via context menu (see below), for example, to assign hardware resources to instantiated function blocks.

Symbols

The function library is displayed in the **Function Browser** with the following symbols:

Symbol	Description
📁	Indicates a function class.
🗗	Indicates a function block type.
▣	Indicates a function block instance.
🔗	Indicates an instantiated function block that is used in the active ConfigurationDesk application.

Hardware Assignment

Purpose

To manage the assignment of hardware resources to the function block.

The following context menu items are available:

- Assign First Fit on page 146
- Assign Channel Set – <Channel Set> on page 146
- Assign Channels on page 146
- Delete Assignment (Channel Set) on page 152
- Clear Channels on page 147

Related topics

References

Assign First Fit... 146

Import Custom Function from Archive

Purpose

To import one or several custom function type(s) from an archive file.

Result

ConfigurationDesk opens an Open dialog for you to select the archive file to be imported.

Description

A custom function block can be used to include user-specific code in a ConfigurationDesk application. A custom function consists of an XML file, two header files (.h files) and a C++ source code file (CPP file).

The submenu of the Import Custom Function from Archive command offers two options:

To Project Folder Select this options to import the custom function files to the project-specific custom functions directory `<DocumentsFolder>\<Project>\CustomFunctions`.

To Search Path Select this option to import the custom function files to the global custom functions directory. The default global custom functions directory is: `<DocumentsFolder>\UserFiles`. You can find and configure the global custom functions directory on the Configuration page of the ConfigurationDesk Options dialog.

Open dialog

Lets you select the archive file to be imported.

Related topics

Basics

ConfigurationDesk Custom I/O Function Implementation Guide

References

Import ECU Interface Container

Purpose

To import an ECU interface container (EIC) file to the selected ECU Interface Configuration function block.

Result

A dialog opens for you to select the EIC file you want to import to the function block.

Description

Each **ECU Interface Configuration** function block that is available in the signal chain requires exactly one EIC interface container (EIC) file. An EIC file must be unique in the active ConfigurationDesk application, i.e., you can import each EIC file only once.

The imported EIC file is copied to the file system of ConfigurationDesk. This copy is used in the ConfigurationDesk application, i.e., once imported you do not need the original EIC file anymore. When you back up the ConfigurationDesk project, the copied EIC file is automatically included in the generated ZIP file.

After you import an EIC file to an **ECU Interface Configuration** function block, you can implement ECU interfacing in ConfigurationDesk. For more information, refer to ECU Interfacing with SCALEXIO or MicroAutoBox III Systems (ConfigurationDesk Real-Time Implementation Guide).

Related topics

Basics

ECU Interfacing with SCALEXIO or MicroAutoBox III Systems (ConfigurationDesk Real-Time Implementation Guide)

References

Update ECU Interface Container.. 163

New – <Function Block>

Purpose

To add a new function block to the ConfigurationDesk application in the global working view.

Result

You added a function block to the ConfigurationDesk application.

Description

On adding the function block, ConfigurationDesk:

- Instantiates a new function block.

  That means: The function block is added to the **Function Browser** and to the signal chain in the global working view.

- Renames the added function block by using the default function block name of the function block with an increased index.

New – Multiple <Function Block>

Purpose

To add multiple new function blocks to the ConfigurationDesk application in the global working view.

Result

ConfigurationDesk opens a dialog to enter the number of function blocks you want to add.

Description

On adding the function blocks, ConfigurationDesk:

- Instantiates the new function blocks.

  That means: The function blocks are added to the **Function Browser** and to the signal chain in the global working view.
- Renames the added function blocks according to the name pattern in the <Function Block> dialog.

Create Multiple <Function Block> dialog

Name pattern Lets you define the name pattern for the new function blocks. You can use the backslash \ as placeholder to add an incremented number to the name of each new function block. This avoids duplicate names.

> **Tip**
>
> The names of function blocks have to be unique.

Number of Instances Lets you define the number of function blocks to add to the Function Browser and the signal chain in the global working view.

Open Equivalent Circuit Diagram

Purpose

To show circuit diagrams with details of the electrical characteristics of the real-time hardware resources.

Result

ConfigurationDesk opens an Equivalent Circuit Diagram dialog which provides use-case-dependent circuit diagrams for different channel types to help you choose a suitable channel set of the respective type in the function block configuration.

Equivalent Circuit Diagram dialog

Lets you select a circuit diagram from a drop-down list. The drop-down list provides use-case-dependent circuit diagrams for different channel types.

Opacity Lets you set the opacity of the Equivalent Circuit Diagram window as follows:

- Slide to right: The opacity of the Equivalent Circuit Diagram window increases. Elements in the window become more visible, elements behind the window become less visible.
- Slide to left: The opacity of the Equivalent Circuit Diagram window decreases. Elements in the window become less visible, elements behind the window become more visible.

Note

The opacity setting takes effect when you click once outside the Equivalent Circuit Diagram window.

Related topics

HowTos

How to View Circuit Diagrams of Hardware Resources (ConfigurationDesk Real-Time Implementation Guide 📖)

Reload Custom Function Definitions

Purpose

To update all static custom function block types and their instances.

Result

All static custom function types and their instances are updated. Custom function types for which files were added to the project-specific or the global custom functions directory are added to the function library.

Description

A custom function block can be used to include user-specific code in a ConfigurationDesk application. A custom function consists of an XML file, two header files (.h files) and a C++ source code file (CPP file).

After adding, modifying, or replacing a custom function XML file, you must update the custom function block type in ConfigurationDesk.

Note

After modifying the XML file, you must also increment the `Version` of the `CustomFunctionBlock` element in it.

Tip

To update specific custom function types, use the Reload Definition for '<CustomFunctionType(s)>' on page 159 command.

Related topics

Basics

Basics on Implementing Custom Function Blocks (ConfigurationDesk Custom I/O Function Implementation Guide 📖)

References

Reload Definition for '<CustomFunctionType(s)>'

Purpose

To update one or more selected static custom function block type(s) and its/their instances.

Result

The selected static custom function block type(s) are updated.

Description

A custom function block can be used to include user-specific code in a ConfigurationDesk application. A custom function consists of an XML file, two header files (.h files) and a C++ source code file (CPP file).

After adding, modifying, or replacing a custom function XML file, you must update the custom function block type in ConfigurationDesk.

Note

After modifying the XML file, you must also increment the `Version` of the `CustomFunctionBlock` element in it.

The **Reload Definition** command allows you to update the custom function block type without having to close the project or ConfigurationDesk. All instances of the function block type are also updated automatically.

Tip

- You can select several custom function block types and update them at once (Reload Definitions for Selected Custom Function Types).
- You can update all static custom function block types using the Reload Custom Function Definitions on page 158 command.

Related topics

Basics

File Types and Directories for Custom Function Blocks (ConfigurationDesk Custom I/O Function Implementation Guide 📖)

References

Configuration Page	480
Create Custom Function Code	149
Create Custom Function Type Definition	150
Import Custom Function from Archive	155
Reload Custom Function Definitions	158

Reload Ini File from Archive

Purpose

To update an FPGA custom function block type and all instances.

Result

ConfigurationDesk opens the Reload custom function type from archive dialog for selecting the modified or the older version of the FPGA model INI file to be reloaded.

Description

FPGA custom function blocks contain the functionality of an FPGA application that must be defined with the RTI FPGA Programming Blockset.

If you copy a modified FPGA model INI file to the project-specific `CustomFunctions` folder, you must reload the FPGA model INI file in ConfigurationDesk.

All FPGA model INI files that reside in the project-specific `CustomFunctions` folder will be available in the selection.

If you export the FPGA model INI file via the FPGA_SETUP_BL block dialog, the export mechanism copies the FPGA model INI file to the `CustomFunctions` folder of the selected project.

If you import the FPGA model INI file via ConfigurationDesk, the FPGA model INI file is available for the Reload custom function type from archive dialog after you copied the file to the `CustomFunctions` folder.

Reload custom function type from archive dialog

Lets you select the archive file to be reloaded. The file must be located in the project-specific custom function folder.

The project-specific `CustomFunctions` folder is:

```
<ProjectRootDirectory>\<Project>\CustomFunctions
```

Related topics

HowTos

How to Update FPGA Custom Function Blocks (ConfigurationDesk I/O Function Implementation Guide 🕮)

Select Assigned Hardware Channel

Purpose

To select and show hardware channels that are assigned to specific function blocks.

Result

ConfigurationDesk selects and shows you the assigned hardware channels in the Hardware Resource Browser.

Related topics

HowTos

How to Select and Show Assigned Elements (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Select Assigned Function.. 71

Set to Channel Type Default Values

Purpose

To set the property settings of the selected function block(s) to default values, which are based on the channel type of the assigned hardware resource(s).

Result

ConfigurationDesk sets the property settings of the selected function block(s) to default values, which are based on the channel type of the assigned hardware resources. Note, that every channel type provides its own default values.

The hardware resource assignment is preserved and therefore not cleared.

Setting properties to default values can affect the number and availability of ports at the function block. However, if ports are already mapped, they are not deleted. A mapped port continues to exist until you remove its mapping line(s).

Show Model

Purpose

To show the selected FPGA custom function block in the corresponding FPGA model.

Result

If not already open, MATLAB and the corresponding FPGA model open. The **FPGA SETUP_BL** block is highlighted in the FPGA model. If the FPGA model of the selected FPGA custom function block is no longer available or has been moved to a another folder, an error message is displayed in MATLAB.

Note

MATLAB/Simulink must be started with the Xilinx System Generator, otherwise the Xilinx System Generator Blockset cannot be displayed correctly.

Related topics

HowTos

How to Open the FPGA Model in MATLAB/Simulink (ConfigurationDesk I/O Function Implementation Guide 📖)

Start FlexRay Configuration Tool

Purpose

To start the FlexRay Configuration Tool and open an existing configuration project from within ConfigurationDesk to view and manage the FlexRay configuration.

Result

The **Open Project** dialog opens for you to select an existing FlexRay configuration project (*.prj). Then the FlexRay Configuration Tool starts with the selected project.

Description

If you already opened a FlexRay configuration project via this command for the FlexRay function block, the FlexRay Configuration Tool starts with the previously selected project without displaying the selection dialog again.

Update ECU Interface Container

Purpose

To update the ECU interface container (EIC) file in the selected **ECU Interface Configuration** function block.

Result

A dialog opens for you to select the EIC file you want to use for updating the EIC file in the function block.

Description

If an ECU interface container changed after you imported the container to an **ECU Interface Configuration** function block and a new version of the related EIC file is available, you can update the EIC file in the function block. If you do so, the function block's model interface, the I/O events, and the requirements regarding the ECU interface are derived from the new EIC file:

- Elements that are unambiguously identified in both EIC files are updated in the function block.
- Elements that are available only in the new EIC file are added to the function block.
- Elements that are available only in the old EIC file become obsolete. Depending on the elements (e.g., function ports, I/O events) and their configuration in the signal chain, they are deleted or become unresolved.

> **Tip**
>
> If you work in the **Temporary** or a user-defined working view, not all update effects might be displayed. Open the **Global** working view to view all update effects.

For more information, refer to Basics on Updating ECU Interface Containers in ConfigurationDesk Applications (ConfigurationDesk Real-Time Implementation Guide).

Related topics

Basics

ECU Interfacing with SCALEXIO or MicroAutoBox III Systems (ConfigurationDesk Real-Time Implementation Guide 🕮)

Model Interface Management

Objective

ConfigurationDesk lets you specify the model interface via model port blocks.

Where to go from here

Information in this section

Information in other sections

Specifying the Model Interface (ConfigurationDesk Real-Time Implementation Guide 📖)

Model Topology

Objective

To implement a real-time application in ConfigurationDesk, you need a model topology which contains information on the interface to the behavior model.

Where to go from here

Information in this section

Add Model

Purpose

To add a model topology, a Simulink model, Simulink implementation container, bus simulation container, V-ECU implementation, or FMU to the active ConfigurationDesk application.

Description

ConfigurationDesk opens the Add Model dialog.

Add Model/Import Model Topology dialog

Application process options Lets you specify the assignment of the selected model implementations to the created application processes. You can select one of the following options:

- Create one preconfigured application process for each model (default).

  If you select this option, ConfigurationDesk creates a separate application process for each model implementation that is to be added to the active ConfigurationDesk application.
- Create one application process for all models and optimize configuration.

  If you select this option, ConfigurationDesk creates one application process and assigns all the selected model implementations to it. ConfigurationDesk optimizes the configuration of the new application process by grouping runnable functions in tasks, and specifying the task priorities. The default name of the new application process is ApplicationProcess_<n>. n is an increasing number.
- Create no application process.

  If you select this option, ConfigurationDesk adds the selected model implementations to the ConfigurationDesk application without creating application processes.

If the executable application does not contain a processing unit application, the preconfigured application process is created in a new processing unit application. If the executable application contains exactly one processing unit application, the preconfigured application process is created in the existing processing unit application.

Note

For the Create one preconfigured application process for each model option and the Create one application process for all models and optimize configuration option the following applies:

- For Simulink models, these options are available only if you select the Analyze model (including task information) checkbox. If the model analysis fails, the preconfigured application process is not created.
- These option are not available if the executable application contains more than one processing unit application.

Add model Lets you select a model implementation or a model topology that you want to add to the active ConfigurationDesk application. If you already opened behavior models in Simulink, you can select them directly from the Models open in Simulink list. You can also click Add model from file. If you do, a standard Open dialog opens for you to select the file type and the file you want to import. You can select one of the following file types:

- Simulink model (`*.slx`;`*.mdl`)

Tip

- It can be useful to save the referenced Simulink model file directly to a project folder that is configured to display MDL/SLX files. Refer to Add Folder on page 29. You can use the folder to save data belonging to your project, such as model files necessary for implementing your real-time application .
- You find a Models folder with an appropriate configuration in the project folder by default.

- Simulink implementation container (`*.sic`)
- Bus simulation container (`*.bsc`)
- V-ECU implementation container (`*.vecu`)
- Functional Mock-up Unit (`*.fmu`)
- (Not available in the Add Model dialog) Model communication description file (`*.mcd`)
- (Not available in the Add Model dialog) Model topology file (`*.mtfx`)

List of selected model(s) Lists the selected model(s) that you want to add to the ConfigurationDesk application. If you click OK, all the selected models are added to the ConfigurationDesk application at once.

The following options are visible if you select one or more models that you want to add to the ConfigurationDesk application. You can specify them separately for each selected model.

Analyze model (including task information) (Only available for MDL, SLX, MTFX, and MCD files) Lets you analyze the model interface of your behavior model. If you select this checkbox, a complete analysis of the selected model is performed, and the changes are made available in ConfigurationDesk. For more information, refer to Analyze Simulink Model (Including Task Information) on page 171.

> **Note**
>
> If the model analysis fails, no behavior model is added to the ConfigurationDesk application. A dialog displays error messages and refers to the MATLAB Command Window which provides more information on the errors that have occurred in the Simulink behavior models. While you correct the errors in the behavior models, the **Add Model** dialog remains open and keeps its configuration. Then, you can add the behavior models to the ConfigurationDesk application again without having to select them again.

Model initialization command (Only available for Simulink models and MCD files) Lets you enter a MATLAB command that is executed before the behavior model is opened by ConfigurationDesk, for example, to prepare data in the MATLAB Base Workspace that is referenced by the behavior model.

ConfigurationDesk lets you specify locations of scripts, data, etc. via relative paths in the **Model initialization command** edit field. This enables you to transfer the ConfigurationDesk project to another path. Refer to How to Back up and Transfer a Project (ConfigurationDesk Real-Time Implementation Guide 🕮). For this purpose, ConfigurationDesk provides the following path macros:

Path Macro Name	Description
`%ProjectRoot%`	Expands to the root directory of the current project.
`%ApplicationRoot%`	Expands to the root directory of the currently active ConfigurationDesk application.
`%ModelRoot%`	Expands to the directory the selected Simulink model is stored in.

Result

ConfigurationDesk adds the model topology information to the active ConfigurationDesk application. The model topology is also displayed at the bottom of the hierarchical structure in the **Model Browser**.

Analyze

Purpose

To execute a default command for analyzing a behavior model or to access a submenu with different commands for this purpose.

Description

The following commands are available:

- Analyze Simulink Model (Including Task Information) on page 171

> **Tip**
>
> This command is the default command that is executed if you click the main icon part of the split button. It is also available in the submenu.

- Analyze Simulink Model (Model Interface Only) on page 172

Analyze Simulink Model (Including Task Information)

Purpose

To perform a complete model analysis for the selected behavior model.

Result

If MATLAB is not already open, it is started, the behavior model opens, and the model interface and the elements provided by the behavior model are analyzed. By analogy with the **Analyze Simulink Model (Model Interface Only)** on page 172 command, the values of the model port block and model port properties specified in the behavior model are transferred to ConfigurationDesk. Existing values of the corresponding properties specified in ConfigurationDesk are overwritten. Additionally, all predefined elements provided by the behavior model are analyzed and displayed in the **Components** folder of the executable application:

- Runnable functions
- Events
- Tasks

The model topology of your active ConfigurationDesk application is updated with the properties of the analyzed behavior model. Model port blocks and model ports which do not have an identity are automatically given one. For details on ID handling, refer to Basics on Model Port Block IDs and Signal IDs (Model Interface Package for Simulink - Modeling Guide 🕮).

Note

- If you have connected several MATLAB installations to your dSPACE installation, you must select one of them as the preferred connection. For details, refer to Introduction to Connecting a MATLAB Installation (Managing dSPACE Software Installations 🕮).
- In the following case, the results of the model analysis and the code generation differ:
  - The signal of an In Bus Element block is specified by a Simulink.Bus object.
  - The In Bus Element block is connected to a Bus Selector block that selects a subset of the bus signals.
  - No other signal is used.

  In the above case, a model port block with the complete bus hierarchy is displayed in ConfigurationDesk after a model analysis. However, after code generation, a model port block with only those bus signals that are selected by the Bus Selector block is displayed in ConfigurationDesk.

Description

The Analyze Simulink Model (Including Task Information) command analyzes not only the model topology but also the elements (tasks, events, runnable functions) a behavior model provides. You can use these elements to model your executable application (= real-time application). For details, refer to Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide 🕮).

Related topics

Basics

Analyzing Simulink Behavior Models (ConfigurationDesk Real-Time Implementation Guide 🕮)
Data Interchange Between ConfigurationDesk and Behavior Model (ConfigurationDesk Real-Time Implementation Guide 🕮)
Handling the Model Interface (ConfigurationDesk Real-Time Implementation Guide 🕮)
Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide 🕮)
Transferring the ConfigurationDesk Model Interface to a New Simulink Interface Model (ConfigurationDesk Real-Time Implementation Guide 🕮)

Analyze Simulink Model (Model Interface Only)

Purpose

To analyze the model interface of a behavior model and make changes available in ConfigurationDesk.

Result

If MATLAB is not already open, it is started, the behavior model opens and its interface is analyzed. The model topology of your active ConfigurationDesk application is updated with the properties of the analyzed behavior model. Model port blocks and model ports which do not have an identity are automatically given one. For details on ID handling, refer to Basics on Model Port Block IDs and Signal IDs (Model Interface Package for Simulink - Modeling Guide 🕮).

> **Note**
>
> - If you connected several MATLAB installations to your dSPACE installation, you must select one of them as the preferred connection. For details, refer to Introduction to Connecting a MATLAB Installation (Managing dSPACE Software Installations 🕮).
> - Performing an analysis of a model that contains root-level In Bus Element and Out Bus Element blocks requires an initialization of the model. Therefore, an analysis of the model interface via the **Analyze Simulink Model (Model Interface Only)** command is not possible in this case.

Description

The values of the model port block and model port properties specified in the behavior model are transferred to ConfigurationDesk. Existing values of the corresponding properties in ConfigurationDesk are discarded.

Depending on the modifications in your behavior model, the display of model port blocks in ConfigurationDesk can change. For details, refer to Effects of Changing Model Interfaces of Simulink Behavior Models (ConfigurationDesk Real-Time Implementation Guide 🕮).

If you analyze the interface of a behavior model that uses blocks from the RTI CAN MultiMessage Blockset or the RTI LIN MultiMessage Blockset, ConfigurationDesk creates one Configuration Port block for each identified **ControllerSetup** block. If you analyze the interface of a behavior model that uses blocks from the automatically generated FlexRay model, ConfigurationDesk creates one Configuration Port block for each identified **FLEXRAYCONFIG UPDATE** block in the Simulink model. The created Configuration Port blocks are displayed in the **Model Browser** and can be dragged to a working view, where you can map them to the respective **CAN**, **LIN**, or **FlexRay** function blocks. For further information, refer to the relevant topic:

- Building the Signal Chain for CAN Bus Communication (ConfigurationDesk Real-Time Implementation Guide 🕮)
- Building the Signal Chain for LIN Bus Communication (ConfigurationDesk Real-Time Implementation Guide 🕮)
- Building the Signal Chain for FlexRay Communication (ConfigurationDesk Real-Time Implementation Guide 🕮)

Related topics

Basics

Analyzing Simulink Behavior Models (ConfigurationDesk Real-Time Implementation Guide 📖)
Data Interchange Between ConfigurationDesk and Behavior Model (ConfigurationDesk Real-Time Implementation Guide 📖)
Handling the Model Interface (ConfigurationDesk Real-Time Implementation Guide 📖)
Transferring the ConfigurationDesk Model Interface to a New Simulink Interface Model (ConfigurationDesk Real-Time Implementation Guide 📖)

Create Inverse Block

Purpose

To create inverse model port blocks prepared for model communication.

Result

For each selected model port block ⍰, a new model port block is created in the signal chain ⍰. The created model port blocks have the following characteristics:

- They have the same names as the original model port blocks.
- They have the same port groups and port names as the original model port blocks.
- They have the inverse data direction of the original model port blocks.
- If the original model port blocks are unmapped, the new inverse model port blocks are automatically mapped to the original model port blocks. Otherwise, the inverse model port blocks are unmapped, so you have to map them manually.
- They are unresolved.

Related topics

Basics

Simplified Preparation of Model Interfaces for Model Communication (ConfigurationDesk Real-Time Implementation Guide 📖)

HowTos

How to Create Inverse Model Port Blocks for Model Communication (ConfigurationDesk Real-Time Implementation Guide 📖)

Delete Connected Function and Model Port Blocks (Including Simulink)

Purpose

To delete the selected function blocks or model port blocks, including the blocks that are mapped to them, in ConfigurationDesk. The representations of the affected model port blocks in the Simulink behavior model are deleted as well.

Result

Depending on the selected elements, ConfigurationDesk performs the following actions:

Selected Element	Result
Function block	ConfigurationDesk deletes the function block and all the model port blocks that are mapped to it. If Runnable Function blocks are deleted, the tasks to which the affected runnable function and I/O event are assigned are deleted as well if no other elements are assigned to them.
Model port block	ConfigurationDesk deletes the model port block and all the function blocks that are mapped to the model port block. If the model port block is a Runnable Function block, the task to which the related runnable function and I/O event are assigned is deleted as well if no other elements are assigned to it. Other model port blocks that are mapped to the affected function blocks are not deleted.

If not already open, MATLAB and the corresponding Simulink model open. Simulink deletes the model port blocks representing the model port blocks that are deleted in the ConfigurationDesk model interface.

Both model interfaces are synchronized, thus it is not necessary to analyze the Simulink model in ConfigurationDesk or to propagate the changes from ConfigurationDesk to the Simulink model.

Note

You cannot undo this command in Simulink.

Related topics

Basics

Data Interchange Between ConfigurationDesk and Behavior Model (ConfigurationDesk Real-Time Implementation Guide 📖)
Generate Simulink Model Interface – Selected Blocks – <Model Name>.................................. 183
Handling the Model Interface (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Generate Simulink Model Interface - Selected Blocks - New Model.. 184
Propagate to ConfigurationDesk Model Interface.. 188
Propagate to Simulink Model.. 189

Generate New Simulink Model Interface

Purpose

To generate a new Simulink interface model that contains newly created model port blocks.

Result

If not already open, MATLAB opens. Simulink generates a new interface model that contains newly created model port blocks. The model port blocks are created according to the selected elements in ConfigurationDesk for which the command is executed:

Selected Element	Result
Function block	ConfigurationDesk creates new model port blocks for all function ports and event ports of the function block, regardless of whether the ports are already mapped to model port blocks. The newly created model port blocks are mapped to the function block and propagated to the generated Simulink interface model. The number and structure of the created model port blocks depend on the setting specified for the **Model port block structure** property of the related function block type.
Model port block	ConfigurationDesk propagates only the model port block to the generated Simulink interface model.

Structure information, such as the organization of blocks in subsystems, is included in the generation of the Simulink interface model. The new model port blocks are automatically initialized with the default values for all MATLAB-specific properties (e.g., sample time) defined in the **Model Interface Blockset**.

Note

- This command always creates a new interface model that contains the created model port blocks.
- If you connected several MATLAB installations to your dSPACE installation, you must select one of them as the preferred connection. For more information, refer to Introduction to Connecting a MATLAB Installation (Managing dSPACE Software Installations 🕮).

Related topics

Basics

Data Interchange Between ConfigurationDesk and Behavior Model (ConfigurationDesk Real-Time Implementation Guide 📖)
Handling the Model Interface (ConfigurationDesk Real-Time Implementation Guide 📖)
Transferring the ConfigurationDesk Model Interface to a New Simulink Interface Model (ConfigurationDesk Real-Time Implementation Guide 📖)

HowTos

How to Transfer Unresolved Model Port Blocks to a Simulink Behavior Model via an Interface Model (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Function Block Type Properties (ConfigurationDesk Function Block Properties 📖)
Generate Simulink Model Interface – All Unresolved Blocks – New Model.................................179
Generate Simulink Model Interface - Selected Blocks - New Model...184

Generate Simulink Model Interface

Purpose

To access different commands for generating the model interface.

Result

Different commands for generating the model interface are available in a submenu.

Description

Depending on the context, the commands available in the submenu are:

- Generate Simulink Model Interface – All Unresolved Blocks – <Model Name> on page 178
- Generate Simulink Model Interface – All Unresolved Blocks – New Model on page 179
- Generate Simulink Model Interface – Selected Blocks – <Model Name> on page 183
- Generate Simulink Model Interface - Selected Blocks - New Model on page 184

Generate Simulink Model Interface – All Unresolved Blocks – <Model Name>

Purpose

To generate all unresolved model port blocks into the selected Simulink model.

Result

If not already open, MATLAB and the selected Simulink model open. All the unresolved model port blocks are generated into the selected target Simulink model. Each model port block is generated into the subsystem hierarchy specified by its full path in the hierarchy of the model topology, including the model which is currently providing the model port block. If a model port block is not part of the model topology (because it was generated via the **Extend Signal Chain - Create Suitable Model Port Block** command), it is generated into the root level of the target Simulink model. The new model port blocks are automatically initialized with the default values for all MATLAB-specific properties (for example, sample time) defined in the **Model Interface Blockset**. After the model port blocks are generated into the selected Simulink model, ConfigurationDesk automatically executes the **Analyze Simulink Model (Model Interface Only)** command. The generated model port blocks are then available in the model topology that is displayed in the **Model Browser**.

Note

- This command is only available if at least one Simulink model is added to the executable application.
- If the target Simulink model already contains a model port block with the same block identifier as a block which has been selected for this operation, the latter block is ignored and an informative message is displayed in the MATLAB Command Window.
- If the target Simulink model already contains a block with the same full name (i.e., including the subsystem path) as a block selected for this operation, the latter block is renamed in order to make the name unique.

Related topics

Basics

Data Interchange Between ConfigurationDesk and Behavior Model (ConfigurationDesk Real-Time Implementation Guide)
Handling the Model Interface (ConfigurationDesk Real-Time Implementation Guide)
Transferring the ConfigurationDesk Model Interface to a New Simulink Interface Model (ConfigurationDesk Real-Time Implementation Guide)

HowTos

How to Transfer Unresolved Model Port Blocks to a Simulink Behavior Model via an Interface Model (ConfigurationDesk Real-Time Implementation Guide)

References

Analyze Simulink Model (Model Interface Only).. 172
Generate Simulink Model Interface – All Unresolved Blocks – New Model................................. 179

Generate Simulink Model Interface – All Unresolved Blocks – New Model

Purpose

To generate all unresolved blocks into a new interface model.

Result

If not already open, MATLAB opens. A new model, called the interface model, is generated. The interface model contains all the unresolved blocks. Structure information such as the organization of blocks in subsystems is included in the generation. The new model port blocks are automatically initialized with the default values for all MATLAB-specific properties (for example, sample time) defined in the **Model Interface Blockset**.

Note

- This command always creates a new interface model that contains the created model port blocks.
- If you connected several MATLAB installations to your dSPACE installation, you must select one of them as the preferred connection. For more information, refer to Introduction to Connecting a MATLAB Installation (Managing dSPACE Software Installations).

Description

Unresolved blocks are blocks which are not contained in the behavior model that is linked to your ConfigurationDesk application.

Next step

You can copy model port blocks and their identities from the interface model and paste them into an existing behavior model. For details, refer to Adding the Generated Model Interface to Your Behavior Model via an Interface Model (Model Interface Package for Simulink - Modeling Guide).

Related topics

Basics

Data Interchange Between ConfigurationDesk and Behavior Model (ConfigurationDesk Real-Time Implementation Guide)
Handling the Model Interface (ConfigurationDesk Real-Time Implementation Guide)
Transferring the ConfigurationDesk Model Interface to a New Simulink Interface Model (ConfigurationDesk Real-Time Implementation Guide)

HowTos

How to Transfer Unresolved Model Port Blocks to a Simulink Behavior Model via an Interface Model (ConfigurationDesk Real-Time Implementation Guide)

References

Generate Simulink Model Interface - Selected Blocks - New Model.. 184

Generate Simulink Model Interface - Runnable Function Blocks and Tasks - <Application Process Name>

Purpose

To create preconfigured tasks for the function block's I/O events in the selected application process and to create related **Runnable Function** blocks in a new Simulink interface model.

Result

If not already open, MATLAB opens. A new interface model is generated. It contains a **Runnable Function** block that triggers a function-call subsystem for each I/O event provided by the function block. The new **Runnable Function** block is configured with the default values for all properties defined in the **Model Interface Blockset**. The name of the **Runnable Function** block and the runnable function is set to `<FunctionBlockName>\<eventName>`.

In ConfigurationDesk, a new task is created in the selected application process. The task has a runnable function and the I/O event assigned. However, the runnable function assigned to the task is unresolved.

Next steps

The runnable function assigned to the new task is initially unresolved. You must perform the following steps to resolve the runnable function and therefore be able to use it in your ConfigurationDesk application:

1. Copy the **Runnable Function** block and the function-call subsystem from the interface model to your behavior model. For details, refer to Adding the Generated Model Interface to Your Behavior Model via an Interface Model (Model Interface Package for Simulink - Modeling Guide).
2. Perform a complete model analysis of the behavior model containing the **Runnable Function** block and the function-call subsystem. For details, refer to Analyze Simulink Model (Including Task Information) on page 171.

Related topics

References

Generate Simulink Model Interface - Runnable Function Blocks and Tasks - In Related Application Process

Purpose

To create preconfigured tasks for the function block's I/O events in the application process that the function block was assigned to via model port mapping, and to create related **Runnable Function** blocks in a new Simulink interface model.

Result

If not already open, MATLAB opens. A new interface model is generated. The interface model contains a **Runnable Function** block that triggers a function-call subsystem for each I/O event provided by the function block. The new **Runnable Function** block is configured with the default values for all properties defined in the **Model Inteface Blockset**. The name of the **Runnable Function** block and the runnable function is set to `<FunctionBlockName>\<eventName>`.

In ConfigurationDesk, a new task is created in the related application process. The task has the runnable function and the I/O event assigned to it. However, the runnable function assigned to the task is unresolved.

Next steps

The runnable function assigned to the new task is initially unresolved. You must perform the following steps to resolve the runnable function and therefore be able to use it in your ConfigurationDesk application:

1. Copy the **Runnable Function** block and the function-call subsystem from the interface model to your behavior model. For details, refer to Adding the Generated Model Interface to Your Behavior Model via an Interface Model (Model Interface Package for Simulink - Modeling Guide).
2. Perform a complete model analysis of the behavior model containing the **Runnable Function** block and the function-call subsystem. For details, refer to Analyze Simulink Model (Including Task Information) on page 171.

Related topics

References

Generate Simulink Model Interface - Runnable Function Blocks and Tasks - <Application Process Name>........ 180
Generate Simulink Model Interface - Runnable Function Blocks and Tasks - Runnable Function Blocks only........ 182

Generate Simulink Model Interface - Runnable Function Blocks and Tasks - Runnable Function Blocks only

Purpose

To generate an interface model containing **Runnable Function** blocks.

Result

If not already open, MATLAB opens. A new interface model is generated. The interface model contains a **Runnable Function** block that triggers a function-call subsystem. The new **Runnable Function** block is configured with the default values for all properties defined in the **Model Interface Blockset**. The name of the **Runnable Function** block and the runnable function is set to `<FunctionBlockName>\<eventName>`.

Next step

You must perform the following steps to be able to use the runnable function for modeling an asynchronous task in the ConfigurationDesk application:

1. Copy the **Runnable Function** block and (optionally) the function-call subsystem to your behavior model.
2. In ConfigurationDesk, execute the **Analyze Simulink Model (Including Task Information)** command from the context menu of the behavior model you copied the **Runnable Function** block to.

   The runnable function is now available for task modeling in the **Executable Application** table view.

Related topics

References

Generate Simulink Model Interface – Selected Blocks – <Model Name>

Purpose

To generate the selected model port blocks into the selected Simulink model.

Result

If not already open, MATLAB and the selected Simulink model open. All the selected model port blocks are generated into the selected target Simulink model. Each model port block is generated into the subsystem hierarchy specified by its full path in the hierarchy of the model topology, including the model which is currently providing the model port block. If a model port block is not part of the model topology (because it was generated via the **Extend Signal Chain - Create Suitable Model Port Block** command), it is generated into the root level of the target Simulink model. The new model port blocks are automatically initialized with the default values for all MATLAB-specific properties (for example, sample time) defined in the **Model Interface Blockset**. After the model port blocks are generated into the selected Simulink model, ConfigurationDesk automatically executes the **Analyze Simulink Model (Model Interface Only)** command. The generated model port blocks are then available in the model topology that is displayed in the **Model Browser**.

Note

- This command is only available if at least one Simulink model is added to the executable application.
- If the target Simulink model already contains a model port block with the same block identifier as a block which has been selected for this operation, the latter block is ignored and an informative message is displayed in the MATLAB Command Window.
- If the target Simulink model already contains a block with the same full name (i.e., including the subsystem path) as a block selected for this operation, the latter block is renamed in order to make the name unique.

Related topics

Basics

Data Interchange Between ConfigurationDesk and Behavior Model (ConfigurationDesk Real-Time Implementation Guide 🕮)
Handling the Model Interface (ConfigurationDesk Real-Time Implementation Guide 🕮)
Transferring the ConfigurationDesk Model Interface to a New Simulink Interface Model (ConfigurationDesk Real-Time Implementation Guide 🕮)

HowTos

How to Transfer Unresolved Model Port Blocks to a Simulink Behavior Model via an Interface Model (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Analyze Simulink Model (Model Interface Only).. 172
Generate Simulink Model Interface – All Unresolved Blocks – New Model........................ 179

Generate Simulink Model Interface - Selected Blocks - New Model

Purpose

To generate selected blocks into a new interface model.

Result

If not already open, MATLAB opens. A new model, called the interface model, is generated. The interface model contains all the selected model port blocks ⧉. Structure information such as the organization of blocks in subsystems is included in the generation. The new model port blocks are automatically initialized with the default values for all MATLAB-specific properties (for example, sample time) defined in the **Model Interface Blockset**.

Note

- This command always creates a new interface model that contains the created model port blocks.
- If you connected several MATLAB installations to your dSPACE installation, you must select one of them as the preferred connection. For more information, refer to Introduction to Connecting a MATLAB Installation (Managing dSPACE Software Installations 🕮).

Next step

You can copy model port blocks and their identities from the interface model and paste them into an existing behavior model ⧉. For details, refer to Adding the Generated Model Interface to Your Behavior Model via an Interface Model (Model Interface Package for Simulink - Modeling Guide 🕮).

Related topics

Basics

Data Interchange Between ConfigurationDesk and Behavior Model (ConfigurationDesk Real-Time Implementation Guide 🕮)
Handling the Model Interface (ConfigurationDesk Real-Time Implementation Guide 🕮)
Transferring the ConfigurationDesk Model Interface to a New Simulink Interface Model (ConfigurationDesk Real-Time Implementation Guide 🕮)

HowTos

How to Transfer Unresolved Model Port Blocks to a Simulink Behavior Model via an Interface Model (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Generate Simulink Model Interface – All Unresolved Blocks – New Model.............................. 179

Import (Model Topology)

Purpose

To access different commands that let you import a model topology.

Result

Several commands for importing a model topology are available in a submenu.

Description

The following commands are available in the submenu:

- Add Model on page 168
- Import / Replace Topology (Model Topology) on page 46

Model Browser

Purpose

To display and manage the model topology of an active ConfigurationDesk application.

Description

The **Model Browser** provides access to all the model port blocks available in the behavior model which is linked to a ConfigurationDesk application. The model elements are displayed in a hierarchy, starting with the model root. Below that are all the subsystems of the behavior model which contain model port blocks. The model port blocks themselves are also displayed.

You can drag model port blocks from the **Model Browser** to a working view. The model port blocks automatically become a part of your real-time application and are used for code generation.

Model port blocks which are used in an application can be easily removed from it if they are no longer needed. You can delete them by right-clicking them in the **Model Browser** and selecting **Delete from Application** from the context menu.

Symbols

The model topology is displayed with the following symbols:

Symbol	Description
▣	Indicates a Simulink behavior model.
▣	Indicates a Simulink implementation container.
▣	Indicates a bus simulation container.
▬	Indicates a V-ECU implementation.
fmu	Indicates a Functional Mock-up Unit.
⊞	Indicates a model subsystem.
▶	Indicates a data port block with data inports.
▶	Indicates a data port block with data outports.
▣	Indicates a Configuration Port block.
▶	Indicates a data inport.
▶	Indicates a structured data inport.
▶	Indicates a data outport.
▶	Indicates a structured data outport.
▶	Indicates a configuration port.
⛓	▪ Indicates a model that is assigned to an application process. or ▪ Indicates a model port block that is used in your active ConfigurationDesk application.
⛓!	▪ Indicates a model which is missing in the model topology but from which model port blocks are still used in the ConfigurationDesk application. or ▪ Indicates a model port block that is unresolved. This means that it is used in your active ConfigurationDesk application but is not contained in the behavior model which is linked to your application.

Symbol	Description
⋈	Indicates a model port block which resides with the same identity in two or more behavior models which are linked to the active ConfigurationDesk application and are assigned to application processes. If the block is part of the signal chain, the model port block is considered as an unresolved block.
🗎	Indicates data inconsistencies between the local copy of a V-ECU implementation container and the file system.

Related topics

Basics

Handling the Model Interface (ConfigurationDesk Real-Time Implementation Guide 🕮)

Model-Function Mapping Browser

Purpose

To simplify the work with Simulink behavior models.

Description

The Model-Function Mapping Browser in ConfigurationDesk and the commands for the remote access to ConfigurationDesk in Simulink behavior models allow a convenient connection of the Simulink behavior models to I/O functionality in ConfigurationDesk. For more information on working with the Model-Function Mapping Browser, refer to User-Friendly Connection of ConfigurationDesk and Simulink Models (ConfigurationDesk Real-Time Implementation Guide 🕮).

Open Model in Simulink

Purpose

To open a Simulink model in MATLAB/Simulink.

Description

The Simulink model is opened in MATLAB/Simulink. If MATLAB is not already open, it is started. If a model initialization command is specified for the Simulink model, it is executed before the Simulink model is opened.

Related topics

Basics

Handling MATLAB via ConfigurationDesk (ConfigurationDesk Real-Time Implementation Guide 🕮)

HowTos

How to Initialize Simulink Behavior Models (ConfigurationDesk Real-Time Implementation Guide 🕮)

Propagate

Purpose

To execute a default command for propagating changes from the ConfigurationDesk model interface to the Simulink behavior model or to access a submenu with different commands for this purpose.

Description

The following commands are available:

- Propagate to Simulink Model on page 189

  Tip

  This command is the default command that is executed if you click the main icon part of the split button. It is also available in the submenu.

- Propagate to ConfigurationDesk Model Interface on page 188
- Generate New Simulink Model Interface on page 176

Propagate to ConfigurationDesk Model Interface

Purpose

To update model port blocks and create new model port blocks for unmapped or partially mapped function blocks.

Result

ConfigurationDesk updates the model port blocks that are mapped to the selected function block and/or creates new model port blocks:

- If model port blocks that are mapped to the function block have unmapped model ports, these model ports are deleted.

- If function ports of the function block are not mapped yet, they are mapped to model ports. New model port blocks are created and model port blocks that are already mapped to the selected function block are updated with new model ports, depending on the following data:
  - The setting specified for the **Model port block structure** property of the related function block type.
  - The functions of the function block to which the unmapped function ports belong.
  - The port type of the unmapped function ports.
- If event ports of the function block are not mapped yet, new Runnable Function blocks are created and mapped to the event ports.

Related topics

Basics

Data Interchange Between ConfigurationDesk and Behavior Model (ConfigurationDesk Real-Time Implementation Guide 🕮)
Generate Simulink Model Interface – All Unresolved Blocks – <Model Name>........................ 178
Generate Simulink Model Interface – Selected Blocks – <Model Name>................................ 183
Handling the Model Interface (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Delete Connected Function and Model Port Blocks (Including Simulink)................................ 175
Function Block Type Properties (ConfigurationDesk Function Block Properties 🕮)
Generate New Simulink Model Interface... 176
Propagate to Simulink Model.. 189

Propagate to Simulink Model

Purpose

To update the ConfigurationDesk model interface according to the selected elements and propagate the changes to the Simulink behavior model in order to keep the model interfaces synchronized.

Result

If not already open, MATLAB and the corresponding Simulink behavior model open.

Depending on the selected element, ConfigurationDesk performs the following actions:

Selected Element	Result
Function block	ConfigurationDesk updates the model port blocks that are mapped to the function block and/or creates new model port blocks: ▪ If model port blocks that are mapped to the function block have unmapped model ports, these model ports are deleted. ▪ If function ports of the function block are not mapped yet, they are mapped to model ports. New model port blocks are created and model port blocks that are already mapped to the selected function block are updated with new model ports, depending on the following data: ▪ The setting specified for the **Model port block structure** property of the related function block type. ▪ The functions of the function block to which the unmapped function ports belong. ▪ The port type of the unmapped function ports. ▪ If event ports of the function block are not mapped yet, new Runnable Function blocks are created and mapped to the event ports. The related runnable functions and I/O events are assigned to tasks. If required, new tasks are created for this purpose. Then, all the model port blocks that are mapped to the function block are propagated to the Simulink model.
Model port block	▪ If the model port block is mapped to a function block, the ConfigurationDesk model interface is updated according to the related function block (as described above). Then, all the model port blocks that are mapped to the function block are propagated to the Simulink model interface. ▪ If the model port block is not mapped to a function block, only the model port block is propagated to the Simulink model interface.
Subsystem	The model port blocks that belong to the subsystem are propagated to the Simulink model interface.
Model	All the model port blocks that belong to the model are propagated to the Simulink model interface.

Note

The following applies if you use the **Propagate to Simulink Model** command:

- You cannot undo the changes in the Simulink model. To prevent a propagate operation from accidentally changing parts of a Simulink model, you can protect specific subsystems by setting them to read-only in Simulink. When analyzed, these subsystems are marked with a lock symbol in ConfigurationDesk.
- Possible formatting of the model port blocks in Simulink, e.g., the color, is lost.
- This command cannot be used for Configuration Port blocks.

Related topics

Basics

Data Interchange Between ConfigurationDesk and Behavior Model (ConfigurationDesk Real-Time Implementation Guide 🕮)

Generate Simulink Model Interface – Selected Blocks – <Model Name>.................................. 183

Handling the Model Interface (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Function Block Type Properties (ConfigurationDesk Function Block Properties 🕮)

Generate Simulink Model Interface – All Unresolved Blocks – New Model.............................. 179

Generate Simulink Model Interface - Selected Blocks - New Model... 184

Propagate to ConfigurationDesk Model Interface.. 188

Reload

Purpose

To make modifications on a V-ECU implementation container, an FMU, or a Simulink implementation container known to ConfigurationDesk.

Result

The Simulink implementation container, Functional Mock-up Unit, or V-ECU implementation used in your ConfigurationDesk application is updated with its latest modifications.

Description

Updating Simulink implementation containers If there are any modifications in the Simulink implementation container (SIC file ⍰), you can update it in ConfigurationDesk. This is necessary if a new Simulink implementation container was generated because the model was changed in Simulink. The **Reload** command deletes the Simulink implementation container from the ConfigurationDesk application ⍰ and reads and copies it again from its original location, and adds it to the ConfigurationDesk application.

Updating FMUs If there are any modifications in the FMU (FMU file ⍰), you can update it in ConfigurationDesk. This is necessary if a new FMU was generated because the model was changed in the modeling tool. The **Reload** command deletes the FMU from the ConfigurationDesk application and reads and copies it from its original location again.

Updating V-ECU implementations When you import a V-ECU implementation container, ConfigurationDesk generates a local copy of the V-ECU implementation container in its file system. To make modifications on the V-ECU implementation container known to ConfigurationDesk, you must update the V-ECU implementation in your ConfigurationDesk application. The **Reload** command tries to find the V-ECU implementation container at a file location relative to the ConfigurationDesk application's root directory. If that is not successful, ConfigurationDesk tries to find the V-ECU implementation container at the file location it was originally imported from. ConfigurationDesk then updates the data in the file system and in the ConfigurationDesk application with the data from the V-ECU implementation container. If the V-ECU implementation cannot be updated successfully, the model topology remains unchanged.

> **Note**
>
> In certain situations, data inconsistencies between the file system and the local copy of the V-ECU implementation container may occur. To avoid data loss, you must familiarize yourself with the reasons for data inconsistencies and how you can correct them. For details, refer to Updating V-ECU Implementations Used in a ConfigurationDesk Application (ConfigurationDesk Real-Time Implementation Guide).

Related topics

Basics

Updating FMUs in ConfigurationDesk (ConfigurationDesk Real-Time Implementation Guide)
Updating Simulink Implementation Containers in ConfigurationDesk (ConfigurationDesk Real-Time Implementation Guide)
Updating V-ECU Implementations Used in a ConfigurationDesk Application (ConfigurationDesk Real-Time Implementation Guide)

Replace Model

Purpose

To replace a resolved model implementation in the model topology with a another model implementation.

Description

ConfigurationDesk opens the Replace Model dialog.

Replace Model dialog

Add model Lets you select only one model implementation to replace the selected model implementation. If you already opened behavior models in Simulink, you can select a model directly from the Models open in Simulink list. You can also click Add model from file. If you do, a standard Open dialog opens for you to select the file type and the file you want to import. You can select one of the following file types:

- Simulink model (`*.slx`;`*.mdl`)

  Tip

  - It can be useful to save the referenced Simulink model file directly to a project folder that is configured to display MDL/SLX files. Refer to Add Folder on page 29. You can use the folder to save data belonging to your project, such as model files necessary for implementing your real-time application.
  - You find a **Models** folder with an appropriate configuration in the project folder by default.

- Simulink implementation container (`*.sic`)
- Bus simulation container (`*.bsc`)
- V-ECU implementation container (`*.vecu`)
- Functional Mock-up Unit (`*.fmu`)

Analyze model (including task information) (Available only for Simulink models) Lets you analyze the model interface of your behavior model. If you select this checkbox, a complete analysis of the selected model is performed, and the changes are made available in ConfigurationDesk. For more information, refer to Analyze Simulink Model (Including Task Information) on page 171.

Note

If the model analysis fails, the old behavior model is not replaced by a new one. A dialog displays error messages and refers to the MATLAB Command Window which provides more information on the errors that occurred in the Simulink behavior model. While you correct the errors in the behavior models, the **Replace Model** dialog remains open and keeps its configuration. Afterwards, you can replace the old model with the new one without having to select the new model again.

Model initialization command (Available only for Simulink models) Lets you enter a MATLAB command that is executed before the behavior model is opened by ConfigurationDesk, for example, to prepare data in the MATLAB Base Workspace that is referenced by the behavior model.

ConfigurationDesk lets you specify locations of scripts, data, etc. via relative paths in the **Model initialization command** edit field. This enables you to transfer the ConfigurationDesk project to another path. Refer to How to Back up and Transfer a Project (ConfigurationDesk Real-Time Implementation Guide 📖). For this purpose, ConfigurationDesk provides the following path macros:

Path Macro Name	Description
`%ProjectRoot%`	Expands to the root directory of the current project.
`%ApplicationRoot%`	Expands to the root directory of the currently active ConfigurationDesk application.

Path Macro Name	Description
`%ModelRoot%`	Expands to the directory the selected Simulink model is stored in.

Result

If you click **OK**, the old model implementation is removed from the model topology and replaced by the new one. If the old model implementation was assigned to an application process, this assignment is deleted and the new model implementation is assigned to the application process.

Related topics

Basics

Replacing Model Implementations (ConfigurationDesk Real-Time Implementation Guide 🕮)

Show in Simulink Model

Purpose

To show the selected model port block in the corresponding behavior model.

Result

If not already open, MATLAB and the corresponding behavior model open. The selected model port block is highlighted in the behavior model. If the selected element is no longer available in the behavior model or has been moved to a different subsystem, an error message is displayed in MATLAB.

Note

- If you have connected several MATLAB installations to your dSPACE installation, you must select one of them as the preferred connection. For details, refer to Introduction to Connecting a MATLAB Installation (Managing dSPACE Software Installations 🕮).
- The **Show in Simulink Model** command does not synchronize the model interface between ConfigurationDesk and MATLAB.
- If a model port block corresponds to In Bus Element and Out Bus Element blocks, the **Show in Simulink Model** command highlights all the In Bus Element or Out Bus Element blocks that belong to the related bus signal.

Related topics

Basics

Switching Between Model Port Blocks in ConfigurationDesk and in Simulink (ConfigurationDesk Real-Time Implementation Guide 🕮)

Model Port Block Properties

Objective

The properties of selected model port blocks or model ports can be displayed in the **Properties Browser** or in a table window.

Where to go from here

Information in this section

Model Port Block Properties

Objective

Let you display the properties of a model port block.

Data Interchange

The properties of model port blocks specified in the behavior model are transferred to the model topology by importing a model topology or by analyzing the model interface. The data types specified in the behavior model are automatically converted to the corresponding data types supported by ConfigurationDesk. The model port blocks that are created in ConfigurationDesk by extending the signal chain receive their properties from the function blocks that they are connected to. For further information, refer to Data Interchange Between ConfigurationDesk and Behavior Model (ConfigurationDesk Real-Time Implementation Guide 🕮).

> **Tip**
>
> ConfigurationDesk lets you specify global options for extending the signal chain. Refer to Specifying Options for Creating Model Port Blocks (ConfigurationDesk Real-Time Implementation Guide 🕮).

Name

(Available only if the name of the model port block is configurable) Displays the name of the block. If the model port block was added to the signal chain via a function block and has not yet been resolved, you can edit the name. The name can consist of 1 or more characters.

Note

ConfigurationDesk does not support Unicode characters in model port block names. If model port blocks are created via the **Extend Signal Chain** command, the model port block names are derived from the associated function block names. In this case, make sure that no Unicode characters are used in function block names.

Source type

Displays the type of the source providing the model port blocks. The following source types are available:

Source Type	Description
Simulink model	The selected model port block is provided by a Simulink model that you added to your ConfigurationDesk application.
Simulink implementation container	The selected model port block is provided by a Simulink implementation container that you added to your ConfigurationDesk application.
FMU	The selected model port block is provided by an FMU that you added to your ConfigurationDesk application.
V-ECU implementation	The selected model port block is provided by a V-ECU implementation that you added to your ConfigurationDesk application.

If model port blocks are created via the **Extend Signal Chain** command, the field is left empty.

Source name

Displays the name of the source providing the selected model port block.

If model port blocks are created via the **Extend Signal Chain** command, the field is left empty.

Path

Displays the path of the model port block within the behavior model.

Block ID

Displays the ID of the model port block.

Background color

Lets you change the background color for the selected model port block.

The following colors are available:

- Light blue
- White
- Red
- Green

- Blue
- Cyan
- Magenta
- Yellow
- Gray
- Orange
- Dark green
- Beige

Related topics

Basics

Specifying Options for Creating Model Port Blocks (ConfigurationDesk Real-Time Implementation Guide 🕮)
Specifying the Model Interface (ConfigurationDesk Real-Time Implementation Guide 🕮)

Model Port Properties

Objective

Let you display the properties of a model port.

Data Interchange

The properties of ports of model port blocks specified in the behavior model are transferred to the model topology by importing a model topology or by analyzing the model interface. The data types specified in the behavior model are automatically converted to the corresponding data types supported by ConfigurationDesk. The ports of model port blocks that are created in ConfigurationDesk by extending the signal chain receive their properties from the function blocks that they are connected to. For further information, refer to Data Interchange Between ConfigurationDesk and Behavior Model (ConfigurationDesk Real-Time Implementation Guide 🕮).

Tip

ConfigurationDesk lets you specify global options for extending the signal chain. Refer to Specifying Options for Creating Model Port Blocks (ConfigurationDesk Real-Time Implementation Guide 🕮).

Name

(Available only if the name of the model port is configurable) Displays the name of the model port. If the related model port block was added to the signal chain via a function block and has not yet been resolved, you can edit the model port name.

Signal ID

Displays the ID of the signal.

Description

Displays a short description of the port.

Port type

Displays the port type of the model port. Port types define the data direction of a port. Possible model port types are: **In**, **Out**.

Data type

Displays the data type (e.g., Int8) for the data that passes through the data port.

This property applies to data ports.

Data width

Displays the vector size of the data that passes through the data port. The width value is of integer type, the possible range is 1 … 2^{31}-1 (1 means scalar). For variable-size signals, **Data width** displays the maximum number of signal elements.

This property applies to data ports.

Variable-size

Displays whether the signal of the data port is variable-size. The number of elements of variable-size signals varies during run time. The maximum number of elements is displayed by the **Data width** property.

Unit

Displays the physical unit for the data that passes through the data port. ConfigurationDesk does not use this property, for example, for calculations.

This property applies to data ports.

Related topics

Basics

Specifying Options for Creating Model Port Blocks (ConfigurationDesk Real-Time Implementation Guide 📖)
Specifying the Model Interface (ConfigurationDesk Real-Time Implementation Guide 📖)

Model Communication

Objective

To configure model communication.

Where to go from here

Information in this section

Information in other sections

Setting Up Model Communication (ConfigurationDesk Real-Time Implementation Guide 📖)

Model Communication Browser

Purpose

To show the Data Outport and Data Inport blocks from selected working views and the mapping lines between them.

Description

For more information on the Model Communication Browser, refer to Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 📖).

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Open in Model Communication Browser 269

Model Communication Package Table

Purpose

To configure and manage communication packages.

Description

The **Model Communication Package** table view provides access to all the model communication packages. Each communication package is displayed as a hierarchy. A communication package can contain one or more Data Inport blocks. For each Data Inport block, the data inports involved in model communication is/are displayed.

The protocol of each communication package is displayed. You can set the **Protocol** property to either **Non-Blocking** (default) or **Blocking**. The behavior model containing each Data Inport block is displayed.

By default, ConfigurationDesk assigns all the Data Inport blocks that are involved in model communication to the **Standard Communication Package**. You cannot rename or delete this package, but you can create your own communication packages and rename them. If you delete one of your own packages, its Data Inport blocks are automatically reassigned to the **Standard Communication Package**. You can drag Data Inport blocks from one communication package to another.

The **Model Communication Package** table view displays the model communication packages with the following symbols:

Symbols	Description
📁	Indicates a communication package.
▶	Indicates a data port block with data inports.
➔	Indicates a data inport.
➔	Indicates a structured data inport.
🔗	Indicates a model port block that is used in the active ConfigurationDesk application.
🔗❗	Indicates a model port block that is used in your active ConfigurationDesk application but is unresolved. For details on unresolved model port blocks, refer to Characteristics of Model Port Blocks (ConfigurationDesk Real-Time Implementation Guide 📖).

Related topics

Basics

Characteristics of Model Port Blocks (ConfigurationDesk Real-Time Implementation Guide 📖)
Setting Up Model Communication (ConfigurationDesk Real-Time Implementation Guide 📖)

New – Model Communication Package

Purpose

To create a user communication package. New model communication packages are numbered consecutively.

Description

After you have added a communication package, you can drag Data Inport blocks from other communication packages to this new one.

Tip

The names of model communication packages have to be unique.

Related topics

HowTos

How to Create Communication Packages (ConfigurationDesk Real-Time Implementation Guide)

References

Model Communication Package Table..201

New – Multiple Model Communication Packages

Purpose

To create multiple user communication packages at once.

Result

ConfigurationDesk opens a **Create Multiple Model Communication Packages** dialog.

Description

After you have added multiple communication packages, you can drag Data Inport blocks from other communication packages to the new ones and configure each communication package individually.

Create Multiple Model Communication Package dialog

Lets you enter the number of packages you want to add.

Name pattern Lets you define the name pattern for the new communication package. You can use the backslash \ to add an incremented number to the name of each new package. This avoids duplicate names.

Tip

The names of model communication packages have to be unique.

Number of Instances Lets you define the number of model communication packages to add to the **Model Communication Package** table view.

Related topics

HowTos

How to Create Communication Packages (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Model Communication Package Table..201

Signal Chain Configuration

Objective

The commands that let you configure the signal chain and its elements are accessible at several places in a ConfigurationDesk application.

Where to go from here

Information in this section

Zoom In / Zoom Out...254
To zoom in and out of working views.

100 %

Purpose

To display all elements in a working view with default zoom settings.

Result

The default zoom settings for all elements are restored if applicable.

Description

The command must be applied to each working view separately.

Related topics

Basics

Handling the Signal Chain in Working Views (ConfigurationDesk Real-Time Implementation Guide)

References

Zoom...253

Add Connected Elements

Purpose

To add all the element's connected elements from the **Global** working view to the current working view.

Result

All the element's connected elements are added from the **Global** working view to the current working view.

Description

If an element in your current user-defined working view has elements connected to it in the **Global** working view, you can use this command to add them all to the current working view. This is an easy way to add missing elements to a user-defined working view.

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide)

Add Removed Ports

Purpose

To add all ports of a selected device block to the current working view.

Result

All ports of the selected device block are added to the current working view.

Description

If you apply this command to device blocks representing devices with ports that are not part of the signal chain, these ports are added to the signal chain. This means that they are also added to the **Global** working view if you apply the command in a user-defined or the **Temporary** working view.

Related topics

Basics

Basics on Device Blocks (ConfigurationDesk Real-Time Implementation Guide)
Handling the Signal Chain in Working Views (ConfigurationDesk Real-Time Implementation Guide)

References

Delete from Application...223
Remove from this Working View..273
Remove from Working View – <Working View>..273

Align Blocks Mode

Purpose

To display mapped ports side by side in working views.

Result

As many mapped ports as possible are displayed side by side. Diagonal mapping lines are avoided whenever possible.

Description

The mode must be set for each working view separately.

This is the default alignment mode when a working view was just opened.

Related topics

Basics

Vertical Alignment of Blocks (ConfigurationDesk Real-Time Implementation Guide)

References

Push Blocks Up Mode / Push Up Mode.. 244

Autofit to Width

Purpose

To permanently display signal chain elements with optimal width in a working view.

Result

The elements are stretched horizontally so that their width exactly matches the width of the working view.

Description

The mode must be set for each working view separately.

Autofit to Width is a display mode. This means that whenever you make changes in a working view, ConfigurationDesk automatically executes the Fit to Width command.

The Autofit to Width mode is indicated by the highlighted ribbon command and a checkmark to the left of the context menu command. If you execute the command again, the mode is disabled and the highlighting as well as the checkmark disappear.

Related topics

Basics

Handling the Signal Chain in Working Views (ConfigurationDesk Real-Time Implementation Guide)

References

Fit to Width Mode.. 235

Clear

Purpose

To remove all entries from the Build Log Viewer or the Find Results Viewer.

Result

ConfigurationDesk deletes all the entries in the Build Log Viewer or the Find Results Viewer.

Related topics

HowTos

How to Find Elements of a ConfigurationDesk Application (ConfigurationDesk Real-Time Implementation Guide)

References

Find Results Viewer..234

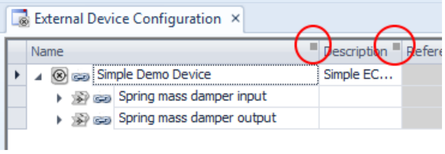
Clear All Column Filters

Purpose

To clear all the column filters set in the current pane.

Result

All the elements are displayed. The columns are marked with a gray square.



External Device Configuration ×

Name	Description	Refere
Simple Demo Device	Simple EC…	
Spring mass damper input		
Spring mass damper output		

Related topics

Basics

Using Display Filters (ConfigurationDesk Real-Time Implementation Guide)

References

Clear Column Filter..214
Set Column Filter...246

Clear All Filters

Purpose

To disable all currently active filters in a pane.

Result

All active filters in the current pane are disabled and all elements in the pane are displayed.

Related topics

Basics

Using Display Filters (ConfigurationDesk Real-Time Implementation Guide 📖)

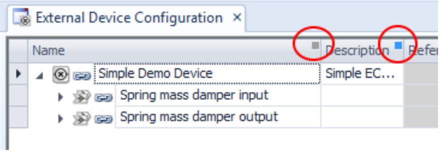
Clear Column Filter

Purpose

To clear a filter set for a column of the current pane.

Result

The elements in the selected column are no longer filtered. The column is marked with a gray square. Filters in other columns remain active.



Related topics

Basics

Using Display Filters (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Clear All Column Filters..213
Set Column Filter..246

Collapse

Purpose

To collapse all the subelements of the selected element.

Result

The subelements of the selected element are hidden.

Description

If a hierarchy consists of many elements, it may be useful to reduce its complexity by hiding subelements.

> **Tip**
>
> When you save a ConfigurationDesk application, the expand/collapse states of hierarchies and blocks are preserved.

Related topics

References

Collapse (Ribbon)

Purpose

To execute collapse commands depending on the context.

Result

The result depends on the currently active pane or selected elements:

- If elements of a hierarchy are selected: Refer to Collapse on page 215.
- If a pane containing hierarchies is active: Refer to Collapse All on page 216.
- If a working view is active or elements in a working view are selected: All blocks and mapping lines in the working view are collapsed in two stages. For details, refer to Collapsing and Expanding Blocks (ConfigurationDesk Real-Time Implementation Guide 🕮).

> **Tip**
>
> When you save a ConfigurationDesk application, the expand/collapse states of hierarchies and blocks are preserved.

Related topics

Basics

Collapsing and Expanding Blocks (ConfigurationDesk Real-Time Implementation Guide 🕮)

Collapse All

Purpose

To collapse all elements in the active pane.

Result

Only the top elements of the hierarchies in the active pane are displayed.

> **Tip**
>
> When you save a ConfigurationDesk application, the expand/collapse states of hierarchies and blocks are preserved.

Related topics

References

Collapse.. 215
Expand.. 224
Expand All.. 225

Configure Filters

Purpose

To activate or deactivate filters for the pane that currently has the focus.

Result

ConfigurationDesk opens the **Filter Settings** dialog, which lets you activate or deactivate filters that are available for the pane that currently has the focus.

Description

All display filters have the following common characteristics:

- A filter setting affects each pane separately.
- You can use one filter or multiple filters in combination. Only the elements matching all filter criteria are displayed.

- In a hierarchy, the higher-level elements of elements matching a filter are also displayed.
- In a hierarchy, ConfigurationDesk lets you display all the subelements of elements matching the filter criteria, even if the subelements themselves do not match the filter criteria. To do so, select the **Include Subelements** checkbox in the **Filter Settings** dialog.

Filter Settings dialog

Associate working view Lets you filter for signal chain elements from a specific working view.

General

- **Include subelements**: Lets you display the subelements of elements matching the filter criteria, even if the subelements themselves do not match the filter criteria.

Functions

- **Used**: Lets you filter for function blocks used in the signal chain.
- **Supported**: Lets you filter for function block types in the function library hierarchy that are supported by the hardware topology of the active ConfigurationDesk application.

Elements

- **Used**: Lets you filter for elements that are used in the signal chain of the active ConfigurationDesk application.
- **Unresolved**: Lets you filter for signal chain elements that are unresolved in the active ConfigurationDesk application.
- **Unused**: Lets you filter for elements that are not used in the signal chain of the active ConfigurationDesk application.
- **User-changed bus elements**: Lets you filter for modified communication matrix elements, i.e., elements that contain user-defined settings.

Related topics

Basics

Using Display Filters (ConfigurationDesk Real-Time Implementation Guide)

References

Associated Working View..258
Filter for Changes / Filter for Changes to Communication Matrices..447
Filter for Supported Functions / Filter for Functions Supported by Hardware Topology............152
Filter for Unresolved Elements / Unresolved Elements..231
Filter for Unused Elements / Unused Elements..232
Filter for Used Elements / Used Elements..232
Filter for Used Functions..153

Copy

Purpose

To copy different types of elements from a ConfigurationDesk application to the Clipboard.

Result

The selected elements are copied to the Clipboard.

Description

The following specific copy commands are available in a ConfigurationDesk application:

- Copy (Device Topology) on page 109
- Copy (Device Connector) on page 108
- Copy (Function Blocks) on page 149
- Copy (Working Views) on page 260

Create Mapping Lines by Name

Purpose

To map function and model ports of selected blocks or of all function blocks and model port blocks in a working view in one step if the port names match.

Description

The command is applied to the current selection selection or, if no selection was made, to the current working view.

You can specify whether you want to create mapping lines between function ports and model ports or between model ports of different models.

IO <-> Model *Available only in the Signal Chain Browser*:

ConfigurationDesk maps function ports to model ports with matching names according to the mapping rules.

Model <->Model *Available in the Signal Chain Browser and the Model Communication Browser*:

ConfigurationDesk maps model ports to model ports with matching names according to the mapping rules.

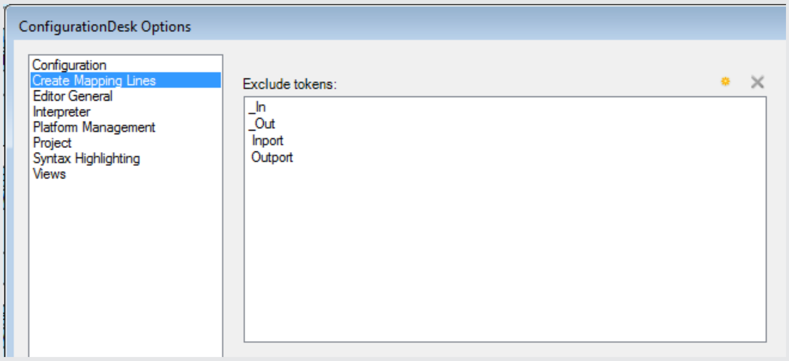
If there is more than one possible match, ConfigurationDesk tries to determine the best one by going through the higher-level elements (e.g., function port groups or structured data ports). In this case, ConfigurationDesk starts at the ports and compares their names up to the related function block or model port block. The largest number of identically named hierarchy elements is considered to be the best match.

Note

- If you execute the command for a complete working view after right-clicking a free area, the function block and model port block names have to match for a mapping to be created.
- ConfigurationDesk creates as many mappings as possible according to the best match principle.
- Existing mappings are not changed.
- Configuration ports are ignored.
- Conflicts caused by created mappings are ignored.

Tip

On the Create Mapping Lines page of the ConfigurationDesk Options dialog, you can define exclude tokens, i.e., strings that are ignored when ConfigurationDesk tries to determine a best match. A number of strings are already excluded by default:



Exclude tokens:

_In
_Out
Inport
Outport

Use the ✹ and ✕ buttons to add or remove tokens.

Related topics

Basics

Model Port Mapping (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Create Mapping Lines Page..481

Cut

Purpose

To cut different types of elements from a ConfigurationDesk application to the Clipboard.

Result

The selected elements are cut to the Clipboard.

Description

The following specific cut commands are available in a ConfigurationDesk application:

- Cut (Device Topology) on page 111
- Cut (Device Connector) on page 111
- Cut (Working Views) on page 261

Decrease Height

Purpose

To decrease the height of all elements in a working view.

Result

The elements appear shorter and the labels smaller in the working view.

Description

The command must be applied to each working view separately.

Related topics

Basics

Handling the Signal Chain in Working Views (ConfigurationDesk Real-Time Implementation Guide)

Decrease Width

Purpose

To decrease the width of all elements in a working view.

Result

The elements appear narrower in the working view.

Description

The command must be applied to each working view separately.

Related topics

Basics

Handling the Signal Chain in Working Views (ConfigurationDesk Real-Time Implementation Guide)

Default Height

Purpose

To display all elements with default height in a working view.

Result

The elements are stretched vertically to their original size (100%) in the working view.

Description

The command must be applied to each working view separately.

Related topics

Basics

Handling the Signal Chain in Working Views (ConfigurationDesk Real-Time Implementation Guide)

Default Width

Purpose

To display all elements with their default widths in a working view.

Result

The elements are stretched horizontally to their original size (100%) in the working view.

Description

The command must be applied to each working view separately.

Delete

Purpose

To access different deletion commands available for the current selection.

Description

Some or all of the following commands are available in a submenu depending on the selected elements:

- Delete from Application on page 223
- Delete from Topology on page 223
- Delete Completely on page 222

Delete Assignment

Purpose

To delete specific assignments in ConfigurationDesk.

Description

The following specific **Delete Assignment** commands are available in a ConfigurationDesk application:

- Delete Assignment (Table Items) on page 284
- Delete Assignment (Channel) on page 65

Delete Completely

Purpose

To delete selected elements from the signal chain and from topologies of the active ConfigurationDesk application.

Result

The selected elements are deleted from the signal chain and from topologies of the active ConfigurationDesk application.

Related topics

References

Delete from Application..223
Delete from Topology..223

Delete from Application

Purpose

To delete one or more selected elements from the active ConfigurationDesk application.

Result

The selected elements are deleted from the ConfigurationDesk application.

Description

If you delete topology elements from the application, they are still available in the corresponding topology.

Related topics

References

Delete Completely..222
Delete from Topology..223

Delete from Topology

Purpose

To delete selected signal chain elements (and their subelements) from a topology.

Result

ConfigurationDesk deletes the selected topology elements (and their subelements) from the topology.

In the following cases, elements become unresolved:

- They are used in the signal chain.

or

- They are part of the calculated external cable harness.

This is shown as follows: . To resolve the elements again, you have to undo the delete operation or add the elements to the topology afterwards.

Related topics

References

Delete Completely..222
Delete from Application...223

Expand

Purpose

To expand all the collapsed subelements of the selected element.

Result

The hidden subelements of the selected element are now displayed.

> **Tip**
>
> When you save a ConfigurationDesk application, the expand/collapse states of hierarchies and blocks are preserved.

Related topics

References

Expand (Ribbon)

Purpose

To execute expand commands depending on the context.

Result

The result depends on the currently active pane or selected elements:

- If elements of a hierarchy are selected: Refer to Expand on page 224.
- If a pane containing hierarchies is active: Refer to Expand All on page 225.
- If a working view is active or elements in a working view are selected: All blocks and mapping lines in the working view are expanded in two stages (if they are collapsed). For details, refer to Collapsing and Expanding Blocks (ConfigurationDesk Real-Time Implementation Guide).

> **Tip**
>
> When you save a ConfigurationDesk application, the expand/collapse states of hierarchies and blocks are preserved.

Related topics

Basics

Collapsing and Expanding Blocks (ConfigurationDesk Real-Time Implementation Guide 🕮)

Expand All

Purpose

To expand all elements in the active pane.

Result

All elements of the hierarchies in the active pane are displayed.

> **Tip**
>
> When you save a ConfigurationDesk application, the expand/collapse states of hierarchies and blocks are preserved.

Related topics

References

Extend Signal Chain

Purpose

To access different commands for adding function blocks or model port blocks to the signal chain.

Result

Different commands for extending the signal chain are available in a submenu.

Description

Depending on the context, the commands available in the submenu are:

- Extend Signal Chain – <Function Block> on page 226
- Extend Signal Chain – Create Suitable Model Port Block on page 226

Extend Signal Chain – <Function Block>

Purpose

To add a function block to the signal chain.

Result

ConfigurationDesk added a new function block instance to the signal chain and mapped the device ports and signal ports. In the Global working view, the block is displayed at the bottom of the function section.

Description

ConfigurationDesk offers you suitable function block types you can select from.

After you have selected a functionality, ConfigurationDesk automatically:

- Instantiates new function blocks.
- Maps the device port(s) to the first suitable signal port of the corresponding function block(s).
- Transfers several configuration settings (for example, the allowed failure classes) from a device port to the mapped signal port.

Related topics

HowTos

How to Add Function Blocks to the Signal Chain via Device Ports (ConfigurationDesk Real-Time Implementation Guide)
How to Assign Reference Ports (ConfigurationDesk Real-Time Implementation Guide)

References

Port Properties.. 135

Extend Signal Chain – Create Suitable Model Port Block

Purpose

To create model port blocks suitable for the selected function blocks.

Result

Model port blocks suitable for the selected function blocks are created.

Description

Depending on the selected function blocks, ConfigurationDesk tries to add one or more model port blocks with a suitable data direction to the signal chain. It also automatically maps the model ports to the function ports from which they were created. This means that you do not have to map ports yourself, so you avoid mapping errors. ConfigurationDesk performs mapping according to specific mapping rules. For details on mapping rules, refer to Rules for Model Port Mapping (ConfigurationDesk Real-Time Implementation Guide 📖). The data type of the new model ports depends on the setting of the **Model port data type** property for the function block type. Other model port properties such as port width and unit are taken from the corresponding function port. The new model port blocks are named according to fixed rules. For details, refer to Basics on Adding Model Port Blocks to the Signal Chain (ConfigurationDesk Real-Time Implementation Guide 📖).

The Global working view displays the new model port blocks in the last position in the model section. They are unresolved.

Note

The **Extend Signal Chain - Create suitable model port block** command does not apply to Configuration Port blocks. They can only be added to the signal chain via the **Model Browser**. For details, refer to How to Add Model Port Blocks to the Signal Chain via the Model Browser (ConfigurationDesk Real-Time Implementation Guide 📖).

Related topics

Basics

Basics on Adding Model Port Blocks to the Signal Chain (ConfigurationDesk Real-Time Implementation Guide 📖)
Creating a Logical Signal Chain (ConfigurationDesk Getting Started 📖)
Methods for Model Port Mapping (ConfigurationDesk Real-Time Implementation Guide 📖)

HowTos

How to Add Model Port Blocks to the Signal Chain via Function Blocks (ConfigurationDesk Real-Time Implementation Guide 📖)

Fill With: / Fill with one of:

Purpose

To configure a property for multiple elements at once with the same value in a table.

Result

Multiple elements are configured with the same property value.

Description

The **Fill** commands let you save time by configuring a property value for multiple elements at once in a table.

Depending on the property you want to configure, there are two different variants of the command that are available in the context menu of properties after selecting multiple elements in a table:

- If the property supports various values, such as strings or numerical values, the **Fill With:** command lets you transfer a property value from one element to the other selected elements. You have to right-click the value you want to transfer and select **Fill With: <value>** from the context menu. The value is transferred to all the selected elements.
- If the property has a limited number of supported values, such as a list of values, the **Fill with one of** command lets you select a value from a list of available values. It does not matter from which element you right-click the property to open the context menu, because the values available in the list are the same for each property.

Tip

You can select multiple elements by pressing `Ctrl` while selecting.

Related topics

Basics

Introduction to Tables (ConfigurationDesk Real-Time Implementation Guide 🕮)

Filter for Connected Ports

Purpose

To display only mapped ports.

Result

Only ports that are mapped to at least one other port are displayed. In working views, blocks that do not have any mapped ports are displayed without block structure.

Description

The command must be set separately for each pane. The active filter is indicated by a checkmark to the left.

Related topics

Basics

Filtering Ports (ConfigurationDesk Real-Time Implementation Guide)

References

Filter for Unconnected Ports.. 230

Filter for Inports

Purpose

To display only inports.

Result

Only inports are displayed.

Description

The filter affects each pane separately. The active filter is indicated by a checkmark to the left.

If you filter model ports, subsystems or models containing only outports are also no longer displayed.

Related topics

Basics

Available Display Filters (ConfigurationDesk Real-Time Implementation Guide)

Filter for Mappable Ports

Purpose

To display only those ports in working views that can be mapped to other ports.

Result

Only ports that can be mapped to other ports are displayed in working views.

Description

The filter is active by default. This also means that function ports for which the **Model access** property is set to **Disabled** are hidden.

The command must be set separately for the **Signal Chain Browser** or the **Model Communication Browser**. The active filter is indicated by a checkmark to the left.

Related topics

Basics

Filtering Ports (ConfigurationDesk Real-Time Implementation Guide 🕮)

Filter for Outports

Purpose

To display only outports.

Result

Only outports are displayed.

Description

The filter affects each pane separately. The active filter is indicated by a checkmark to the left.

If you filter model ports, subsystems or models containing only inports are also no longer displayed.

Related topics

Basics

Available Display Filters (ConfigurationDesk Real-Time Implementation Guide 🕮)

Filter for Unconnected Ports

Purpose

To display only unmapped ports.

Result

Only ports that are unmapped are displayed. In working views, blocks that do not have any unmapped ports are displayed without block structure.

Description

The command must be set separately for each pane. The active filter is indicated by a checkmark to the left.

Related topics

Basics

Filtering Ports (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Filter for Unresolved Elements / Unresolved Elements

Purpose

To display only elements that are unresolved in the active ConfigurationDesk application.

Result

If you set this filter, only elements that are unresolved in the active ConfigurationDesk application are displayed. In context menus, the active filter is indicated by a checkmark to the left. It is also highlighted on the Home ribbon. If the unresolved elements belong to higher-level elements in a hierarchy, the higher-level elements are also displayed.

Description

The filter affects each pane separately.

The term 'unresolved' has different meanings depending on the context and the elements it refers to:

Signal chain elements In panes containing a hierarchy of signal chain elements (except for model topology elements), such as the **Hardware Resource Browser**, 'unresolved' refers to elements that are used in the signal chain of the active ConfigurationDesk application but are missing in the respective topology.

Model port blocks In the model topology hierarchy, 'unresolved' refers to model port blocks that are used in the signal chain of the active ConfigurationDesk application but are not contained in the behavior model which is linked to the application.

Elements of the external wiring information In panes focusing on elements of the external wiring information, such as the External Device Connectors table, 'unresolved' refers to elements that are referenced by the calculated external cable harness but not provided by the device topology or simulator hardware topology.

Related topics

Basics

Using Display Filters (ConfigurationDesk Real-Time Implementation Guide 🕮)

Filter for Unused Elements / Unused Elements

Purpose

To display only elements that are part of the active ConfigurationDesk application, but not used in the signal chain.

Result

If you set this filter, only those elements of the active ConfigurationDesk application that are not used in the signal chain are displayed. In context menus, the active filter is indicated by a checkmark to the left. It is also highlighted on the Home ribbon.

Description

The filter affects each pane separately.

> **Note**
>
> In panes focusing on elements of the external wiring information, such as the External Device Connectors table, 'used' refers to elements that are part of the calculated external cable harness.

Related topics

Basics

Using Display Filters (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Filter for Used Elements / Used Elements..232

Filter for Used Elements / Used Elements

Purpose

To display only elements that are used in the signal chain of the active ConfigurationDesk application.

Result

If you set this filter, only elements that are used in the signal chain of the active ConfigurationDesk application are displayed. In context menus, the active filter is indicated by a checkmark to the left. It is also highlighted on the Home ribbon.

Description

The filter affects each pane separately.

> **Note**
>
> In panes focusing on elements of the external wiring information, such as the External Device Connectors table, 'used' refers to elements that are part of the calculated external cable harness.

Related topics

Basics

Using Display Filters (ConfigurationDesk Real-Time Implementation Guide)

References

Find

Purpose

To find specific elements of the currently active ConfigurationDesk application.

Result

ConfigurationDesk opens the Find (Global) dialog, which lets you enter a search string.

Description

The name of every element in the active ConfigurationDesk application whose properties can be accessed in the Properties Browser is included in the search. This includes, for example, device topology elements, tasks, or build configuration sets. It does not include, for example, mapping lines.

The results of searches that you performed via Find (Global) dialog are displayed in the Find Results Viewer on page 234.

Find (Global) dialog

To enter a search string and perform a search.

Find what Lets you enter a search string. Only searches containing at least one character are performed.

Match whole word only Lets you limit the results to matches where no letter or numeral precedes of follows the search string.

Match case Lets you limit the results to matches which have the same case as the text entered in Find what.

Regular expression Lets you use the text entered in Find what as a regular expression: e.g., if you enter `Out|In`, you limit the results to matches which contain the words *Out* or *In*.

Find Starts the search.

Close Lets you close the dialog.

Related topics

HowTos

How to Find Elements of a ConfigurationDesk Application (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Find Results Viewer..234

Find Results Viewer

Purpose

To display the results of searches you performed via Find command.

Description

For more information on the Find Results Viewer, refer to How to Find Elements of a ConfigurationDesk Application (ConfigurationDesk Real-Time Implementation Guide 🕮).

Related topics

HowTos

How to Find Elements of a ConfigurationDesk Application (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Find...233

Fit to Height Mode

Purpose

To adjust the display of signal chain elements to the height of a working view.

Result

The elements are stretched vertically so that their overall height exactly matches the height of the working view if possible.

Description

The command must be applied to each working view separately.

Related topics

Basics

Handling the Signal Chain in Working Views (ConfigurationDesk Real-Time Implementation Guide)

Fit to Width Mode

Purpose

To adjust the display of signal chain elements to the width of a working view.

Result

The elements are stretched horizontally so that their overall width exactly matches the width of the working view if possible.

Description

The command must be applied to each working view separately.

Related topics

Basics

Handling the Signal Chain in Working Views (ConfigurationDesk Real-Time Implementation Guide)

Generate Working Views

Purpose

To create new working views and working view groups according to the structure of used signal chain elements.

Result

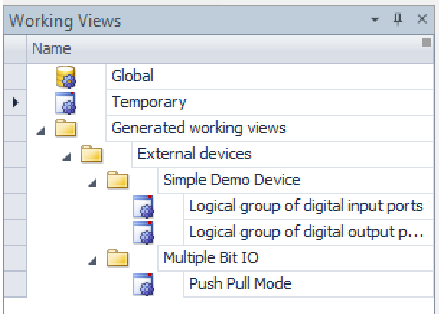
In the Working View Manager, working views and working view groups are created according to the structure of used signal chain elements in the pane you executed the command from.

Tip

If you execute the command from the Working View Manager, new working views and working view groups are created according to the structure of used signal chain elements from all topology browsers and the Function Browser.

Description

For example, if you execute the command from the External Device Browser, working view groups and working views are created according to the structure of used devices and device port groups:



Related topics

HowTos

How to Generate Working Views from Signal Chain Elements (ConfigurationDesk Real-Time Implementation Guide 📖)

Highlight

Purpose

To access different commands to set the highlighting mode in the signal chain.

Result

Different commands for setting the highlighting mode in the signal chain are available in a submenu.

Description

The commands available in the submenu are:

- Highlight – None on page 238
- Highlight – Build on page 237
- Highlight – All on page 237

Highlight – All

Purpose

To highlight all signal chain elements in working views that are involved in a conflict.

Result

All signal chain elements (blocks, ports, lines) are highlighted orange if they cause a warning conflict and red if they cause an error conflict.

> **Tip**
>
> The command must be set separately for the Signal Chain Browser or the Model Communication Browser.

Related topics

Basics

Highlighting Signal Chain Conflicts (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Highlight – Build..237
Highlight – None..238

Highlight – Build

Purpose

To highlight all signal chain elements in working views that are involved in conflicts that have an effect on the build process.

Result

Signal chain elements (blocks, ports, lines) are highlighted orange if they cause a warning conflict and red if they cause an error conflict.

> **Tip**
>
> The command must be set separately for the Signal Chain Browser or the Model Communication Browser.

Related topics

Basics

Highlighting Signal Chain Conflicts (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Highlight – All..237
Highlight – None..238

Highlight – None

Purpose

To highlight no signal chain elements in working views, even if they are involved in a conflict.

Result

Signal chain elements are not highlighted.

> **Tip**
>
> The command must be set separately for the Signal Chain Browser or the Model Communication Browser.

Related topics

Basics

Highlighting Signal Chain Conflicts (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Highlight – All..237
Highlight – Build..237

Increase Height

Purpose

To increase the height of all elements in a working view.

Result

The elements appear taller and the labels larger in the working view.

Description

The command must be applied to each working view separately.

Related topics

Basics

Handling the Signal Chain in Working Views (ConfigurationDesk Real-Time Implementation Guide)

Increase Width

Purpose

To increase the width of all elements in a working view.

Result

The elements appear broader in the working view.

Description

The command must be applied to each working view separately.

Related topics

Basics

Handling the Signal Chain in Working Views (ConfigurationDesk Real-Time Implementation Guide)

New

Purpose

To access different commands for adding one or more elements.

Result

Different commands for creating one or more elements are available in a submenu.

Order Blocks

Purpose

To rearrange the blocks of a signal chain in a working view so that as many mapping lines as possible are horizontal.

Result

- In the **Signal Chain Browser**: Function blocks and model port blocks are rearranged so that as many mapping lines as possible are horizontal. The device blocks on the left are not rearranged.
- In the **Model Communication Browser**: Model port blocks are rearranged so that as many mapping lines as possible are horizontal.

Note

- Ports are not rearranged.
- Reference ports are not taken into account.

Related topics

Basics

Ordering Blocks (ConfigurationDesk Real-Time Implementation Guide)

Paste Before

Purpose

To paste copied elements from the Clipboard before the current selection.

Result

The elements from the Clipboard are pasted before the current selection.

Description

The following specific Paste Before commands are available in a ConfigurationDesk application:

- Paste Before (Device Connector) on page 122
- Paste Before (Device Topology) on page 123
- Paste Before (Working Views) on page 270

Paste Into

Purpose

To paste copied elements from the Clipboard into the current selection.

Result

The elements from the Clipboard are pasted into the current selection.

Description

The following specific Paste Into commands are available in a ConfigurationDesk application:

- Paste Into (Signal Chain) on page 241
- Paste Into (Device Connector) on page 124
- Paste Into (Device Topology) on page 124
- Paste Into (Working Views) on page 271

Paste Into (Signal Chain)

Purpose

To paste copied device topology elements or function blocks to the current working view.

Result

The copied elements are pasted to the current working view. A suffix is added to identify the copy. Pasting them to a user-defined or the **Temporary** working view also pastes them as new instances to the **Global** working view.

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Copy (Device Topology).. 109
Copy (Function Blocks).. 149

Paste Multiple Before

Purpose

To paste multiple instances of copied elements from the Clipboard before the current selection.

Result

ConfigurationDesk opens a **Paste Multiple** dialog for you to define the number and names of instances to be pasted.

Description

The following specific Paste Multiple Before commands are available in a ConfigurationDesk application:

- Paste Multiple Before (Device Connector) on page 125
- Paste Multiple Before (Device Topology) on page 126
- Paste Multiple Before (Working Views) on page 271

Paste Multiple Into

Purpose

To paste multiple instances of copied elements from the Clipboard into the current selection.

Result

ConfigurationDesk opens a **Paste Multiple** dialog for you to define the number and names of instances to be pasted.

Description

The following specific Paste Multiple Into commands are available in a ConfigurationDesk application:

- Paste Multiple Into (Signal Chain) on page 243
- Paste Multiple Into (Device Connector) on page 127

- Paste Multiple Into (Device Topology) on page 128
- Paste Multiple Into (Working Views) on page 272

Paste Multiple Into (Signal Chain)

Purpose

To paste multiple instances of copied device topology elements or function blocks.

Result

ConfigurationDesk opens a Paste multiple dialog.

Paste multiple dialog

Name pattern Lets you define the name pattern for the pasted elements. You can use the backslash \ to add an incremented number to the name of each pasted element. This avoids duplicate names.

Use original names The original name of the copied element(s) is used for all pasted instances. A suffix and an incremented number are added to identify the copies and to avoid duplicate names.

Number of Instances Lets you define the number of pasted elements.

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Copy (Device Topology)..109
Copy (Function Blocks)...149

Properties Browser

Purpose

To access the properties of selected elements.

Description

For more information on the Properties Browser, refer to Configuring Elements with the Properties Browser (ConfigurationDesk Real-Time Implementation Guide 🕮).

Related topics

Basics

Configuring Elements with the Properties Browser (ConfigurationDesk Real-Time Implementation Guide 🕮)

Push Blocks Up Mode / Push Up Mode

Purpose

To move blocks towards the top in working views.

Result

The blocks move towards the top, regardless of port positions and diagonal mapping lines.

The mode must be set for each working view separately.

Related topics

Basics

Vertical Alignment of Blocks (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Align Blocks Mode..211

Remove (Find Results Viewer)

Purpose

To remove the selected search result from the **Find Results Viewer**.

Result

ConfigurationDesk removes the selected search result from the Find Results Viewer.

Related topics

HowTos

How to Find Elements of a ConfigurationDesk Application (ConfigurationDesk Real-Time Implementation Guide)

References

Find Results Viewer.. 234

Select Connected Elements

Purpose

To add elements which are connected to selected elements in the signal chain to the selection.

Result

All the elements connected to the selected elements are also selected and highlighted.

Description

You can select elements in a working view and then use this command to select all their connected elements. This is an easy way to select a set of elements that belong together.

> **Tip**
>
> You can select multiple elements by pressing `Ctrl` while selecting.

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide)

Select Elements by Type / Select

Purpose

To select all elements of a specific type from a pane or selection.

Result

All elements of the selected type are selected. Their properties are displayed in the **Properties Browser** .

Description

Depending on the current pane or selection, the selectable element types are available in a submenu. The number of elements of the type is indicated in parentheses.

Related topics

HowTos

How to Select Elements of the Same Type (ConfigurationDesk Real-Time Implementation Guide 📖)

Set Column Filter

Purpose

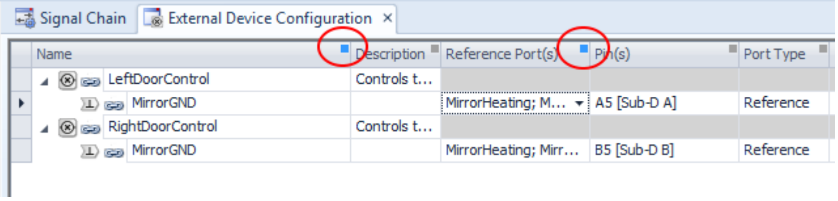
To filter the elements displayed in columns.

Result

ConfigurationDesk opens the **Column Filter Expression** dialog, which lets you enter a filter string.

Description

ConfigurationDesk displays the elements that match the regular expression (and their higher-level elements). The filtered columns are marked with a blue square.



Name	Description	Reference Port(s)	Pin(s)	Port Type
LeftDoorControl	Controls t...			
MirrorGND		MirrorHeating; M...	A5 [Sub-D A]	Reference
RightDoorControl	Controls t...			
MirrorGND		MirrorHeating; Mirr...	B5 [Sub-D B]	Reference

Column Filter Expression dialog

Filter expression Lets you enter a filter string.

Match whole word only Lets you limit the filter to matches where no letter or numeral precedes of follows the filter string.

Match case Lets you make the filter case-sensitive.

Regular expression Lets you disable the support of regular expressions. This lets you use filter strings containing regular expression metacharacters.

Inverse Lets you filter for elements *not* matching the filter string.

Related topics

HowTos

How to Filter Columns by Text and Regular Expressions (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Show

Purpose

To show selected elements in specific panes.

Result

Selected elements are shown in a specific pane depending on the command you select from the submenu.

Description

The available commands in the submenu depend on the types of elements you selected. If necessary, ConfigurationDesk opens the pane and switches to a different view set.

Related topics

Basics

Accessing Elements (ConfigurationDesk Real-Time Implementation Guide 🕮)

Show External Devices Column

Purpose

To define the visibility of the External Devices column in a working view.

Result

Depending on the current setting, the **External Devices** column is displayed or hidden in a working view.

Description

The command applies to each working view separately. The current setting is indicated by a checkmark to the left. The **External Devices** column is displayed by default.

If the **External Devices** is hidden and the working view contains device port mappings, the mapped signal ports are black.

You can specify the default setting for the command on the Configuration Page on page 480 of the **ConfigurationDesk Options** dialog.

Related topics

Basics

Customizing the Display of the Working View Columns (ConfigurationDesk Real-Time Implementation Guide 🕮)

Show in <Browser Name> Browser

Purpose

To show selected elements or components in a specific browser.

Result

The selected elements or components are shown in the appropriate browser. If necessary, ConfigurationDesk opens the browser and switches to a different view set. The selected elements are highlighted in the browser.

For the **Signal Chain Browser** and the **Model Communication Browser**, you have to select the working view you want to show the elements in from a submenu.

Related topics

Basics

Accessing Elements (ConfigurationDesk Real-Time Implementation Guide 🕮)
Using Working Views (ConfigurationDesk Real-Time Implementation Guide 🕮)

Show in <Table Name> Table

Purpose

To show selected elements in a specific table.

Result

The selected elements are shown in the appropriate table. If necessary, ConfigurationDesk opens the table and switches to a different view set. The selected elements are highlighted in the table.

Show Selection

Purpose

To show selected elements in the pane that currently has the focus.

Result

If some or all of the selected elements are available in the pane that currently has the focus, the following effects occur:

- The selection is displayed in the pane as far as possible depending on the size of the pane.
- (for topology browsers and tables) The nodes containing the selected elements are expanded in the hierarchy.
- (for the Signal Chain Browser and Model Communication Browser) If necessary, collapsed blocks are expanded to display the elements.

Signal Chain Browser

Purpose

To open working views, such as the Global working view or user-defined working views.

Description

For more information on the Signal Chain Browser, refer to Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 🕮).

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Open in Signal Chain Browser..270

Transfer Settings

Purpose

To access different commands for transferring settings from device ports to mapped signal ports in working views.

Result

Different commands for transferring settings are available in a submenu.

Description

The commands available in the submenu are:

- Transfer Settings – All Settings on page 250
- Transfer Settings – Allowed Failure Classes on page 251
- Transfer Settings – Load Rejection on page 252

Related topics

Basics

Basics on Transferring Port Settings (ConfigurationDesk Real-Time Implementation Guide)

Transfer Settings – All Settings

Purpose

To transfer device port settings to mapped signal ports in working views.

Result

You transferred the following device port property settings to mapped signal ports:

- Device ports from devices with the ECU device type:
  - Load description was transferred.
  - Allowed failure classes were transferred.
- Device ports from devices with the Load device type:
  - Load description was transferred.
  - Allowed failure classes were transferred.
  - Load rejection setting was transferred.

Description

Restrictions of receiving transfer settings for signal ports The following function ports have restrictions on receiving settings:

- Signal reference ports cannot receive settings from device ports.
- Load description is available only for signal inports, so it is transferred only to signal inports.
- Allowed failure classes can only be transferred to ports in function blocks for which Failure simulation is set to Required.
- Only signal ports which are mapped directly to the transferring device ports can receive the following property settings:
  - Allowed failure classes
  - Load description
- Only signal ports for external loads from a function blocks with load signals which are mapped to the transferring device ports can receive Load rejection settings.

Transferring settings from several device ports mapped to one signal port Identical settings of the device ports are transferred to the signal port of the function block.

If the settings are not identical, the following rules apply:

- If you are transferring the settings from several elements you have selected while pressing the **Ctrl** key, the setting of the last element you have selected is transferred.
- If you are transferring the settings from device blocks or port groups, the settings of the last port in the block or group is transferred.

Sometimes the rules above apply in combination. For example, if you have selected a device port and then a port group while pressing the **Ctrl** key, the setting of the last port in the port group is transferred.

Related topics

Basics

Handling Loads (ConfigurationDesk Real-Time Implementation Guide 🕮)
Specifying Failure Simulation in ConfigurationDesk (ConfigurationDesk Real-Time Implementation Guide 🕮)

HowTos

How to Transfer Port Settings from Device Ports to Signal Ports (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Port Properties.. 135

Transfer Settings – Allowed Failure Classes

Purpose

To transfer settings of failure classes from device ports to mapped signal ports in working views.

Result

You transferred the allowed failure classes of device ports to mapped signal ports.

Description

Restrictions of receiving transfer settings for signal ports The following function ports have restrictions on receiving settings:

- Signal reference ports cannot receive settings from device ports.
- Load description is available only for signal inports, so it is transferred only to signal inports.

- Allowed failure classes can only be transferred to ports in function blocks for which **Failure simulation** is set to **Required**.
- Only signal ports which are mapped directly to the transferring device ports can receive the following property settings:
  - Allowed failure classes
  - Load description
- Only signal ports for external loads from a function blocks with load signals which are mapped to the transferring device ports can receive **Load rejection** settings.

Transferring settings from several device ports mapped to one signal port Identical settings of the device ports are transferred to the signal port of the function block.

If the settings are not identical, the following rules apply:

- If you are transferring the settings from several elements you have selected while pressing the **`Ctrl`** key, the setting of the last element you have selected is transferred.
- If you are transferring the settings from device blocks or port groups, the settings of the last port in the block or group is transferred.

Sometimes the rules above apply in combination. For example, if you have selected a device port and then a port group while pressing the **`Ctrl`** key, the setting of the last port in the port group is transferred.

Related topics

Basics

Specifying Failure Simulation in ConfigurationDesk (ConfigurationDesk Real-Time Implementation Guide 🕮)

HowTos

How to Transfer Port Settings from Device Ports to Signal Ports (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Port Properties.. 135

Transfer Settings – Load Rejection

Purpose

To transfer the **Load rejection** setting of device ports to signal ports of mapped function blocks in working views.

Result

You transferred the Load rejection setting of device ports to signal ports of mapped function blocks.

Description

To transfer the settings helps you to simplify configuration work.

Only device ports of devices with the Load device type have the Load rejection property. For details on device port properties, refer to Port Properties on page 135.

Restrictions of receiving transfer settings for signal ports The following function ports have restrictions on receiving settings:

- Only signal ports for external loads from a function blocks with load signals which are mapped to the transferring device ports can receive Load rejection settings.

Transferring settings from several device ports mapped to one signal port Identical settings of the device ports are transferred to the signal port of the function block.

If the settings are not identical, the following rules apply:

- If you are transferring the settings from several elements you have selected while pressing the `Ctrl` key, the setting of the last element you have selected is transferred.
- If you are transferring the settings from device blocks or port groups, the settings of the last port in the block or group is transferred.

Sometimes the rules above apply in combination. For example, if you have selected a device port and then a port group while pressing the `Ctrl` key, the setting of the last port in the port group is transferred.

Related topics

Basics

Basics on Load Rejection (ConfigurationDesk Real-Time Implementation Guide 📖)

HowTos

How to Transfer Port Settings from Device Ports to Signal Ports (ConfigurationDesk Real-Time Implementation Guide 📖)

Zoom

Purpose

To access different commands to adjust the size of elements in a working view.

Result

Different commands for adjusting the size of elements in a working view are available in a submenu.

Description

The commands available in the submenu are:

- Fit to Height Mode on page 235
- Default Height on page 221
- Increase Height on page 239
- Increase Width on page 239
- Fit to Width Mode on page 235
- Default Width on page 221
- Decrease Height on page 220
- Decrease Width on page 220

Related topics

References

100 %..210

Zoom In / Zoom Out

Purpose

To zoom in and out of working views.

Result

The size of signal chain elements displayed in the working view is increased / decreased.

Working View Management

Objective

A working view is a specific view of the signal chain. To concentrate on specific signal chain elements, ConfigurationDesk allows you to create and manage your own working views.

Where to go from here

Information in this section

Add Connected Elements (Temporary Working View)

Purpose

To add the element and all its connected elements from the **Global** working view to the **Temporary** working view.

Result

The selected element and all signal chain elements connected to it are added to the **Temporary** working view.

Description

If an element has elements connected to it in the **Global** working view, you can use this command to add them all to the **Temporary** working view. This is an easy way add elements to the **Temporary** working view. If the element itself is not yet part of the **Temporary** working view, it is also added.

The **Temporary** working view is a special working view that always exists, and cannot be renamed or removed from the application (like the **Global** working view). You can use it for drafting a signal chain segment (like a notepad).

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 🕮)

Add to Working View

Purpose

To add copied device topology elements, function blocks, or model topology elements to the current working view.

Result

The copied element(s) are added to the current working view. Be aware of the following specifics:

- Elements that are already used in the signal chain can only be added to user-defined working views or to the **Temporary** working view.

- Elements that are not used in the signal chain and have been copied from a topology browser, such as a device from the **External Device Browser**, can also be added to the **Global** working view. Adding them to a user-defined or the **Temporary** working view adds them to the **Global** working view as well.

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Associated Working View

Purpose

To display only signal chain elements from a specific working view in a pane.

Result

Only the signal chain elements from the selected working view are displayed in the current topology browser or table.

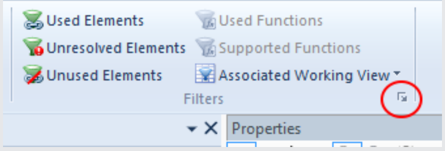
Description

To remove the filter, select <not filtered> from the list of working views.

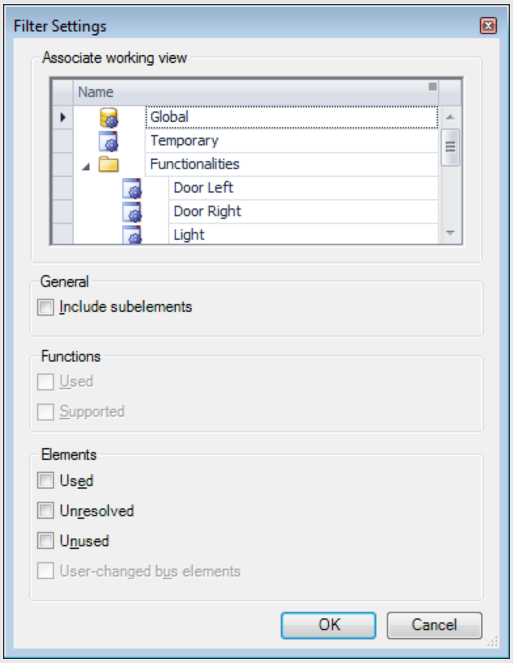
Tip

- If you select an empty associated working view, no signal chain elements will be displayed in the pane.
- You can also open the **Filter Settings** dialog via the dialog launcher in the **Filters** ribbon group to select an associated working view.

1.



2.



Related topics

Basics

Using Display Filters (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Configure Filters...216

Clear Working View

Purpose

To delete all the elements which are contained in a working view.

Result

ConfigurationDesk deletes all the elements from a working view. They are still available in the signal chain and in the **Global** working view.

Description

The command is available in different contexts:

- In the **Signal Chain Browser** or **Model Communication Browser**, you can use it to delete the elements from the current working view.
- In the **Working View Manager** or **Model Communication Browser**, you can use it to delete the elements from a selected working view.
- In the **Temporary Working View** submenu, that is available in multiple panes, you can use it to delete the elements from the **Temporary** working view.

You cannot use the command to delete elements from the **Global** working view.

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 🕮)

Copy (Working Views)

Purpose

To copy the selected working views or working view groups to the Clipboard.

Result

You copied the selected working views or working view groups to the Clipboard.

Tip

You can select multiple working views or working view groups by pressing `Ctrl` while selecting.

Description

After you copied the working views or working view groups to the Clipboard, you can:

- Paste them into the top-level of the **Working View Manager** or into other working view groups.
- Paste them before other working views or working view groups in the **Working View Manager**.

Related topics

Basics

Using Working Views (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Cut (Working Views)

Purpose

To cut the selected working views or working view groups to the Clipboard.

Result

You removed the selected working views or working view groups and placed it in the Clipboard.

Tip

You can select multiple working views or working view groups by pressing `Ctrl` while selecting.

Description

After you cut the working views or working view groups to the Clipboard, you can:

- Paste them into the top-level of the **Working View Manager** or into other working view groups.
- Paste them before other working views or working view groups in the **Working View Manager**.

Related topics

Basics

Using Working Views (ConfigurationDesk Real-Time Implementation Guide)

References

Paste Before (Working Views)... 270
Paste Into (Working Views)... 271

Import/Export – Add Signal Chain to Working View

Purpose

To add the signal chain elements from the selected CAFX file (ConfigurationDesk application fragment file) to the selected working view as new elements.

Result

An **Import from ConfigurationDesk Application Fragment XML** dialog opens for you to select the CAFX file containing the signal chain elements you want to add.

Description

After opening the selected CAFX file:

- All signal chain elements from the CAFX file and their configurations are added to the selected working view as new elements.
- The added signal chain elements are also added to the **Global** working view.
- Adding the elements can cause conflicts (e.g., duplicate name conflicts). Exception: Hardware resource assignment conflicts are avoided by removing channel assignments from imported function blocks if the respective channels are already assigned to existing function blocks.
- Device block elements are also added to the **External Device Browser** maintaining their state (resolved or unresolved).
- Model port blocks are added as new, unresolved blocks with a new identity (ID). They are not displayed in the Model Browser. Refer to Characteristics of Model Port Blocks (ConfigurationDesk Real-Time Implementation Guide).
- Assigned hardware resources are added to the **Hardware Resource Browser** as unresolved elements if necessary.

Related topics

HowTos

How to Export and Import Elements from a Working View (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Import/Export – Export Signal Chain from Working View

Purpose

To export the signal chain elements from the selected working view to a CAFX file (ConfigurationDesk application fragment file).

Result

An **Export to ConfigurationDesk Application Fragment XML** dialog opens for you to provide the name and location for the CAFX file containing the signal chain elements from the working view that you want to export.

Description

All signal chain elements from the selected working view with their mapping lines and configurations are written to the CAFX file.

Related topics

HowTos

How to Export and Import Elements from a Working View (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Import/Export – Merge Signal Chain into Working View

Purpose

To merge the signal chain elements in the selected working view with the ones from a selected CAFX file (ConfigurationDesk application fragment file).

Result

An Import from ConfigurationDesk Application Fragment XML dialog opens for you to select the CAFX file containing the signal chain elements you want to merge with existing elements.

Description

After opening the selected CAFX file:

- The signal chain elements from the CAFX file that do not already exist in the selected working view and their configurations are added to the working view.
- The added signal chain elements that do not already exist in the active ConfigurationDesk application are also added to the Global working view.
- If signal chain elements exist both in the CAFX file and in the selected working view, the elements from the working view and their configurations are replaced by the elements from the CAFX file and their configurations.
- Hardware resource assignment conflicts are avoided by removing channel assignments from already existing function blocks if the respective channels are already assigned to existing function blocks.
- Device block elements that do not already exist in the active ConfigurationDesk application are also added to the **External Device Browser** maintaining their state (resolved or unresolved).
- The identity (ID) of model port blocks is preserved. If there is a matching model port block in a connected behavior model, a model port block is resolved and displayed in the Model Browser. Refer to Characteristics of Model Port Blocks (ConfigurationDesk Real-Time Implementation Guide).
- Assigned hardware resources are added to the **Hardware Resource Browser** as unresolved elements if necessary.

Related topics

HowTos

How to Export and Import Elements from a Working View (ConfigurationDesk Real-Time Implementation Guide)

References

Import/Export – Add Signal Chain to Working View.. 262
Import/Export – Export Signal Chain from Working View.. 263
Import/Export – Replace Signal Chain in Working View.. 264

Import/Export – Replace Signal Chain in Working View

Purpose

To replace the signal chain elements in the selected working view with the ones from a selected CAFX file (ConfigurationDesk application fragment file).

Result

An **Import from ConfigurationDesk Application Fragment XML** dialog opens for you to select the CAFX file containing the signal chain elements you want to import.

Description

After opening the selected CAFX file:

- The signal chain elements in the selected working view are replaced by the signal chain elements from the CAFX file and their configurations.
- The added signal chain elements that do not already exist in the active ConfigurationDesk application are also added to the **Global** working view.
- Elements that are removed from the working view are also removed from the **Global** working view.
- Hardware resource assignment conflicts are avoided by removing channel assignments from imported function blocks if the respective channels are already assigned to existing function blocks.
- Device block elements that do not already exist in the active ConfigurationDesk application are also added to the **External Device Browser** maintaining their state (resolved or unresolved).
- The identity (ID) of model port blocks is preserved. If there is a matching model port block in a connected behavior model, a model port block is resolved and displayed in the Model Browser. Refer to Characteristics of Model Port Blocks (ConfigurationDesk Real-Time Implementation Guide).
- Assigned hardware resources are added to the **Hardware Resource Browser** as unresolved elements if necessary.

Related topics

HowTos

How to Export and Import Elements from a Working View (ConfigurationDesk Real-Time Implementation Guide)

References

Import/Export – Add Signal Chain to Working View........ 262
Import/Export – Export Signal Chain from Working View........ 263
Import/Export – Merge Signal Chain into Working View........ 263

Import/Export (Working Views)

Purpose

To access different commands for importing or exporting signal chain elements via a CAFX (ConfigurationDesk application fragment) file.

Result

Different commands for importing or exporting signal chain elements via a CAFX file are available in a submenu.

Description

The commands available in the submenu are:

- Import/Export – Add Signal Chain to Working View on page 262
- Import/Export – Export Signal Chain from Working View on page 263
- Import/Export – Merge Signal Chain into Working View on page 263
- Import/Export – Replace Signal Chain in Working View on page 264

Related topics

HowTos

How to Export and Import Elements from a Working View (ConfigurationDesk Real-Time Implementation Guide)

New – Multiple Working View Groups

Purpose

To add multiple working view groups to the Working View Manager.

Result

ConfigurationDesk opens a Create Multiple Working View Groups dialog.

Description

Each new working view group is empty. You can add new working view groups or working views to it.

Create Multiple Working Views/Working View Groups dialog

Name pattern Lets you define the name pattern for the new working views or working view groups. You can use the backslash \ as placeholder to add an incremented number to the name of each new working view or working view group.

Number of Instances Lets you define the number of working views or working view groups to add to the Working View Manager.

Related topics

HowTos

How to Create Working Views and Working View Groups (ConfigurationDesk Real-Time Implementation Guide 📖)

References

New – Working View Group..268

New – Multiple Working Views

Purpose

To add multiple working views to the **Working View Manager**.

Result

ConfigurationDesk opens a **Create Multiple Working Views** dialog.

Description

Each new working view is empty and not opened. You can copy and add contents from other working views to them.

Create Multiple Working Views/Working View Groups dialog

Name pattern Lets you define the name pattern for the new working views or working view groups. You can use the backslash \ as placeholder to add an incremented number to the name of each new working view or working view group.

Number of Instances Lets you define the number of working views or working view groups to add to the **Working View Manager**.

Related topics

HowTos

How to Create Working Views and Working View Groups (ConfigurationDesk Real-Time Implementation Guide 📖)

References

New – Working View..268

New – Working View

Purpose

To add an empty working view to the Working View Manager.

Result

A new working view is added to the **Working View Manager**.

Description

The new working view is empty, not opened, and has a default name. You can rename it, and copy and add contents from other working views to it.

Related topics

HowTos

How to Create Working Views and Working View Groups (ConfigurationDesk Real-Time Implementation Guide)

References

New – Multiple Working Views..267

New – Working View Group

Purpose

To add an empty working view group to the Working View Manager.

Result

A new working view group is added to the **Working View Manager**.

Description

The new working view group is empty and has a default name. You can rename it, and add working view groups or working views to it.

Related topics

HowTos

How to Create Working Views and Working View Groups (ConfigurationDesk Real-Time Implementation Guide)

References

New – Multiple Working View Groups..266

Open in Model Communication Browser

Purpose

To open a selected working view in the Model Communication Browser.

Result

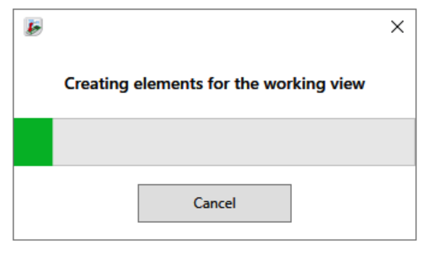
The model port blocks of the selected working view are shown in the Model Communication Browser. The Data Inport blocks and the Data Outport blocks are shown in two different columns so that mappings between them can easily be identified.

Description

The Model Communication Browser is opened or brought to the front if necessary.

Tip

If you open a large working view with many signal chain elements, a progress bar is displayed while the working view is prepared.



Creating elements for the working view

Click Cancel to stop the opening process. The working view is not opened.

Related topics

Basics

Setting Up Model Communication (ConfigurationDesk Real-Time Implementation Guide 🕮)

HowTos

How to Open Working Views in the Model Communication Browser (ConfigurationDesk Real-Time Implementation Guide 🕮)

Open in Signal Chain Browser

Purpose

To open a selected working view in the Signal Chain Browser.

Result

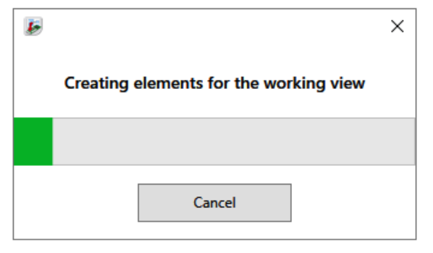
The selected working view is opened in the **Signal Chain Browser**.

Description

The Signal Chain Browser is opened or brought to the front if necessary.

Tip

If you open a large working view with many signal chain elements, a progress bar is displayed while the working view is prepared.



Creating elements for the working view

Click Cancel to stop the opening process. The working view is not opened.

Related topics

HowTos

How to Open Working Views in the Signal Chain Browser (ConfigurationDesk Real-Time Implementation Guide)

References

Working View Manager..274

Paste Before (Working Views)

Purpose

To paste a copied or cut working view or working view group before the current selection.

Result

The copied or cut working view or working view group is pasted before the current selection. A suffix is added to identify the copy.

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide)

References

Copy (Working Views)..260
Cut (Working Views)..261

Paste Into (Working Views)

Purpose

To paste a copied or cut working view or working view group into the currently selected working view group.

Result

The copied or cut working view or working view group is pasted into the currently selected working view group. A suffix is added to identify the copy.

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide)

References

Copy (Working Views)..260
Cut (Working Views)..261

Paste Multiple Before (Working Views)

Purpose

To paste a copied working view or working view group multiple times before the current selection.

Result

ConfigurationDesk opens a Paste Multiple dialog.

Paste Multiple dialog

Name pattern Lets you define the name pattern for the new working views or working view groups. You can use the backslash \ to add an incremented number to each name.

Use original names The name of the copied working view or working view group is used for all new instances. A suffix and an incremented number are added to identify the copies and to avoid duplicate names.

Number of instances Lets you specify the number of working views or working view groups to be pasted.

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Copy (Working Views)..260

Paste Multiple Into (Working Views)

Purpose

To paste a copied working view or working view group multiple times into the currently selected working view group.

Result

ConfigurationDesk opens a Paste Multiple dialog.

Paste Multiple dialog

Name pattern Lets you define the name pattern for the new working views or working view groups. You can use the backslash \ to add an incremented number to each name.

Use original names The name of the copied working view or working view group is used for all new instances. A suffix and an incremented number are added to identify the copies and to avoid duplicate names.

Number of instances Lets you specify the number of working views or working view groups to be pasted.

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Copy (Working Views)..260

Remove from this Working View

Purpose

To remove selected elements from the current working view.

Result

The selected elements are removed from the current working view.

Description

This command is available only for user-defined working views and the **Temporary** working view.

Elements removed from a user-defined or the **Temporary** working view are not deleted from the ConfigurationDesk application and are still displayed in the **Global** working view.

Ports that were removed from a working view can be restored via the **Add Removed Ports** command.

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide)

References

Add Removed Ports...211
Remove from Working View – <Working View>.............273

Remove from Working View – <Working View>

Purpose

To remove selected signal chain elements from a selected working view.

Result

The selected elements are removed from the selected working view.

Description

The working views which contain the selected element(s) are displayed in a submenu from which you can select the working view that you want to remove them from.

Elements removed from a user-defined or the **Temporary** working view are not deleted from the ConfigurationDesk application and are still displayed in the **Global** working view.

Ports that were removed from a working view can be restored via the **Add Removed Ports** command.

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Temporary Working View

Purpose

To access different commands affecting the **Temporary** working view.

Result

Several commands for the **Temporary** working view are available in a submenu.

Description

The **Temporary** working view is a special working view that always exists, and cannot be renamed or removed from the application (like the **Global** working view). You can use it for drafting a signal chain segment (like a notepad).

The commands available in the submenu are:

- Add Connected Elements (Temporary Working View) on page 257
- Extend Signal Chain – <Function Block> on page 226
- Extend Signal Chain – Create Suitable Model Port Block on page 226
- Clear Working View on page 260

Working View Manager

Purpose

To manage the working views of the active ConfigurationDesk application.

Description

For more information on working views and the **Working View Manager**, refer to Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 🕮).

Related topics

Basics

Basics on Working Views (ConfigurationDesk Real-Time Implementation Guide 🕮)

Executable Application and Task Modeling

Objective

ConfigurationDesk provides methods of modeling executable applications and tasks.

Where to go from here

Information in this section

Information in other sections

Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide 📖)

Executable Application Table View

Objective

The executable application table view provides a flexible way of modeling an executable application (i.e., a real-time application) and the tasks used in it.

Where to go from here

Information in this section

Assign Event

Purpose

To assign an event to a task.

Description

This context menu command lets you assign an event to a task that is assigned to an application process. If you select the Assign Event command, ConfigurationDesk lists the available events in a submenu.

Note

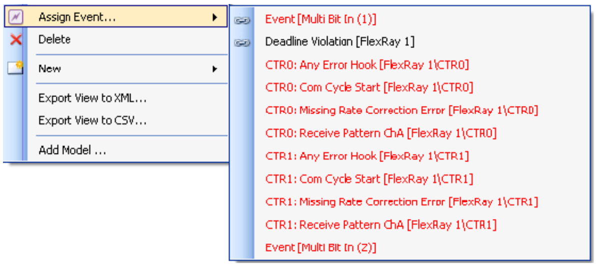
The following events are excluded:

- Unresolved events
- Events that are already preconfigured by other components
- Events whose task assignment is predefined by a component, for example, software events which are provided by a Simulink model component
- Events provided by a model

The events are listed in the submenu with information on the providing components and the event groups (if available) as follows:

```
<event name> [<component name>\<event group name (optional)>]
```

Events that are already assigned to a task are marked with a chain icon. Possible assignments that would lead to the build aborting are colored red. The following illustration shows an example:


Assign Event...
Delete
New
Export View to XML...
Export View to CSV...
Add Model ...
Event [Multi Bit In (1)]
Deadline Violation [FlexRay 1]
CTR0: Any Error Hook [FlexRay 1\CTR0]
CTR0: Com Cycle Start [FlexRay 1\CTR0]
CTR0: Missing Rate Correction Error [FlexRay 1\CTR0]
CTR0: Receive Pattern ChA [FlexRay 1\CTR0]
CTR1: Any Error Hook [FlexRay 1\CTR1]
CTR1: Com Cycle Start [FlexRay 1\CTR1]
CTR1: Missing Rate Correction Error [FlexRay 1\CTR1]
CTR1: Receive Pattern ChA [FlexRay 1\CTR1]
Event [Multi Bit In (2)]

If you select a red-colored event, a conflict is shown in the **Conflicts Viewer**.

Result

The selected event is assigned to the task.

Related topics

Basics

Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide 🕮)

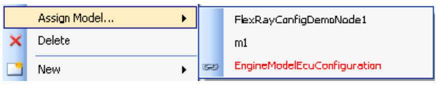
Assign Model

Purpose

To assign a model implementation to an application process.

Description

This context menu command lets you assign a model implementation (which you added to your active ConfigurationDesk application) to the selected application process. If you select the **Assign Model** command, ConfigurationDesk lists all the resolved model implementations of the ConfigurationDesk application in a submenu. Model implementations that are already assigned to other application processes are marked with a chain icon. Possible assignments that would lead to the build aborting are colored red.



If you select a red-colored model implementation, a conflict is shown in the **Conflicts Viewer**.

Result

The selected model implementation is assigned to the application process.

Related topics

Basics

Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide 🕮)

Assign Runnable Function

Purpose

To assign a runnable function to a task.

Description

This context menu command lets you assign a runnable function to the selected task. If you select the **Assign Runnable Function** command, ConfigurationDesk lists the available runnable functions in a submenu, followed by the name of the providing component in brackets:

Note

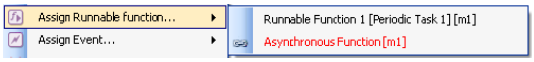
The following runnable functions are excluded:

- Unresolved runnable functions
- Runnable functions whose task assignment is predefined by a component, for example, runnable functions which are assigned to a software event triggered task provided by a Simulink model component.
- Runnable functions provided by a model

The runnable functions are listed in the submenu with information on the related tasks (if available) and the names of the providing components as follows:

```
<runnable function name> [<component name>]
```

Runnable functions that are already assigned to a task are marked with a chain icon. Possible assignments that would lead to the build aborting are colored red. The following illustration shows an example:


Assign Runnable function...
Assign Event...
Runnable Function 1 [Periodic Task 1] [m1]
Asynchronous Function [m1]

If you select a red-colored runnable function, a conflict is shown in the Conflicts Viewer.

Result

The selected runnable function is assigned to the task.

Related topics

Basics

Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide 🕮)

Create Preconfigured Application Process

Purpose

To access different commands for creating a preconfigured application process.

Result

Different commands for creating a preconfigured application process are available in a submenu.

Description

The commands available in the submenu are:

- Create Preconfigured Application Process - In Existing Processing Unit Application on page 282
- Create Preconfigured Application Process - In New Processing Unit Application on page 283

Create Preconfigured Application Process - In Existing Processing Unit Application

Purpose

To have ConfigurationDesk create a preconfigured application process automatically in an existing processing unit application.

Result

An application process is created in an existing processing unit application. The behavior model is assigned to the application process. The tasks that are predefined by the model are adopted, including the events assigned to them and the runnable functions they execute. If a runnable function is not assigned to a predefined task, a new task is created and the runnable function is assigned to it. If the runnable function has a task priority restriction, the new task is configured accordingly. If the runnable function has a cycle time restriction, a timer event is assigned to the new task and configured accordingly. A DAQ raster name (name = task name) and Real-Time Testing are specified for the fastest periodic task if there is exactly one such task.

Description

The **Create preconfigured application process - In existing processing unit application** command lets you select an existing processing unit application from a context menu list to create the preconfigured application process in. As a result, the application process contains preconfigured elements provided by the behavior model. You only have to model and configure tasks triggered by I/O events to complete your executable application. For details, refer to Modeling Asynchronous Tasks (ConfigurationDesk Real-Time Implementation Guide).

> **Note**
>
> The existing processing unit application can already be assigned to a processing unit, but it does not have to be. For details on assigning processing units to processing unit applications, refer to Processing Resource Assignment Table on page 299.

Create Preconfigured Application Process - In New Processing Unit Application

Purpose

To have ConfigurationDesk create a preconfigured application process automatically in a new processing unit application.

Result

An application process is created in a new processing unit application. The behavior model is assigned to the application process. The tasks that are predefined by the model are adopted, including the events assigned to them and the runnable functions they execute. If a runnable function is not assigned to a predefined task, a new task is created and the runnable function is assigned to it. If the runnable function has a task priority restriction, the new task is configured accordingly. If the runnable function has a cycle time restriction, a timer event is assigned to the new task and configured accordingly. A DAQ raster name (name = task name) and Real-Time Testing are specified for the fastest periodic task if there is exactly one such task.

Description

The **Create preconfigured application process - In new processing unit application** command creates a new processing unit application with a new application process. As a result, the application process contains preconfigured elements provided by the behavior model. You only have to model and configure tasks triggered by I/O events to complete your executable application. For details, refer to Modeling Asynchronous Tasks (ConfigurationDesk Real-Time Implementation Guide).

Delete Assignment (Table Items)

Purpose

To delete an assignment of an item to another item in an executable application, a build configuration set, or a model communication package.

Result

The assignment of the selected item to another item is removed.

Related topics

Basics

Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide)

Executable Application Table

Purpose

To model executable applications (i.e., real-time applications) and the tasks used in them.

Executable Application

Priority Each task is assigned a priority according to its relative importance. The RTOS preempts a low-priority task so that a high-priority task is given a turn (preemptive multitasking). When the high-priority task has been executed, the suspended low-priority task resumes execution.

Note

For user-defined tasks and tasks provided by Simulink behavior models and Functional Mock-up Units (FMUs), lower values indicate higher task priorities. For tasks provided by V-ECU implementations, it is the other way round: Higher values indicate higher task priorities.

Element type Displays the type of the element of the executable application. The element type is read-only.

DAQ Raster Name To measure variables (e.g., task information variables) stored in the TRC file synchronously with a task, for example, in dSPACE's ControlDesk, the DAQ service must be enabled for that task. The sample time of the task defines the measurement raster or DAQ raster.

In a real-time application, each application process can have up to 31 tasks that have their DAQ service enabled. A task's DAQ service is enabled if the **DAQ Raster Name** field is not empty. If you have ConfigurationDesk create a preconfigured application process automatically, the fastest periodic task has its DAQ service enabled.

Real-Time Testing Using Real-Time Testing, you can execute tests synchronously with a task. This is especially useful for FlexRay as the RTT services must be synchronous with a FlexRay application task. If you have ConfigurationDesk create a preconfigured application process automatically, Real-Time Testing is enabled for the fastest periodic task. You can enable RTT for exactly one task. For details on performing RTT, refer to Real-Time Testing Guide.

Note

The **Real-time testing** property must be enabled for exactly one task in each application process. If you do not use Real-Time Testing, the Real-Time Testing calls in the generated code do not influence the performance of the real-time application.

Number of Accepted Overruns You can specify the number of overruns that can occur for a specific task before the application stops. 0 means that the application stops at the first overrun. -1 means that an unlimited number of overruns is allowed. The default is 0.

Jitter and latency optimization Lets you specify the jitter and latency run-time behavior of the task. The following settings are possible:

Setting	Description
Standard (full functionality)	(Default setting for newly created tasks) The task is created as a standard task. This setting should be the first choice for newly created tasks, because it includes no limitations regarding the contents of an application process.
Low jitter, low latency	The task has low jitter and low latency. You should use **Low jitter, low latency** to reduce jitter and latency if required. If you use this setting, you must keep in mind the limitations described in Limitations for Low jitter, low latency tasks on page 286.

Setting	Description
No jitter, low latency	The task has no jitter and low latency which lets you achieve smaller sample times for real-time applications. You should configure a task as No jitter, low latency only in cases with high requirements concerning run time, jitter, and latency. If you do, you must keep in mind the limitations described in Limitations for No jitter, low latency tasks on page 286.

Limitations for Low jitter, low latency tasks The following limitations apply to application processes that contain tasks with Low jitter, low latency:

- The Low jitter, low latency tasks must have a higher priority than tasks with a different Jitter and Latency Optimization setting.
- A Low jitter, low latency task can only trigger other Low jitter, low latency tasks. In reverse, it can also only be triggered by Low jitter, low latency tasks.
- Real-Time Testing is not supported for Low jitter, low latency tasks.

> **Note**
>
> If the preconditions described above are not fulfilled, a conflict is shown in the Conflicts Viewer.

- It is forbidden to use system calls within a Low jitter, low latency task, because they re-introduce jitter.
- It is not possible to use Low jitter, low latency tasks on core 0 of a DS6001 Processor Board. A task configured in this way is automatically executed in Standard (full functionality) mode.
- Behavior models containing blocks from the MotionDesk Blockset cannot be executed in a task configured as Low jitter, low latency.
- Ethernet functionality is not supported for application processes containing a task configured as Low jitter, low latency.
- It cannot be guaranteed that tasks configured as Low jitter, low latency are compatible with dSPACE solutions.

Limitations for No jitter, low latency tasks The following limitations apply to application processes that contain a task that is configured as No jitter, low latency:

- The task must be the only task in the application process, and it must be triggered by a timer event. Otherwise, a conflict is shown in the Conflicts Viewer.
- It is not recommended to use system calls within a No jitter, low latency task, because they re-introduce jitter.
- The background task of the application process is not executed.
- It cannot be guaranteed that tasks configured as No jitter, low latency are compatible with dSPACE solutions.
- Behavior models containing blocks from the MotionDesk Blockset cannot be executed in a task configured as No jitter, low latency.
- FPGA signal tracing is not supported for application processes containing a task configured as No jitter, low latency.
- Ethernet functionality is not supported for application processes containing a task configured as No jitter, low latency.

Symbols

The hierarchy is displayed with the following symbols:

Symbol	Meaning
📄	Indicates the currently active ConfigurationDesk application.
❐	Indicates a processing unit application.
📁	Indicates a folder containing subelements.
ƒn	Indicates a function block type.
❐	Indicates a Simulink behavior model.
❐	Indicates a Simulink implementation container.
❐	Indicates a bus simulation container.
fmu	Indicates a Functional Mock-up Unit (FMU).
▬	Indicates a V-ECU implementation.
⚙	Indicates an application process.
📋	Indicates a task.
📋	Indicates task group.
ƒ	Indicates a runnable function.
⚡	Indicates an event.
⚡	Indicates an event group.
🔗	Indicates a component or an element provided by a component (task, event, runnable function) that is used in the ConfigurationDesk application. This means that the component or element will be considered when you build the executable application.
🔗❗	Indicates an element (task, event, or runnable function) whose providing executable application component is not assigned to an application process or a model that is assigned to an application process and is no longer part of the model topology.
🔒	Indicates an element (task, event, or runnable function) whose assignment cannot be modified. For tasks, the following applies: You cannot delete the task. The assignment of events and runnable functions assigned to the task cannot be deleted. For events and runnable functions, the following applies: You cannot delete the assignment to a task.
🔒	Indicates a task whose assignments can be modified partially, for example, a task with a non-modifiable

Symbol	Meaning
	runnable function assignment, but with a modifiable event assignment.

Related topics

Basics

Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide)

New - Application Process

Purpose

To add a new application process to the processing unit application.

Result

A new application process is added to the processing unit application. New application processes are numbered consecutively. You can rename them. The names must be unique.

Description

ConfigurationDesk lets you add one or more application processes to your processing unit application. An application process executes one or more task. For detailed information, refer to Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide).

Related topics

Basics

Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide)

References

New - Multiple Application Processes..290

New - Application Process (Providing Default Task)

Purpose

To give the processing unit application a new application process that provides a default task.

Result

A new application process is added to the processing unit application. The application process contains a periodic default task that has an assigned resolved runnable function named **Communication Step Function** and a timer event with the default configuration (**Period** = 0.001 s and **Offset** = 0 s). Real-Time Testing and the **Periodic Task 1** DAQ raster name are specified for the default task.

The event assigned to the default task can be changed. However, the runnable function cannot be changed, as indicated by a lock symbol (🔒). The task has a grayed-out lock symbol (🔒).

Description

Application processes that provide a default task do not need to have an assigned model implementation. You can use them for bringing simulators into service or for wire testing, for example.

Related topics

Basics

Introduction to Application Processes Without Behavior Models (ConfigurationDesk Real-Time Implementation Guide)

HowTos

How to Create Application Processes That Provide Default Tasks (ConfigurationDesk Real-Time Implementation Guide)

References

New - Multiple Application Processes (Providing Default Task).. 291

New – Delayed Event

Purpose

To add a new delayed event to a task.

Result

A new delayed event with the default configuration is added to the task. New delayed events are numbered consecutively. You can rename them. Their names must be unique.

Description

ConfigurationDesk lets you add default events to a task to trigger it with a delay in relation to another task. For basic information, refer to Configuring Delayed

Tasks (ConfigurationDesk Real-Time Implementation Guide 📖). For more information on delayed event properties, refer to Delayed Event Properties on page 317.

Related topics

Basics

Configuring Delayed Tasks (ConfigurationDesk Real-Time Implementation Guide 📖)

New - Multiple Application Processes

Purpose

To add several application processes to your processing unit application at once.

Result

ConfigurationDesk opens a Create Multiple Application Process dialog.

Create Multiple <Application and Task Modeling Element> dialog

Name pattern Lets you define the name pattern for the new elements (e.g., application processes). You can use the backslash \ as a placeholder to add an incremented number to the name of each new element. This prevents duplicate names.

Tip

- The names of application processes, timer events, and delayed events must be unique.
- The names of processing unit applications must be unique within the executable application.
- The names of tasks must be unique within an application process.

Number of instances Lets you define the number of elements to create.

Related topics

Basics

Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide 📖)

References

New - Multiple Application Processes (Providing Default Task)

Purpose

To give the processing unit application multiple application processes that provide a default task.

Result

The selected number of application processes is added to the processing unit application. Each application process contains a periodic default task that has an assigned resolved runnable function named **Communication Step Function** and a timer event with a default configuration (**Period** = 0.001 s and **Offset** = 0 s). The application processes and the timer events are numbered consecutively.

For each application process the following applies: Real-Time Testing and the **Periodic Task 1** DAQ raster name are specified for the default task.

The event assigned to the default task can be changed. However, the runnable function cannot be changed, as indicated by a lock symbol (🔒). The task has a grayed-out lock symbol (🔒).

Description

Application processes that provide a default task do not need to have an assigned model implementation. You can use them for bringing simulators into service or for wire testing, for example.

Create Multiple <Application and Task Modeling Element> dialog

Name pattern Lets you define the name pattern for the new elements (e.g., application processes). You can use the backslash \ as a placeholder to add an incremented number to the name of each new element. This prevents duplicate names.

Tip

- The names of application processes, timer events, and delayed events must be unique.
- The names of processing unit applications must be unique within the executable application.
- The names of tasks must be unique within an application process.

Number of instances Lets you define the number of elements to create.

Related topics

Basics

Introduction to Application Processes Without Behavior Models (ConfigurationDesk Real-Time Implementation Guide 📖)

HowTos

How to Create Application Processes That Provide Default Tasks (ConfigurationDesk Real-Time Implementation Guide 📖)

References

New - Application Process (Providing Default Task)..288

New – Multiple Delayed Events

Purpose

To add multiple delayed events to a task at once.

Result

ConfigurationDesk opens the **Create Multiple Delayed Events** dialog.

Description

The new delayed events are created with the default configuration. For more information, refer to Configuring Delayed Tasks (ConfigurationDesk Real-Time Implementation Guide 📖). For more information on delayed event properties, refer to Delayed Event Properties on page 317.

Create Multiple <Application and Task Modeling Element> dialog

Name pattern Lets you define the name pattern for the new elements (e.g., application processes). You can use the backslash \ as a placeholder to add an incremented number to the name of each new element. This prevents duplicate names.

> **Tip**
>
> - The names of application processes, timer events, and delayed events must be unique.
> - The names of processing unit applications must be unique within the executable application.
> - The names of tasks must be unique within an application process.

Number of instances Lets you define the number of elements to create.

Related topics

Basics

Configuring Delayed Tasks (ConfigurationDesk Real-Time Implementation Guide 🕮)

New - Multiple Processing Unit Applications

Purpose

To add several processing unit applications to an executable application at once.

Result

ConfigurationDesk opens the Create Multiple Processing Unit Application dialog that allows you to create several processing unit applications in an executable application.

Description

The new processing unit applications are added to the executable application. They are numbered consecutively. You can assign application processes to them, or you can delete or rename them. In addition, you can assign processing units to processing unit applications.

Note

- The names of the processing unit applications must be unique within an executable application.
- As a precondition for assigning processing units to processing unit applications, a hardware topology must exist in the **Hardware Resource Browser**. For details, refer to How to Register Hardware for Multi-PU Applications and Add it to a ConfigurationDesk Application (ConfigurationDesk Real-Time Implementation Guide 🕮).

For details on working with processing unit applications, refer to Creating Multi-Processing-Unit Applications With ConfigurationDesk (ConfigurationDesk Real-Time Implementation Guide 🕮).

Create Multiple <Application and Task Modeling Element> dialog

Name pattern Lets you define the name pattern for the new elements (e.g., application processes). You can use the backslash \ as a placeholder to add an incremented number to the name of each new element. This prevents duplicate names.

Tip

- The names of application processes, timer events, and delayed events must be unique.
- The names of processing unit applications must be unique within the executable application.
- The names of tasks must be unique within an application process.

Number of instances Lets you define the number of elements to create.

New - Multiple Tasks

Purpose

To add several tasks to an application process at once.

Result

ConfigurationDesk opens a **Create Multiple Tasks** dialog.

Description

The new tasks are created with the default configuration. For details on task modeling, refer to Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide 🕮). For details on task properties, refer to Task Properties on page 311.

Create Multiple <Application and Task Modeling Element> dialog

Name pattern Lets you define the name pattern for the new elements (e.g., application processes). You can use the backslash \ as a placeholder to add an incremented number to the name of each new element. This prevents duplicate names.

Tip

- The names of application processes, timer events, and delayed events must be unique.
- The names of processing unit applications must be unique within the executable application.
- The names of tasks must be unique within an application process.

Number of instances Lets you define the number of elements to create.

Related topics

Basics

Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide 🕮)

New – Multiple Timer Events

Purpose

To add several timer events to a task at once.

Result

ConfigurationDesk opens the **Create Multiple Timer Events** dialog.

Description

The new timer events are created with the default configuration. For details on task modeling, refer to Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide). For details on timer-event properties, refer to Event Properties on page 316.

Create Multiple <Application and Task Modeling Element> dialog

Name pattern Lets you define the name pattern for the new elements (e.g., application processes). You can use the backslash \ as a placeholder to add an incremented number to the name of each new element. This prevents duplicate names.

> **Tip**
>
> - The names of application processes, timer events, and delayed events must be unique.
> - The names of processing unit applications must be unique within the executable application.
> - The names of tasks must be unique within an application process.

Number of instances Lets you define the number of elements to create.

Related topics

Basics

Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide)

New - Processing Unit Application

Purpose

To add a new processing unit application to an executable application.

Result

A new processing unit application is added to the executable application.

Description

ConfigurationDesk adds a new processing unit application to the executable application. You can assign application processes to the processing unit application, or you can delete or rename it. In addition, you can assign processing units to processing unit applications.

> **Note**
>
> - The names of the processing unit applications must be unique within an executable application.
> - As a precondition for assigning processing units to processing unit applications, a hardware topology must exist in the **Hardware Resource Browser**. For details on how to add a hardware topology, refer to How to Import a Hardware Topology (ConfigurationDesk Real-Time Implementation Guide).

New - Task

Purpose

To add a new task to an application process.

Result

You added a new task with the default configuration to the application process .

Description

The New - Task command lets you create a task that is executed in an application process. The task name must be unique within the application process. You can assign different event types to the task to model either periodic or asynchronous tasks. For details on task modeling, refer to Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide). For details on the task properties, refer to Task Properties on page 311.

Related topics

Basics

Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide)

New - Timer Event

Purpose

To add a new timer event to a task.

Result

A new timer event with the default configuration is added to the task. New timer events are numbered consecutively. You can rename them. Their names must be unique.

Description

ConfigurationDesk lets you add timer events to a task to trigger it periodically. For basic information, refer to Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide). For details on timer-event properties, refer to Event Properties on page 316.

Related topics

Basics

Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide)

Optimize Configuration

Purpose

To optimize the configuration of application processes that have multiple model implementations assigned.

Result

ConfigurationDesk automatically optimizes the configuration of newly created application processes if they are added via the Add Model dialog and the Create one application process for all models and optimize configuration option is selected. Additionally, you can trigger the optimization via the Optimize Configuration context command of the application process. Optimizing the application process configuration comprises the following steps:

1. Grouping runnable functions in tasks.
2. Specifying the execution order of the runnable functions within a task.
3. Specifying the task priorities.

If you set up or changed model communication or added additional model implementations to the application process, you must optimize the configuration by selecting the Optimize Configuration command from the context menu of the application process. If you added additional runnable functions to a task only, and you do not want ConfigurationDesk to make changes to the whole application process, you can let ConfigurationDesk order the runnable functions within the task based on the data flow by selecting the Optimize Configuration command from the context menu of the task.

Description

The Optimize Configuration command optimizes the configuration of application processes according to the rules described below.

Rules for grouping runnable functions in tasks

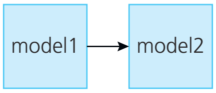
Runnable functions that have the same cycle time restriction (indicated by the Period property in the Properties Browser) are combined in the same task. If there is already a task that is triggered with the required sample time, the runnable functions are added to that task. Otherwise, ConfigurationDesk creates a new task with a suitable new timer event, and assigns the runnable functions to that task.

Rules for specifying the execution order of runnable functions

For tasks to which more than one runnable function are assigned, ConfigurationDesk specifies the execution order of the runnable functions. The execution order depends on the connection of the models providing the runnable functions. In the following examples, model1 provides the rf1 runnable function, and model2 provides the rf2 runnable function.

model1 and model2 are not connected ConfigurationDesk specifies the execution order of rf1 and rf2 according to the alphabetical order of the related model names.

model1 sends data to model2 Refer to the following illustration:

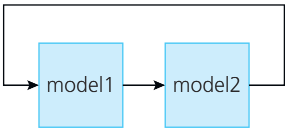


In the preceding illustration, rf1 must be executed before rf2. Thus, ConfigurationDesk specifies the following execution order:

rf1: 1

rf2: 2

model1 and model2 are connected in a closed loop Refer to the following illustration.



In the preceding example, the execution order depends on the setting of the Protocol property of the related model communication package. ConfigurationDesk interrupts the closed loop where the model communication package is specified as non-blocking. If the loop is interrupted before model1, rf1 is executed before rf2, for example.

Note

If the Protocol property of all model communication packages in a closed loop is set to blocking, ConfigurationDesk cannot specify an execution order for the runnable functions and aborts the optimization process. A dialog with an error message is then displayed. Configurations that have been carried out to this point are reversed.

Rules for specifying the task priorities

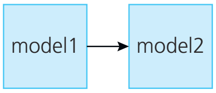
The specification of the task priorities depends on the sample times with that the related tasks are triggered.

Tasks are triggered with different sample times Tasks with a smaller sample time get a higher priority.

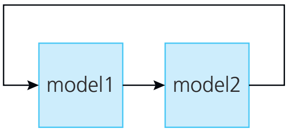
Tasks are triggered with the same sample times The priority of the tasks depends on the specified communication of the models providing the tasks. In the following examples, model1 provides the task1 task, and model2 provides the task2 task

- model1 and model2 are not connected:

  ConfigurationDesk sorts model1 and model2 in an alphabetical order according to their model names in the Model-Function Mapping Browser. The task priorities are specified according to the specified model order.
- model1 sends data to model2:

  task1 gets a higher priority than task2.
- model1 and model2 are connected in a closed loop:

  ConfigurationDesk breaks up the loop at a connection where the model communication is configured as non-blocking. If the loop is disconnected before model1, task1 gets a higher priority than task2.

Two tasks contain at least one runnable functions that is provided by the same model The task with the runnable function that has a smaller priority restriction gets a smaller priority value.

Processing Resource Assignment Table

Purpose

To configure and inspect the processing resources in an executable application. This table is useful especially for multi-processing-unit applications.

Description

The Processing Resource Assignment table gives you an overview of all the processing unit applications, their assigned application processes and processing units.

Processing Unit Assignment

Assigned Processing Unit Lets you assign a processing unit to each processing unit application from the Assigned Processing Unit list. Processing units that are already assigned to a processing unit application are marked red.

> **Note**
>
> - In a multi-processing-unit application, each processing unit application must have a processing unit assigned. For more information on multi-processing-unit applications, refer to Creating Multi-Processing-Unit Applications With ConfigurationDesk (ConfigurationDesk Real-Time Implementation Guide).
> - As a precondition for assigning processing units to processing unit applications, a hardware topology must exist in the **Hardware Resource Browser**. For more information, refer to How to Register Hardware for Multi-PU Applications and Add it to a ConfigurationDesk Application (ConfigurationDesk Real-Time Implementation Guide).

Symbols

The hierarchy is displayed with the following symbols:

Symbol	Meaning
⧉	Indicates a processing unit application.
▣	Indicates an application process.

Task Configuration Table

Purpose

To configure the tasks of an executable application.

Description

As soon as you have modeled your executable application, the Task Configuration table view gives you a clear overview of all the tasks and task groups it contains. You can inspect and configure the properties of the tasks and task groups (if they are configurable). If a task is modeled to be triggered by a timer event, you can inspect and configure the event's Period and Offset properties.

Task Configuration

Priority Each task is assigned a priority according to its relative importance. The RTOS preempts a low-priority task so that a high-priority task can be performed (preemptive multitasking). When the high-priority task has been executed, the suspended low-priority task resumes execution.

> **Note**
>
> For user-defined tasks and tasks provided by Simulink behavior models and Functional Mock-up Units (FMUs), lower values indicate higher task priorities. For tasks provided by V-ECU implementations, it is the other way round: Higher values indicate higher task priorities.

DAQ Raster Name To measure variables (e.g., task information variables) stored in the TRC file synchronously with a task, for example, in dSPACE's ControlDesk, the DAQ service must be enabled for that task. The sample time of the task defines the measurement raster or DAQ raster.

In a real-time application, each application process can have up to 31 tasks that have their DAQ service enabled. A task's DAQ service is enabled if the **DAQ Raster Name** field is not empty. If you specify to create a predefined application process automatically, the fastest periodic task has its DAQ service enabled.

Real-Time Testing Using Real-Time Testing, you can execute tests synchronously with a task. This is especially useful for FlexRay, as the RTT services must be synchronous with a FlexRay application task. If you specify to create a predefined application process automatically, Real-Time Testing is enabled for the fastest periodic task. You can enable RTT for exactly one task. For details on performing RTT, refer to Real-Time Testing Guide 🕮.

> **Note**
>
> The **Real-time testing** property must be enabled for exactly one task in each application process. If you do not use Real-Time Testing, the Real-Time Testing calls in the generated code do not influence the performance of the real-time application.

Period Lets you define the period of the timer event, i.e., the time interval at which the event is generated periodically.

Value / Range	▪ All values >= 0 s (0 s means, the task is triggered only once after simulation start) ▪ 0.001 s (default)
Description	—
Dependencies	—

Offset Lets you define an offset for the timer event, i.e., the delay time between the simulation starting and the timer event occurring.

Value / Range	▪ 0 s … <max. value of double> s ▪ 0 s (default)

Description	—
Dependencies	—

Number of Accepted Overruns You can specify the number of overruns that can occur for a specific task before the application stops. 0 means that the application stops at the first overrun. -1 means that an unlimited number of overruns is allowed. The default is 0.

Jitter and latency optimization Lets you specify the jitter and latency run-time behavior of the task. The following settings are possible:

Setting	Description
Standard (full functionality)	(Default setting for newly created tasks) The task is created as a standard task. This setting should be the first choice for newly created tasks, because it includes no limitations regarding the contents of an application process.
Low jitter, low latency	The task has low jitter and low latency. You should use Low jitter, low latency to reduce jitter and latency if required. If you use this setting, you must keep in mind the limitations described in Limitations for Low jitter, low latency tasks on page 302.
No jitter, low latency	The task has no jitter and low latency which lets you achieve smaller sample times for real-time applications. You should configure a task as No jitter, low latency only in cases with high requirements concerning run time, jitter, and latency. If you do, you must keep in mind the limitations described in Limitations for No jitter, low latency tasks on page 303.

Limitations for Low jitter, low latency tasks The following limitations apply to application processes that contain tasks with Low jitter, low latency:

- The Low jitter, low latency tasks must have a higher priority than tasks with a different Jitter and Latency Optimization setting.
- A Low jitter, low latency task can only trigger other Low jitter, low latency tasks. In reverse, it can also only be triggered by Low jitter, low latency tasks.
- Real-Time Testing is not supported for Low jitter, low latency tasks.

> **Note**
>
> If the preconditions described above are not fulfilled, a conflict is shown in the Conflicts Viewer.

- It is forbidden to use system calls within a Low jitter, low latency task, because they re-introduce jitter.
- It is not possible to use Low jitter, low latency tasks on core 0 of a DS6001 Processor Board. A task configured in this way is automatically executed in Standard (full functionality) mode.
- Behavior models containing blocks from the MotionDesk Blockset cannot be executed in a task configured as Low jitter, low latency.
- Ethernet functionality is not supported for application processes containing a task configured as Low jitter, low latency.
- It cannot be guaranteed that tasks configured as Low jitter, low latency are compatible with dSPACE solutions.

Limitations for No jitter, low latency tasks The following limitations apply to application processes that contain a task that is configured as No jitter, low latency:

- The task must be the only task in the application process, and it must be triggered by a timer event. Otherwise, a conflict is shown in the Conflicts Viewer.
- It is not recommended to use system calls within a No jitter, low latency task, because they re-introduce jitter.
- The background task of the application process is not executed.
- It cannot be guaranteed that tasks configured as No jitter, low latency are compatible with dSPACE solutions.
- Behavior models containing blocks from the MotionDesk Blockset cannot be executed in a task configured as No jitter, low latency.
- FPGA signal tracing is not supported for application processes containing a task configured as No jitter, low latency.
- Ethernet functionality is not supported for application processes containing a task configured as No jitter, low latency.

Symbols

The hierarchy is displayed with the following symbols:

Symbol	Meaning
⚙	Indicates an application process.
📋	Indicates a task.
🔒	Indicates a task whose assignment cannot be modified. You cannot delete the task. The assignment of events and runnable functions assigned to the task cannot be deleted.
🔒	Indicates a task whose assignments can be modified partially, for example, a task with a non-modifiable runnable function assignment, but with a modifiable event assignment.

Related topics

Basics

Modeling Executable Applications and Tasks (ConfigurationDesk Real-Time Implementation Guide 🕮)

Executable Application Components Properties

Objective

The Properties Browser lets you display and configure elements of the executable application.

Where to go from here

Information in this section

Processing Unit Application Properties

Objective

Lets you configure and display the properties of the selected processing unit application.

Name

Lets you specify the name of the selected processing unit application. The names of all processing unit applications within an executable application must be unique. Otherwise, a conflict is displayed in the **Conflicts Viewer**.

Assigned processing unit

Lets you assign a processing unit to the selected processing unit application. In an executable application, each processing unit application must have a separate processing unit assigned. Otherwise, a conflict is shown in the Conflicts Viewer.

Processing units that are already assigned to other processing unit applications are marked red in the drop-down list.

Model Implementation Properties

Objective

Lets you configure and display the properties of the selected model implementation.

Model implementation types

ConfigurationDesk lets you add the following model implementation types to your ConfigurationDesk application:

- Bus simulation container
- Functional Mock-up Unit
- Simulink implementation container
- Simulink model
- V-ECU implementation container

Bus simulation container properties

Source type Displays the source type of the model implementation. For bus simulation containers, the source type is Bus simulation container.

Model location Displays the path to the folder the bus simulation container resides in.

Content Lists the following properties of the bus simulation container:

- The name of the communication matrix
- The clusters
- The restbus participants

Format version Displays the format version of the bus simulation container.

Exporting tool Displays the tool, and its version, the bus simulation container was exported from.

Functional Mock-up Unit properties

Source type Displays the source type of the model implementation. For FMUs, the source type is FMU.

Model location Displays the path to the folder the FMU resides in.

Format version Displays the format version of the FMI standard the FMU complies with.

Exporting tool Displays the tool, and its version, the FMU was exported from.

Precompiled for Displays the following information:

- The platforms the FMU was precompiled for
- The ConfigurationDesk versions the FMU was precompiled with

Simulink implementation container properties

Source type Displays the source type of the model implementation. For Simulink implementation containers, the source type is Simulink implementation container.

Model location Displays the path to the folder the Simulink implementation container resides in.

Format version Displays the format version of the Simulink implementation container.

Exporting tool Displays the tool, and its version, the Simulink implementation container was exported from.

Precompiled for Displays the following information:

- The platforms the Simulink implementation container was precompiled for
- The ConfigurationDesk versions the Simulink implementation container was precompiled with

Simulink model properties

Source type Displays the source type of the model implementation. For Simulink models, the source type is Simulink model.

Model location Displays the path to the folder the behavior model resides in.

Initialization command Lets you specify a MATLAB command that initializes the behavior model. Initialization commands are executed before the behavior model is opened, for example, to prepare data in the MATLAB base workspace that is referenced by the behavior model.

ConfigurationDesk lets you specify locations of scripts, data, etc. via relative paths in the Initialization command edit field. This enables you to transfer the ConfigurationDesk project to another path (refer to How to Back up and Transfer a Project (ConfigurationDesk Real-Time Implementation Guide 🕮)). For this purpose, ConfigurationDesk provides the following path macros:

Path Macro Name	Description
%ProjectRoot%	Expands to the root directory of the current project
%ApplicationRoot%	Expands to the root directory of the currently activated ConfigurationDesk application
%ModelRoot%	Expands to the directory the selected Simulink model is stored in

Skip model code generation Runs the build process without performing a model analysis and model code generation for the behavior model. The build process can be completed successful only if model code was already generated for the behavior model during a prior build process or via the Generate Model Code command and is still available.

Note

- The model code must be of the same dSPACE Release with which the real-time application is built.
- You cannot use this option to move the generated model code together with the project or transfer it to another system.
- If you want to use model code in several projects, on different systems, and across several dSPACE Releases, you must use SIC files for this purpose.

V-ECU implementation container properties

Source type Displays the source type of the model implementation. For V-ECU implementation containers, the source type is V-ECU implementation.

Model location Displays the path to the folder the V-ECU implementation container resides in.

Autostart Displays if Autostart of the V-ECU implementation is enabled or disabled. This property is read-only.

V-ECU implementations are configured and exported with SystemDesk or TargetLink. In SystemDesk, you can configure the Autostart property of a V-ECU implementation to be enabled or disabled.

Autostart	Description
Enabled	The V-ECU implementation starts immediately when the real-time application starts.
Disabled	The V-ECU implementation waits when the real-time application starts.

Memory page support Displays if the V-ECU implementation provides a memory segments/pages. This property is read-only.

Supported platforms (Only available for V-ECU implementation containers that are based on VECU files) Displays the simulation platforms that are supported by a V-ECU implementation container. This property is read-only.

Virtual bypassing support Displays if the V-ECU implementation supports virtual bypassing. Virtual bypassing allows you to bypass variables or functions of the V-ECU implementation while the real-time application is running on the real-time hardware. This property is read-only.

Format version Displays the V-ECU implementation container version.

Exporting tool Displays the dSPACE product, the V-ECU implementation container was exported from.

Application Process Properties

Objective

Lets you configure and display the properties of the application process.

Name

Lets you specify the name of the selected application process.

Provide default task

Lets you transform the application process into an application process that provides a default task. Such application processes do not need to have an assigned model implementation. You can use them for tests, such as wire testing.

If you select the **Provide default task** checkbox for an existing application process, the following applies: If a suitable periodic task already exists, that task is converted into the default task. If there is no task, a new default task is created.

The default task has an assigned resolved runnable function named **Communication Step Function** and a timer event with the default configuration (**Period** = 0.001 s and **Offset** = 0 s). If no other task in the application process has Real-Time Testing enabled, Real-Time Testing is enabled for the default task. If no task has a DAQ raster name specified, the **Periodic Task 1** DAQ raster name is specified for the default task

The events assigned to the default task can be changed. However, the runnable function cannot be changed, as indicated by a lock symbol (🔒). The task has a grayed-out lock symbol (🔒).

After testing, you can clear the **Provide default task** checkbox in the **Properties Browser** and assign the desired behavior model to the application process. The default task then is converted into a standard user task. The runnable function is replaced with a base rate runnable function when the behavior model is assigned.

Note

You can select the **Provide default task** checkbox for application processes that already exist in your executable application. If the application process has an assigned model implementation, a conflict is shown in the **Conflicts Viewer**. You have to resolve the conflict before you start the build process. Otherwise, the build process will be aborted with an error message.

Optimize configuration automatically

Lets you activate or deactivate the automatic configuration optimization of an application process. If this checkbox is selected, the configuration optimization is performed automatically at the following points in time:

- During each build process before the conflicts are calculated.
- After you perform the following actions:
  - You reload a model container.
  - You analyze a model implementation.
  - You delete a model.
  - You delete a model assignment.
  - You assign a model to an application process.
  - You replace a model.

> **Note**
>
> You can use the Optimize configuration automatically checkbox on the Configuration page of the Options dialog to specify the default behavior of the application process property. If you select this checkbox, the Optimize configuration automatically checkbox is selected for each newly created application process by default.

Local real-time testing access only

> **Note**
>
> This property is not available for application processes that a V-ECU implementation is assigned to.

Lets you increase the number of available communication channels in a multi-processing-unit system by deactivating the real-time testing access to TRC variables of other application processes.

If the Local real-time testing access only checkbox is selected, real-time testing access is possible only to TRC variables of the application process.

If the Local real-time testing access only checkbox is cleared (default setting), real-time testing access is possible to the TRC variables of all other application processes that support real-time testing access.

XCP service

(Only available for application processes that have a Simulink model/SIC file with Variable description file type = A2L or V-ECU implementation containers assigned) Lets you activate or deactivate the XCP service for the selected application process. If you clear this checkbox, the CFD_I_XCP license is not required for the build process. In this case, no A2L file is created for the application process.

XCP service IP address

(Only available for application processes that have a Simulink model/SIC file with Variable description file type = A2L or V-ECU implementation containers assigned) Lets you specify the IP address of the XCP service in IPv4 notation: Each number of the local IP address must be in the range 0 … 255, e.g., 192.168.140.6.

XCP service port

(Only available for application processes that have a Simulink model/SIC file with Variable description file type = A2L or V-ECU implementation containers assigned) Lets you specify an XCP service port for the application process. XCP service ports must be unique for application processes with assigned V-ECU implementations.

Runnable Function Properties

Objective

Displays the properties of the selected runnable function. The runnable function properties are read-only.

Source type

Displays the type of the source providing the runnable function. The following source types are available:

Source Type	**Description**
Simulink model	The selected runnable function is provided by a Simulink model that you added to your ConfigurationDesk application.
Simulink implementation container	The selected runnable function is provided by a Simulink implementation container that you added to your ConfigurationDesk application.
Functional Mock-up Unit	The selected runnable function is provided by a Functional Mock-up Unit that you added to your ConfigurationDesk application.
V-ECU implementation	The selected runnable function is provided by a V-ECU implementation that you added to your ConfigurationDesk application.
I/O function	The selected runnable function is provided by an I/O function you added to your ConfigurationDesk application.

Source name

Displays the name of the source providing the selected runnable function.

Execution order

Displays the execution order of runnable functions within a task. A suitable execution order of runnable functions is specified by ConfigurationDesk automatically, when you create a new application process that has multiple model implementations assigned. If you set up model communication, and/or added additional runnable functions to a task, you can have ConfigurationDesk specify a suitable execution order for the runnable functions within a task via the Optimize Configuration command from the context menu of the task. You can also manually specify an execution order, if required.

Period

Displays the cycle time restriction of the runnable function. A cycle time restriction indicates a sample time the runnable function requires to achieve correct results.

Priority

Displays the task priority restriction of the runnable function. When configuring the tasks the runnable functions are assigned to, you must make sure that the relationship between the task priorities matches the relationship between the predefined priorities, even if the absolute priority values differ from the predefined values. For details, refer to Basics on Modeling Tasks in ConfigurationDesk (ConfigurationDesk Real-Time Implementation Guide 📖).

Description

Displays a description of the calculations the runnable function performs.

Task Properties

Objective

Lets you configure and display properties of the selected task.

Name

(Available only if the name of the task is configurable) This is the name of the task which is used as an alias, for example, in task-specific messages, or later in dSPACE's experiment tools such as ControlDesk. You can also use the task name to find the task information variables (e.g., Task Call Counter) in the TRC file.

Source type

Displays the type of the source providing the task. The following source types are available:

Source Type[1)]	Description
Simulink model	The selected task is provided by a Simulink model you added to your ConfigurationDesk application.
V-ECU implementation	The selected task is provided by a V-ECU implementation you added to your ConfigurationDesk application.
User	The selected task is a user-defined task.

Source Type[1]	Description
I/O function	The selected task is provided by an I/O function you added to your ConfigurationDesk application.

[1] The source type is read-only.

Source name

Displays the name of the source providing the selected task. For user-defined tasks, the source name is left empty. The source name is read-only.

Number of accepted overruns

You can specify the number of overruns that can occur for a specific task before the application stops. 0 means that the application stops at the first overrun. -1 means that an unlimited number of overruns is allowed. The default is 0.

Priority

Each task is assigned a priority according to its relative importance. The RTOS preempts a low-priority task so that a high-priority task is given a turn (preemptive multitasking). When the high-priority task has been executed, the suspended low-priority task resumes execution.

> **Note**
>
> For user-defined tasks and tasks provided by Simulink behavior models and Functional Mock-up Units (FMUs), lower values indicate higher task priorities. For tasks provided by V-ECU implementations, it is the other way round: Higher values indicate higher task priorities.

Runnable functions such as those provided by a behavior model can have a task priority restriction. If you configure the tasks you want to assign the runnable functions to, you must make sure that the relationship between the task priorities matches the relationship between the predefined priorities, though the absolute priority values can differ from the predefined values.

Scheduling policy

The scheduling policy determines the interaction between tasks that have the *same* priority:

- First come, first served (FCFS), or
- Last come, first served (LCFS)

An LCFS task preempts an FCFS task and also another LCFS task. ConfigurationDesk assigns the FCFS scheduling policy to all tasks except FlexRay tasks (LCFS).

The task scheduling policy is read-only.

Jitter and latency optimization

Lets you specify the jitter and latency run-time behavior of the task. The following settings are possible:

Setting	Description
Standard (full functionality)	(Default setting for newly created tasks) The task is created as a standard task. This setting should be the first choice for newly created tasks, because it includes no limitations regarding the contents of an application process.
Low jitter, low latency	The task has low jitter and low latency. You should use Low jitter, low latency to reduce jitter and latency if required. If you use this setting, you must keep in mind the limitations described in Limitations for Low jitter, low latency tasks on page 313.
No jitter, low latency	The task has no jitter and low latency which lets you achieve smaller sample times for real-time applications. You should configure a task as No jitter, low latency only in cases with high requirements concerning run time, jitter, and latency. If you do, you must keep in mind the limitations described in Limitations for No jitter, low latency tasks on page 313.

Limitations for Low jitter, low latency tasks The following limitations apply to application processes that contain tasks with Low jitter, low latency:

- The Low jitter, low latency tasks must have a higher priority than tasks with a different Jitter and Latency Optimization setting.
- A Low jitter, low latency task can only trigger other Low jitter, low latency tasks. In reverse, it can also only be triggered by Low jitter, low latency tasks.
- Real-Time Testing is not supported for Low jitter, low latency tasks.

> **Note**
>
> If the preconditions described above are not fulfilled, a conflict is shown in the Conflicts Viewer.

- It is forbidden to use system calls within a Low jitter, low latency task, because they re-introduce jitter.
- It is not possible to use Low jitter, low latency tasks on core 0 of a DS6001 Processor Board. A task configured in this way is automatically executed in Standard (full functionality) mode.
- Behavior models containing blocks from the MotionDesk Blockset cannot be executed in a task configured as Low jitter, low latency.
- Ethernet functionality is not supported for application processes containing a task configured as Low jitter, low latency.
- It cannot be guaranteed that tasks configured as Low jitter, low latency are compatible with dSPACE solutions.

Limitations for No jitter, low latency tasks The following limitations apply to application processes that contain a task that is configured as No jitter, low latency:

- The task must be the only task in the application process, and it must be triggered by a timer event. Otherwise, a conflict is shown in the Conflicts Viewer.
- It is not recommended to use system calls within a No jitter, low latency task, because they re-introduce jitter.

- The background task of the application process is not executed.
- It cannot be guaranteed that tasks configured as **No jitter, low latency** are compatible with dSPACE solutions.
- Behavior models containing blocks from the MotionDesk Blockset cannot be executed in a task configured as **No jitter, low latency**.
- FPGA signal tracing is not supported for application processes containing a task configured as **No jitter, low latency**.
- Ethernet functionality is not supported for application processes containing a task configured as **No jitter, low latency**.

DAQ raster name

To measure variables (e.g., task information variables) stored in the TRC file synchronously with a task, for example, in dSPACE's ControlDesk, the DAQ service must be enabled for that task. The sample time of the task defines the measurement raster or DAQ raster.

A real-time application can have up to 31 tasks per application process that have their DAQ service enabled. At least one DAQ service must be enabled per application process. A task's DAQ service is enabled if the **DAQ raster name** field is not empty. If you have ConfigurationDesk create a preconfigured application process automatically, the fastest periodic task has its DAQ service enabled.

Assigned event

Lets you select one or more events you want to assign to the task.

The **Properties Browser** of the selected task lets you inspect and configure the properties of the assigned events. For a description of the event properties, refer to Event Properties on page 316.

Assigned runnable function

Lets you select one or more runnable functions you want to assign to the task.

The **Properties Browser** of the selected task lets you inspect the properties of the assigned runnable functions. For a description of the runnable function properties, refer to Runnable Function Properties on page 310.

Real-time testing

Using Real-Time Testing (RTT), you can execute tests synchronously with a task. This is especially useful for FlexRay as the RTT services must be synchronous with a FlexRay application task. If you have ConfigurationDesk create a preconfigured application process automatically, the **Real-time testing** property is enabled for the fastest periodic task. You can enable RTT for exactly one task. For details on performing RTT, refer to Real-Time Testing Guide 🕮.

Note

The **Real-time testing** property must be enabled for exactly one task in each application process. If you do not use Real-Time Testing, the Real-Time Testing calls in the generated code do not influence the performance of the real-time application.

Bus monitoring

Lets you enable or disable the bus monitoring service for the selected task.

Note

- Bus monitoring must be activated for exactly one task in each application process.
  If bus monitoring is activated for more than one task or for no task in an application process, a conflict is displayed in the **Conflicts Viewer**. You must resolve this conflict before you start the build process. Otherwise, the build process is aborted.
- Bus monitoring must only be specified for periodic tasks.
  If you activate bus monitoring for an asynchronous task, a warning is displayed via a conflict in the **Conflicts Viewer**. You are recommended to resolve this conflict before you start the build process. Otherwise, unexpected results might be displayed in the bus monitoring device in the experiment software.

Task stack size mode

Lets you specify the task stack size mode for each task. The following table shows the possible settings and their definitions:

Setting	Definition
Default	Specifies the default value for the task stack size. The default value is 128 kB.
Custom	Lets you select a custom task stack size from the **Task stack size** list.

Task stack size

If you selected **Custom** from the **Task stack size mode** list, you can select a custom task stack size mode from the **Task stack size** list. Possible values are:

- 128 kB
- 256 kB
- 512 kB
- 1024 kB
- 2048 kB

Event Properties

Objective

Lets you configure and display the properties of the selected event.

Name

(Available only if the name of the event is configurable) Indicates the name of the event. You can edit the name of user-defined timer event.

Source type

Displays the type of the source providing the timer event. The following source types are available:

Source Type	Description
Simulink model	The selected event is provided by a Simulink model you added to your ConfigurationDesk application.
V-ECU implementation	The selected event is provided by a V-ECU implementation you added to your ConfigurationDesk application.
User	The selected event is a user-defined event.
I/O function	The selected event is provided by an I/O function you added to your ConfigurationDesk application.

The source type is read-only.

Source name

Displays the name of the source providing the selected timer event. For user-defined timer event, the source name is left empty. The source name is read-only.

Event type

The following event types are available:

- Timer events are periodic events with a sampling rate and an optional offset.
- I/O events are asynchronous events triggered by I/O functions.
- Software events are parts of predefined tasks provided by the behavior model. They are available in ConfigurationDesk after model analysis.

Send event via Gigalink

Lets you select to send timer events via Gigalink connection to another Gigalink member to achieve event-based synchronization of applications running on two or more systems. Synchronized communication via Gigalink is supported between SCALEXIO systems and between a SCALEXIO system and a PHS-bus-based system. Up to eight events can be used for each Gigalink port.

For basic information on Gigalink communication, refer to Basics on Gigalink Communication (ConfigurationDesk Real-Time Implementation Guide).

Gigalink number

Lets you specify a Gigalink port number for a Gigalink connection. You must enter a value between 1 and 8.

Channel number

Lets you specify a Gigalink channel number for a Gigalink connection. You must enter a value between 1 and 8.

Period

Lets you define the period of the timer event, i.e., the time interval at which the event is generated periodically.

Value / Range	▪ All values >= 0 s (0 s means, the task is triggered only once after simulation start) ▪ 0.001 s (default)
Description	—
Dependencies	—

Offset

Lets you define an offset for the timer event, i.e., the delay time between the simulation starting and the timer event occurring.

Value / Range	▪ 0 s … <max. value of double> s ▪ 0 s (default)
Description	—
Dependencies	—

Delayed Event Properties

Objective

Lets you configure and display the properties of the selected delayed event.

Name

Indicates the name of the event. You can edit the name of delayed events.

Source type

Displays the source type of the event (user). A delayed event is always a user-defined event. The source type of delayed events is read-only.

Event type

Displays the type of the event (delayed event). The event type is read-only.

Source task

Lets you specify a source task for the delayed task. The source task is the task that triggers the delayed event. As soon as the delay time expires after the execution of the source task has started, the source task triggers the delayed event to execute the delayed task.

All tasks in the same application process as the delayed task can be specified as a source task, except for the delayed task itself. Invalid source tasks are highlighted in red in the **Source task** list in the **Properties Browser**.

Delay

Lets you specify a delay for the delayed task. The delay specifies the time span from the start of the execution of the source task to the trigger of the delayed event in seconds. You can specify a semicolon-separated list of multiple floating-point values > 0. The default value is 0.001. If multiple values are specified, the delayed event triggers the delayed task multiple times after the start of the execution of the source task.

Note

You have to select a delay value that triggers the delayed task before the source task is triggered two more times. For periodically executed source tasks, this means that the delay of the delayed task must be smaller than 2 * period of the timer event that triggers the source task.
For example, if the period of the timer event that triggers the source task is 0.001 s, the delay of the delayed event must be < 0.002 s.

Resolving Conflicts

Objective

ConfigurationDesk allows for flexible configuration without strict constraints. This lets you work more freely, but it can lead to conflicting configuration settings. ConfigurationDesk automatically detects conflicts and provides the **Conflicts Viewer** to display and help resolve them. Before you build a real-time application, you have to resolve at least the most severe conflicts to get proper build results.

Conflicts Viewer

Purpose

To display the configuration conflicts that exist in the active ConfigurationDesk application. You can resolve most of the conflicts directly in the **Conflicts Viewer**.

Description

For more information on the **Conflicts Viewer**, refer to Basics on Resolving Conflicts (ConfigurationDesk Real-Time Implementation Guide 🕮).

Related topics

Basics

Resolving Conflicts (ConfigurationDesk Real-Time Implementation Guide 🕮)

Build and Download Management

Objective

ConfigurationDesk lets you manage the build process of real-time applications and their download to the connected platform.

Where to go from here

Information in this section

Information in other sections

Building Real-Time Applications (ConfigurationDesk Real-Time Implementation Guide 🕮)

Downloading and Executing Real-Time Applications (ConfigurationDesk Real-Time Implementation Guide 🕮)

Build Configuration Table

Purpose

To create build configuration sets and to configure build settings, for example, build options, or the build and download behavior.

The Build Configuration table lets you specify the following build settings:

- Global Build Settings: The task startup behavior, the download behavior, and the trace file options.
- Build Configuration Set options: C/C++ compiler settings and custom code options that apply to the build process of real-time applications.
- Build Configuration Set - Bus Custom Code Set options: Bus custom code options that let you include user code implementations in a real-time application or in bus simulation containers if you work with bus implementation software, such as the Bus Manager or the RTI CAN MultiMessage Blockset.

If you select the elements described above in the Build Configuration table, you can access the respective build settings in the Properties Browser.

Tip

The Build Configuration table also provides access to the properties of model implementations, and application processes. Refer to the following topics:

- Model Implementation Properties on page 305
- Application Process Properties on page 308

Global Build Settings

ConfigurationDesk lets you configure the task startup behavior of periodic tasks in real-time applications as well as the download behavior after a successful build process.

The following settings can be configured:

Time-scaled period Lets you specify a time period [0;T] with T >= 0.0 seconds. The period, including the trigger times of periodic tasks within it, is scaled by the factor you define at Time Scale Factor. During the scaled time period [0; T * Time scale factor], periodic tasks are not executed in real time, but have more time to complete their computations. This is useful to avoid task overruns in periodic tasks during the first simulation steps after the simulation is started, for example, overruns caused by a cold cache. The period is entered as a double.

Time scale factor Lets you specify the factor by which to scale the time period. The scale factor must be >= 1.0 and is entered as a double.

Download real-time application after build If the state of the ConfigurationDesk application is Matching platform connected (displayed in the status bar), the download starts immediately after the build process is completed. If Start real-time application is not selected, the downloaded real-time application's state is Stopped. The specified setting is saved with the current ConfigurationDesk application, i.e., the specified setting does not affect other ConfigurationDesk applications and is not modified if you change the default build or download behavior in the ConfigurationDesk Options dialog.

Unload a loaded application Unloads a loaded application from the connected platform and loads the new application. If the option is not set, the download is stopped if another application is loaded and running on the platform. The specified setting is saved with the current ConfigurationDesk application, i.e., the specified setting does not affect other ConfigurationDesk applications and is not modified if you change the default build or download behavior in the ConfigurationDesk Options dialog.

Start real-time application Automatically starts the real-time application when the download is completed. The specified setting is saved with the current ConfigurationDesk application, i.e., the specified setting does not affect other ConfigurationDesk applications and is not modified if you change the default build or download behavior in the ConfigurationDesk Options dialog.

Insert model name TRC file group Lets you specify to add a new model group to the TRC file.

If multiple models are assigned to the same application process, ConfigurationDesk automatically generates additional model groups in the TRC file. If you want an additional model group in TRC files for application processes to which only one model is assigned, you can select the Insert model name TRC file group checkbox.

Build configuration set options

ConfigurationDesk provides build configuration sets for you to set compiler options for the build process. Depending on the platform you are using, specific optimization options are preset. Compiler options are model-specific build settings that apply to all behavior models assigned to a build configuration set.

You can specify the following options:

Name (Available only for newly created build configuration sets) Lets you specify the name of the build configuration set.

Macros to be defined Lets you specify a list of macro definitions (#define) that are defined. The macros are entered as a blank-separated list, i.e., `MYMACRO_A=4 MYMACRO_B`.

Note

To perform ECU interfacing, some preconfigured macros are available. For more information, refer to Preconfigured Preprocessor Macros for ECU Interfacing (ConfigurationDesk Real-Time Implementation Guide).

Macros to be undefined Lets you specify a list of macro definitions that are to be deleted (#undef). The macros are entered as a blank-separated list, i.e., `MYMACRO_X MYMACRO_Y MYMACRO_Z`.

C compiler options Lets you specify C compiler options other than the optimization options.

C++ compiler options Lets you specify C++ compiler options other than the optimization options.

Compiler optimization set Lets you choose between the following options:

Option	Description
Default	The default set of compiler optimizations is used (`-O2 -finline-functions`). Note the following points: ▪ The compiler optimizes the execution time, not the code size. ▪ Optimization is performed without causing exceedingly long build times. Therefore, this is not necessarily the highest possible optimization level.
Custom	A user-defined set of compiler optimizations is used (defined via **User-defined compiler optimization options**).
None	Compiler optimization is not used.

User-defined compiler optimization options Lets you specify a set of compiler optimization options if the **Compiler optimization set** property is set to **Custom**.

For information on compiler options and optimization, refer to the documentation of the compiler.

Note

- Compiler options always apply to the entire application process, i.e., they cannot be specified separately for individual source files.
- To compile very large models in less time, it is sometimes necessary to reduce optimization options or to disable them completely.
- Do not specify optimization levels higher than the default.

  Do not specify compiler options that have an effect on data widths and memory alignments.

  For example, do not use the following options:
  - GNU GCC C Compiler:
    - `-fpack-struct`
    - `-fshort-enums`
    - Do not use the `__attribute__ ((__packed__))` directive for struct and enum definitions within your source code for variables that will be included in the variable description file.

Search paths Lets you specify a list of search paths. You can enter absolute or relative paths. Relative paths must be relative to the application root directory. During the build process, the paths are used by the make tool for searching source, header and library files. You must enter the search paths as a list, separated by semicolons. For example, `D:\\prj\appl\sourcefiles; .\headerfiles`.

You can also click the ... button to open a dialog that lets you add and remove search paths. Here, you can access a standard Browse For Folder dialog to select folders as search paths.

Custom source files Lets you specify a list of source files that must be compiled and linked to the final real-time application. You can enter file names or absolute or relative paths. Relative paths must be relative to the application root directory. The source files are entered as a list separated by semicolons. For example, `custcode1.cpp; .\sourcefiles\custcode2.cpp`.

You can also click the ... button to open a dialog that lets you add and remove custom source files. Here, you can access a standard Open dialog to select files.

Custom libraries Lets you specify a list of libraries that must be linked to the final real-time application. You can enter library names or absolute or relative paths. Relative paths must be relative to the application root directory. The library files are entered as a list separated by semicolons. For example, `libcustom1.a; .\libs\libcustom2.a`.

You can also click the ... button to open a dialog that lets you add and remove custom libraries. Here, you can access a standard Open dialog to select library files.

Build configuration set - bus custom code options

If you work with bus implementation software, such as the Bus Manager, the RTI CAN MultiMessage Blockset, or the FlexRay Configuration Package, you can use the bus custom code options provided by ConfigurationDesk to include user code implementations in a real-time application and/or in bus simulation containers.

Base directory Lets you specify a base directory for the files of user code implementations that must be added to the active ConfigurationDesk application. The base directory can be specified as a relative path or as an absolute path. In case of a relative path, the path must be relative to the application root directory. If no base directory is specified, the application root directory is used as the base directory for all source files, include files, and directories with a relative path.

Source files Lets you specify source files (C, CPP) of user code implementations that must be added to the active ConfigurationDesk application. When you build a real-time application or generate bus simulation containers, the code of the source files of user code implementations is compiled and included in the real-time application or bus simulation container, respectively. Source files can be specified as a relative path or as an absolute path. If you use a relative path, the path must be relative to the directory specified in the Base directory property. The files must be specified as a semicolon-separated list.

Include files and directories Lets you specify include files (H, HPP) and include directories of user code implementations that must be added to the active ConfigurationDesk application. When you build a real-time application or generate bus simulation containers, the code of the include files of user code implementations is compiled and included in the real-time application or bus simulation containers, respectively. If the include files that must be included are specified as relative paths in the related source files, you must also specify the related include directories. Include files and directories can be specified as a relative path or as an absolute path. If you use a relative path, the path must be relative to the directory specified on the **Base directory** property. The include files and directories must be specified as a semicolon-separated list. The specified include directories must end with the \ or / character. Consider the following example:

```
./custom/inc/;./custom/inc/a/header.h;./custom/inc/b/code.h
```

> **Note**
>
> All user code implementations that are added to one build configuration set apply to all application processes that are assigned to this build configuration set. You must manually ensure that each application process uses all modules of the Bus Custom Code interface that are used in the user code implementations (e.g., the `DsBusCustomCode_PduUserCode` module). If an application process does not use a required module, the following applies:
>
> - When you build a real-time application, the build process aborts.
> - When you generate bus simulation containers, a BSC file is generated for the application process but this file is invalid.

Non-build files (Applicable only to the generation of bus simulation containers) Lets you specify non-build files of user code implementations, such as copyright information or configuration files, that must be added to the active ConfigurationDesk application. When you generate bus simulation containers, the specified non-build files are included into the generated bus simulation containers. Non-build files can be specified as a relative path or as an absolute path. If you use a relative path, the path must be relative to the directory specified on the **Base directory** property. The files must be specified as a semicolon-separated list.

> **Note**
>
> Specified non-build files are included only in generated bus simulation containers, not in real-time applications.

Symbols

The build configuration hierarchy is displayed with the following symbols:

Symbol	Meaning
⛁	Indicates the branch for global build settings which apply to all build configuration sets.
▦	Indicates a build configuration set.
⚙	Indicates an application process.

Symbol	Meaning
	Indicates a behavior model.
	Indicates a Simulink implementation container.
	Indicates a bus simulation container.
fmu	Indicates a Functional Mock-up Unit (FMU).
	Indicates a V-ECU implementation.

Related topics

Basics

Specifying Options for the Build Process (ConfigurationDesk Real-Time Implementation Guide 🕮)
Working with User Code (ConfigurationDesk Bus Manager Implementation Guide 🕮)
Working with User Code (Bus Manager (Stand-Alone) Implementation Guide 🕮)

References

New – Build Configuration Set.. 332
New - Multiple Build Configuration Sets.. 332

Build Log Viewer

Purpose

To display all kinds of messages generated during the build process.

Result

The messages generated during the build process are shown.

Description

During the build process, messages in the **Build Log Viewer** indicate the progress of the build process in chronological order.

Messages in the **Build Log Viewer** are not saved when you close the project. However, ConfigurationDesk provides a context menu which lets you copy the available messages to the clipboard or save them in a text file for further use.

Related topics

Basics

Details on the Build Process (ConfigurationDesk Real-Time Implementation Guide 🕮)

Clear Complete Flash Memory

Purpose

To clear the flash memory of the selected platform in whole.

Result

ConfigurationDesk completely clears the flash memory of the selected platform. The real-time processor must be reset for this. If an application is running on the selected platform when you call the command, you are asked to unload the running application.

Clear Flash

Purpose

To clear the flash memory of the selected multiprocessor platform, in whole or in part.

Description

ConfigurationDesk opens the **Clear Flash** dialog, which lets you completely or partly clear the flash memory of the selected multiprocessor platform. The real-time processors must be reset for this. If an application is running on the selected platform when the **Clear Flash** dialog opens, you are asked to stop the running RTPs or unload the running applications, respectively.

The dialog displays all connected processing units and/or processor boards that belong to the selected platform and the applications currently loaded to their flash memories. If more than one clearing option is possible for the multiprocessor platform, you can choose the appropriate option. Finally you can select the processing units or processor boards for which the clear flash option is to be applied and start the clearing.

Clear Flash dialog

To completely or partly clear the flash memory of the selected multiprocessor platform.

Clear options Lets you select the clearing option to be applied to the multiprocessor platform. The available clearing options are offered for selection. If only one clearing option is possible, it is selected automatically and you cannot change it.

Processor Board/Processing Unit Displays the processor boards and/or processing units belonging to the multiprocessor system.

Application in Flash Displays for each processor board or processing unit the path of the application that is loaded to the flash memory. If there is no application in the flash memory or after clearing the flash memory, 'No application loaded.' is displayed.

Select Lets you select the individual processor boards or processing units to which the selected clearing option is to be applied.

Clear Lets you execute the selected clearing option for the selected processing units/processor boards.

Generate Model Code

Purpose

To start the model code generation including complete analysis.

Result

You generated the model code. The model topology as well as predefined elements to model an executable application (runnable functions, events, tasks) are updated in ConfigurationDesk.

Description

After the model code generation is finished, its results can be found at `<ConfigurationDesk Application>\Components\<model>`.

The model code generation includes a complete model analysis (refer to Analyze Simulink Model (Including Task Information) on page 171).

Results from a previous model code generation or build process are overwritten. The generated model code is used during a later build process for models for which **Skip model code generation** is enabled (see Build Configuration Table on page 323). For details on the build process and build results, refer to Building Real-Time Applications (ConfigurationDesk Real-Time Implementation Guide).

Related topics

Basics

Building Real-Time Applications (ConfigurationDesk Real-Time Implementation Guide)

References

Analyze Simulink Model (Including Task Information)..171
Build Configuration Table...323

Load to Matching Platform

Purpose

To load a real-time application to the connected platform if the active ConfigurationDesk application is in **Matching platform connected** state.

Result

The real-time application of the active ConfigurationDesk application is loaded to the connected platform. The real-time application is not started automatically.

Description

ConfigurationDesk checks if a real-time application is already loaded on the connected platform. If there is, you are prompted to unload it and load the new real-time application. After the selected real-time application is loaded, it is not started automatically, which is indicated by a 🟥 symbol.

Related topics

HowTos

How to Download a Real-Time Application (ConfigurationDesk Real-Time Implementation Guide)

Load to Platform <Platform> / Download to <Platform>

Purpose

To download the real-time application onto the selected platform.

Result

The real-time application is downloaded.

Description

If several platforms are connected to ConfigurationDesk, a submenu of the context menu displays all the platforms which are compatible to the real-time application and to which the real-time application can be loaded.

Related topics

HowTos

How to Download a Real-Time Application (ConfigurationDesk Real-Time Implementation Guide)

New – Build Configuration Set

Purpose

To add a new build configuration set to the Build Configuration table.

Result

A new build configuration set is added to the Build Configuration table. New build configuration sets are numbered consecutively. You can rename them. Their names must be unique. Otherwise, a conflict is generated.

Description

ConfigurationDesk provides build configuration sets for you to specify the C/C++ compiler options for the build process. For basic information on build configuration sets, refer to Specifying Options for the Build Process (ConfigurationDesk Real-Time Implementation Guide 🕮). For details on the properties of the build configuration sets, refer to Build Configuration Table on page 323.

Related topics

Basics

Specifying Options for the Build Process (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Build Configuration Table..323
New - Multiple Build Configuration Sets..332

New - Multiple Build Configuration Sets

Purpose

To add several build configuration sets to the Build Configuration table at once.

Result

ConfigurationDesk opens a Create Multiple Build Configuration Set dialog.

Description

For basic information on build configuration sets, refer to Specifying Options for the Build Process (ConfigurationDesk Real-Time Implementation Guide 🕮). For details on the properties of the build configuration sets, refer to Build Configuration Table on page 323.

Create Multiple Build Configuration Set dialog

Name pattern Lets you define the name pattern for the new build configuration sets. You can use the backslash \ as placeholder to add an incremented number to the name of each new build configuration set. This avoids duplicate names.

> **Note**
>
> The names of build configuration sets have to be unique.

Number of instances Lets you define the number of build configuration sets to add to the Build Configuration table.

Related topics

Basics

Specifying Options for the Build Process (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Build Configuration Table..323
New – Build Configuration Set..332

Real-Time Application - Load

Purpose

To load a real-time application to the RAM of the selected platform without starting it automatically.

Result

The selected real-time application is loaded to the RAM of the selected platform. The real-time application is not started automatically.

> **Note**
>
> When an application was built with Release 2017-B, loading it to a SCALEXIO system does not require a SCALEXIO_RTLIB license on the host PC as was required in previous releases.
>
> When an application was built with Release 2017-A or earlier, loading it to a SCALEXIO system is not possible if you are using an experiment software of Release 2017-B.
>
> Refer to Basics on Experimenting Using a SCALEXIO System (SCALEXIO – Hardware and Software Overview 🕮).

Description

To download a real-time application to the RAM of the selected platform, right-click the desired platform and select the **Real-Time Application - Load** command from the context menu. The **Select Real-Time Application** dialog opens for you to select the RTA or SDF file of the real-time application that you want to load. After you select the RTA or SDF file and click **Open**, ConfigurationDesk checks if a real-time application is already loaded on the selected platform. If there is, you are prompted to unload it and load the new real-time application. After the selected real-time application is loaded, it is not started automatically, which is indicated by a 🟥 symbol.

Related topics

HowTos

How to Download a Real-Time Application (ConfigurationDesk Real-Time Implementation Guide)

Real-Time Application - Load and Start

Purpose

To load a real-time application to the RAM of the selected platform and start it automatically.

Result

The selected real-time application is loaded to the RAM of the selected platform. The real-time application is started automatically.

Note

When an application was built with Release 2017-B, loading it to a SCALEXIO system does not require a SCALEXIO_RTLIB license on the host PC as was required in previous releases.

When an application was built with Release 2017-A or earlier, loading it to a SCALEXIO system is not possible if you are using an experiment software of Release 2017-B.

Refer to Basics on Experimenting Using a SCALEXIO System (SCALEXIO – Hardware and Software Overview).

Description

To download a real-time application to the RAM of the selected platform, right-click the desired platform and select the **Real-Time Application - Load and Start** command from the context menu. The **Select Real-Time Application** dialog opens for you to select the RTA or SDF file of the real-time application that you want to load. After you select the RTA or SDF file and click **Open**, ConfigurationDesk checks if a real-time application is already loaded on the selected platform. If there is, you are prompted to unload it and load the new real-time application. After the selected real-time application is loaded, it is started automatically, which is indicated by a ▶ symbol.

Related topics

HowTos

How to Download a Real-Time Application (ConfigurationDesk Real-Time Implementation Guide 📖)

Real-Time Application - Load to Flash

Purpose

To load an application to the flash memory and the RAM of the selected platform without starting it automatically.

Result

The selected real-time application is loaded to the flash memory and the RAM of the selected platform. The real-time application is not started automatically.

> **Note**
>
> An application that was already in the RAM is overwritten when you load an application to the flash memory.

Description

To download a real-time application to the flash memory and the RAM of the selected platform, right-click the desired platform and select the **Real-Time Application - Load to Flash** command from the context menu. The **Select Real-Time Application** dialog opens for you to select the RTA or SDF file of the real-time application that you want to load. After you select the RTA or SDF file and click **Open**, ConfigurationDesk checks if a real-time application is already loaded on the selected platform. If there is, you are prompted to unload it and load the new real-time application. After the selected real-time application is loaded, it is not started automatically, which is indicated by a F symbol. If you reboot the platform, the real-time application in the flash memory is started automatically.

Related topics

HowTos

How to Download a Real-Time Application (ConfigurationDesk Real-Time Implementation Guide 📖)

Real-Time Application - Load to Flash and Start

Purpose

To load an application to the flash memory and the RAM of the selected platform and start it automatically.

Result

The selected real-time application is loaded to the flash memory and the RAM of the selected platform. The real-time application is started automatically.

> **Note**
>
> An application that was already in the RAM is overwritten when you load an application to the flash memory.

Description

To download a real-time application to the flash memory and the RAM of the selected platform and start it automatically, right-click the desired platform and select the **Real-Time Application - Load to Flash and Start** command from the context menu. The **Select Real-Time Application** dialog opens for you to select the RTA or SDF file of the real-time application that you want to load. After you select the RTA or SDF file and click **Open**, ConfigurationDesk checks if a real-time application is already loaded on the selected platform. If there is, you are prompted to unload it and load the new real-time application. The selected real-time application is then loaded to the flash memory and the RAM of the selected platform, and started automatically, which is indicated by a F symbol. If you reboot the platform, the real-time application in the flash memory is started automatically.

Related topics

HowTos

How to Download a Real-Time Application (ConfigurationDesk Real-Time Implementation Guide)

Reload (Real-Time Application)

Purpose

To reload the loaded application to the RAM without starting it automatically.

Result

The application that is loaded on the platform is reloaded to the RAM. The real-time application is not started automatically.

Description

The application is reloaded to the RAM from the same file path that was previously used to downloaded it. real-time applications can be reloaded independently of their state (Running or Stopped). After the selected real-time application is reloaded, it is not started automatically, which is indicated by a 🟥 symbol.

Reload and Start (Real-Time Application)

Purpose

To reload the loaded application to the RAM and start it automatically.

Result

The application that is loaded on the platform is reloaded to the RAM and started automatically.

Description

The application is reloaded to the RAM from the same file path that was previously used to downloaded it. real-time applications can be reloaded independently of their state (Running or Stopped). After the selected real-time application is reloaded, it is started automatically, which is indicated by a ▶ symbol.

Reload to Flash

Purpose

To reload the loaded application to the flash memory without starting it automatically.

Result

The application that is loaded on the platform is reloaded to the flash memory. The real-time application is not started automatically.

> **Note**
>
> An application that was already in the flash memory is overwritten when you reload the application to the flash memory.

Description

The application is reloaded to the flash memory from the same file path that was previously used to downloaded it. real-time applications can be reloaded independently of their state (Running or Stopped). After the selected real-time application is reloaded, it is not started automatically, which is indicated by a 🟥F symbol. If you reboot the platform, the real-time application in the flash memory is started automatically.

Related topics

HowTos

How to Download a Real-Time Application (ConfigurationDesk Real-Time Implementation Guide)

Reload to Flash and Start

Purpose

To reload the loaded application to the flash memory and start it automatically.

Result

The application that is loaded on the platform is reloaded to the flash memory and started automatically.

> **Note**
>
> An application that was already in the flash memory is overwritten when you reload the application to the flash memory.

Description

The application is reloaded to the flash memory from the same file path that was previously used to downloaded it. Real-time applications can be reloaded independently of their state (Running or Stopped). After the selected real-time application is reloaded, it is started automatically, which is indicated by a F symbol.

Related topics

HowTos

How to Download a Real-Time Application (ConfigurationDesk Real-Time Implementation Guide)

Start (Real–Time Application)

Purpose

To start the execution of the real-time application on the selected platform.

Result

The real-time application is started, which is indicated by a green arrow next to the application's name in the **Platform Manager**.

Description

If a real-time application is already downloaded to the selected platform and is stopped, it can be started.

Start Build

Purpose

To start the build process.

Result

The build process is started.

> **Note**
>
> If MATLAB is not running when you start the build process, it is opened automatically, if required. If you have connected several MATLAB installations to your dSPACE installation, you must select one of them as the preferred connection. For details, refer to Introduction to Connecting a MATLAB Installation (Managing dSPACE Software Installations).

Description

During the build process, messages in the **Build Log Viewer** indicate the progress of the build process in chronological order.

Related topics

HowTos

How to Start the Build Process (ConfigurationDesk Real-Time Implementation Guide)

References

Stop Build.. 340

Stop (Real–Time Application)

Purpose

To stop execution of the real-time application running on the selected platform.

Result

The execution of the real-time application is stopped, which is indicated by a red square next to the application's name in the **Platform Manager**.

Description

When the real-time application is stopped on the selected platform, it can be started or unloaded, for example.

Stop Build

Purpose

To stop the build process.

Description

Requests the build process to be stopped. The build process terminates before the next build step. Existing build results from a previous build process are not overwritten.

Related topics

HowTos

How to Start the Build Process (ConfigurationDesk Real-Time Implementation Guide)

References

Start Build...339

Unload (Real–Time Application)

Purpose

To unload the real-time application from the selected platform.

Result

The real-time application is unloaded. It is not displayed in the **Platform Manager** any longer.

Description

Real-time applications can be unloaded independently of their state (Running or Stopped).

Bus Manager Handling

Introduction

The Bus Manager lets you configure bus communication for simulation, inspection, and/or manipulation purposes. Via the **Buses Browser** and the Bus Manager-specific tables, you can import communication matrices and handle bus configurations. Bus configurations let you implement bus communication in the signal chain to simulate, inspect, and/or manipulate it during run time. Additionally, bus configurations let you specify gateways to exchange CAN frames between two communication clusters. **Bus Configuration** function blocks provide a graphical view on the ports of bus configurations.

Where to go from here

Information in this section

Add CAN Filter Rule

Purpose

To add a CAN filter rule to a selected filter that is available for a frame capture or frame gateway in a bus configuration.

Result

A CAN filter rule is added to the selected filter.

Description

Frame captures and frame gateways that are available for a bus configuration provide filters. By adding one or more CAN filter rules to a filter, you can specify the CAN frames to be captured or gatewayed.

For more information on filtering CAN frames for capturing/gatewaying, refer to:

- Bus Manager in ConfigurationDesk: Specifying Filters for Frame Captures and Frame Gateways (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Specifying Filters for Frame Captures and Frame Gateways (Bus Manager (Stand-Alone) Implementation Guide)

For basic information on frame captures and gateways, refer to:

- Bus Manager in ConfigurationDesk: Specifying CAN Frame Captures and Gateways (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Specifying CAN Frame Captures and Gateways (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Add Feature - Filter Control 360
Add Feature - Frame Gateway Direction 363
Add Frame Gateway 384

Add Communication Matrix

Purpose

To add a communication matrix to a ConfigurationDesk application.

Result

An Add Communication Matrix dialog opens for you to select the communication matrix you want to add.

Description

An application can contain multiple communication matrices. The communication matrices can have identical names but they must differ in content.

You can select one communication matrix at a time to add it to the application. If a communication matrix with identical content already exists within the application, the selected communication matrix is not added. Via the **Buses Browser**, you can access all the communication matrix elements, display their properties, and assign the communication matrix or parts of it to bus configurations.

You can filter for communication matrix elements. If an added communication matrix is not displayed or you are missing communication matrix elements, make sure that all the filter options are deactivated. For an overview of the available filter options for communication matrix elements, refer to:

- Bus Manager in ConfigurationDesk: Working with Communication Matrices (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Working with Communication Matrices (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

Basics

Elements of the Bus Manager (Bus Manager (Stand-Alone) Implementation Guide)
Elements of the Bus Manager (ConfigurationDesk Bus Manager Implementation Guide)

References

Elements and Symbols of Bus Configurations and Communication Matrices........................ 429

Add Element to Communication Matrix

Purpose

To access different commands for adding elements to the selected communication matrix element.

Result

Different commands for adding elements to the selected communication matrix element are available in a submenu.

Description

Depending on the selected element and the available communication matrix elements, the commands available in the submenu are:

- Add Element to Communication Matrix - Bus ISignal on page 351
- Add Element to Communication Matrix - Bus ISignal IPDU (RX) on page 351
- Add Element to Communication Matrix - Bus ISignal IPDU (TX) on page 352
- Add Element to Communication Matrix - Cyclic Timing on page 353

- Add Element to Communication Matrix - Time Offset on page 353
- Add Element to Communication Matrix - Time Period on page 354

Add Element to Communication Matrix - Bus ISignal

Purpose

To add a user-defined ISignal to a selected CAN PDU of a communication matrix.

Result

A user-defined ISignal is added to the selected CAN PDU of the communication matrix.

Description

The user-defined ISignal is added to all instances of the selected PDU, regardless of the direction (TX or RX). For example, if you add an ISignal to a TX instance of a PDU, the ISignal is added to all instances of the PDU, including the RX instances. The direction of the ISignal is adjusted automatically. The ISignal is not automatically assigned to bus configurations, even if instances of the related PDU are assigned.

The user-defined ISignal is added with a default name, i.e., User_defined_ISignal_<number>. Required elements such as a computation method and an ISignal-to-IPDU mapping are added automatically. For more information, refer to:

- Bus Manager in ConfigurationDesk: Adding ISignals to CAN PDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Adding ISignals to CAN PDUs (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Filter for Changes / Filter for Changes to Communication Matrices..447
Undo All Changes..452
Undo Changes to Communication Matrix...453

Add Element to Communication Matrix - Bus ISignal IPDU (RX)

Purpose

To add a user-defined RX ISignal IPDU to a selected CAN channel of a communication matrix.

Result

A user-defined RX ISignal IPDU is added to the selected CAN channel of the communication matrix.

Description

The user-defined RX ISignal IPDU is added with a default name, i.e., User_defined_IPDU_<number>. Required elements such as a frame and frame triggering are added automatically. The ISignal IPDU is not automatically assigned to bus configurations. For more information, refer to:

- Bus Manager in ConfigurationDesk: Adding ISignal IPDUs to CAN Channels (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Adding ISignal IPDUs to CAN Channels (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Related topics

References

Filter for Changes / Filter for Changes to Communication Matrices..447
Undo All Changes..452
Undo Changes to Communication Matrix..453

Add Element to Communication Matrix - Bus ISignal IPDU (TX)

Purpose

To add a user-defined TX ISignal IPDU to a selected CAN channel of a communication matrix.

Result

A user-defined TX ISignal IPDU is added to the selected CAN channel of the communication matrix.

Description

The user-defined TX ISignal IPDU is added with a default name, i.e., User_defined_IPDU_<number>. Required elements such as a frame and frame triggering are added automatically. The ISignal IPDU is not automatically assigned to bus configurations. For more information, refer to:

- Bus Manager in ConfigurationDesk: Adding ISignal IPDUs to CAN Channels (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Adding ISignal IPDUs to CAN Channels (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Related topics

References

Filter for Changes / Filter for Changes to Communication Matrices..447
Undo All Changes...452
Undo Changes to Communication Matrix..453

Add Element to Communication Matrix - Cyclic Timing

Purpose

To add a Cyclic Timing element to a selected basic PDU.

Result

A Cyclic Timing element is added to the selected basic PDU. Additionally, a Time Offset and Time Period element are added automatically.

Description

If the communication matrix does not specify a cyclic transmission for a basic PDU, you can add a Cyclic Timing element to specify cyclic transmission. Via the related Time Offset and Time Period elements, you can specify a time offset (delay time) and a time period (cycle time), respectively.

When you add a Cyclic Timing element to a basic PDU, it affects all instances of the PDU, independently from the direction (TX or RX). For more information, refer to:

- Bus Manager in ConfigurationDesk: Adding Cyclic Timing Elements to Basic PDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Adding Cyclic Timing Elements to Basic PDUs (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Filter for Changes / Filter for Changes to Communication Matrices..447
Undo All Changes...452
Undo Changes to Communication Matrix..453

Add Element to Communication Matrix - Time Offset

Purpose

To add a Time Offset element to a selected Cyclic Timing element of a basic PDU.

Result

A **Time Offset** element is added to the selected **Cyclic Timing** element of the basic PDU.

Description

If the communication matrix does not specify a time offset (delay time) for the cyclic transmission of a basic PDU, you can add a **Time Offset** element to the **Cyclic Timing** element of the PDU to specify the time offset.

When you add a **Time Offset** element to the **Cyclic Timing** element of a basic PDU, it affects all instances of the PDU, independently from the direction (TX or RX). For more information, refer to:

- Bus Manager in ConfigurationDesk: Adding Cyclic Timing Elements to Basic PDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Adding Cyclic Timing Elements to Basic PDUs (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Filter for Changes / Filter for Changes to Communication Matrices.. 447
Undo All Changes.. 452
Undo Changes to Communication Matrix.. 453

Add Element to Communication Matrix - Time Period

Purpose

To add a **Time Period** element to a selected **Cyclic Timing** element of a basic PDU.

Result

A **Time Period** element is added to the selected **Cyclic Timing** element of the basic PDU.

Description

If the communication matrix does not specify a time period (cycle time) for the cyclic transmission of a basic PDU, you can add a **Time Period** element to the **Cyclic Timing** element of the PDU to specify the time period.

When you add a **Time Period** element to the **Cyclic Timing** element of a basic PDU, it affects all instances of the PDU, independently from the direction (TX or RX). For more information, refer to:

- Bus Manager in ConfigurationDesk: Adding Cyclic Timing Elements to Basic PDUs (ConfigurationDesk Bus Manager Implementation Guide)

- Bus Manager (stand-alone): Adding Cyclic Timing Elements to Basic PDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Related topics

References

Add Element to Working View

Purpose

To access different commands for adding the selected bus configuration to a working view.

Result

Different commands for adding the selected bus configuration to a working view are available in a submenu.

Description

The commands available in the submenu are:

- Add Element to Working View - <Working View> on page 355
- Add Element to Working View - Create and Add on page 356

Add Element to Working View - <Working View>

Purpose

To add the selected bus configuration to the selected working view.

Result

The selected bus configuration is added to the selected working view.

Description

The bus configuration is added to the working view but is not highlighted in it. The working view remains closed if it was not opened yet. If the working view is open in the background, its position remains unchanged, i.e., it is not displayed in the foreground.

Add Element to Working View - Create and Add

Purpose

To create a new working view and add the selected bus configuration to it.

Result

A new working view containing the selected bus configuration is added to the **Working View Manager**.

Description

The new working view has a default name. It is added to the Working View Manager but not opened in the **Signal Chain Browser**. You can rename the working view, and copy & paste contents of other working views to it.

Add Feature

Purpose

To access different commands for adding bus configuration features to a selected bus configuration element.

Result

Different commands for adding bus configuration features to the selected bus configuration element are available in a submenu.

Description

Depending on the selected element, the commands available in the submenu are:

- Add Feature - Bus Configuration Enable on page 357
- Add Feature - Communication Controller Enable on page 358
- Add Feature - Counter Signal on page 359
- Add Feature - Filter Control on page 360
- Add Feature - Frame Access on page 361
- Add Feature - Frame Capture Data on page 362
- Add Feature - Frame Gateway Direction on page 363
- Add Feature - Frame Length on page 364
- Add Feature - GTS Time Base Data on page 364
- Add Feature - GTS Transmission Control on page 365
- Add Feature - GTS Validation on page 366
- Add Feature - ISignal Group End-to-End Protection Status on page 367
- Add Feature - ISignal Offset Value on page 368
- Add Feature - ISignal Overwrite Value on page 369
- Add Feature - ISignal Value on page 370
- Add Feature - J1939 Network Management Enable on page 371

- Add Feature - LIN Schedule Table on page 371
- Add Feature - LIN Wake-Up on page 372
- Add Feature - PDU Cyclic Timing Control on page 373
- Add Feature - PDU Enable on page 374
- Add Feature - PDU Length on page 375
- Add Feature - PDU Raw Data on page 375
- Add Feature - PDU RX Interrupt on page 376
- Add Feature - PDU RX Status on page 377
- Add Feature - PDU Trigger on page 378
- Add Feature - PDU User Code on page 378
- Add Feature - SecOC on page 379
- Add Feature - SecOC Authenticator Invalidation on page 380
- Add Feature - SecOC Freshness Overwrite Value on page 381
- Add Feature - Suspend Frame Transmission on page 382

Add Feature - Bus Configuration Enable

Purpose

To add the **Bus Configuration Enable** feature to a selected bus configuration.

Result

The **Bus Configuration Enable** feature is added to the selected bus configuration. This has the following effects:

- In the **Bus Configurations** table, a **Bus Configuration Enable** node is added to the bus configuration.
- An Enable function port is available for the bus configuration.

Description

Adding this feature to a selected bus configuration lets you completely or partly enable and disable the bus configuration via the added function port.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Enabling and Disabling Bus Configurations (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Enabling and Disabling Bus Configurations (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - Communication Controller Enable

Purpose

To add the **Communication Controller Enable** feature to a communication controller that is selected in a bus configuration.

Result

The **Communication Controller Enable** feature is added to the selected communication controller. This has the following effects:

- In the bus configuration, two function ports are available for the communication controller:
  - Communication Controller Enable function port
  - Communication Controller Enable State function port
- In the **Bus Configurations** table, a **Communication Controller Enable** node is added to the communication controller.

Description

Adding this feature to a selected communication controller lets you enable and disable the communication controller.

Each network node of a communication cluster has one communication controller (CAN communication controller, LIN slave communication controller, or LIN master communication controller). Disabling a communication controller disables the communication of the related network node, i.e., the network node does not transmit frames on the bus and received frames are not evaluated. If you disable a LIN master communication controller, the LIN master is disabled and therefore the communication of the entire LIN cluster stops. Disabling a LIN slave or CAN communication controller does not directly affect other network nodes.

The Communication Controller Enable function port lets you enable and disable the communication controller. The Communication Controller Enable State function port provides the state of the controller to a behavior model, for example.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Enabling and Disabling Communication Controllers (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Enabling and Disabling Communication Controllers (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)
Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - Counter Signal

Purpose

To add the **Counter Signal** feature to an ISignal that is selected in a bus configuration.

Result

The **Counter Signal** feature is added to the selected ISignal. This has the following effects:

- If the ISignal is an RX ISignal, a Counter State function port is available for the ISignal in the bus configuration.
- In the **Bus Configurations** table, a **Counter Signal** node is added to the ISignal.
- In the **Bus Simulation Features** table, the **Maximum Value**, **Minimum Value**, **Increment Value**, **Initial Value**, and **Step Length** columns are enabled.

Description

Adding this feature to a selected ISignal lets you configure the ISignal as a counter signal. You can use counter signals to detect lost PDUs or test PDU counters that are implemented in the ECU under test, for example.

You cannot use the **Counter Signal** feature in parallel with the following features:

- **ISignal Value** (applies only to TX ISignals and TX PDUs)
- **PDU Raw Data** (applies only to TX ISignals and TX PDUs)
- **Frame Access**

If you add the **Counter Signal** feature to an ISignal that is affected by any of these features, conflicts occur.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Working with Counter Signals (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Working with Counter Signals (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - Filter Control

Purpose

To add the Filter Control feature to a selected filter that is available for a frame capture or frame gateway in a bus configuration.

Result

The Filter Control feature is added to the selected filter. This has the following effects:

- An Enable function port is available for the filter in the bus configuration.
- In the Bus Configurations table, a Filter Control node is added to the filter.

Description

Adding this feature to a selected filter lets you specify the filter mode of the filter. Additionally, you can enable and disable the filter via the added function port.

For more information on the Filter Control feature, refer to:

- Bus Manager in ConfigurationDesk: Controlling Filters of Frame Captures and Frame Gateways (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Controlling Filters of Frame Captures and Frame Gateways (Bus Manager (Stand-Alone) Implementation Guide)

For basic information on frame captures and gateways, refer to:

- Bus Manager in ConfigurationDesk: Specifying CAN Frame Captures and Gateways (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Specifying CAN Frame Captures and Gateways (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

Basics

Add Frame Capture.......... 383

References

Add CAN Filter Rule.......... 349
Add Feature - Frame Gateway Direction.......... 363
Add Frame Gateway.......... 384
Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - Frame Access

Purpose

To add the Frame Access feature to a frame of a CAN PDU that is selected in a bus configuration.

Result

The Frame Access feature is added to the selected PDU. This has the following effects:

- Depending on the direction of the PDU (TX, RX), the following function ports are available for the PDU in the bus configuration:

Function Ports Available for	
TX PDUs	**RX PDUs**
Trigger function port	State function port
TX Length function port	RX Length function port
TX Raw Data function port	RX Raw Data function port
Identifier function port	Identifier function port
Extended Addressing function port	Extended Addressing function port
CAN FD Frame Support function port	CAN FD Frame Support function port
Bit Rate Switch function port	Bit Rate Switch function port

- In the Bus Configurations table, a Frame Access node is added to the PDU.
- Depending on the direction of the PDU, the Trigger Mode and/or Maximum Length columns are enabled in the Bus Simulation Features table.

Description

Adding this feature to a selected PDU lets you configure settings of the related frame, and access frame settings via the related function ports during run time.

Note

- Cyclic timings that are specified in the communication matrix are ignored. Instead, the transmission of the PDU is triggered only according to the settings specified for the Frame Access feature.
- Bit patterns for unused bits that are specified in the communication matrix are ignored. Instead, 0 is used as the bit pattern for unused bits. This applies to the PDU to which you add the Frame Access feature and to its included PDUs.

You cannot use the Frame Access feature with any other PDU- or ISignal-related bus configuration feature in parallel. If you add the Frame Access feature to a PDU that is affected by another feature, conflicts occur.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Accessing CAN Frame Settings (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Accessing CAN Frame Settings (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties 🕮)
Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties 🕮)

Add Feature - Frame Capture Data

Purpose

To add the **Frame Capture Data** feature to a frame capture that is selected in a bus configuration.

Result

The **Frame Capture Data** feature is added to the selected frame capture. This has the following effects:

- The following function ports are available for the frame capture in the bus configuration:
  - State function port
  - Time function port
  - Length function port
  - Raw Data function port
  - Identifier function port
  - Extended Addressing function port
  - CAN FD Frame Support function port
  - Bit Rate Switch function port
- In the **Bus Configurations** table, a **Frame Capture Data** node is added to the frame capture.

Description

Adding this feature to a selected frame capture lets you access the data of all captured frames via the related function ports during run time.

For more information on the **Frame Capture Data** feature, refer to:

- Bus Manager in ConfigurationDesk: Accessing the Data of Captured Frames (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Accessing the Data of Captured Frames (Bus Manager (Stand-Alone) Implementation Guide 🕮)

For basic information on frame captures, refer to:

- Bus Manager in ConfigurationDesk: Capturing CAN Frames (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Capturing CAN Frames (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Related topics

References

Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - Frame Gateway Direction

Purpose

To add the **Frame Gateway Direction** feature to a selected frame gateway of a bus configuration .

Result

The **Frame Gateway Direction** feature is added to the selected frame gateway. This has the following effects:

- A Direction function port is available for the frame gateway in the bus configuration.
- In the **Bus Configurations** table, a **Frame Gateway Direction** node is added to the frame gateway.

Description

Adding this feature to a selected frame gateway lets you specify the gateway direction or disable the gateway via the added function port .

For more information on the **Frame Gateway Direction** feature, refer to:

- Bus Manager in ConfigurationDesk: Specifying the Direction of CAN Frame Gateways (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Specifying the Direction of CAN Frame Gateways (Bus Manager (Stand-Alone) Implementation Guide)

For basic information on frame gateways, refer to:

- Bus Manager in ConfigurationDesk: Specifying CAN Gateways (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Specifying CAN Gateways (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Add CAN Filter Rule..349
Add Feature - Filter Control..............................360
Add Frame Gateway..384
Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - Frame Length

Purpose

To add the **Frame Length** feature to a frame of a PDU that is selected in a bus configuration.

Result

The **Frame Length** feature is added to the selected PDU. This has the following effects:

- The following function ports are available for the PDU in the bus configuration:
  - Current Countdown Value function port
  - Countdown Start Value function port
  - Enable function port
  - Enable State function port
  - Length function port
- In the **Bus Configurations** table, a **Frame Length** node is added to the PDU.
- In the **Bus Manipulation Features** table, the **Countdown Start Value**, **Padding Value**, and **Length** columns are enabled.

Description

Adding this feature to a selected PDU lets you manipulate the payload length of the related frame permanently or temporarily.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Manipulating the Payload Length of CAN Frames (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Manipulating the Payload Length of CAN Frames (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)
Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - GTS Time Base Data

Purpose

To add the **GTS Time Base Data** feature to a global time domain that is selected in a bus configuration.

Result

The **GTS Time Base Data** feature is added to the selected global time domain. This has the following effects:

- In the bus configuration, the following function ports are available for the global time domain:
  - Time function port
  - Status function port (only for time masters)
  - Synchronized to Gateway function port (only for time slaves)
  - User Bytes function port
  - User Data Length function port
- In the **Bus Configurations** table, a **GTS Time Base Data** node is added to the global time domain.

Description

Adding this feature to a selected global time domain lets you access the time base data of the related time master or time slave via the added function ports.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Accessing the Time Base Data of Time Masters and Time Slaves (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Accessing the Time Base Data of Time Masters and Time Slaves (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - GTS Transmission Control

Purpose

To add the **GTS Transmission Control** feature to a TX global time domain that is selected in a bus configuration.

Result

The **GTS Transmission Control** feature is added to the selected TX global time domain. This has the following effects:

- In the bus configuration, the following function ports are available for the TX global time domain:
  - Period function port
  - Trigger function port
- In the **Bus Configurations** table, a **GTS Transmission Control** node is added to the TX global time domain.

Description

Adding this feature to a selected TX global time domain lets you control the timing of time synchronization via the added function ports.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Controlling the Timing of Time Synchronization (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Controlling the Timing of Time Synchronization (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - GTS Validation

Purpose

To add the **GTS Validation** feature to an RX global time domain that is selected in a bus configuration.

Result

The **GTS Validation** feature is added to the selected RX global time domain. This has the following effects:

- In the bus configuration, the following function ports are available for the RX global time domain:
  - Partial Validation function port
  - Result function port
- In the **Bus Configurations** table, a **GTS Validation** node is added to the RX global time domain.

Description

Adding this feature to a selected RX global time domain lets you access the results of validity checks that are performed for the time synchronization data received by time slaves via the added function ports.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Accessing Validity Checks for Time Synchronization Messages (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Accessing Validity Checks for Time Synchronization Messages (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)
Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - ISignal Group End-to-End Protection Status

Purpose

To add the **ISignal Group End-to-End Protection Status** feature to an end-to-end-protected ISignal group that is selected in a bus configuration.

Result

The **ISignal Group End-to-End Protection Status** feature is added to the selected ISignal group. This has the following effects:

- In the bus configuration, two function ports are available for the ISignal group:
  - Calculated CRC function port
  - State function port
- In the **Bus Configurations** table, a **ISignal Group End-to-End Protection Status** node is added to the ISignal group.

Description

Adding this feature to a selected ISignal group lets you observe the status of the received end-to-end protection information via the added function ports during run time.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Observing the Status of Received End-to-End-Protected ISignal Groups (ConfigurationDesk Bus Manager Implementation Guide)

- Bus Manager (stand-alone): Observing the Status of Received End-to-End-Protected ISignal Groups (Bus Manager (Stand-Alone) Implementation Guide 📖)

Related topics

References

Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties 📖)

Add Feature - ISignal Offset Value

Purpose

To add the ISignal Offset Value feature to an ISignal ⧉ that is selected in a bus configuration ⧉.

Result

The ISignal Offset Value feature is added to the selected ISignal. This has the following effects:

- If not already available, a feature switch is added to the ISignal.
- The following function ports ⧉ are available for the ISignal in the bus configuration:
  - Offset Value function port
  - Minimum ISignal Value function port
  - Maximum ISignal Value function port
- In the Bus Configurations table, an ISignal Offset Value node is added to the ISignal.
- In the Bus Manipulation Features table, the following columns are enabled:
  - Offset Value
  - Minimum ISignal Value
  - Maximum ISignal Value
  - Recalculate End-to-End Protection (Offset) (only if the ISignal is included in an end-to-end-protected ISignal group)
  - Recalculate SecOC Information (Offset) (only if the ISignal is included in a PDU that is involved in secure onboard communication)

Description

Adding this feature to a selected ISignal lets you manipulate the ISignal by adding an offset value to the original ISignal value.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Adding Offset Values to ISignal Values (ConfigurationDesk Bus Manager Implementation Guide 📖)

- Bus Manager (stand-alone): Adding Offset Values to ISignal Values (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties 🕮)

Add Feature - ISignal Overwrite Value

Purpose

To add the **ISignal Overwrite Value** feature to an ISignal that is selected in a bus configuration.

Result

The **ISignal Overwrite Value** feature is added to the selected ISignal. This has the following effects:

- If not already available, a feature switch is added to the ISignal.
- The Overwrite Value function port is available for the ISignal in the bus configuration.
- In the **Bus Configurations** table, an **ISignal Overwrite Value** node is added to the ISignal.
- In the **Bus Manipulation Features** table, the following columns are enabled:
  - **Overwrite Value**
  - **Recalculate End-to-End Protection (Overwrite)** (only if the ISignal is included in an end-to-end-protected ISignal group)
  - **Recalculate SecOC Information (Overwrite)** (only if the ISignal is included in a PDU that is involved in secure onboard communication)

Description

Adding this feature to a selected ISignal lets you manipulate the ISignal by overwriting the original ISignal value with a user-defined value.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Overwriting ISignal Values (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Overwriting ISignal Values (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - ISignal Value

Purpose

To add the **ISignal Value** feature to an ISignal that is selected in a bus configuration.

Result

The **ISignal Value** feature is added to the selected ISignal. This has the following effects:

- Depending on the direction of the ISignal (TX or RX), a TX ISignal Value or RX ISignal Value function port is available for the ISignal in the bus configuration.
- In the **Bus Configurations** table, an **ISignal Value** node is added to the ISignal.

Description

Adding this feature to a selected ISignal lets you access the ISignal value via the added function port. Via the function port properties, you can specify or change the ISignal value.

You cannot use the **ISignal Value** feature in parallel with the following features:

- **Counter Signal** (applies only to TX ISignals and TX PDUs)
- **PDU Raw Data** (applies only to TX ISignals and TX PDUs)
- **Frame Access**

If you add the **ISignal Value** feature to an ISignal that is affected by any of these features, conflicts occur.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Working with ISignal Values (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Working with ISignal Values (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties 🕮)
Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties 🕮)

Add Feature - J1939 Network Management Enable

Purpose

To add the **J1939 Network Management Enable** feature to a communication controller that participates in J1939 communication and that is selected in a bus configuration.

Result

The **J1939 Network Management Enable** feature is added to the selected communication controller. This adds a **J1939 Network Management Enable** node to the communication controller in the **Bus Configurations** table.

Description

Adding this feature to a selected communication controller lets you enable and disable J1939 network management for the related network node and specify the enable state.

For more information on the **J1939 Network Management Enable** feature, refer to:

- Bus Manager in ConfigurationDesk: Enabling and Disabling J1939 Network Management (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Enabling and Disabling J1939 Network Management (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Add Feature - LIN Schedule Table

Purpose

To add the **LIN Schedule Table** feature to a LIN master communication controller that is selected in a bus configuration.

Result

The **LIN Schedule Table** feature is added to the selected LIN master communication controller. This has the following effects:

- In the bus configuration, a LIN Schedule Index function port is available for the communication controller.

- In the **Bus Configurations** table, a **LIN Schedule Table** node is added to the communication controller.

Description

Adding this feature to a selected LIN master communication controller lets you specify an initial LIN schedule table via the added function port. If you enable model access and/or test automation support for the function port, you can change the active LIN schedule table during run time.

> **Note**
>
> The default settings of the function port disable the LIN communication. To enable the LIN communication, you must either specify an initial schedule table or enable model access and/or test automation support for the function port.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Working with LIN Schedule Tables (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Working with LIN Schedule Tables (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - LIN Wake-Up

Purpose

To add the **LIN Wake-Up** feature to a LIN master or LIN slave communication controller that is selected in a bus configuration.

Result

The **LIN Wake-Up** feature is added to the selected communication controller. This has the following effects:

- In the bus configuration, the following function ports are available for the communication controller:
  - Wake-Up function port
  - Wake-Up State function port
- In the **Bus Configurations** table, a **LIN Wake-Up** node is added to the communication controller.

Description

Adding this feature to a selected communication controller lets you send wake-up signals via a LIN bus.

The Wake-Up function port lets you send wake-up signals via the related communication cluster. The Wake-Up State function port indicates whether the related communication controller detected a wake-up signal.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Sending Wake-Up Signals on a LIN Bus (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Sending Wake-Up Signals on a LIN Bus (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties 🕮)
Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties 🕮)

Add Feature - PDU Cyclic Timing Control

Purpose

To add the PDU Cyclic Timing Control feature to a basic PDU that is selected in a bus configuration.

Result

The PDU Cyclic Timing Control feature is added to the selected basic PDU. This has the following effects:

- In the bus configuration, two function ports are available for the PDU:
  - PDU Timing Control Period function port
  - PDU Timing Control Offset function port
- In the Bus Configurations table, a PDU Cyclic Timing Control node is added to the PDU.

Description

Adding this feature to a selected basic PDU lets you control the cyclic transmission of the PDU via the added function ports.

You cannot use the PDU Cyclic Timing Control and the Frame Access feature in parallel. If you add the PDU Cyclic Timing Control feature to a PDU that is affected by the Frame Access feature, conflicts occur.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Controlling the Cyclic Timing of CAN PDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Controlling the Cyclic Timing of CAN PDUs (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - PDU Enable

Purpose

To add the **PDU Enable** feature to a basic PDU that is selected in a bus configuration.

Result

The **PDU Enable** feature is added to the selected basic PDU. This has the following effects:

- In the bus configuration, a PDU Enable function port is available for the PDU.
- In the **Bus Configurations** table, a **PDU Enable** node is added to the PDU.

Description

Adding this feature to a selected basic PDU lets you enable and disable the transmission of the PDU via the added function port.

You cannot use the **PDU Enable** and the **Frame Access** feature in parallel. If you add the **PDU Enable** feature to a PDU that is affected by the **Frame Access** feature, conflicts occur.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Enabling and Disabling the Transmission of PDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Enabling and Disabling the Transmission of PDUs (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - PDU Length

Purpose

To add the **PDU Length** feature to a CAN basic PDU that is selected in a bus configuration.

Result

The **PDU Length** feature is added to the selected PDU. This has the following effects:

- Depending on the direction of the PDU (TX or RX), a TX Length or RX Length function port is available for the PDU in the bus configuration.
- In the **Bus Configurations** table, a **PDU Length** node is added to the PDU.

Description

Adding this feature to a selected basic PDU lets you access the payload length of the PDU via the added function port.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Accessing the Payload Length of CAN PDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Accessing the Payload Length of CAN PDUs (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)
Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - PDU Raw Data

Purpose

To add the **PDU Raw Data** feature to a PDU that is selected in a bus configuration.

Result

The **PDU Raw Data** feature is added to the selected PDU. This has the following effects:

- A PDU Raw Data function port is is available for the PDU in the bus configuration.
- In the **Bus Configurations** table, a **PDU Raw Data** node is added to the PDU.
- If you add the feature to a TX basic PDU, the **Access Mode** column in the **Bus Simulation Features** table is enabled.

Description

Adding this feature to a selected PDU lets you access the payload of the PDU in raw data format via the added function port.

You cannot use the **PDU Raw Data** feature in parallel with the **Frame Access** feature. If you add the **PDU Raw Data** feature to a PDU that is affected by the **Frame Access** feature, conflicts occur.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Accessing the Payload of PDUs in Raw Data Format (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Accessing the Payload of PDUs in Raw Data Format (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - PDU RX Interrupt

Purpose

To add the **PDU RX Interrupt** feature to a CAN RX PDU that is selected in a bus configuration.

Result

The **PDU RX Interrupt** feature is added to the selected CAN RX PDU. This has the following effects:

- A PDU RX Interrupt Enable function port is is available for the PDU in the bus configuration.
- A PDU Received event port is available for the PDU in the bus configuration.
- In the **Bus Configurations** table, a **PDU RX Interrupt** node is added to the PDU.

Description

Adding this feature to a selected CAN RX PDU lets you use RX interrupts to trigger the execution of functions in a behavior model.

You cannot use the **PDU RX Interrupt** feature in parallel with the **Frame Access** feature. If you add both features to a PDU, conflicts occur.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Triggering the Execution of Functions in a Behavior Model via RX Interrupts (ConfigurationDesk Bus Manager Implementation Guide)

- Bus Manager (stand-alone): Triggering the Execution of Functions in a Behavior Model via RX Interrupts (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties 🕮)
Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties 🕮)

Add Feature - PDU RX Status

Purpose

To add the **PDU RX Status** feature to an RX PDU that is selected in a bus configuration.

Result

The **PDU RX Status** feature is added to the selected RX PDU. This has the following effects:

- In the bus configuration, four function ports are available for the PDU:
  - PDU Counter function port
  - PDU State function port
  - PDU Time function port
  - PDU Delta Time function port
- In the **Bus Configurations** table, a **PDU RX Status** node is added to the PDU.

Description

Adding this feature to a selected RX PDU lets you observe the reception status of the PDU via the added function ports during run time.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Observing the Status of Received PDUs (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Observing the Status of Received PDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Related topics

References

Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties 🕮)

Add Feature - PDU Trigger

Purpose

To add the **PDU Trigger** feature to a basic PDU that is selected in a bus configuration.

Result

The **PDU Trigger** feature is added to the selected basic PDU. This has the following effects:

- In the bus configuration, a PDU Trigger function port is available for the PDU.
- In the **Bus Configurations** table, a **PDU Trigger** node is added to the PDU.
- In the **Bus Simulation Features** table, the **Trigger Mode** column is enabled.

Description

Adding this feature to a selected basic PDU lets you specify a user-defined trigger and trigger the transmission of the PDU at run time.

You cannot use the **PDU Trigger** and the **Frame Access** feature in parallel. If you add the **PDU Trigger** feature to a PDU that is affected by the **Frame Access** feature, conflicts occur.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Specifying User-Defined Triggers for Transmitting PDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Specifying User-Defined Triggers for Transmitting PDUs (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - PDU User Code

Purpose

To add the **PDU User Code** feature to a PDU that is selected in a bus configuration.

Result

The **PDU User Code** feature is added to the selected PDU. This has the following effects:

- If the PDU is an RX PDU, a Result function port is available for the PDU in the bus configuration.

- If the PDU is assigned to the **Manipulation** part, the following function ports are available:
  - Current Countdown Value function port
  - Countdown Start Value function port
  - Enable function port
  - Enable State function port
- In the **Bus Configurations** table, a **PDU User Code** node is added to the PDU.
- In the **Bus Simulation Features**, **Bus Inspection Features**, or **Bus Manipulation Features** table, the **User Code ID** column is enabled.
- In the **Bus Manipulation Features** table, the **Recalculate SecOC Information (User Code)** column is enabled if the PDU is involved in secure onboard communication.

Description

Adding this feature to a selected PDU lets you apply user code to the PDU, for example, to write checksum values that are calculated according to user-specific algorithms to ISignals of the PDU.

You cannot use the **PDU User Code** and the **Frame Access** feature in parallel. If you add the **PDU User Code** feature to a PDU that is affected by the **Frame Access** feature, conflicts occur.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Applying User Code to PDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Applying User Code to PDUs (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)
Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - SecOC

Purpose

To add the **SecOC** feature to an RX secured IPDU that is selected in a bus configuration.

Result

The SecOC feature is added to the selected secured IPDU. This has the following effects:

- Depending on the related bus configuration part, the following function ports are available for the PDU:

Bus Configuration Part	Function Port
Simulated ECUs	▪ Enable Verification function port ▪ State function port
Inspection	▪ Enable Verification function port ▪ State function port ▪ Authenticator Value function port ▪ Freshness Value function port ▪ Calculated Freshness Value function port

- In the Bus Configurations table, a SecOC node is added to the PDU.

Description

Adding this feature to a selected RX secured IPDU lets you verify its authentication information via OEM-specific verification algorithms that are available in user code implementations.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Verifying the Authentication Information of Received Secured IPDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Verifying the Authentication Information of Received Secured IPDUs (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)
Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - SecOC Authenticator Invalidation

Purpose

To add the SecOC Authenticator Invalidation feature to a TX secured IPDU that is selected in a bus configuration.

Result

The SecOC Authenticator Invalidation feature is added to the selected secured IPDU. This has the following effects:

- The following function ports are available for the PDU in the bus configuration:
  - Current Countdown Value function port
  - Countdown Start Value function port
  - Enable function port
  - Enable State function port
- In the Bus Configurations table, a SecOC Authenticator Invalidation node is added to the PDU.
- In the Bus Manipulation Features table, the Countdown Start Value and Recalculate SecOC Information (Authenticator Invalidation) columns are enabled.

Description

Adding this feature to a TX secured IPDU lets you invalidate its authenticator permanently or temporarily.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Invalidating the Authenticator of Secured IPDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Invalidating the Authenticator of Secured IPDUs (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)
Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - SecOC Freshness Overwrite Value

Purpose

To add the SecOC Freshness Overwrite Value feature to a secured IPDU that is selected in a bus configuration.

Result

The **SecOC Freshness Overwrite Value** feature is added to the selected PDU. This has the following effects:

- The following function ports are available for the PDU in the bus configuration:
  - Current Countdown Value function port
  - Countdown Start Value function port
  - Enable function port
  - Enable State function port
  - Freshness Overwrite Value function port
- In the **Bus Configurations** table, a **SecOC Freshness Overwrite Value** node is added to the PDU.
- In the **Bus Manipulation Features** table, the **Countdown Start Value** and **Freshness Overwrite Value** columns are enabled.

Description

Adding this feature to a selected secured IPDU lets you overwrite the freshness value permanently or temporarily.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Overwriting the Freshness Value of Secured IPDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Overwriting the Freshness Value of Secured IPDUs (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)
Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Feature - Suspend Frame Transmission

Purpose

To add the **Suspend Frame Transmission** feature to a frame of a PDU that is selected in a bus configuration.

Result

The **Suspend Frame Transmission** feature is added to the selected PDU. This has the following effects:

- The following function ports are available for the PDU in the bus configuration:
  - Current Countdown Value function port
  - Countdown Start Value function port
  - Enable function port
  - Enable State function port
- In the **Bus Configurations** table, a **Suspend Frame Transmission** node is added to the PDU.
- In the **Bus Manipulation Features** table, the **Countdown Start Value** column is enabled.

Description

Adding this feature to a selected PDU lets you manipulate the transmission of the related frame by suspending the transmission permanently or temporarily.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Suspending the Transmission of Frames (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Suspending the Transmission of Frames (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Function Inport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)
Function Outport Properties (Bus Configuration) (ConfigurationDesk Function Block Properties)

Add Frame Capture

Purpose

To add a frame capture to the **Inspection** part of a bus configuration.

Result

A frame capture is added to the related bus configuration. This has the following effects:

- In the **Bus Configurations** table, a **Frame Capture (<number>)** node is available for the **Inspection** part of the bus configuration.

- The following elements are added to the frame capture:
  - The **Frame Capture Data** feature.
  - The **CAN Cluster Filter** filter.
- The **Filter Control** feature is added to the filter.
- In the **Bus Access Requests** table, a bus access request is available for the frame capture.

Description

The frame capture lets you capture data of received CAN frames. You can specify the maximum number of frames that can be captured in one sampling step and specify filters to capture specific frames. The **Frame Capture Data** feature lets you access the captured data at run time. Via the **Filter Control** feature, you can control the specified filter, e.g., enable or disable the filter at run time.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Capturing CAN Frames (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Capturing CAN Frames (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Add CAN Filter Rule..349
Add Feature - Filter Control..360
Add Feature - Frame Capture Data..362

Add Frame Gateway

Purpose

To add a frame gateway to the **Gateways** part of a bus configuration.

Result

A frame gateway is added to the related bus configuration. This has the following effects:

- In the **Bus Configurations** table, a **Frame Gateway (<number>)** node is available for the **Gateways** part of the bus configuration.
- The following elements are added to the frame gateway:
  - The **Frame Gateway Direction** feature.
  - The **CAN Cluster 1 Filter** and **CAN Cluster 2 Filter** filters.
- The **Filter Control** feature is added to each filter.
- In the **Bus Access Requests** table, two bus access requests are available for the frame gateway.

Description

The frame gateway lets you specify a gateway to exchange CAN communication between two communication clusters. Via the **Frame Gateway Direction** feature, you can specify the gateway direction or disable the gateway. Additionally, you can specify filters to gateway specific frames. Via the **Filter Control** feature, you can control the specified filters, e.g., enable or disable a filter at run time.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Specifying CAN Gateways (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Specifying CAN Gateways (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Assign - <Communication Matrix> / Communication Matrix Assignment - <Communication Matrix>

Purpose

To replace a currently assigned communication matrix in a bus configuration.

Result

Subelements of the selected communication matrix node that can be identified unambiguously in the newly assigned communication matrix are replaced. Elements that cannot be identified unambiguously are moved to a communication matrix node that summarizes these deprecated elements of the bus configuration.

Description

The Bus Manager compares the names of the currently assigned elements and their position in the bus hierarchy with the newly assigned communication matrix. The currently assigned elements are replaced if:

- The elements are found in both the bus configuration and the newly assigned communication matrix.
- The following definitions match:
  - Communication cluster (not compared for elements of DBC and LDF communication matrices)

- ECU (only compared for communication matrix elements that are assigned to the **Simulated ECUs** part)
- Frame
- Channel
- Direction (only compared for TX and RX elements, such as PDUs or ISignals)
- Element name (e.g., PDU name, ISignal name)

> **Note**
>
> Depending on the replaced elements, some of their settings are reset to the default values or lost (such as user-defined settings of communication matrix elements).

Subelements of the selected communication matrix node that are not unambiguously identified in the newly assigned communication matrix cannot be replaced. These elements are moved to a new communication matrix node named **<communication matrix node name> (deprecated <number>)**.

Elements that are part of the newly assigned communication matrix but not of the bus configuration are not assigned.

Due to the changes between the replaced and newly assigned communication matrices, the replacement might result in conflicting bus configurations.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Replacing Assigned Communication Matrices (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Replacing Assigned Communication Matrices (Bus Manager (Stand-Alone) Implementation Guide)

Assign - <Function Block> / Bus Access Assignment - Assignable Accesses - <Function Block>

Purpose

To assign the selected bus access request to a bus access by selecting an existing suitable bus function block (**CAN**, **LIN**).

Result

The bus access request is assigned to the selected bus function block (**CAN**, **LIN**). The assigned hardware resource of the bus function block is used for the bus communication of the related communication cluster.

Description

If you implement bus communication in a real-time application, each communication cluster that is assigned to a bus configuration requires access to bus channels of real-time hardware. To specify the hardware access, a suitable bus function block (**CAN**, **LIN**) is needed. For example, to specify a CAN channel for a CAN cluster, a **CAN** function block must be implemented in the signal chain.

This command lets you select a bus function block to specify the bus access for the selected bus access request. The settings of the bus function block remain unchanged. To avoid conflicts, you must ensure that settings such as the baud rates of the bus access request and the bus function block match.

The bus function block is assigned to the selected bus access request regardless of whether it is already assigned to another bus access request. If the bus function block is assigned to multiple bus access requests, conflicts might occur.

For details on assigning bus accesses to bus access requests, refer to Specifying the Hardware Access (ConfigurationDesk Bus Manager Implementation Guide).

Assign - Assign First Fit / Bus Access Assignment - Assign First Fit

Purpose

To assign the selected bus access request to a bus access by assigning the first suitable bus function block (**CAN**, **LIN**).

Result

The first suitable bus function block (**CAN**, **LIN**) is assigned to the selected bus access request. The assigned hardware resource of the bus function block is used for the bus communication of the related communication cluster.

Description

If you implement bus communication in a real-time application, each communication cluster that is assigned to a bus configuration requires access to bus channels of real-time hardware. To specify the hardware access, a suitable bus function block (**CAN**, **LIN**) is needed. For example, to specify a CAN channel for a CAN cluster, a **CAN** function block must be implemented in the signal chain.

This command automatically assigns the first suitable bus function block to the selected bus access request. The settings of the bus function block remain unchanged. To avoid conflicts, you must ensure that settings such as the baud rates of the bus access request and the bus function block match.

The bus function block is assigned to the selected bus access request regardless of whether it is already assigned to another bus access request. If the bus function block is assigned to multiple bus access requests, conflicts might occur.

For details on assigning bus accesses to bus access requests, refer to Specifying the Hardware Access (ConfigurationDesk Bus Manager Implementation Guide).

Assign - Create and Assign Suitable Access / Bus Access Assignment - Create and Assign Suitable Access

Purpose

To assign the selected bus access request to a bus access by adding a new suitable bus function block (CAN, LIN) to the signal chain and assigning it to the bus access request.

Result

A suitable bus function block (CAN, LIN) is added to the signal chain and assigned to the bus access request.

Description

If you implement bus communication in a real-time application, each communication cluster that is assigned to a bus configuration requires access to bus channels of real-time hardware. To specify the hardware access, a suitable bus function block (CAN, LIN) is needed. For example, to specify a CAN channel for a CAN cluster, a CAN function block must be implemented in the signal chain.

This command adds a new suitable bus function block to the signal chain and assigns it to the selected bus access request. The default name of the bus function block is <communication cluster name> (<number>). Default settings such as the baud rate are taken from the communication matrix. To complete the bus access assignment for the selected bus access request, you must assign a hardware resource in the bus function block.

For details on assigning bus accesses to bus access requests, refer to Specifying the Hardware Access (ConfigurationDesk Bus Manager Implementation Guide 🕮).

Assign - Delete Assignment / Bus Access Assignment - Delete Assignment

Purpose

To remove a bus access from the selected bus access request.

Result

The assigned bus access is removed from the selected bus access request. The bus function block (CAN, LIN) that was used for the bus access remains unchanged.

Assign Automatically

Purpose

To assign suitable bus accesses to all the bus access requests of the active ConfigurationDesk application in one step.

Result

All bus access requests of the active ConfigurationDesk application are assigned to suitable bus accesses.

Description

If you implement bus communication in a real-time application, each communication cluster that is assigned to a bus configuration requires access to bus channels of real-time hardware. To specify the hardware access, a suitable bus function block (**CAN**, **LIN**) is needed. For example, to specify a CAN channel for a CAN cluster, a **CAN** function block must be implemented in the signal chain.

This command automatically assigns suitable bus function blocks to all the bus access requests of the active ConfigurationDesk application that are not assigned to a bus access yet.

Depending on the bus function blocks that are available in the signal chain, existing bus function blocks are used or new bus function blocks are added to the signal chain.

If existing bus function blocks are used, some of their settings are adapted to settings specified in the communication matrix. In this case, conflicts might occur (e.g., if a bus function block is assigned to several bus access requests with different settings).

If multiple suitable bus function blocks are available in the signal chain and their names are ambiguous, no bus function blocks can be assigned to the relevant bus access requests. In this case, the **Message Viewer** displays an error message.

For more information on assigning bus accesses to bus access requests and for details on the automatic bus access assignment, refer to Specifying the Hardware Access (ConfigurationDesk Bus Manager Implementation Guide).

Automatic Bus Access Assignment

Purpose

To assign suitable bus accesses to all the lower-level bus access requests of the selected bus configuration element in one step.

Result

All lower-level bus access requests of the selected bus configuration element are assigned to suitable bus accesses.

Description

If you implement bus communication in a real-time application, each communication cluster that is assigned to a bus configuration requires access to bus channels of real-time hardware. To specify the hardware access, a suitable bus function block (**CAN**, **LIN**) is needed. For example, to specify a CAN channel for a CAN cluster, a **CAN** function block must be implemented in the signal chain.

This command automatically assigns suitable bus function blocks to all the lower-level bus access requests of the selected bus configuration element that are not assigned to a bus access yet.

Depending on the bus function blocks that are available in the signal chain, existing bus function blocks are used or new bus function blocks are added to the signal chain.

If existing bus function blocks are used, some of their settings are adapted to settings specified in the communication matrix. In this case, conflicts might occur (e.g., if a bus function block is assigned to several bus access requests with different settings).

If multiple suitable bus function blocks are available in the signal chain and their names are ambiguous, no bus function blocks can be assigned to the relevant bus access requests. In this case, the **Message Viewer** displays an error message.

For more information on assigning bus accesses to bus access requests and for details on the automatic bus access assignment, refer to Specifying the Hardware Access (ConfigurationDesk Bus Manager Implementation Guide).

Bus Access Assignment

Purpose

To access different commands for handling the bus access assignment.

Result

Different commands for handling the bus access assignment are available in a submenu.

Description

The commands available in the submenu are:

- Assign - <Function Block> / Bus Access Assignment - Assignable Accesses - <Function Block> on page 386
- Assign - Create and Assign Suitable Access / Bus Access Assignment - Create and Assign Suitable Access on page 388
- Assign - Assign First Fit / Bus Access Assignment - Assign First Fit on page 387
- Assign - Delete Assignment / Bus Access Assignment - Delete Assignment on page 388

Bus Access Requests Table

Purpose

To access bus access requests of a ConfigurationDesk application and assign them to bus accesses.

Description

The **Bus Access Requests** table allows you to:

- Display all the bus access requests of all the bus configurations of a ConfigurationDesk application.
- Assign bus access requests to bus accesses.
- Assign hardware resources to bus function blocks (**CAN**, **LIN**) that specify the bus accesses.
- Select elements to access their properties in the **Properties Browser**.
- Remove bus accesses from bus access requests.
- Remove communication clusters, bus systems, or communication matrices from bus configurations.
- Remove bus configurations from the application.

Symbols

The elements are displayed with various symbols. For an overview of all the available symbols, refer to Elements and Symbols of Bus Configurations and Communication Matrices on page 429.

Columns

The **Bus Access Requests** table provides the following columns:

Name Displays the names of bus configuration elements. You can change some of the displayed names , e.g., the names of communication matrix nodes. Changing the names directly affects the TRC file. Refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Configurations (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Basics on Bus Configurations (Bus Manager (Stand-Alone) Implementation Guide)

Element Type Displays the type of bus configuration elements, e.g., **Bus ECU**, **Bus Network Node**, or **Bus ISignal IPDU**.

Bus Configuration Displays the name of the bus configuration for which bus access requests and related higher-level elements are available. You can change the displayed name.

Bus Configuration Part Displays the bus configuration part (**Simulated ECUs**, **Inspection**, **Manipulation**, **Gateways**) for which bus access requests are available.

Communication Cluster Displays the communication cluster for which bus access requests and related higher-level elements are available. If the communication cluster is specified in a DBC or LDF communciation matrix, you can change the displayed name.

Communication Matrix Displays the name of the communication matrix node for which bus access requests and related higher-level elements are available. You can change the displayed name.

Bus System Displays the bus system (CAN, LIN) for which bus access requests and related higher-level elements are available.

Channel Displays the name of the bus channel of real-time hardware that is assigned as the hardware resource to bus function blocks (CAN, LIN) that specify the bus accesses.

Related topics

Basics

Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide)

Bus Configuration Ports Table

Purpose

To access and configure function ports and event ports of bus configurations.

Description

The Bus Configuration Ports table allows you to:

- Display all the function ports and event ports of the bus configurations of a ConfigurationDesk application.
- Distinguish the ports by means of different columns.
- Select ports to access their properties in the **Properties Browser**.
- Configure properties of function ports.

> **Tip**
>
> Event ports have no configurable properties.

Symbols

The bus configuration ports are displayed with the following symbols:

Symbol	Description
fx	Indicates a function inport.
fx	Indicates a function outport.
▶	Indicates an event port.

Columns

The Bus Configuration Ports table provides the following columns:

Name Displays the port names.

Part Displays the bus configuration part (Simulated ECUs, Inspection, Manipulation, or Gateways) for which a port is available.

Communication Cluster Displays the communication cluster for which a port is available in a bus configuration.

ECU Displays the ECU for which a port is available in a bus configuration.

Feature Displays the bus configuration feature for which a port is available. For an overview of the available bus configuration features, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Configuration Features (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Basics on Bus Configuration Features (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Model Access Lets you enable and disable model access for function ports.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: Model access is enabled for the function port. ▪ Disabled: Model access is disabled for the function port.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Configuring Function Ports for Bus Configuration Features (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Configuring Function Ports for Bus Configuration Features (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Connected Model Ports Lets you map model ports to function ports and event ports via drag & drop, and displays the mapped ports. You can map data inports to function outports, data outports to function inports, and runnable function ports to event ports. If a function port's model access is disabled, it is enabled automatically when you map a data port.

Test Automation Support Lets you enable and disable test automation support for function ports.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: Test automation support is enabled for the function port. ▪ Disabled: Test automation support is disabled for the function port.
Dependencies	–

More information	▪ Bus Manager in ConfigurationDesk: Configuring Function Ports for Bus Configuration Features (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Configuring Function Ports for Bus Configuration Features (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Related TX ECUs Displays all the ECUs that transmit the bus configuration element (e.g., ISignal, PDU) for which a port is available. The displayed ECUs are derived from the communication matrix and not from the element's configuration in the bus configuration.

Related RX ECUs Displays all the ECUs that receive the bus configuration element for which a port is available. The displayed ECUs are derived from the communication matrix and not from the element's configuration in the bus configuration.

Bus System Displays the bus system (CAN, LIN) for which a port is available.

Bus Configuration Displays the bus configuration for which a port is available.

Direction Displays the direction (RX or TX) of the bus configuration element for which a port is available.

Port Type Displays the port type (In or Out) of a port.

Original Communication Matrix Displays the file name of the communication matrix that specifies the bus configuration element for which a port is available.

Related topics

Basics

Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide 🕮)

Bus Configurations Table

Purpose

To access and configure bus configurations of a ConfigurationDesk application.

Description

The Bus Configurations table allows you to:

- Drag communication matrix elements to bus configurations to implement bus communication in the signal chain.
- Specify gateways to exchange CAN communication between two communication clusters.
- Specify frame captures to capture CAN frames that are received on the bus.

- Display all the elements of all the bus configurations of a ConfigurationDesk application.
- Add bus configuration features to bus configuration elements.
- Select elements to access their properties in the **Properties Browser**.
- Remove elements from bus configurations.
- Remove bus configurations from the application.

Symbols

The bus configuration elements are displayed with various symbols. For an overview of all the available symbols, refer to Elements and Symbols of Bus Configurations and Communication Matrices on page 429.

Columns

The Bus Configurations table provides the following columns:

Name Displays the names of bus configuration elements. You can change some of the displayed names , e.g., the names of communication matrix nodes. Changing the names directly affects the TRC file. Refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Configurations (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Basics on Bus Configurations (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Direction Displays the direction (RX or TX) of bus configuration elements, e.g., of ISignal IPDUs or ISignals.

Element Type Displays the type of bus configuration elements, e.g., Bus ECU, Bus Network Node, or Bus ISignal IPDU.

Related Clusters Displays the communication clusters a bus configuration element is a member of. The displayed communication clusters depend on the communication matrix:

- AUTOSAR and FIBEX communication matrices:

  The displayed clusters are derived from the communication matrix and not from the element's configuration in the bus configuration. For example, if an ECU is assigned to a bus configuration in the context of one communication cluster but the ECU is a member of two clusters, both clusters are displayed in this column.
- DBC and LDF communication matrices:

  DBC and LDF communication matrices specify the communication of only one communication cluster. The displayed name is the name of the communication cluster in the bus configuration. You can change the cluster name via the Properties Browser when you select an assigned PDU, for example. If you do this, the displayed name is adjusted accordingly.

Related TX ECUs Displays all the ECUs that transmit a bus configuration element. The displayed ECUs are derived from the communication matrix and not from the element's configuration in the bus configuration. For example, if a

PDU is assigned to a bus configuration in the context of one transmitting ECU but the PDU is transmitted by two ECUs, both ECUs are displayed in this column.

Related RX ECUs Displays all the ECUs that receive a bus configuration element. The displayed ECUs are derived from the communication matrix and not from the element's configuration in the bus configuration. For example, if a PDU is assigned to a bus configuration in the context of one receiving ECU but the PDU is received by two ECUs, both ECUs are displayed in this column.

Connected Model Ports Lets you map model ports to Bus Configuration function ports via drag & drop and displays the mapped model ports. You can map model inports to function outports and model outports to function inports. If a function port's model access is disabled, it is enabled automatically when you map a model port.

Related topics

Basics

Bus Configuration Tables (Bus Manager (Stand-Alone) Implementation Guide)
Bus Configuration Tables (ConfigurationDesk Bus Manager Implementation Guide)
Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide)

Bus Inspection Features Table

Purpose

To access and configure PDU-related and signal-related bus configuration features of a ConfigurationDesk application for inspection purposes.

Description

The Bus Inspection Features table allows you to:

- Display all the PDUs and ISignals of all the bus configurations of a ConfigurationDesk application that are used for inspection purposes.
- Display all the PDU-related and signal-related bus configuration features that are available for inspection purposes.
- Add bus configuration features to PDUs and ISignals for inspection purposes.
- Configure bus configuration features.
- Select PDUs and ISignals to access their properties in the **Properties Browser**.
- Remove PDUs and ISignals from the Inspection node of bus configurations.

For more information on bus configuration features, refer to:

- Bus Manager in ConfigurationDesk: Working with Bus Configuration Features (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Working with Bus Configuration Features (Bus Manager (Stand-Alone) Implementation Guide)

Symbols

The PDUs and ISignals are displayed with various symbols. For an overview of all the available symbols, refer to Elements and Symbols of Bus Configurations and Communication Matrices on page 429.

Subviews

The Bus Inspection Features table provides the following subviews:

Subview	Purpose
PDU Features	Provides a list of all the PDUs that are assigned to the Inspection parts of all bus configurations, and lets you access the PDU-related bus inspection features. This is the default view.
Signal Features	Provides a list of all the ISignals that are assigned to the Inspection parts of all bus configurations, and lets you access the signal-related bus inspection features.

Columns

The Bus Inspection Features table provides various columns. Depending on the selected subview, the following columns are available:

Name Displays the names of bus configuration elements. You can change some of the displayed names , e.g., the names of communication matrix nodes. Changing the names directly affects the TRC file. Refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Configurations (ConfigurationDesk Bus Manager Implementation Guide 📖)
- Bus Manager (stand-alone): Basics on Bus Configurations (Bus Manager (Stand-Alone) Implementation Guide 📖)

Element Type Displays the type of bus configuration elements, e.g., Bus ECU, Bus Network Node, or Bus ISignal IPDU.

Direction Displays the direction (RX or TX) of bus configuration elements, e.g., of ISignal IPDUs or ISignals.

Bus Configuration Displays the name of the bus configuration to which the PDUs or ISignals are assigned. You can change the displayed name.

Communication Matrix Displays the name of the communication matrix node for which the PDUs or ISignals are available. You can change the displayed name.

Communication Cluster Displays the name of the communication cluster for which the PDUs or ISignals are assigned to the bus configuration.

PDU Displays the name of the PDU for which the ISignals are assigned to the bus configuration.

PDU RX Status Lets you enable and disable the PDU RX Status feature for PDUs.

Value	▪ Enabled ▪ Disabled

Description	▪ Enabled: The **PDU RX Status** feature is enabled for the related PDU. This adds the following function ports to the related bus configuration: ▪ PDU Counter function port ▪ PDU State function port ▪ PDU Time function port ▪ PDU Delta Time function port Additionally, a **PDU RX Status** node is available in the **Bus Configurations** table. ▪ Disabled: The **PDU RX Status** feature is disabled for the related PDU. The function ports are deleted from the related bus configuration, regardless of whether they are mapped to model ports. The **PDU RX Status** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Observing the Status of Received PDUs (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Observing the Status of Received PDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

PDU Raw Data Lets you enable and disable the **PDU Raw Data** feature for PDUs.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The **PDU Raw Data** feature is enabled for the related PDU. This adds a RX PDU Raw Data function port to the related bus configuration. Additionally, a **PDU Raw Data** node is available in the **Bus Configurations** table. ▪ Disabled: The **PDU Raw Data** feature is disabled for the related PDU. The function port is deleted from the related bus configuration, regardless of whether it is mapped to a model port. The **PDU Raw Data** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Accessing the Payload of PDUs in Raw Data Format (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Accessing the Payload of PDUs in Raw Data Format (Bus Manager (Stand-Alone) Implementation Guide 🕮)

PDU User Code Lets you enable and disable the **PDU User Code** feature for PDUs.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The **PDU User Code** feature is enabled for the related PDU. This adds a Result function port to the related bus configuration. Additionally, the **User Code ID** column is enabled and a **PDU User Code** node is available in the **Bus Configurations** table. ▪ Disabled: The **PDU User Code** feature is disabled for the related PDU. The function port is deleted from the related bus configuration, regardless of whether it is mapped to a model port. The **User Code ID** column is disabled and the **PDU User Code** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Applying User Code to PDUs (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Applying User Code to PDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

User Code ID Lets you specify the user code ID if you enabled the **PDU User Code** feature for a PDU.

Value	All characters
Description	The user code ID is required to reference the related user code file. It must match the user code ID that is specified for the `DS_Bus_Custom_Feature_Name` attribute in the related user code file.
Dependencies	Available only if **PDU User Code** is set to **Enabled**.
More information	▪ Bus Manager in ConfigurationDesk: Applying User Code to PDUs (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Applying User Code to PDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

SecOC Lets you enable and disable the **SecOC** feature for RX secured IPDUs.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The **SecOC** feature is enabled for the related secured IPDU. This adds the following function ports to the related bus configuration: ▪ Enable Verification function port ▪ State function port

	▪ Authenticator Value function port ▪ Freshness Value function port ▪ Calculated Freshness Value function port Additionally, a **SecOC** node is available in the **Bus Configurations** table. ▪ Disabled: The **SecOC** feature is disabled for the related secured IPDU. The function ports are deleted from the related bus configuration, regardless of whether they are mapped to model ports. The **SecOC** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Verifying the Authentication Information of Received Secured IPDUs (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Verifying the Authentication Information of Received Secured IPDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

ISignal Value Lets you enable and disable the **ISignal Value** feature for ISignals.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The **ISignal Value** feature is enabled for the related ISignal. This adds an RX ISignal function port to the related bus configuration. Additionally, an **ISignal Value** node is available in the **Bus Configurations** table. ▪ Disabled: The **ISignal Value** feature is disabled for the related ISignal. The function port is deleted from the related bus configuration, regardless of whether it is mapped to a model port. The **ISignal Value** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Working with ISignal Values (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Working with ISignal Values (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Related Clusters Displays the communication clusters a bus configuration element is a member of. The displayed communication clusters depend on the communication matrix:

- AUTOSAR and FIBEX communication matrices:

  The displayed clusters are derived from the communication matrix and not from the element's configuration in the bus configuration. For example, if an

ECU is assigned to a bus configuration in the context of one communication cluster but the ECU is a member of two clusters, both clusters are displayed in this column.

- DBC and LDF communication matrices:

  DBC and LDF communication matrices specify the communication of only one communication cluster. The displayed name is the name of the communication cluster in the bus configuration. You can change the cluster name via the Properties Browser when you select an assigned PDU, for example. If you do this, the displayed name is adjusted accordingly.

Related TX ECUs Displays all the ECUs that transmit a bus configuration element. The displayed ECUs are derived from the communication matrix and not from the element's configuration in the bus configuration. For example, if a PDU is assigned to a bus configuration in the context of one transmitting ECU but the PDU is transmitted by two ECUs, both ECUs are displayed in this column.

Related RX ECUs Displays all the ECUs that receive a bus configuration element. The displayed ECUs are derived from the communication matrix and not from the element's configuration in the bus configuration. For example, if a PDU is assigned to a bus configuration in the context of one receiving ECU but the PDU is received by two ECUs, both ECUs are displayed in this column.

Related topics

Basics

Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide 🕮)

Bus Manipulation Features Table

Purpose

To access and configure PDU-related and signal-related bus configuration features of a ConfigurationDesk application for manipulation purposes.

Description

The Bus Manipulation Features table allows you to:

- Display all the PDUs and ISignals of all the bus configurations of a ConfigurationDesk application that are used for manipulation purposes.
- Display all the PDU-related and signal-related bus configuration features that are available for manipulation purposes.
- Add bus configuration features to PDUs and ISignals for manipulation purposes.
- Configure bus configuration features.
- Select PDUs and ISignals to access their properties in the Properties Browser.

- Remove PDUs and ISignals from the **Manipulation** node of bus configurations.

For more information on bus configuration features, refer to:

- Bus Manager in ConfigurationDesk: Working with Bus Configuration Features (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Working with Bus Configuration Features (Bus Manager (Stand-Alone) Implementation Guide)

Symbols

The PDUs and ISignals are displayed with various symbols. For an overview of all the available symbols, refer to Elements and Symbols of Bus Configurations and Communication Matrices on page 429.

Subviews

The **Bus Manipulation Features** table provides the following subviews:

Subview	Purpose
PDU Features	Provides a list of all the PDUs that are assigned to the **Manipulation** parts of all bus configurations, and lets you access the PDU-related bus manipulation features. This is the default view.
Signal Features	Provides a list of all the ISignals that are assigned to the **Manipulation** parts of all bus configurations, and lets you access the signal-related bus manipulation features.

Columns

The **Bus Manipulation Features** table provides various columns. Depending on the selected subview, the following columns are available:

Name Displays the names of bus configuration elements. You can change some of the displayed names , e.g., the names of communication matrix nodes. Changing the names directly affects the TRC file. Refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Configurations (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Basics on Bus Configurations (Bus Manager (Stand-Alone) Implementation Guide)

Element Type Displays the type of bus configuration elements, e.g., **Bus ECU**, **Bus Network Node**, or **Bus ISignal IPDU**.

Direction Displays the direction (RX or TX) of bus configuration elements, e.g., of ISignal IPDUs or ISignals.

Bus Configuration Displays the name of the bus configuration to which the PDUs or ISignals are assigned. You can change the displayed name.

Communication Matrix Displays the name of the communication matrix node for which the PDUs or ISignals are available. You can change the displayed name.

Communication Cluster Displays the name of the communication cluster for which the PDUs or ISignals are assigned to the bus configuration.

PDU Displays the name of the PDU for which the ISignals are assigned to the bus configuration.

Countdown Start Value Lets you specify a countdown start value for all bus manipulation features of the selected bus configuration element.

Value	0 … $\text{UInt32}_{\text{max}}$
Description	If one or more bus manipulation features of the selected bus configuration element are temporarily enabled, the countdown start value determines how often the bus configuration element or its related elements are manipulated.
Dependencies	Available only if Frame Length, Suspend Frame Transmission, PDU User Code, SecOC Authenticator Invalidation, or SecOC Freshness Overwrite Value is set to Enabled.
More information	▪ Bus Manager in ConfigurationDesk: Basics on Bus Manipulation Features (ConfigurationDesk Bus Manager Implementation Guide 📖) ▪ Bus Manager (stand-alone): Basics on Bus Manipulation Features (Bus Manager (Stand-Alone) Implementation Guide 📖)

Suspend Frame Transmission Lets you enable and disable the Suspend Frame Transmission feature for frames.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The Suspend Frame Transmission feature is enabled for the frame of the selected PDU. This adds the following function ports to the related bus configuration: ▪ Current Countdown Value function port ▪ Countdown Start Value function port ▪ Enable function port ▪ Enable State function port Additionally, the Countdown Start Value column is enabled and a Suspend Frame Transmission node is available in the Bus Configurations table. ▪ Disabled: The Suspend Frame Transmission feature is disabled for the frame of the selected PDU: ▪ The function ports are deleted from the related bus configuration, regardless of whether they are mapped to model ports. ▪ The Countdown Start Value column is disabled only if Frame Length, PDU User Code, SecOC Authenticator Invalidation, and SecOC Freshness Overwrite Value are set to Disabled as well. ▪ The Suspend Frame Transmission node is deleted from the Bus Configurations table.

Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Suspending the Transmission of Frames (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Suspending the Transmission of Frames (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Frame Length Lets you enable and disable the **Frame Length** feature for frames.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The **Frame Length** feature is enabled for the frame of the selected PDU. This adds the following function ports to the related bus configuration: ▪ Current Countdown Value function port ▪ Countdown Start Value function port ▪ Enable function port ▪ Enable State function port ▪ Length function port Additionally, the following columns are enabled: ▪ **Countdown Start Value** ▪ **Padding Value** ▪ **Length** In the **Bus Configurations** table, a **Frame Length** node is available. ▪ Disabled: The **Frame Length** feature is disabled for the frame of the selected PDU: ▪ The function ports are deleted from the related bus configuration, regardless of whether they are mapped to model ports. ▪ The **Padding Value** and **Length** columns are disabled. ▪ The **Countdown Start Value** column is disabled only if **Suspend Frame Transmission**, **PDU User Code**, **SecOC Authenticator Invalidation**, and **SecOC Freshness Overwrite Value** are set to **Disabled** as well. ▪ The **Frame Length** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Manipulating the Payload Length of CAN Frames (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Manipulating the Payload Length of CAN Frames (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Padding Value Lets you specify a padding value if you enabled the **Frame Length** feature for a frame.

Value	0 … 255
Description	If padding bytes are added to the related frame, the specified value is used as the padding byte value.
Dependencies	Available only if **Frame Length** is set to **Enabled**.
More information	▪ Bus Manager in ConfigurationDesk: Manipulating the Payload Length of CAN Frames (ConfigurationDesk Bus Manager Implementation Guide 📖) ▪ Bus Manager (stand-alone): Manipulating the Payload Length of CAN Frames (Bus Manager (Stand-Alone) Implementation Guide 📖)

Length Lets you specify a frame length if you enabled the **Frame Length** feature for a frame.

Value	0 … 64 bytes
Description	The specified length is used as the frame length at run time. The valid length depends on the bus protocol: ▪ Classic CAN frames: 0 byte ... 8 bytes ▪ CAN FD frames: 0 byte ... 64 bytes If you specify a length that is invalid according to the bus protocol, conflicts occur.
Dependencies	Available only if **Frame Length** is set to **Enabled**.
More information	▪ Bus Manager in ConfigurationDesk: Manipulating the Payload Length of CAN Frames (ConfigurationDesk Bus Manager Implementation Guide 📖) ▪ Bus Manager (stand-alone): Manipulating the Payload Length of CAN Frames (Bus Manager (Stand-Alone) Implementation Guide 📖)

PDU User Code Lets you enable and disable the **PDU User Code** feature for PDUs.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The **PDU User Code** feature is enabled for the PDU. This adds the following function ports to the related bus configuration: ▪ Current Countdown Value function port ▪ Countdown Start Value function port ▪ Enable function port ▪ Enable State function port

	Additionally, the Countdown Start Value, User Code ID, and Recalculate SecOC Information (User Code) columns are enabled and a PDU User Code node is available in the Bus Configurations table. ▪ Disabled: The PDU User Code feature is disabled for the PDU: ▪ The function ports are deleted from the related bus configuration, regardless of whether they are mapped to model ports. ▪ The User Code ID and Recalculate SecOC Information (User Code) columns are disabled. ▪ The Countdown Start Value column is disabled only if Suspend Frame Transmission, Frame Length, SecOC Authenticator Invalidation, and SecOC Freshness Overwrite Value are set to Disabled as well. ▪ The PDU User Code node is deleted from the Bus Configurations table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Applying User Code to PDUs (ConfigurationDesk Bus Manager Implementation Guide 📖) ▪ Bus Manager (stand-alone): Applying User Code to PDUs (Bus Manager (Stand-Alone) Implementation Guide 📖)

User Code ID Lets you specify the user code ID if you enabled the PDU User Code feature for a PDU.

Value	All characters
Description	The user code ID is required to reference the related user code file. It must match the user code ID that is specified for the `DS_Bus_Custom_Feature_Name` attribute in the related user code file.
Dependencies	Available only if PDU User Code is set to Enabled.
More information	▪ Bus Manager in ConfigurationDesk: Applying User Code to PDUs (ConfigurationDesk Bus Manager Implementation Guide 📖) ▪ Bus Manager (stand-alone): Applying User Code to PDUs (Bus Manager (Stand-Alone) Implementation Guide 📖)

Recalculate SecOC Information (User Code) Lets you enable and disable the recalculation of authentication information of an affected secured IPDU if you enabled the PDU User Code feature for a PDU that is involved in secure onboard communication.

Value	Checkbox: - Selected - Cleared
Description	To prevent invalid authentication information that results from using the **PDU User Code** feature, the authentication information can be recalculated before the related secured IPDU is transmitted.
Dependencies	Available only if **PDU User Code** is set to **Enabled**.
More information	- Bus Manager in ConfigurationDesk: Recalculating End-to-End Protection and Authentication Information (ConfigurationDesk Bus Manager Implementation Guide 🕮) - Bus Manager (stand-alone): Recalculating End-to-End Protection and Authentication Information (Bus Manager (Stand-Alone) Implementation Guide 🕮)

SecOC Authenticator Invalidation Lets you enable and disable the **SecOC Authenticator Invalidation** feature for secured IPDUs.

Value	- Enabled - Disabled
Description	- Enabled: The **SecOC Authenticator Invalidation** feature is enabled for the secured IPDU. This adds the following function ports to the related bus configuration: - Current Countdown Value function port - Countdown Start Value function port - Enable function port - Enable State function port Additionally, the **Countdown Start Value** and **Recalculate SecOC Information (Authenticator Invalidation)** columns are enabled and a **SecOC Authenticator Invalidation** node is available in the **Bus Configurations** table. - Disabled: The **SecOC Authenticator Invalidation** feature is disabled for the secured IPDU: - The function ports are deleted from the related bus configuration, regardless of whether they are mapped to model ports. - The **Recalculate SecOC Information (Authenticator Invalidation)** column is disabled. - The **Countdown Start Value** column is disabled only if **Suspend Frame Transmission**, **Frame Length**, **PDU User Code**, and **SecOC Freshness Overwrite Value** are set to **Disabled** as well. - The **SecOC Authenticator Invalidation** node is deleted from the **Bus Configurations** table.
Dependencies	–

More information	- Bus Manager in ConfigurationDesk: Invalidating the Authenticator of Secured IPDUs (ConfigurationDesk Bus Manager Implementation Guide 📖) - Bus Manager (stand-alone): Invalidating the Authenticator of Secured IPDUs (Bus Manager (Stand-Alone) Implementation Guide 📖)

Recalculate SecOC Information (Authenticator Invalidation) Lets you enable and disable to recalculate the authentication information of an affected secured IPDU if you enabled the SecOC Authenticator Invalidation feature for a PDU that is involved in secure onboard communication.

Value	Checkbox: - Selected - Cleared
Description	To prevent invalid authentication information that results from using the SecOC Authenticator Invalidation feature, the authentication information can be recalculated before the related secured IPDU is transmitted.
Dependencies	Available only if SecOC Authenticator Invalidation is set to Enabled.
More information	- Bus Manager in ConfigurationDesk: Invalidating the Authenticator of Secured IPDUs (ConfigurationDesk Bus Manager Implementation Guide 📖) - Bus Manager (stand-alone): Invalidating the Authenticator of Secured IPDUs (Bus Manager (Stand-Alone) Implementation Guide 📖)

SecOC Freshness Overwrite Value Lets you enable and disable the SecOC Freshness Overwrite Value feature for secured IPDUs.

Value	- Enabled - Disabled
Description	- Enabled: The SecOC Freshness Overwrite Value feature is enabled for the secured IPDU. This adds the following function ports to the related bus configuration: - Current Countdown Value function port - Countdown Start Value function port - Enable function port - Enable State function port - Freshness Overwrite Value function port

	Additionally, the Countdown Start Value and Freshness Overwrite Value columns are enabled and a SecOC Freshness Overwrite Value node is available in the Bus Configurations table. ▪ Disabled: The SecOC Freshness Overwrite Value feature is disabled for the secured IPDU: ▪ The function ports are deleted from the related bus configuration, regardless of whether they are mapped to model ports. ▪ The Freshness Overwrite Value column is disabled. ▪ The Countdown Start Value column is disabled only if Suspend Frame Transmission, Frame Length, PDU User Code, and SecOC Authenticator Invalidation are set to Disabled as well. ▪ The SecOC Freshness Overwrite Value node is deleted from the Bus Configurations table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Overwriting the Freshness Value of Secured IPDUs (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Overwriting the Freshness Value of Secured IPDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Freshness Overwrite Value Lets you specify a freshness overwrite value if you enabled the SecOC Freshness Overwrite Value feature for a secured IPDU.

Value	0 … UInt64$_{max}$
Description	The specified freshness overwrite value overwrites the original freshness value at run time and the authenticator is recalculated by using the specified freshness overwrite value. If the authentication information that is included in the secured IPDU consists of the authenticator and the freshness value, the specified freshness overwrite value is included in the secured IPDU. However, depending on the Freshness value TX length setting of the secured IPDU, the value might be truncated.
Dependencies	Available only if SecOC Freshness Overwrite Value is set to Enabled.
More information	▪ Bus Manager in ConfigurationDesk: Overwriting the Freshness Value of Secured IPDUs (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Overwriting the Freshness Value of Secured IPDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Feature Switch Lets you specify the active bus manipulation feature for the selected ISignal.

Value	▪ 0: None ▪ 1: <bus manipulation feature 1> (permanently) ▪ 2: <bus manipulation feature 1> (temporarily) ▪ n: <bus manipulation feature m> (permanently) ▪ n+1: <bus manipulation feature m> (temporarily)
Description	If at least one bus manipulation feature is enabled for the selected ISignal, the feature switch lets you specify the initially active feature and whether it manipulates the ISignal permanently or temporarily. If the feature switch is available, the following function ports are added to the related bus configuration: ▪ Current Countdown Value function port ▪ Countdown Start Value function port ▪ Feature Switch function port ▪ Selected Feature function port Additionally, a **Feature Switch** node is available in the **Bus Configurations** table.
Dependencies	Available only if **ISignal Overwrite Value** or **ISignal Offset Value** is set to **Enabled**.
More information	▪ Bus Manager in ConfigurationDesk: Basics on Bus Manipulation Features (ConfigurationDesk Bus Manager Implementation Guide 📖) ▪ Bus Manager (stand-alone): Basics on Bus Manipulation Features (Bus Manager (Stand-Alone) Implementation Guide 📖)

ISignal Overwrite Value Lets you enable and disable the **ISignal Overwrite Value** feature for ISignals.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The **ISignal Overwrite Value** feature is enabled for the related ISignal. This adds an Overwrite Value function port to the related bus configuration and enables the **Overwrite Value** column. Additionally, the **Feature Switch** column is enabled and an **ISignal Overwrite Value** node is available in the **Bus Configurations** table. If the ISignal is included in an end-to-end-protected ISignal group, the **Recalculate End-to-End Protection (Overwrite)** column is also enabled. ▪ Disabled: The **ISignal Overwrite Value** feature is disabled for the related ISignal. The function port is deleted from the related bus configuration, regardless of whether it is mapped to a model port. Additionally, the **Overwrite Value** and **Recalculate End-to-End Protection (Overwrite)** columns

	are disabled. The Feature Switch column is disabled only if ISignal Offset Value is set to Disabled as well. The ISignal Overwrite Value node is deleted from the Bus Configurations table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Overwriting ISignal Values (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Overwriting ISignal Values (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Overwrite Value Lets you specify an overwrite value if you enabled the ISignal Overwrite Value feature for an ISignal.

Value	Depends on the ISignal's physical base data type and the ISignal's length
Description	The overwrite value overwrites the original ISignal value at run time.
Dependencies	Available only if ISignal Overwrite Value is set to Enabled.
More information	▪ Bus Manager in ConfigurationDesk: Overwriting ISignal Values (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Overwriting ISignal Values (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Recalculate End-to-End Protection (Overwrite) Lets you enable and disable to recalculate the end-to-end protection algorithms if you enabled the ISignal Overwrite Value feature for an ISignal that is included in an end-to-end-protected ISignal group.

Value	Checkbox: ▪ Selected ▪ Cleared
Description	To prevent invalid end-to-end protection checksum or counter values that result from using the ISignal Overwrite Value feature, the end-to-end protection algorithms can be recalculated before the related ISignal group is transmitted.
Dependencies	Available only if ISignal Overwrite Value is set to Enabled.
More information	▪ Bus Manager in ConfigurationDesk: Recalculating End-to-End Protection and Authentication Information (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Recalculating End-to-End Protection and Authentication Information (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Recalculate SecOC Information (Overwrite) Lets you enable and disable to recalculate the authentication information of an affected secured IPDU if you enabled the ISignal Overwrite Value feature for an ISignal that is involved in secure onboard communication.

Value	Checkbox: ▪ Selected ▪ Cleared
Description	To prevent invalid authentication information that results from using the ISignal Overwrite Value feature, the authentication information can be recalculated before the related secured IPDU is transmitted.
Dependencies	Available only if ISignal Overwrite Value is set to Enabled.
More information	▪ Bus Manager in ConfigurationDesk: Recalculating End-to-End Protection and Authentication Information (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Recalculating End-to-End Protection and Authentication Information (Bus Manager (Stand-Alone) Implementation Guide 🕮)

ISignal Offset Value Lets you enable and disable the ISignal Offset Value feature for ISignals.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The ISignal Offset Value feature is enabled for the related ISignal. This adds the following function ports to the related bus configuration: ▪ Offset Value function port ▪ Minimum ISignal Value function port ▪ Maximum ISignal Value function port Additionally, the following columns are enabled: ▪ Feature Switch ▪ Offset Value ▪ Minimum ISignal Value ▪ Maximum ISignal Value ▪ Recalculate End-to-End Protection (Offset) (only for ISignals that are included in an end-to-end-protected ISignal group) In the Bus Configurations table, an ISignal Offset Value node is available. ▪ Disabled: The ISignal Offset Value feature is disabled for the related ISignal. The function ports are deleted from the related bus configuration, regardless of whether they are mapped to model ports. Additionally, the Offset Value, Minimum ISignal Value, Maximum ISignal Value, and Recalculate End-to-End Protection (Offset) columns are disabled. The Feature Switch column is disabled only if ISignal Overwrite Value is set to Disabled as well. The ISignal Offset Value node is deleted from the Bus Configurations table.

Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Adding Offset Values to ISignal Values (ConfigurationDesk Bus Manager Implementation Guide 📖) ▪ Bus Manager (stand-alone): Adding Offset Values to ISignal Values (Bus Manager (Stand-Alone) Implementation Guide 📖)

Offset Value Lets you specify an offset value if you enabled the **ISignal Offset Value** feature for an ISignal.

Value	▪ Depends on the ISignal's physical base data type and the ISignal's length ▪ If the physical base data type is an unsigned integer type, the value range is derived from the related integer type (e.g., with a physical base data type of UInt8, the value range is Int8_{min} … Int8_{max}).
Description	The offset value is added to the original ISignal value at run time.
Dependencies	Available only if **ISignal Offset Value** is set to **Enabled**.
More information	▪ Bus Manager in ConfigurationDesk: Adding Offset Values to ISignal Values (ConfigurationDesk Bus Manager Implementation Guide 📖) ▪ Bus Manager (stand-alone): Adding Offset Values to ISignal Values (Bus Manager (Stand-Alone) Implementation Guide 📖)

Minimum ISignal Value Lets you specify the minimum value of an ISignal if you enabled the **ISignal Offset Value** feature for the ISignal.

Value	Depends on the ISignal's physical base data type and the ISignal's length
Description	The minimum ISignal value is the lower limit of the ISignal value.
Dependencies	Available only if **ISignal Offset Value** is set to **Enabled**.
More information	▪ Bus Manager in ConfigurationDesk: Adding Offset Values to ISignal Values (ConfigurationDesk Bus Manager Implementation Guide 📖) ▪ Bus Manager (stand-alone): Adding Offset Values to ISignal Values (Bus Manager (Stand-Alone) Implementation Guide 📖)

Maximum ISignal Value Lets you specify the maximum value of an ISignal if you enabled the **ISignal Offset Value** feature for the ISignal.

Value	Depends on the ISignal's physical base data type and the ISignal's length
Description	The maximum ISignal value is the upper limit of the ISignal value.
Dependencies	Available only if **ISignal Offset Value** is set to **Enabled**.

More information	▪ Bus Manager in ConfigurationDesk: Adding Offset Values to ISignal Values (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Adding Offset Values to ISignal Values (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Recalculate End-to-End Protection (Offset) Lets you enable and disable to recalculate the end-to-end protection algorithms if you enabled the **ISignal Offset Value** feature for an ISignal that is included in an end-to-end-protected ISignal group.

Value	Checkbox: ▪ Selected ▪ Cleared
Description	To prevent invalid end-to-end protection checksum or counter values that result from using the **ISignal Offset Value** feature, the end-to-end protection algorithms can be recalculated before the related ISignal group is transmitted.
Dependencies	Available only if **ISignal Offset Value** is set to **Enabled**.
More information	▪ Bus Manager in ConfigurationDesk: Recalculating End-to-End Protection and Authentication Information (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Recalculating End-to-End Protection and Authentication Information (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Recalculate SecOC Information (Offset) Lets you enable and disable to recalculate the authentication information of an affected secured IPDU if you enabled the **ISignal Offset Value** feature for an ISignal that is involved in secure onboard communication.

Value	Checkbox: ▪ Selected ▪ Cleared
Description	To prevent invalid authentication information that results from using the **ISignal Offset Value** feature, the authentication information can be recalculated before the related secured IPDU is transmitted.
Dependencies	Available only if **ISignal Offset Value** is set to **Enabled**.
More information	▪ Bus Manager in ConfigurationDesk: Recalculating End-to-End Protection and Authentication Information

	(ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Recalculating End-to-End Protection and Authentication Information (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Related Clusters Displays the communication clusters a bus configuration element is a member of. The displayed communication clusters depend on the communication matrix:

- AUTOSAR and FIBEX communication matrices:

  The displayed clusters are derived from the communication matrix and not from the element's configuration in the bus configuration. For example, if an ECU is assigned to a bus configuration in the context of one communication cluster but the ECU is a member of two clusters, both clusters are displayed in this column.

- DBC and LDF communication matrices:

  DBC and LDF communication matrices specify the communication of only one communication cluster. The displayed name is the name of the communication cluster in the bus configuration. You can change the cluster name via the Properties Browser when you select an assigned PDU, for example. If you do this, the displayed name is adjusted accordingly.

Related TX ECUs Displays all the ECUs that transmit a bus configuration element. The displayed ECUs are derived from the communication matrix and not from the element's configuration in the bus configuration. For example, if a PDU is assigned to a bus configuration in the context of one transmitting ECU but the PDU is transmitted by two ECUs, both ECUs are displayed in this column.

Related RX ECUs Displays all the ECUs that receive a bus configuration element. The displayed ECUs are derived from the communication matrix and not from the element's configuration in the bus configuration. For example, if a PDU is assigned to a bus configuration in the context of one receiving ECU but the PDU is received by two ECUs, both ECUs are displayed in this column.

Related topics

Basics

Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide 🕮)

Bus Simulation Features Table

Purpose

To access and configure PDU-related and signal-related bus configuration features of a ConfigurationDesk application for simulation purposes.

Description

The Bus Simulation Features table allows you to:

- Display all the PDUs and ISignals of all the bus configurations of a ConfigurationDesk application that are used for simulation purposes.
- Display all the PDU-related and signal-related bus configuration features that are available for simulation purposes.
- Add bus configuration features to PDUs and ISignals for simulation purposes.
- Configure bus configuration features.
- Select PDUs and ISignals to access their properties in the **Properties Browser**.
- Remove PDUs and ISignals from the Simulated ECUs node of bus configurations.

For more information on bus configuration features, refer to:

- Bus Manager in ConfigurationDesk: Working with Bus Configuration Features (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Working with Bus Configuration Features (Bus Manager (Stand-Alone) Implementation Guide)

Symbols

The PDUs and ISignals are displayed with various symbols. For an overview of all the available symbols, refer to Elements and Symbols of Bus Configurations and Communication Matrices on page 429.

Subviews

The Bus Simulation Features table provides the following subviews:

Subview	Purpose
PDU Features	Provides a list of all the PDUs that are assigned to the Simulated ECUs parts of all bus configurations, and lets you access the PDU-related bus simulation features. This is the default view.
Signal Features	Provides a list of all the ISignals that are assigned to the Simulated ECUs parts of all bus configurations, and lets you access the signal-related bus simulation features.

Columns

The Bus Simulation Features table provides various columns. Depending on the selected subview, the following columns are available:

Name Displays the names of bus configuration elements. You can change some of the displayed names , e.g., the names of communication matrix nodes. Changing the names directly affects the TRC file. Refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Configurations (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Basics on Bus Configurations (Bus Manager (Stand-Alone) Implementation Guide)

Element Type Displays the type of bus configuration elements, e.g., Bus ECU, Bus Network Node, or Bus ISignal IPDU.

Direction Displays the direction (RX or TX) of bus configuration elements, e.g., of ISignal IPDUs or ISignals.

Bus Configuration Displays the name of the bus configuration to which the PDUs or ISignals are assigned. You can change the displayed name.

Communication Matrix Displays the name of the communication matrix node for which the PDUs or ISignals are available. You can change the displayed name.

ECU Displays the name of the ECU for which the PDUs or ISignals are assigned to the bus configuration.

PDU Displays the name of the PDU for which the ISignals are assigned to the bus configuration.

PDU Raw Data Lets you enable and disable the PDU Raw Data feature for PDUs.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The PDU Raw Data feature is enabled for the related PDU. This adds a PDU Raw Data function port to the related bus configuration and enables the Access Mode column. Additionally, a PDU Raw Data node is available in the Bus Configurations table. ▪ Disabled: The PDU Raw Data feature is disabled for the related PDU. The function port is deleted from the related bus configuration, regardless of whether it is mapped to a model port. The Access Mode column is disabled and the PDU Raw Data node is deleted from the Bus Configurations table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Accessing the Payload of PDUs in Raw Data Format (ConfigurationDesk Bus Manager Implementation Guide) ▪ Bus Manager (stand-alone): Accessing the Payload of PDUs in Raw Data Format (Bus Manager (Stand-Alone) Implementation Guide)

Access Mode Displays the access mode if you enabled the PDU Raw Data feature for a PDU. For TX basic PDUs, you can specify the access mode.

Value	▪ Read ▪ Write
Description	▪ Read: The payload of the PDU is read in raw data format.

	▪ Write: Available only for TX basic PDUs. Raw data is written to the payload of the TX basic PDU.
Dependencies	Available only if **PDU Raw Data** is set to **Enabled**.
More information	▪ Bus Manager in ConfigurationDesk: Accessing the Payload of PDUs in Raw Data Format (ConfigurationDesk Bus Manager Implementation Guide 📖) ▪ Bus Manager (stand-alone): Accessing the Payload of PDUs in Raw Data Format (Bus Manager (Stand-Alone) Implementation Guide 📖)

PDU Length Lets you enable and disable the **PDU Length** feature for PDUs.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The **PDU Length** feature is enabled for the related PDU. Depending on the direction of the PDU (TX or RX), this adds a TX Length or RX Length function port to the related bus configuration. Additionally, a **PDU Length** node is available in the **Bus Configurations** table. ▪ Disabled: The **PDU Length** feature is disabled for the related PDU. The function port is deleted from the related bus configuration, regardless of whether it is mapped to a model port. The **PDU Length** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Accessing the Payload Length of CAN PDUs (ConfigurationDesk Bus Manager Implementation Guide 📖) ▪ Bus Manager (stand-alone): Accessing the Payload Length of CAN PDUs (Bus Manager (Stand-Alone) Implementation Guide 📖)

PDU User Code Lets you enable and disable the **PDU User Code** feature for PDUs.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The **PDU User Code** feature is enabled for the related PDU. If the PDU is an RX PDU, this adds a Result function port to the related bus configuration. Additionally, the **User Code ID** column is enabled and a **PDU User Code** node is available in the **Bus Configurations** table. ▪ Disabled: The **PDU User Code** feature is disabled for the related PDU. If applicable, the Result function port is deleted from the related bus configuration, regardless of whether it is

	mapped to a model port. The **User Code ID** column is disabled and the **PDU User Code** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	- Bus Manager in ConfigurationDesk: Applying User Code to PDUs (ConfigurationDesk Bus Manager Implementation Guide 🕮) - Bus Manager (stand-alone): Applying User Code to PDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

User Code ID Lets you specify the user code ID if you enabled the **PDU User Code** feature for a PDU.

Value	All characters
Description	The user code ID is required to reference the related user code file. It must match the user code ID that is specified for the `DS_Bus_Custom_Feature_Name` attribute in the related user code file.
Dependencies	Available only if **PDU User Code** is set to **Enabled**.
More information	- Bus Manager in ConfigurationDesk: Applying User Code to PDUs (ConfigurationDesk Bus Manager Implementation Guide 🕮) - Bus Manager (stand-alone): Applying User Code to PDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

SecOC Lets you enable and disable the **SecOC** feature for RX secured IPDUs.

Value	- Enabled - Disabled
Description	- Enabled: The **SecOC** feature is enabled for the related secured IPDU. This adds the following function ports to the related bus configuration: - Enable Verification function port - State function port Additionally, a **SecOC** node is available in the **Bus Configurations** table. - Disabled: The **SecOC** feature is disabled for the related secured IPDU. The function ports are deleted from the related bus configuration, regardless of whether they are mapped to model ports. The **SecOC** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	- Bus Manager in ConfigurationDesk: Verifying the Authentication Information of Received Secured IPDUs

	(ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Verifying the Authentication Information of Received Secured IPDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

PDU Trigger Lets you enable and disable the **PDU Trigger** feature for PDUs.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The **PDU Trigger** feature is enabled for the related PDU. This adds a PDU Trigger function port to the related bus configuration and enables the **Trigger Mode** column. Additionally, a **PDU Trigger** node is available in the **Bus Configurations** table. ▪ Disabled: The **PDU Trigger** feature is disabled for the related PDU. The function port is deleted from the related bus configuration, regardless of whether it is mapped to a model port. The **Trigger Mode** column is disabled only if **Frame Access** is set to **Disabled** as well. The **PDU Trigger** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Specifying User-Defined Triggers for Transmitting PDUs (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Specifying User-Defined Triggers for Transmitting PDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Trigger Mode Lets you specify the trigger mode if you enabled the **PDU Trigger** feature for a PDU or the **Frame Access** feature for a frame of a TX PDU.

Value	▪ Positive edge ▪ Level-triggered
Description	▪ Positive edge: The transmission of the PDU or frame is triggered once each time the value of the related Trigger function port changes from 0 to 1. ▪ Level-triggered: The transmission of the PDU or frame is triggered continuously as long as the value of the related Trigger function port is set to 1.
Dependencies	Available only if **PDU Trigger** or **Frame Access** is set to **Enabled**.
More information	▪ Bus Manager in ConfigurationDesk: Triggering the Transmission of PDUs and Frames via User-Defined Triggers

	(ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Triggering the Transmission of PDUs and Frames via User-Defined Triggers (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Frame Access Lets you enable and disable the **Frame Access** feature for frames.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The **Frame Access** feature is enabled for the frame of the selected PDU. Depending on the direction of the PDU (TX or RX), this adds the following function ports to the related bus configuration: ▪ Trigger function port (available only for TX PDUs) ▪ State function port (available only for RX PDUs) ▪ TX Length function port (for TX PDUs) or RX Length function port (for RX PDUs) ▪ TX Raw Data function port (for TX PDUs) or RX Raw Data function port (for RX PDUs) ▪ Extended Addressing function port ▪ CAN FD Frame Support function port ▪ Bit Rate Switch function port Additionally, the following columns are enabled: ▪ **Maximum Length** ▪ **Trigger Mode** (only for TX PDUs) In the **Bus Configurations** table, a **Frame Access** node is available. ▪ Disabled: The **Frame Access** feature is disabled for the frame of the selected PDU. The function ports are deleted from the related bus configuration, regardless of whether they are mapped to model ports. Additionally, the **Maximum Length** column is disabled. The **Trigger Mode** column is disabled only if **PDU Trigger** is set to **Disabled** as well. The **Frame Access** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Accessing CAN Frame Settings (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Accessing CAN Frame Settings (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Maximum Length Lets you specify the maximum length for a frame if you enabled the Frame Access feature for its related PDU.

Value	1 … 8, 12, 16, 20, 24, 32, 48, and 64 bytes
Description	The specified length is the maximum length that can be used for the frame at run time.
Dependencies	Available only if Frame Access is set to Enabled.
More information	▪ Bus Manager in ConfigurationDesk: Accessing CAN Frame Settings (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Accessing CAN Frame Settings (Bus Manager (Stand-Alone) Implementation Guide 🕮)

PDU Cyclic Timing Control Lets you enable and disable the PDU Cyclic Timing Control feature for PDUs.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The PDU Cyclic Timing Control feature is enabled for the related PDU. This adds the following function ports to the related bus configuration: ▪ PDU Timing Control Period function port ▪ PDU Timing Control Offset function port Additionally, a PDU Cyclic Timing Control node is available in the Bus Configurations table. ▪ Disabled: The PDU Cyclic Timing Control feature is disabled for the related PDU. The function ports are deleted from the related bus configuration, regardless of whether they are mapped to model ports. The PDU Cyclic Timing Control node is deleted from the Bus Configurations table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Controlling the Cyclic Timing of CAN PDUs (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Controlling the Cyclic Timing of CAN PDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

PDU Enable Lets you enable and disable the PDU Enable feature for PDUs.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The PDU Enable feature is enabled for the related PDU. This adds a PDU Enable function port to the related bus

	configuration. Additionally, a **PDU Enable** node is available in the **Bus Configurations** table. ▪ Disabled: The **PDU Enable** feature is disabled for the related PDU. The function port is deleted from the related bus configuration, regardless of whether it is mapped to a model port. The **PDU Enable** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Enabling and Disabling the Transmission of PDUs (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Enabling and Disabling the Transmission of PDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

PDU RX Interrupt Lets you enable and disable the **PDU RX Interrupt** feature for PDUs.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The **PDU RX Interrupt** feature is enabled for the related PDU. This adds the following ports to the related bus configuration: ▪ PDU RX Interrupt Enable function port ▪ PDU Received event port Additionally, a **PDU RX Interrupt** node is available in the **Bus Configurations** table. ▪ Disabled: The **PDU RX Interrupt** feature is disabled for the related PDU. The ports are deleted from the related bus configuration, regardless of whether they are mapped to model ports. The **PDU RX Interrupt** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Triggering the Execution of Functions in a Behavior Model via RX Interrupts (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Triggering the Execution of Functions in a Behavior Model via RX Interrupts (Bus Manager (Stand-Alone) Implementation Guide 🕮)

PDU RX Status Lets you enable and disable the **PDU RX Status** feature for PDUs.

Value	▪ Enabled ▪ Disabled

Description	▪ Enabled: The **PDU RX Status** feature is enabled for the related PDU. This adds the following function ports to the related bus configuration: ▪ PDU Counter function port ▪ PDU State function port ▪ PDU Time function port ▪ PDU Delta Time function port Additionally, a **PDU RX Status** node is available in the **Bus Configurations** table. ▪ Disabled: The **PDU RX Status** feature is disabled for the related PDU. The function ports are deleted from the related bus configuration, regardless of whether they are mapped to model ports. The **PDU RX Status** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Observing the Status of Received PDUs (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Observing the Status of Received PDUs (Bus Manager (Stand-Alone) Implementation Guide 🕮)

ISignal Value Lets you enable and disable the **ISignal Value** feature for ISignals.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The **ISignal Value** feature is enabled for the related ISignal. Depending on the direction of the ISignal (TX⧉ or RX⧉), this adds a TX ISignal Value function port or RX ISignal function port to the related bus configuration. Additionally, an **ISignal Value** node is available in the **Bus Configurations** table. ▪ Disabled: The **ISignal Value** feature is disabled for the related ISignal. The function port is deleted from the related bus configuration, regardless of whether it is mapped to a model port. The **ISignal Value** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Working with ISignal Values (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Working with ISignal Values (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Counter Signal Lets you enable and disable the **Counter Signal** feature for ISignals.

Value	▪ Enabled ▪ Disabled
Description	▪ Enabled: The **Counter Signal** feature is enabled for the related ISignal. If the ISignal is an RX ISignal, this adds a Counter State function port to the related bus configuration. Additionally, the following columns are enabled: ▪ **Minimum Value** ▪ **Maximum Value** ▪ **Increment Value** ▪ **Initial Value** ▪ **Step Length** In the **Bus Configurations** table, a **Counter Value** node is available. ▪ Disabled: The **Counter Signal** feature is disabled for the related ISignal. If applicable, the Counter State function port is deleted from the related bus configuration, regardless of whether it is mapped to a model port. Additionally, the related columns are disabled and the **Counter Value** node is deleted from the **Bus Configurations** table.
Dependencies	–
More information	▪ Bus Manager in ConfigurationDesk: Working with Counter Signals (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Working with Counter Signals (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Minimum Value Lets you specify the minimum value for a counter if you enabled the **Counter Signal** feature for an ISignal.

Value	▪ Depends on the ISignal's physical and coded base data types, and the ISignal's length ▪ Maximum value range: $\text{Int32}_{\text{min}}$ … $\text{Int32}_{\text{max}}$
Description	If a counter value reaches the minimum value, the counter continues counting, starting with the specified **Maximum Value**.
Dependencies	Available only if **Counter Signal** is set to **Enabled**.
More information	▪ Bus Manager in ConfigurationDesk: Working with Counter Signals (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Working with Counter Signals (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Maximum Value Lets you specify the maximum value for a counter if you enabled the **Counter Signal** feature for an ISignal.

Value	▪ Depends on the ISignal's physical and coded base data types, and the ISignal's length ▪ Maximum range: $\text{Int32}_{\text{min}}$ … $\text{Int32}_{\text{max}}$
Description	If a counter value reaches the maximum value, the counter continues counting, starting with the specified **Minimum Value**.
Dependencies	Available only if **Counter Signal** is set to **Enabled**.
More information	▪ Bus Manager in ConfigurationDesk: Working with Counter Signals (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Working with Counter Signals (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Increment Value Lets you specify the increment value for a counter if you enabled the **Counter Signal** feature for an ISignal.

Value	-(**Maximum Value** - **Minimum Value**) ... +(**Maximum Value** - **Minimum Value**)
Description	The increment value determines the step size for incrementing the counter value.
Dependencies	Available only if **Counter Signal** is set to **Enabled**.
More information	▪ Bus Manager in ConfigurationDesk: Working with Counter Signals (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Working with Counter Signals (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Initial Value Lets you specify the initial value for a counter if you enabled the **Counter Signal** feature for an ISignal.

Value	**Minimum Value** … **Maximum Value**
Description	The specified value is used as the counter's initial value directly after the executable application is started. Depending on the setting of the bus configuration's **Initial value usage** property, the specified initial value is set only once at first application start or every time the application starts, including when the application state switches from stopped to running.
Dependencies	Available only if **Counter Signal** is set to **Enabled**.
More information	▪ For more information on the **Counter Signal** feature, refer to: ▪ Bus Manager in ConfigurationDesk: Working with Counter Signals (ConfigurationDesk Bus Manager Implementation Guide 🕮)

	▪ Bus Manager (stand-alone): Working with Counter Signals (Bus Manager (Stand-Alone) Implementation Guide 🕮) ▪ For more information on the **Initial value usage** property, refer to Common Function Block Properties (Bus Configuration) (ConfigurationDesk Function Block Properties 🕮).

Step length Lets you specify the step length for a counter if you enabled the Counter Signal feature for an ISignal.

Value	1 … Int32$_{max}$
Description	The step length determines the frequency for incrementing the counter value, i.e., whether the counter value is incremented with each transmission/reception of the related PDU, with every other transmission/reception etc.
Dependencies	Available only if Counter Signal is set to Enabled.
More information	▪ Bus Manager in ConfigurationDesk: Working with Counter Signals (ConfigurationDesk Bus Manager Implementation Guide 🕮) ▪ Bus Manager (stand-alone): Working with Counter Signals (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Related Clusters Displays the communication clusters a bus configuration element is a member of. The displayed communication clusters depend on the communication matrix:

- AUTOSAR and FIBEX communication matrices:

  The displayed clusters are derived from the communication matrix and not from the element's configuration in the bus configuration. For example, if an ECU is assigned to a bus configuration in the context of one communication cluster but the ECU is a member of two clusters, both clusters are displayed in this column.

- DBC and LDF communication matrices:

  DBC and LDF communication matrices specify the communication of only one communication cluster. The displayed name is the name of the communication cluster in the bus configuration. You can change the cluster name via the Properties Browser when you select an assigned PDU, for example. If you do this, the displayed name is adjusted accordingly.

Related TX ECUs Displays all the ECUs that transmit a bus configuration element. The displayed ECUs are derived from the communication matrix and not from the element's configuration in the bus configuration. For example, if a PDU is assigned to a bus configuration in the context of one transmitting ECU but the PDU is transmitted by two ECUs, both ECUs are displayed in this column.

Related RX ECUs Displays all the ECUs that receive a bus configuration element. The displayed ECUs are derived from the communication matrix and not from the element's configuration in the bus configuration. For example, if a PDU is assigned to a bus configuration in the context of one receiving ECU but the PDU is received by two ECUs, both ECUs are displayed in this column.

Related topics

Basics

Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide)

Buses Browser

Purpose

To display and access the communication matrices of a ConfigurationDesk application.

Description

The Buses Browser provides a hierarchically structured access to all the communication matrices of the active ConfigurationDesk application. You can:

- Add or remove communication matrices to or from the application.
- Select different views of the communication matrices (Clusters or ECUs view).
- Select communication matrix elements and drag them to bus configurations to implement bus communication in the signal chain.
- Use various context menu commands that let you, for example, filter the displayed communication matrices.

Symbols

The communication matrix elements are displayed with various symbols. For an overview of all the available symbols, refer to Elements and Symbols of Bus Configurations and Communication Matrices on page 429.

Columns

The Buses Browser provides the following columns:

Name Displays the name of communication matrix elements.

Direction Displays the direction (RX or TX) of communication matrix elements, e.g., of ISignal IPDUs or ISignals.

Element Type Displays the type of communication matrix elements, e.g., Bus CAN Communication Cluster, Bus Network Node, or Bus ISignal IPDU.

Related topics

Basics

Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide)
Elements of the Bus Manager (Bus Manager (Stand-Alone) Implementation Guide)
Elements of the Bus Manager (ConfigurationDesk Bus Manager Implementation Guide)

Clear (Communication Matrices from Project)

Purpose

To remove all the communication matrices and the associated bus configuration elements from the active ConfigurationDesk application.

Result

All the communication matrices and the associated bus configuration elements are removed from the active application.

> **Note**
>
> The bus communication that is implemented via the Bus Manager is lost completely.

Description

Because all the bus configuration elements that use the communication matrices are removed as well, the entire bus communication that is implemented via the Bus Manager is removed from the active ConfigurationDesk application.

If you do not want to remove all the communication matrices but only individual communication matrices and their associated bus configuration elements, you can use the Delete Completely on page 222 command.

If you removed the communication matrices by mistake, you can undo the action (refer to Undo on page 529).

Elements and Symbols of Bus Configurations and Communication Matrices

Introduction

Bus configurations and communication matrices consist of various elements that can be accessed via the **Buses Browser**, Bus Manager-related tables, and/or the **Properties Browser**. You can identify the elements by their symbols.

Bus configuration

Symbol	⚙
Available in	▪ Bus Configurations table ▪ Bus Access Requests table ▪ Properties Browser

A bus configuration is the main Bus Manager element for implementing bus communication in a ConfigurationDesk application. A bus configuration can be accessed via various tables and its **Bus Configuration** function block. For more information, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Configurations (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Basics on Bus Configurations (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Simulated ECUs

Symbol	🗂
Available in	▪ Bus Configurations table ▪ Properties Browser

This element is one part of a bus configuration. It provides access to all the ECUs and their communication elements that are assigned to the bus configuration for simulation purposes. For more information, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Configurations (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Basics on Bus Configurations (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Inspection

Symbol	🔍
Available in	▪ Bus Configurations table ▪ Properties Browser

This element is one part of a bus configuration. It provides access to the PDUs and ISignals of all the communication clusters that are assigned to the bus configuration for inspection purposes. For more information, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Configurations (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Basics on Bus Configurations (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Manipulation

Symbol	🛠
Available in	▪ Bus Configurations table ▪ Properties Browser

This element is one part of a bus configuration. It provides access to the PDUs and ISignals of all the communication clusters that are assigned to the bus configuration for manipulation purposes. For more information, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Configurations (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Basics on Bus Configurations (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Gateways

Symbol	
Available in	▪ Bus Configurations table ▪ Properties Browser

This element is one part of a bus configuration. It provides access to the frame gateways that are specified for the bus configuration. For more information, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Configurations (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Basics on Bus Configurations (Bus Manager (Stand-Alone) Implementation Guide)

Bus access requests

Symbol	
Available in	▪ Bus Access Requests table ▪ Properties Browser

This element is one part of a bus configuration. It provides access to all the bus access requests of the bus configuration. For more information, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Access Requests (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Basics on Bus Access Requests (Bus Manager (Stand-Alone) Implementation Guide)

Communication matrix

Symbol	
Available in	▪ Bus Configurations table ▪ Bus Access Requests table ▪ Buses Browser ▪ Properties Browser

This element indicates a communication matrix that is available in the ConfigurationDesk application and provides access to information on the communication matrix (e.g., the file format). A communication matrix defines the communication of a bus network. When you assign elements of communication matrices to bus configurations, you can configure the bus communication for simulation or inspection purposes, for example. For more information, refer to:

- Bus Manager in ConfigurationDesk: Working with Communication Matrices (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Working with Communication Matrices (Bus Manager (Stand-Alone) Implementation Guide)

Frame capture

Symbol	
Available in	▪ Bus Configurations table

	▪ Bus Access Requests table ▪ Properties Browser

This element indicates a frame capture that is available for the Inspection part of a bus configuration. For more information, refer to:

- Bus Manager in ConfigurationDesk: Capturing CAN Frames (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Capturing CAN Frames (Bus Manager (Stand-Alone) Implementation Guide)

Frame gateway

Symbol	🟥
Available in	▪ Bus Configurations table ▪ Bus Access Requests table ▪ Properties Browser

This element indicates a frame gateway that is available for the Gateways part of a bus configuration. For more information, refer to:

- Bus Manager in ConfigurationDesk: Specifying CAN Gateways (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Specifying CAN Gateways (Bus Manager (Stand-Alone) Implementation Guide)

Bus system

Symbol	▪ (CAN bus) ▪ (LIN bus)
Available in	▪ Bus Access Requests table ▪ Buses Browser ▪ Properties Browser

This element indicates the bus system of communication clusters that are available in a communication matrix or bus configuration.

Communication cluster

Symbol	▪ (CAN communication cluster, CAN frame capture cluster, or CAN frame gateway cluster) ▪ (LIN communication cluster)
Available in	▪ Bus Configurations table ▪ Bus Access Requests table ▪ Buses Browser ▪ Properties Browser

This element indicates one the following:

- A CAN or LIN communication cluster that is available in a communication matrix or in a bus configuration if a communication matrix element is assigned to the bus configuration. This element provides access to properties of the communication cluster (e.g., the baud rate). Some of these properties can be configured. For more information, refer to:

  - Bus Manager in ConfigurationDesk: Configurable Settings of Communication Clusters (ConfigurationDesk Bus Manager Implementation Guide)
  - Bus Manager (stand-alone): Configurable Settings of Communication Clusters (Bus Manager (Stand-Alone) Implementation Guide)

- A CAN frame capture cluster or CAN frame gateway cluster. Frame capture clusters are available for frame captures of the **Inspection** part of a bus configuration. Frame gateway clusters are available for frame gateways of the **Gateways** part. For more information, refer to:
  - Bus Manager in ConfigurationDesk: Specifying CAN Frame Captures and Gateways (ConfigurationDesk Bus Manager Implementation Guide)
  - Bus Manager (stand-alone): Specifying CAN Frame Captures and Gateways (Bus Manager (Stand-Alone) Implementation Guide)

Bus access request of simulated ECUs

Symbol	
Available in	- Bus Access Requests table - Properties Browser

This element indicates the bus access request of the ECUs of one communication cluster that are assigned to a bus configuration for simulation purposes. This element also provides access to the assigned bus access. For more information, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Access Requests (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Basics on Bus Access Requests (Bus Manager (Stand-Alone) Implementation Guide)

Bus access request for inspection

Symbol	
Available in	- Bus Access Requests table - Properties Browser

This element indicates the bus access request of elements of one communication cluster that are assigned to a bus configuration for inspection purposes. This element also provides access to the assigned bus access. For more information, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Access Requests (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Basics on Bus Access Requests (Bus Manager (Stand-Alone) Implementation Guide)

Bus access request for manipulation

Symbol	
Available in	▪ Bus Access Requests table ▪ Properties Browser

This element indicates the bus access request of elements of one communication cluster that are assigned to a bus configuration for manipulation purposes. This element also provides access to the assigned bus access. For more information, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Access Requests (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Basics on Bus Access Requests (Bus Manager (Stand-Alone) Implementation Guide)

Bus access request for gateways

Symbol	
Available in	▪ Bus Access Requests table ▪ Properties Browser

This element indicates the bus access request of a frame gateway that is added to a bus configuration. This element also provides access to the assigned bus access. For more information, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Access Requests (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Basics on Bus Access Requests (Bus Manager (Stand-Alone) Implementation Guide)

Filter

Symbol	
Available in	▪ Bus Configurations table ▪ Properties Browser

This element indicates a filter that is available for the following elements:

- Frame captures of the Inspection part of a bus configuration.
- Frame gateways of the Gateways part of a bus configuration.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Specifying Filters for Frame Captures and Frame Gateways (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Specifying Filters for Frame Captures and Frame Gateways (Bus Manager (Stand-Alone) Implementation Guide)

CAN filter rule

Symbol	
Available in	▪ Bus Configurations table ▪ Properties Browser

This element indicates a CAN filter rule that is available for a filter. It provides access to the specified filter settings. For more information, refer to:

- Bus Manager in ConfigurationDesk: Specifying Filters for Frame Captures and Frame Gateways (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Specifying Filters for Frame Captures and Frame Gateways (Bus Manager (Stand-Alone) Implementation Guide)

ECU

Symbol	🖥
Available in	▪ Bus Configurations table ▪ Buses Browser ▪ Properties Browser

This element indicates an ECU and provides access to properties of the ECU (e.g., its communication controllers).

Network node

Symbol	🖥
Available in	▪ Buses Browser ▪ Properties Browser

This element indicates a network node and provides access to information on the network node (e.g., its communication controller).

Communication controller

Symbol	▪ (CAN communication controller) ▪ (LIN master communication controller) ▪ (LIN slave communication controller)
Available in	▪ Bus Configurations table ▪ Properties Browser

This element indicates a CAN, LIN master, or LIN slave communication controller and provides access to the properties of the communication controller (e.g., the time base of a LIN master communication controller).

Bus channel

Symbol	▪ (CAN bus channel) ▪ (LIN bus channel)
Available in	▪ Bus Configurations table ▪ Buses Browser ▪ Properties Browser

This element indicates a CAN or LIN bus channel that is specified in a communication matrix for the communication of a network node.

Basic PDU

Symbol	▪ (TX) ▪ (RX)
Available in	▪ Bus Configurations table ▪ Bus Simulation Features table ▪ Bus Inspection Features table ▪ Bus Manipulation Features table ▪ Buses Browser ▪ Properties Browser

This element indicates a basic PDU and its direction (TX, RX), and provides access to properties of the basic PDU (e.g., its frame triggering). Some of these properties can be configured. For more information, refer to:

- Bus Manager in ConfigurationDesk: Configurable Settings of PDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Configurable Settings of PDUs (Bus Manager (Stand-Alone) Implementation Guide)

Multiplexed IPDU

Symbol	▪ (TX) ▪ (RX)
Available in	▪ Bus Configurations table ▪ Bus Simulation Features table ▪ Bus Inspection Features table ▪ Bus Manipulation Features table ▪ Buses Browser ▪ Properties Browser

This element indicates a multiplexed IPDU and its direction (TX, RX), and provides access to properties of the multiplexed IPDU (e.g., its dynamic part). Some of these properties can be configured. For more information, refer to:

- Bus Manager in ConfigurationDesk: Configurable Settings of PDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Configurable Settings of PDUs (Bus Manager (Stand-Alone) Implementation Guide)

Container IPDU

Symbol	▪ (TX) ▪ (RX)
Available in	▪ Bus Configurations table ▪ Bus Simulation Features table ▪ Bus Inspection Features table ▪ Bus Manipulation Features table ▪ Buses Browser ▪ Properties Browser

This element indicates a container IPDU and its direction (TX, RX), and provides access to properties of the container IPDU (e.g., the header type). Some of these properties can be configured. For more information, refer to:

- Bus Manager in ConfigurationDesk: Configurable Settings of PDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Configurable Settings of PDUs (Bus Manager (Stand-Alone) Implementation Guide)

Secured IPDU

Symbol	▪ (TX) ▪ (RX)
Available in	▪ Bus Configurations table ▪ Bus Simulation Features table ▪ Bus Inspection Features table ▪ Bus Manipulation Features table ▪ Buses Browser ▪ Properties Browser

This element indicates a secured IPDU and its direction (TX, RX), and provides access to properties of the secured IPDU (e.g., the header type). Some of these properties can be configured. Secured IPDUs are used in secure onboard communication scenarios.

- For information on secure onboard communication, refer to:
  - Bus Manager in ConfigurationDesk: Aspects of Supported AUTOSAR Features (ConfigurationDesk Bus Manager Implementation Guide)
  - Bus Manager (stand-alone): Aspects of Supported AUTOSAR Features (Bus Manager (Stand-Alone) Implementation Guide)
- For more information on the configurable properties, refer to:
  - Bus Manager in ConfigurationDesk: Configurable Settings of PDUs (ConfigurationDesk Bus Manager Implementation Guide)
  - Bus Manager (stand-alone): Configurable Settings of PDUs (Bus Manager (Stand-Alone) Implementation Guide)

ISignal group

Symbol	▪ (TX) ▪ (RX)
Available in	▪ Bus Configurations table ▪ Buses Browser ▪ Properties Browser

This element indicates an ISignal group and its direction (TX, RX), and provides access to properties of the ISignal group (e.g., its position in a PDU).

ISignal

Symbol	▪ (TX) ▪ (RX)

Available in	▪ Bus Configurations table ▪ Bus Simulation Features table ▪ Bus Inspection Features table ▪ Bus Manipulation Features table ▪ Buses Browser ▪ Properties Browser

This element indicates an ISignal and its direction (TX, RX), and provides access to properties of the ISignal (e.g., the signal length). Some of these properties can be configured. For more information, refer to:

- Bus Manager in ConfigurationDesk: Configurable Settings of ISignals (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Configurable Settings of ISignals (Bus Manager (Stand-Alone) Implementation Guide)

Global time domain

Symbol	▪ ▸🕒 (TX) ▪ ◂🕒 (RX)
Available in	▪ Bus Configurations table ▪ Buses Browser ▪ Properties Browser

This element indicates a global time domain and its direction (TX, RX) , and provides access to properties of the global time domain (e.g., the domain ID). TX global time domains are available for time masters, RX global time domains are available for time slaves. For more information, refer to:

- Bus Manager in ConfigurationDesk: Aspects of Supported AUTOSAR Features (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Aspects of Supported AUTOSAR Features (Bus Manager (Stand-Alone) Implementation Guide)

Bus configuration feature

Symbol	🧩
Available in	▪ Bus Configurations table ▪ Properties Browser

This element indicates a bus configuration feature (e.g., the PDU Raw Data feature) that is added to an element of a bus configuration and provides access to properties of the feature. Some of these properties can be configured. For more information, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Configuration Features (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Basics on Bus Configuration Features (Bus Manager (Stand-Alone) Implementation Guide)

Feature switch

Symbol	
Available in	▪ Bus Configurations table ▪ Properties Browser

This element indicates a feature switch and provides access to properties of the feature switch. A feature switch is available for each ISignal if at least one bus manipulation feature is added to the ISignal. Some properties of the feature switch can be configured. For more information, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Manipulation Features (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Basics on Bus Manipulation Features (Bus Manager (Stand-Alone) Implementation Guide)

Communication connector

Symbol	▪ (CAN communication connector) ▪ (LIN master communication connector) ▪ (LIN slave communication connector)
Available in	Properties Browser

This element indicates a CAN, LIN master, or LIN slave communication connector that is defined in the communication matrix and provides access to the properties of the communication connector (e.g., the internal network address). You can access this element when you select an ECU in the Buses Browser, for example.

LIN schedule table

Symbol	
Available in	Properties Browser

This element indicates a LIN schedule table. If the element is available, you can access it when you select a LIN PDU in the Buses Browser, for example. For more information on LIN schedule tables, refer to:

- Bus Manager in ConfigurationDesk: Working with LIN Schedule Tables (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Working with LIN Schedule Tables (Bus Manager (Stand-Alone) Implementation Guide)

LIN schedule table entry

Symbol	
Available in	Properties Browser

This element indicates an entry of a LIN schedule table. If this element is available, you can access it when you select a LIN PDU in the Buses Browser, for example. For more information on LIN schedule tables, refer to:

- Bus Manager in ConfigurationDesk: Working with LIN Schedule Tables (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Working with LIN Schedule Tables (Bus Manager (Stand-Alone) Implementation Guide)

J1939 transport protocol node

Symbol	
Available in	Properties Browser

This element indicates a J1939 transport protocol node (J1939 TP node) and provides access to the properties of the node (e.g., the transport protocol address). For more information, refer to:

- Bus Manager in ConfigurationDesk: Aspects of Supported CAN Bus Features (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Aspects of Supported CAN Bus Features (Bus Manager (Stand-Alone) Implementation Guide)

J1939 network management node

Symbol	
Available in	Properties Browser

This element indicates a J1939 network management node (J1939 NM node) and provides access to the properties of the node (e.g., the industry group). For more information, refer to:

- Bus Manager in ConfigurationDesk: Aspects of Supported CAN Bus Features (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Aspects of Supported CAN Bus Features (Bus Manager (Stand-Alone) Implementation Guide)

J1939 transport protocol connection

Symbol	
Available in	Properties Browser

This element indicates a J1939 transport protocol connection and provides access to the properties of the J1939 transport protocol connection (e.g., the PGN). For more information, refer to:

- Bus Manager in ConfigurationDesk: Aspects of Supported CAN Bus Features (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Aspects of Supported CAN Bus Features (Bus Manager (Stand-Alone) Implementation Guide)

Time master

Symbol	M
Available in	Properties Browser

This element indicates a time master of a global time domain and provides access to the properties of the time master (e.g., the synchronization period). For more information, refer to:

- Bus Manager in ConfigurationDesk: Aspects of Supported AUTOSAR Features (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Aspects of Supported AUTOSAR Features (Bus Manager (Stand-Alone) Implementation Guide)

Time slave

Symbol	S
Available in	Properties Browser

This element indicates a time slave of a global time domain and provides access to the properties of the time slave (e.g., the follow-up timeout value). For more information, refer to:

- Bus Manager in ConfigurationDesk: Aspects of Supported AUTOSAR Features (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Aspects of Supported AUTOSAR Features (Bus Manager (Stand-Alone) Implementation Guide)

Frame

Symbol	[ ]
Available in	Properties Browser

This element indicates a frame and provides access to properties of the frame (e.g., the frame triggering). You can access this element when you select a PDU in the **Buses Browser**, for example. Some of the frame properties can be configured. For more information, refer to:

- Bus Manager in ConfigurationDesk: Configurable Settings of Frames (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Configurable Settings of Frames (Bus Manager (Stand-Alone) Implementation Guide)

Cyclic timing

Symbol	
Available in	Properties Browser

This element indicates a cyclic timing of a PDU and provides access to properties of the cyclic timing (e.g., the time offset). If this element is available, you can access it when you select a PDU in the **Buses Browser**, for example.

Some properties of the cyclic timing can be configured. For more information, refer to:

- Bus Manager in ConfigurationDesk: Configurable Settings of PDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Configurable Settings of PDUs (Bus Manager (Stand-Alone) Implementation Guide)

Event-controlled timing

Symbol	
Available in	Properties Browser

This element indicates an event-controlled timing of a PDU and provides access to properties of the event-controlled timing (e.g., the minimum delay). If this element is available, you can access it when you select a PDU in the Buses Browser, for example.

Properties of cyclic and event-controlled timings

Symbol	
Available in	Properties Browser

This element indicates properties of a cyclic or event-controlled timing of a PDU (e.g., the time period). If this element is available, you can access it when you select a PDU in the Buses Browser, for example. Some of the properties provide configurable properties themselves. For more information, refer to:

- Bus Manager in ConfigurationDesk: Configurable Settings of PDUs (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Configurable Settings of PDUs (Bus Manager (Stand-Alone) Implementation Guide)

Selector field

Symbol	
Available in	Properties Browser

This element indicates the selector field of a multiplexed IPDU and provides access to properties of the selector field (e.g., the endianness). You can access this element when you select a multiplexed IPDU in the Buses Browser, for example.

Dynamic part

Symbol	
Available in	Properties Browser

This element indicates the dynamic part of a multiplexed IPDU and provides access to properties of the dynamic part (e.g., the segment position). You can access this element when you select a multiplexed IPDU in the Buses Browser, for example.

Dynamic part alternative

Symbol	▣
Available in	Properties Browser

This element indicates the dynamic part alternatives of a multiplexed IPDU and provides access to properties of the related PDUs (e.g., the payload length). If this element is available, you can access it when you select a multiplexed IPDU in the Bus Configurations table, for example.

Static part

Symbol	▮
Available in	Properties Browser

This element indicates the static part of a multiplexed IPDU and provides access to properties of the static part (e.g., the segment position). If this element is available, you can access it when you select a multiplexed IPDU in the Buses Browser, for example.

Segment position

Symbol	⇤
Available in	Properties Browser

This element indicates the segment positions of the dynamic and static parts of a multiplexed IPDU and provides access to the properties of the segment positions (e.g., the start bit position). You can access this element when you select a multiplexed IPDU in the Buses Browser, for example.

Coded data type

Symbol	101 011
Available in	Properties Browser

This element indicates the coded data type of an ISignal and provides access to properties of the coded data type (e.g., the base data type). You can access this element when you select an ISignal in the Buses Browser, for example. Some of the coded data type properties can be configured.

- For more information on the coded data type, refer to:
  - Bus Manager in ConfigurationDesk: Signal Conversion by the Bus Manager (ConfigurationDesk Bus Manager Implementation Guide)
  - Bus Manager (stand-alone): Signal Conversion by the Bus Manager (Bus Manager (Stand-Alone) Implementation Guide)
- For more information on the configurable properties, refer to:
  - Bus Manager in ConfigurationDesk: Configurable Settings of ISignals (ConfigurationDesk Bus Manager Implementation Guide)
  - Bus Manager (stand-alone): Configurable Settings of ISignals (Bus Manager (Stand-Alone) Implementation Guide)

Physical data type

Symbol	■
Available in	Properties Browser

This element indicates the physical data type of an ISignal and provides access to properties of the physical data type (e.g., the base data type). You can access this element when you select an ISignal in the **Buses Browser**, for example. Some of the physical data type properties can be configured.

- For more information on the physical data type, refer to:
  - Bus Manager in ConfigurationDesk: Signal Conversion by the Bus Manager (ConfigurationDesk Bus Manager Implementation Guide)
  - Bus Manager (stand-alone): Signal Conversion by the Bus Manager (Bus Manager (Stand-Alone) Implementation Guide)
- For more information on the configurable properties, refer to:
  - Bus Manager in ConfigurationDesk: Configurable Settings of ISignals (ConfigurationDesk Bus Manager Implementation Guide)
  - Bus Manager (stand-alone): Configurable Settings of ISignals (Bus Manager (Stand-Alone) Implementation Guide)

Computation method

Symbol	ƒ
Available in	Properties Browser

This element indicates the computation method of an ISignal and provides access to properties of the computation method (e.g., the category). If this element is available, you can access it when you select an ISignal in the **Buses Browser**, for example. For more information, refer to:

- Bus Manager in ConfigurationDesk: Signal Conversion by the Bus Manager (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Signal Conversion by the Bus Manager (Bus Manager (Stand-Alone) Implementation Guide)

Computation scale

Symbol	p
Available in	Properties Browser

This element indicates a computation scale of an ISignal and provides access to properties of the computation scale (e.g., the scale type). If this element is available, you can access it when you select an ISignal in the **Buses Browser**, for example. Depending on the settings that are specified in the communication matrix, some computation scale properties can be configured.

- For more information on computation scales, refer to:
  - Bus Manager in ConfigurationDesk: Signal Conversion by the Bus Manager (ConfigurationDesk Bus Manager Implementation Guide)
  - Bus Manager (stand-alone): Signal Conversion by the Bus Manager (Bus Manager (Stand-Alone) Implementation Guide)

- For more information on the configurable properties, refer to:
  - Bus Manager in ConfigurationDesk: Configurable Settings of ISignals (ConfigurationDesk Bus Manager Implementation Guide 🕮)
  - Bus Manager (stand-alone): Configurable Settings of ISignals (Bus Manager (Stand-Alone) Implementation Guide 🕮)

End-to-end protection

Symbol	🛡
Available in	Properties Browser

This element indicates the end-to-end protection of an end-to-end protected ISignal ⧉ group and provides access to properties of the end-to-end protection (e.g., the category of the end-to-end protection profile). If this element is available, you can access it when you select an ISignal group in the Buses Browser, for example. For more information on end-to-end protection, refer to:

- Bus Manager in ConfigurationDesk: Aspects of Supported AUTOSAR Features (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Aspects of Supported AUTOSAR Features (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Position in ISignal IPDU

Symbol	▨
Available in	Properties Browser

This element indicates the position of an end-to-end protected ISignal ⧉ group in an ISignal IPDU and provides access to properties of the position (e.g., the data offset). If this element is available, you can access it when you select an ISignal group in the Buses Browser, for example. For more information on end-to-end protection, refer to:

- Bus Manager in ConfigurationDesk: Aspects of Supported AUTOSAR Features (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Aspects of Supported AUTOSAR Features (Bus Manager (Stand-Alone) Implementation Guide 🕮)

ISignal-to-IPDU mapping

Symbol	⏦
Available in	Properties Browser

This element indicates an ISignal-to-IPDU mapping of an ISignal ⧉ and provides access to the properties of the ISignal-to-IPDU mapping (e.g., the start bit position). You can access this element when you select an ISignal in the Buses Browser, for example.

Function block

Symbol	f
Available in	▪ Bus Access Requests table

	- Properties Browser

This element indicates a function block, e.g, a CAN function block that specifies the bus access for a bus access request.

Function group

Symbol	fx
Available in	Properties Browser

This element indicates a function group, e.g, of PDUs. You can access this element when you select a function port in the Bus Configurations table, for example.

Function

Symbol	fx
Available in	Properties Browser

This element indicates a function. You can access this element when you select a function port in the Bus Configurations table, for example.

Function port group

Symbol	fx
Available in	Properties Browser

This element indicates a group of function ports. You can access this element when you select function port in the Bus Configurations table, for example.

Function inport

Symbol	fx
Available in	- Bus Configurations table - Bus Configuration Ports table - Properties Browser

This element indicates a function inport, e.g., a TX ISignal Value function port or LIN Schedule Index function port.

Function outport

Symbol	fx
Available in	- Bus Configurations table - Bus Configuration Ports table - Properties Browser

This element indicates a function outport, e.g., an RX ISignal Value function port or RX PDU Raw Data function port.

Event port

Symbol	▶
Available in	▪ Bus Configurations table ▪ Bus Configuration Ports table ▪ Properties Browser

This element indicates an event port, e.g., a PDU Received event port.

Related topics

Basics

Basics on the Bus Manager (ConfigurationDesk Bus Manager Implementation Guide)
Basics on the Bus Manager (Bus Manager (Stand-Alone) Implementation Guide)

References

Bus Access Requests Table........391
Bus Configurations Table........394
Bus Inspection Features Table........396
Bus Manipulation Features Table........401
Bus Simulation Features Table........415
Buses Browser........428

Filter for Changes / Filter for Changes to Communication Matrices

Purpose

To display only modified communication matrix elements, i.e., elements that were added by the user or contain user-defined settings.

Result

Only communication matrix elements that were added by the user or contain user-defined settings and their related higher-level elements are displayed.

Description

The active filter is selected on the Home ribbon and indicated by a checkmark in context menus.

For basic information on modifying communication matrices, refer to:

- Bus Manager in ConfigurationDesk: Modifying Communication Matrices (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Modifying Communication Matrices (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

Basics

Using Display Filters (ConfigurationDesk Real-Time Implementation Guide)

References

Add Element to Communication Matrix..350
Undo All Changes..452
Undo Changes to Communication Matrix..453

Generate

Purpose

To generate bus simulation container files (BSC files).

Result

The generation of bus simulation container files (BSC files) is started. Messages in the **Message Viewer** indicate the progress of the generation process in chronological order.

For each application process that fulfills the following conditions, one BSC file is generated:

- The application process contains no or exactly one model implementation, which is a Simulink implementation container (SIC file).
- The application process contains at least one bus configuration for which no conflicts of the **Generate no code** and **Abort BSC file generation** categories exist.

If not available yet, a **Generated Containers** node is added to the active ConfigurationDesk application in the **Project Manager**. The generated BSC files are displayed below this node.

Note

During the generation of bus simulation containers, existing BSC files are deleted from the **Generated Containers** folder without notice. If you want to keep these BSC files, you must store them in another folder before you start the generation process.

Description

Bus simulation containers contain the bus communication that was configured by using the Bus Manager. For example, you can use BSC files in VEOS Player or ConfigurationDesk to implement the configured bus communication in offline simulation applications or real-time applications, respectively.

For more information, refer to:

- Bus Manager in ConfigurationDesk: Generating Bus Simulation Containers (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Generating Bus Simulation Containers (Bus Manager (Stand-Alone) Implementation Guide)

New - Multiple Bus Configurations

Purpose

To add multiple bus configurations to a ConfigurationDesk application.

Result

A dialog opens that lets you enter the number of bus configurations you want to add.

Description

The following bus configuration elements are added to the application:

- New bus configuration nodes and bus configuration part nodes (e.g., **Simulated ECUs** node, **Inspection** node) in the **Bus Configurations** and **Bus Access Requests** tables
- New **Bus Configuration** function blocks in the **Function Browser** and in the global working view

To implement bus communication in the active ConfigurationDesk application and configure it for simulation purposes, you must assign communication matrix elements to the bus configurations.

For more information on working with bus configurations, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Configurations (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Basics on Bus Configurations (Bus Manager (Stand-Alone) Implementation Guide)

Bus Configurations dialog

Name pattern Lets you define the name pattern for the new bus configurations. You can use the backslash \ as a placeholder to add an incremented number to the name of each new bus configuration.

Number of instances Lets you define the number of bus configurations to add to the application.

Related topics

References

Elements and Symbols of Bus Configurations and Communication Matrices.......................... 429

New / New - Bus Configuration

Purpose

To add a new bus configuration to a ConfigurationDesk application.

Result

The following bus configuration elements are added to the application:

- A new bus configuration node and bus configuration part nodes (e.g., Simulated ECUs node, Inspection node) in the Bus Configurations and Bus Access Requests tables
- An instantiated Bus Configuration function block in the **Function Browser** and in the Global working view

Description

The new bus configuration has a unique default name. You can rename the bus configuration (e.g., in the Bus Configurations table).

To configure bus communication (e.g., for simulation purposes), you must assign communication matrix elements to the bus configuration. The bus configuration implements the bus communication in the signal chain and lets you configure it.

For more information on working with bus configurations, refer to:

- Bus Manager in ConfigurationDesk: Basics on Bus Configurations (ConfigurationDesk Bus Manager Implementation Guide 🕮)
- Bus Manager (stand-alone): Basics on Bus Configurations (Bus Manager (Stand-Alone) Implementation Guide 🕮)

Related topics

References

Select Bus Configuration

Purpose

To select the related bus configurations of the selected Bus Configuration function blocks in the Bus Configurations table.

Result

If the Bus Configurations table is not already open, it opens in the Buses view set. The related bus configurations of the selected Bus Configuration function blocks are highlighted in the table.

Select Function Block

Purpose

To select the related Bus Configuration function blocks of the selected bus configurations.

Result

The related Bus Configuration function blocks are highlighted in the Signal Chain Browser and in the Function Browser. If the Function Browser is not already open, it opens in the Signal Chain view set.

Select Related Elements in Bus Configurations Table

Purpose

To select elements related to the selected elements in the Bus Configurations table.

Result

Elements related to the selected elements are selected in the Bus Configurations table.

Description

Which elements are selected in the Bus Configurations table depends on the selected elements, for example:

- If you select an ISignal in the **Buses Browser**, this command selects all related ISignal instances in all the bus configurations of the active ConfigurationDesk application.
- If you select a frame triggering in the **Conflicts Viewer**, this command selects all PDUs that have this frame triggering assigned in all the bus configurations of the active ConfigurationDesk application.

Select Related Elements in Communication Matrices by Clusters View

Purpose

To select elements related to the selected elements in the Communication Matrices by Clusters view of the **Buses Browser**.

Result

Elements related to the selected elements are selected in the Communication Matrices by Clusters view of the Buses Browser.

Description

Which elements are selected in the Communication Matrices by Clusters view depends on the selected elements, for example:

- If you select an RX instance of an ISignal in the Bus Configurations table, this command selects all related TX and RX ISignals of all network nodes in all communication clusters.
- If you select an ECU in the Bus Configurations table, this command selects all related network nodes in all communication clusters.

Select Related Elements in Communication Matrices by ECUs View

Purpose

To select elements related to the selected elements in the Communication Matrices by ECUs view of the **Buses Browser**.

Result

Elements related to the selected elements are selected in the Communication Matrices by ECUs view of the Buses Browser.

Description

Which elements are selected in the Communication Matrices by ECUs view depends on the selected elements, for example:

- If you select an RX instance of an ISignal in the context of one communication cluster in the Bus Configurations table, this command selects all related TX and RX ISignals of all ECUs, regardless of the communication clusters.
- If you select a PDU that is included in a container IPDU in the Bus Configurations table, this command selects all related PDUs in all container IPDUs in all ECUs.

Undo All Changes

Purpose

To undo all the changes of all the communication matrices contained in the active ConfigurationDesk application.

Result

The user-defined settings of all communication matrix elements are reset to their original values. If elements were added by the user, they are removed from the application.

Description

This command affects all the communication matrix elements and their instances, i.e., the elements that are assigned to bus configurations.

For basic information on modifying communication matrices, refer to:

- Bus Manager in ConfigurationDesk: Modifying Communication Matrices (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Modifying Communication Matrices (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Add Element to Communication Matrix........350
Undo Changes to Communication Matrix........453

Undo Changes to Communication Matrix

Purpose

To undo changes of the selected communication matrix element and all its instances.

Result

The user-defined settings of the selected element and all its instances (i.e., the elements that are assigned to bus configurations) are reset to their original values. If the element was added by the user, the element and its related lower-level elements are removed from the active ConfigurationDesk application.

Description

This command affects the selected element in the communication matrix and all the element instances that are assigned to bus configurations of the active ConfigurationDesk application.

For basic information on modifying communication matrices, refer to:

- Bus Manager in ConfigurationDesk: Modifying Communication Matrices (ConfigurationDesk Bus Manager Implementation Guide)
- Bus Manager (stand-alone): Modifying Communication Matrices (Bus Manager (Stand-Alone) Implementation Guide)

Related topics

References

Add Element to Communication Matrix........350
Undo All Changes........452

Automation

Objective

ConfigurationDesk provides the following commands and dialogs for the automation of ConfigurationDesk features.

Where to go from here

Information in this section

Information in other sections

ConfigurationDesk Automating Tool Handling
Gives you detailed information on ConfigurationDesk's automation interface.

Activate (Script)

Purpose

To display the Source Code Editor containing the selected Python script.

Result

The Source Code Editor containing the selected Python script is displayed in the foreground.

Advanced (Automation)

Purpose

To provide access to advanced Source Code Editor commands.

Description

You have access to the following commands:

- Select All on page 474
- Comment out Region on page 460
- Uncomment Region on page 477
- Tabify Region / Tabify Selection on page 475
- Untabify Region / Untabify Selection on page 478
- Tabify Leading Spaces on page 475
- Toggle Fold on page 477
- Toggle all Folds on page 476
- Toggle Bookmark on page 476
- Next Bookmark on page 470
- Previous Bookmark on page 470
- Clear Bookmarks on page 459
- To Uppercase on page 476
- To Lowercase on page 475
- Convert End-of-Line Characters on page 461

Check Syntax

Purpose

To check the syntax of the code in the selected Python script.

Result

If the syntax of your Python code is correct, a dialog informs you that the syntax check finished successfully. Otherwise a dialog shows you the syntax errors in the code.

Description

To avoid error messages concerning syntax errors in your Python scripts when executing, you should check the syntax beforehand directly after editing in the Interpreter or Source Code Editor.

Clear Bookmarks

Purpose

To remove all of the bookmarks in the source code file shown in the Source Code Editor.

Result

The Source Code Editor removes all of the bookmarks in the source code file that you toggled via Toggle Bookmark (see Toggle Bookmark on page 476).

Related topics

HowTos

How to Use Bookmarks in the Source Code Editor (ConfigurationDesk Automating Tool Handling)

References

Toggle Bookmark..476

Clear Namespace

Purpose

To clear all user-defined variables and all the imported modules from the interpreter's namespace since the last application start.

Result

- All the variables that were defined in the Interpreter namespace since the last start of the application are deleted.
- All the imported Python modules are unloaded.

> **Note**
>
> Imported PYD modules, i.e., Python modules programmed in C/C++, and modules used by the Interpreter itself, are *not* unloaded.

Description

You can define variables and import modules when using the Interpreter. To reset the Interpreter to its initial state, use the **Clear Namespace** command. Clearing the namespace is useful if a script error occurred due to user-defined variables or imported modules.

Clear Window

Purpose

To clear the current contents in the **Interpreter**.

Result

The contents of the Interpreter are cleared. The command prompt in the first line remains.

> **Tip**
>
> The commands entered in the Interpreter remain valid and are accessible via the command history.

Close (Script)

Purpose

To close a Python script file that is currently open in the Source Code Editor.

Comment out Region

Purpose

To insert double comment character(s) at the beginning of all the lines selected in the source code file displayed in the Source Code Editor.

> **Note**
>
> A double comment character (##) is inserted to distinguish between a user comment and a comment-out region.

Result

The Source Code Editor places comment character(s) at the beginning of the lines you selected in the source code file.

Convert End-of-Line Characters

Purpose	To convert all end-of-line characters according to operating system conventions.
Description	Operating systems have different conventions for end-of-line characters (CRLF, LF, or CR). The Source Code Editor converts the existing end-of-line characters according to the **End-of-line mode** setting in the **ConfigurationDesk Options** dialog (see Editor General Page on page 482). To display the end-of-line characters in the Source Code Editor, activate **Show end-of-line markers** on the same dialog page.

Export Script

Purpose	To export a Python script.
Result	Opens a standard file dialog to specify path and file name.

Find (Interpreter)

Purpose	To find text in the Interpreter or Source Code Editor.
Result	ConfigurationDesk opens the **Find** dialog for you to locate the text in the source code.
Find dialog	**Find what** Lets you specify the text you wish to find in the code. **Match whole word only** Lets you search only for text that is separated from the surrounding text by whitespace or tabs. **Match case** Lets you search only for expressions which have the same case as the text entered in **Find what**.

Use regular expressions Lets you use regular expressions for your search:

Regular Expression	Purpose	Example
`.`	Matches any single character.	`ho.se` matches horse and house
`\<`	Matches the start of a word.	`\<art` matches artist but not start.
`\>`	Matches the end of a word.	`\>art` matches start but not artist.
`[...]`	Matches a single character that is contained within the brackets. For example, `[abc]` matches `"a"`, `"b"`, or `"c"`. `[a-z]` specifies a range which matches any lowercase letter from `"a"` to `"z"`.	`a[ur]t` matches auto or artist but not alternate
`[^...]`	Matches a single character that is not contained within the brackets. For example, `[^abc]` matches any character other than `"a"`, `"b"`, or `"c"`. `[^a-z]` matches any single character that is not a lowercase letter from `"a"` to `"z"`.	`a[^ur]t` matches alternative but not automobile or artist
`^`	Matches the start of a line.	`^dig` matches digit at the start of a line.
`$`	Matches the end of a line.	`git$` matches digit at the end of a line.
`*`	Matches the preceding character zero or more times.	`di*git` matches dgit, digit, diigit, diiigit, etc.
`+`	Matches the preceding character one or more times.	`di+git` matches digit, diigit, diiigit, etc.
`\x`	Allows you to use a character x that would otherwise have a special meaning.	`2\+2<5` matches 2+2<5

Direction Lets you select Up or Down as the search direction.

Find Next Starts or continues the search.

Go to Line

Purpose To go to any line in the source code shown in the Source Code Editor.

Result The cursor goes to the line number you selected in the dialog.

Import Module

Purpose

To add variable and function definitions of the selected Python module to the Interpreter namespace.

Result

The variables and function definitions in the specified module are added to the namespace of the Interpreter.

If the script is not a PYD module (a module programmed in C/C++), the script variables are reloaded to the Interpreter namespace.

Unlike the `import <module_name>` Python command, the **Import Module** command overwrites the module if it was imported before, i.e., you do not need to clear the Interpreter namespace to reload a module.

Dialog settings

File name Enter the file name of the Python script you want to import or select the file from the respective folder.

Related topics

HowTos

How to Import a Python Module to the Interpreter Namespace (ConfigurationDesk Automating Tool Handling 🕮)

References

Clear Namespace..459

Import Script

Purpose

To import a Python script to the active project.

Result

Opens the **Import Python Script** dialog for you to select a Python script to import.

Dialog settings

File name Enter the file name of the Python script you want to import.

Insert Script

Purpose

To insert a new empty Python script into the active project.

Result

The **Insert Python Script** dialog is opened for you to open a new empty Python script in the working area. The new Python script is inserted into the active project.

If you want to insert a "free floating" Python script into a project, select **File - Import - Python Script**. For more information, refer to Import Script on page 463.

Insert Python Script dialog

Name of new Python script Enter the file name of the Python script you want to insert.

Interpreter

Purpose

To show or hide ConfigurationDesk's **Interpreter**.

Description

ConfigurationDesk provides the **Interpreter** for you to run Python scripts and execute line-based commands.

Editing assistance ConfigurationDesk's **Interpreter** provides editing assistance that helps you complete Python variables, functions, and object attributes automatically.

The editing assistance feature in ConfigurationDesk's **Interpreter** comprises the following aspects:

- Auto completion
- Tooltip
- Member completion

The following table compares the aspects:

Aspect	**Action**	**Meaning**	**Available**
Auto completion	Enter `Ctrl+SPACE` after the beginning letter of a string	Lists all the possible names for you to choose from to complete the string.	Can be enabled or disabled
Tooltip	Enter "(" after a method	Shows a tooltip for the method.	Can be enabled or disabled
Member completion	Enter "." after an object	List the members of the object for you to choose from.	Always available

Note

The auto completion feature does not complete automatically or show you a list to choose from. Python is a dynamic language. The character of an object can only be read after the object has been instantiated. No editing assistance is shown if the string describing the object contains parenthesis.

Syntax highlighting The Interpreter features syntax highlighting for standard input. You can specify different colors and fonts for the syntax tokens on the Syntax Highlighting Page on page 486. This feature makes it easy to distinguish between the syntax tokens in the command lines.

Auto indentation In Python, multi-line commands are introduced by a colon, for example, the command `def f():` starts a command block that defines a function, and their scopes are declared by indentation. Each command line that is typed in is checked by the Interpreter. At the start of a multi-line command, the Interpreter automatically indents the next line. The indent depth of the line depends on the number of control structures or command blocks started. In addition, you can start a multi-line command by pressing the **Shift+ENTER** keys. The indent depth is decreased by entering an empty line or pressing the **BACKSPACE** key. For example, you have to do this to finish an `If` or `Else` branch.

Notation in the Interpreter controlbar

The Interpreter's ">>>" command prompt allows you to enter a command. A command is interpreted when you press the **Enter** key.

When you enter a multi-line command such as an If-clause or a function definition, the command prompt changes to "...".

Prompt	Meaning
>>>	Input line, start of command
...	Continued multi-line command
(Blanks)	Output text

Context menu of the Interpreter control bar

The Interpreter controlbar's context menu provides the following commands.

Command	Purpose
Undo	To reverse the last edit action you made in the Interpreter.
Redo	To reverse the latest undo action in the Interpreter.
Cut	To cut the selected text and add it to the Clipboard.
Copy	To copy the selected text and add it to the Clipboard.
Paste	To paste the current content from the Clipboard to the Interpreter.
Delete	To delete the selected text.
Select all	To select all the text in the Interpreter.

Command	Purpose
View Whitespace	To show or hide the whitespace marks in the Interpreter.
Find	To find text in the Interpreter.
Run script	To run a Python script.
Import script	To insert variable and function definitions of Python scripts or modules in the Interpreter's namespace.
Clear Window	To clear the current contents of the Interpreter.
Clear Namespace	To clear all user-defined variables since the last start of ConfigurationDesk and all imported modules.

Shortcut keys of the Interpreter

The following table shows the shortcuts keys that can be used in the Interpreter control bar:

Shortcut Keys	Purpose
`Ctrl+F`	To open the Find dialog.
`Ctrl+W`	To enable/disable the whitespace view in the Interpreter.
`Ctrl+A`	To select all the current text in the Interpreter. This command is also available as "Select all" in the context menu of the Interpreter.
`Ctrl+Z`	To reverse the last action made in the Interpreter. This command is also available as "Undo" in the context menu of the Interpreter.
`Ctrl+Y`	To reverse the latest undo action in the Interpreter. This command is also available as "Redo" in the context menu of the Interpreter.
`Ctrl+`"Numpad +"	To zoom into the Interpreter control bar.
`Ctrl+`"Numpad -"	To zoom out of the Interpreter control bar.
`Ctrl+` Mouse wheel	To zoom into or out of the Interpreter.
`Backspace`	To decrease the indent depth in multi-line commands or delete the last character.
`Ctrl+Down`	To go one command down in the command stack.
`Ctrl+Up`	To go one command up in the command stack.
`Ctrl+End`	To go to the last command that was typed in at the command prompt.
`Ctrl+Home`	To go to the first command of the command stack.
`Ctrl+Space`	To enter the auto-completion mode for Python variables.
`Ctrl+X`	To cut the selected text to the Clipboard. This command is also available as "Cut" in the context menu of the Interpreter.

Shortcut Keys	Purpose
`Ctrl+C`	To copy the selected text to the Clipboard. This command is also available as "Copy" in the context menu of the Interpreter.
`Ctrl+V`	To paste text from the Clipboard to the current cursor position. This command is also available as "Paste" in the context menu of the Interpreter.
`Ctrl+I`	To open the Import Script dialog. This command is also available as "Import Script" on the Automation ribbon..
`Ctrl+R`	To open the Run Script dialog. This command is also available as "Run script" in the context menu of the Interpreter and on th ribbon.
`Esc`	To delete the current command in the line if the command is available.
`Shift+Enter`	To start a multi-line command.
`Tab`	To increase the indent depth in multi-line commands.

Error tracing

The preferred error handling in Python is exception handling. When an exception occurs, the Interpreter displays a traceback.

Typical errors To correct errors that have occurred during interactive command input, examine the printed traceback. Otherwise, edit the Python source file. These are some typical errors:

```
>>>dir("1", "1")
   Traceback (innermost last):
     File "<interactive input>", line 0, in ?
   TypeError: dir requires at most 1 argument; 2 given
>>>if () && 1:
     File "<string>", line 1
        if () && 1:
               ^
   SyntaxError: invalid syntax
>>>c
   Traceback (innermost last):
     File "<interactive input>", line 0, in ?
   NameError: c
```

Note

If a file path is available, you can double-click it above the error description. ConfigurationDesk opens the file in the Source Code Editor and scrolls to the error position.

Predefined exceptions The following table is an extract from the predefined exceptions (refer to *Built-in Exceptions* in the *Python Library Reference*):

Exception Type	Additional Information	Cause
NameError	Name of the variable	Local or global name not found
SyntaxError	Invalid syntax and an extra line preceded by a "^", which points to the character where the syntax error starts.	Invalid syntax
TypeError	Depends on the specific error.	Built-in operation or function was applied to an object of an inappropriate type.
AttributeError	Depends on the specific error.	Object or variable has no member with the given name.
KeyboardInterrupt	–	The script was interrupted by using the system tray menu.
IOError	Depends on the I/O error.	A file system operation failed: for example, writing a file.
IndexError	Index out of range	Access to an index which is greater than the size of a list or tuple.
KeyError	Wrong key name	Key was not found in a dictionary's set.
ZeroDivisionError, OverflowError, FloatingPointError	–	Mathematical errors

Traceback information The traceback information always has the same structure:

```
Traceback (innermost last):
    File "<filename>", line <line number>, in <function name>
    <command>
<additional line when a syntax error has occurred>
<exception type>: <additional information>
```

`<filename>`	The file name is given whenever possible.
`<line number>`	The line number specifies either the script line in which the error occurred, or the line of the calling function.
`<function name>`	If a function name is given, it relates to the function where the error occurred or to the function that called the incorrect function.
`<command>`	If a command is given, it is the command belonging to the line number specified above.

Command line arguments

You can open, import, and run Python scripts in ConfigurationDesk via command line:

Argument	Description	Example
filename	Opens the Python script in the Source Code Editor. If the path or filename includes spaces it must be enclosed by quotation marks.	`ConfigurationDesk "c:\temp\Frame Access.py"`
/i filename[1), 2)]	Imports the Python script to the namespace of the Interpreter.	`ConfigurationDesk /i "c:\temp\Frame Access.py"`
/r filename [;workingdir [;arguments]][1), 2)]	Runs the Python script in the Interpreter. You can optionally add a working directory and/or further arguments. If the argument contains a semicolon, it must be preceded by a backslash: \;.	▪ `ConfigurationDesk /r "c:\temp\Frame Access.py"` ▪ `ConfigurationDesk /r "c:\temp\Frame Access.py;C:\temp;-a Argument"` ▪ `ConfigurationDesk /r "c:\temp\Frame Access.py;;-a Argument"`

[1)] If the argument includes spaces it must be enclosed by quotation marks.

[2)] Instead of a slash you can also use a minus to prefix the argument, for example -i, or -r.

Related topics

Basics

Basics on the Interpreter (ConfigurationDesk Automating Tool Handling 📖)

New (Script File)

Purpose

To create a new Python script.

Result

A new empty Python script is opened in the working area.

Description

The new script is not inserted into the active project.

- To create a new Python script that is inserted into the active project, refer to Insert Script on page 464.
- To insert a "free" Python script into a project, refer to Import Script on page 463.

Related topics

References

Import Script..463
Insert Script..464

Next Bookmark

Purpose

To move the cursor to the next bookmark in the source code file shown in the Source Code Editor.

Result

The Source Code Editor moves the cursor and scrolls the view to the next bookmark in the source code file. For information on how to toggle bookmarks, refer to Toggle Bookmark on page 476.

Related topics

References

Toggle Bookmark..476

Open File or Project

Purpose

To open a Python script in the Source Code Editor, or to open a ConfigurationDesk project (CDP) file.

Description

The path to the opened project file is not added to the list of project root directories.

Previous Bookmark

Purpose

To move the cursor to the previous bookmark in the source code file shown in the Source Code Editor.

Result

The Source Code Editor moves the cursor and scrolls the view to the previous bookmark in the source code file. This bookmark was toggled via **Toggle Bookmark**.

Related topics

References

Toggle Bookmark..476

Print (Source Code Editor)

Purpose

To print the active Python script.

Result

This invokes standard printing. The specified file is printed according to your specifications.

You can configure and prepare the print process using the **Print Preview**, **Print Setup**, and **Page Setup** commands.

Rename (Python Script)

Purpose

To rename a Python script.

Result

Opens the **Rename Python Script** dialog where you can rename the Python script.

Rename Python Script dialog

New Python script name Lets you enter a new name for the selected Python script. The name has to apply with Windows file name conventions.

Replace

Purpose

To quickly replace an expression in the Source Code Editor.

Result

The expression is replaced with the text you specified.

Description

The **Replace** command is available only for the Source Code Editor.

Find and Replace dialog

Lets you specify search criteria.

Find what Lets you specify the text you wish to find in the source code.

Replace with Lets you specify the expression to replace the text you are searching for.

Match whole word only Indicates whether the text entered above is found only when it is separated from the surrounding text by white space or tabs.

Match case Indicates whether only expressions which have the same case as the text entered in Find what are found.

Use regular Expressions Indicates whether the specified string is a regular expression which the text must match.

In the search string,

- . matches any single character.
- * indicates there are 0, 1 or any number of the previous expression or character.

Find Next Starts or continues the search.

Replace Replaces the found text and then searches for the next occurrence of the text.

Replace All Replaces all occurrences of the find string in the document.

Run

Purpose

To run the selected Python script in the Interpreter.

Run on Load

Purpose

To run the selected Python script in the Interpreter when the project is opened.

Run Script

Purpose

To run a Python script.

Result

The selected Python script is executed in the namespace of the Interpreter. Standard outputs and error outputs are redirected to the Interpreter window.

Dialog settings

Look in Lets you specify the directory where the Python script to be run is stored.

File name Lets you specify the name of the Python script.

Files of type Lets you specify the file type. The selected file type must be Python Files.

Arguments If the Python script needs any command line arguments, you have to enter them here. The list of arguments to be entered depends on the selected script.

Working directory Lets you specify the working directory from which you want to run the Python script. You can use the Browse button to open a standard dialog to select a folder.

Open file in editor Lets you specify to open the selected file in the Source Code Editor when you start running it.

Related topics

HowTos

How to Run Scripts (ConfigurationDesk Automating Tool Handling 📖)

Save (Script File)

Purpose

To save a Python script.

Result

ConfigurationDesk saves the Python script. If the file has not been saved before (or if it is read-only), ConfigurationDesk opens a standard dialog to save the file.

Save As (Script File)

Purpose

To save a file under a new name.

Result

Opens a standard dialog to save the file under a new name.

Script File

Purpose

To create or save a python script.

Description

You have access to the following commands:

- New (Script File) on page 469
- Save (Script File) on page 473
- Save As (Script File) on page 473

Select All

Purpose

To select all the entries in the Interpreter or the Source Code Editor.

Source Code Editor

Description

ConfigurationDesk provides a Source Code Editor for you to open and edit Python scripts.

The Source Code Editor is started if you open or create a Python file inside ConfigurationDesk.

For more information, refer to Using the Source Code Editor (ConfigurationDesk Automating Tool Handling 🕮).

> **Tip**
>
> For executing line-based Python commands interactively, ConfigurationDesk provides the Interpreter. Refer to Interpreter Settings Page on page 484.

Related topics

References

Open File or Project..470

Tabify Leading Spaces

Purpose

To replace the leading whitespaces in the selected source code with a tabulator.

Result

The Source Code Editor replaces the leading whitespaces in the selected text with tabulators according to the number of whitespaces that was set as tabulator width in the **Editor General** page of the **ConfigurationDesk Options** dialog.

Related topics

References

Editor General Page....482

Tabify Region / Tabify Selection

Purpose

To replace the whitespaces in the selected source code with tabulators.

Result

The Source Code Editor replaces the whitespaces in the selected text with tabulators according to the number of whitespaces that was set as tabulator width in the **Editor General** page of the **ConfigurationDesk Options** dialog.

Related topics

References

Editor General Page....482
Untabify Region / Untabify Selection....478

To Lowercase

Purpose

To convert all the characters in the selection to lower case.

Related topics

References

To Uppercase....476

To Uppercase

Purpose

To convert all the characters inside the selection to upper case.

Related topics

References

To Lowercase..475

Toggle all Folds

Purpose

To fold/unfold all the source code parts.

Description

The folding of source code parts must be activated in the ConfigurationDesk Options dialog (see Editor General Page on page 482).

Result

Folds or unfolds all the source code parts in the script. A checkmark is displayed next to the command if all source code parts are folded.

Related topics

References

Toggle Fold...477

Toggle Bookmark

Purpose

To add/remove a bookmark to/from the line where the cursor is currently located.

Result

When you add a bookmark with this feature, it appears as a blue square in the left margin of the line where it was placed. If no margin is specified, the bookmark is indicated by a blue bar highlighting the whole line of text. When you move the cursor to a line which already contains a bookmark and activate the Toggle Bookmark command again, the bookmark is removed.

Related topics

HowTos

How to Use Bookmarks in the Source Code Editor (ConfigurationDesk Automating Tool Handling 🕮)

References

Next Bookmark....................470
Previous Bookmark....................470

Toggle Fold

Purpose

To fold/unfold a selected source code part.

Description

The folding of source code parts must be activated in the ConfigurationDesk Options dialog (see Editor General Page on page 482).

Result

Folds or unfolds the source code part the text cursor is in. A checkmark is displayed next to the command if a source code part is folded.

Related topics

References

Toggle all Folds....................476

Uncomment Region

Purpose

To delete the comment character(s) you placed at the beginning of the line(s) selected in the source code file shown in the Source Code Editor.

Result

The leading comment characters of the selected source code lines are deleted. Further comment characters in the same line remain unchanged.

Untabify Region / Untabify Selection

Purpose	To replace the tabulators in the selected source code with whitespaces.
Result	All leading tabulators in the selection are changed to a series of spaces. The number of spaces per tabulator depends on the tabulator width setting in the Editor General page of the ConfigurationDesk Options dialog (see Editor General Page on page 482).

View Whitespace

Purpose	To toggle the display of whitespaces and tabulators in the source code.
Result	If you selected the View Whitespace option, the Source Code Editor or Interpreter displays the symbols used for spaces in the source code so you can see whether whitespaces or tabulators are used.

Options

Objective

In the **ConfigurationDesk Options** dialog you can specify the global settings of ConfigurationDesk.

Where to go from here

Information in this section

Configuration Page

Purpose

To specify a global search path for static custom functions and to specify certain default settings for ConfigurationDesk applications.

Custom function directory

Displays the global search path in which ConfigurationDesk searches for custom functions. Click the Browse button to select a different path.

Description To use the same static custom functions in different projects, you can specify a global search path outside the project root folder. This way you can apply changes globally. Custom function files can be copied to the specified directory, and the custom functions are available in the **Function Browser** after using the Reload Custom Function Definitions command or the next time you open a ConfigurationDesk project.

Note

- You must copy custom function files directly to the specified directory. Files from subdirectories are ignored.
- ConfigurationDesk lets you import custom function blocks from ZIP files via the Import Custom Function from Archive command.
- If ConfigurationDesk finds two custom function XML files in which the same custom function block type name is specified in the project-specific directory (`<DocumentsFolder>\<Project>\CustomFunctions`) and the global search path, only the custom function from the project-specific directory is available in the project.
- Unlike custom function files from project-specific custom functions directories, custom function files from the global search path are not included in a project backup. You must provide them separately, for example to use them on a different system.

Build and download default

Lets you specify the default behavior for building and downloading real-time applications. The settings apply to all newly added ConfigurationDesk applications. If you change the specified settings, existing ConfigurationDesk applications are not affected by the new settings. Instead, you can specify the build and download behavior individually for each ConfigurationDesk application via the Build Configuration table.

Download real-time application after build If the state of the ConfigurationDesk application is Matching platform connected (displayed in the status bar), the download starts immediately after the build process is completed. If Start real-time application is not selected, the downloaded real-time application's state is Stopped.

Unload a loaded application Unloads a loaded application from the connected platform and loads the new application. If the option is not set, the download is stopped if another application is loaded and running on the platform.

Start real-time application Automatically starts the real-time application when the download is completed.

Signal Chain Browser default

Show External Devices column Lets you specify the default setting for the Show External Devices Column on page 247 command in working views. If the checkbox is cleared, by default the External Devices column is hidden in working views that you open in the Signal Chain Browser.

Application process optimization default

Optimize configuration automatically Lets you specify the default setting for the Optimize configuration automatically property of application processes. If the checkbox is selected, the Optimize configuration automatically checkbox is selected for newly created application processes by default.

Related topics

Basics

ConfigurationDesk Custom I/O Function Implementation Guide
Customizing the Display of the Working View Columns (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Build Configuration Table..323
Import Custom Function from Archive..155
Reload Custom Function Definitions..158
Reload Definition for '<CustomFunctionType(s)>'..159
Show External Devices Column..247

Create Mapping Lines Page

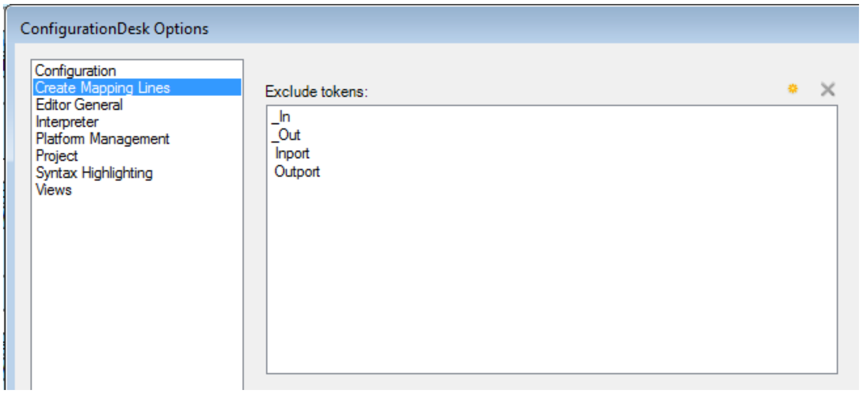
Purpose

To add or remove exclude tokens that are to be ignored by the best match principle for the creation of mapping lines by name.

Exclude tokens

Exclude tokens are strings that are ignored when ConfigurationDesk tries to determine a best match after executing the Create Mapping Lines by Name on page 218 command.

A number of strings are already excluded by default:



ConfigurationDesk Options

Configuration
Create Mapping Lines
Editor General
Interpreter
Platform Management
Project
Syntax Highlighting
Views

Exclude tokens:

_In
_Out
Inport
Outport

Use the ✹ and ✕ buttons to add or remove tokens.

Related topics

References

Create Mapping Lines by Name 218

Editor General Page

Purpose

To specify settings for the Source Code Editor and the Interpreter.

Editor General page

Autocompletion Lets you enable/disable ConfigurationDesk's auto-completion function.

Show tooltips Lets you enable/disable the display of tooltips.

Show function browser Lets you enable/disable the display of a drop-down list at the top of the editor. You can jump to a function in the code via this list.

Highlight brace matching Lets you enable/disable the highlighting of matching braces.

Spaces for tabs Lets you enable/disable tabs-to-spaces conversion.

Tab width Lets you define the tab width.

End-of-line mode Lets you select one of the following end-of-line modes:

Mode	Description
CRLF	Carriage return - line feed, used by MS-DOS and Windows operating systems
LF	Line feed, used by Unix operating systems
CR	Carriage return, used by Macintosh operating systems

The mode is only used for new end-of-line characters that are typed in. Existing end-of-line characters are not converted.

Show line numbers Lets you specify whether to display line numbers.

Show whitespaces Lets you specify whether to display symbols for whitespaces and tabulators.

Show end-of-line markers Lets you specify whether to display end-of-line markers.

Show indentation guides Lets you specify whether to display indentation lines.

Vertical scroll line visible Lets you specify whether to display a vertical scroll line.

Horizontal scroll line visible Lets you specify whether to display a horizontal scroll line.

Line wrapping Lets you enable/disable line wrapping.

Line wrapping - Mode Lets you select one of the following modes for line wrapping:

Mode	Description
Char	The line is wrapped after the last visible character in the current view.
Word	The line is wrapped after complete words only.

Line wrapping - Visual flags Lets you select a visual flag for wrapped lines:

Flag	Description
None	No visual flag is used.
End	A symbol ↵ is displayed at the end of the wrapped line.
Start	A symbol ↳ is displayed at the beginning of the wrapped line.

Line wrapping - Next line indentation Lets you specify the number of characters by which the next line is indented.

Folding Lets you activate folding. If folding is active, you can fold parts of the source code to get a better overview. Folded parts are marked with a right arrow

▶ in the folding margin. Expanded parts are marked with a down arrow ▼. The folding margin must be greater than 0 (see Folding margin).

Folding - Flags Lets you select where the folding line is displayed.

Line number margin Lets you specify the width of the line number margin in pixels.

Folding margin Lets you specify the width of the folding margin in pixels (see Folding).

Marker margin Lets you specify the width of the marker margin in pixels.

Interpreter Settings Page

Purpose

To specify additional search paths for Python script files inside ConfigurationDesk.

Interpreter Settings page

The page lets you specify additional search paths for Python script files.

Interpreter Python path Lists the user-defined Python directories in which the Interpreter searches for modules. The search order can be changed by moving directories up and down the list by the Up/Down buttons.

Buttons

Button	Description
📁	Lets you open a directory selection dialog. After a directory is selected, it is added to the list of user-defined Python directories.
✕	Lets you delete the selected directory from the list.
⬆	Lets you move the selected directory up the search order of the user-defined Python path.
⬇	Lets you move the selected directory down the search order of the user-defined Python path.

System Python path Lists the permanent system Python directories. You cannot change the system Python path in this dialog.

For instructions, refer to How to Specify the Python Path (ConfigurationDesk Automating Tool Handling 📖).

Platform Management Page

Purpose

To specify general settings for seeking the platforms during ConfigurationDesk startup.

Platform Management page

Seek connected platforms on startup Lets you specify whether to search for registered platforms when ConfigurationDesk is started manually (**Platform Seeking**) or via automation (**Platform Seeking via Tool Automation**).

- If the checkbox is selected, ConfigurationDesk scans the recent hardware configuration and searches for registered and connected SCALEXIO systems or MicroAutoBox III systems.
- If the checkbox is cleared, ConfigurationDesk does not search for connected and registered platforms during startup.

Project Page

Purpose

To specify project root directories and to configure project-specific settings.

Project page

Root directories Lets you specify one or more project root directories. To define and work with projects and ConfigurationDesk applications, at least one project root directory must be specified.

📁 `Insert`	To specify a new project root directory. A new line is added to the list of project root directories. You can enter a directory name in the edit field or select a directory via the Browse button. The path name can contain letters, digits, underscores, minus, parentheses, and dots. The name must not contain a dot or whitespace as the first or last character.
✖ `Delete`	To remove the selected directory from the list of project root directories.
↑	To move the selected directory up in the list of project root directories.
↓	To move the selected directory down in the list of project root directories.
🔍	To browse the contents of the selected directory.
`F2`	To edit the selected directory in the list of project root directories.

Note

To avoid errors, adhere to the following rules:

- Do not specify a ConfigurationDesk project folder as a project root folder.
- Do not specify a project root folder in a project root folder.

Automatically load the most recently used application on startup Lets you specify whether to load the most recently used application when ConfigurationDesk is started. The next time ConfigurationDesk is started, the most recently used application is loaded, even if it was closed before ConfigurationDesk was closed.

Starting an application via a desktop shortcut (see Create Shortcut on page 40) is not affected by this setting.

Note

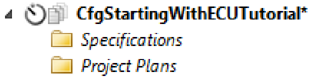
Remember that large projects will take a long time to load.

Only show projects containing applications Activates a filter to show only projects containing applications when you browse for projects via Open Project/Open Project and Application on page 53. The Only show projects containing applications checkbox is selected by default.

Tip

For example, the filter affects ControlDesk projects that contain only experiments. Clear the checkbox to display these projects, or open them via Open File or Project on page 470.

Automatically save project every <x> minutes Lets you specify to save the project at a regular time interval in minutes. If activated, a clock symbol is displayed in the Project Manager next to the project item.



CfgStartingWithECUTutorial*
Specifications
Project Plans

If you activate automatic saving and enter 0 minutes, the project is not saved and no clock symbol is displayed.

Syntax Highlighting Page

Purpose

To alter the settings for the syntax highlighting in the Source Code Editor and the Interpreter.

Syntax Highlighting page

Font Opens the standard Windows Font dialog. You can specify the font, the font color, and the font size.

Background color Lets you select the interpreter's background color.

Bookmarks color Lets you select the color for bookmarks.

Output color Lets you select the text color for output in the Interpreter.

Error color Lets you select the text color for errors in the Interpreter.

Right edge Indicates whether the limit of the right edge is visible in the Source Code Editor or not (right edge visualization).

Right edge - Column Lets you specify the column on which right edge visualization starts.

Right edge - Line Indicates that right edge visualization is a line on the specified column.

Right edge - Background Indicates that right edge visualization is a background color for characters to the right of the specified column.

Right edge - Color Lets you choose a color for right edge visualization (background or line).

Display items Lets you choose a Python item to be configured from the list.

Display items - Text color Lets you choose the text color of the selected display item.

Display items - Background color Lets you choose the background color of the selected Python item.

Display items - Bold Indicates whether the selected Python item is displayed in bold.

Display items - Italic Indicates whether the selected Python item is displayed in italics.

Views Page

Purpose

To alter the settings for workbook tabs and controlbar tabs and to reset the view sets.

Views page

Workbook tabs position If you have opened several sequences and other windows, you have a better overview of them when you use the workbook mode. In this mode, each opened sequence and other windows are represented by a tab. With this setting, you can specify whether the tabs are displayed at the *top* or the *bottom* of the working area.

Controlbar tabs layout You can customize the window layout of ConfigurationDesk. If you have docked several components to a window, it provides tabs to switch between them. With this setting, you can specify the layout of the controlbar tabs:

Layout	Description
Auto-size	Each controlbar tab contains the symbol and the name of the component that it represents. The whole component name is displayed. If the window is too small for displaying all controlbar tabs, you must scroll to the hidden ones.
Compressed	Only the active controlbar tab contains the symbol and name of the component that it represents. The other controlbar tabs are displayed only by their symbols.
Size to Fit	Each controlbar tab contains the symbol and the name of the component that it represents. However, the component names are shortened so that all controlbar tabs are accessible without scrolling.

Show shortcut keys in tooltips If you select this option, the tooltip of a selected command contains information on its shortcut key, if available.

Animate auto hiding of controlbars If you select this option, auto hiding of a window is done with an animation.

Drag controlbars to float Lets you specify whether a docked controlbar can be changed to the floating state by dragging the controlbar's title bar.

Reset all View Sets Lets you reset all view set settings to the state of the first installation.

Note

All the user-defined view sets are deleted.

Related topics

References

Auto Hide..502
Workbook Mode..514

Message Handling

Objective

ConfigurationDesk provides functions to generate error, warning, and general information messages. They are displayed in the Message Viewer and can, for example, be copied or removed.

Where to go from here

Information in this section

Clear Messages

Purpose

To remove all entries from the Message Viewer.

Result

ConfigurationDesk deletes all the entries in the Message Viewer.

Description

Messages generated by the real-time application (for example, for debugging purposes) may increase the number of messages displayed in ConfigurationDesk's Message Viewer very fast. If there are more than 20,000 messages in the Message Viewer, ConfigurationDesk's performance may decrease. To enhance it, delete all existing messages in the Message Viewer.

Related topics

References

Message Viewer....493

Collapse (Messages)

Purpose

To collapse all the subelements of the selected message.

Result

The subelements of the selected message are hidden.

Related topics

References

Message Viewer........ 493

Copy (Messages)

Purpose

To copy the selected messages in the Message Viewer/dSPACE Log to the Clipboard.

Related topics

References

dSPACE Log........ 520
Message Viewer........ 493

Expand (Messages)

Purpose

To expand all the collapsed subelements of the selected message.

Result

The hidden subelements of the selected message are displayed.

Related topics

References

Message Viewer........ 493

Fit All Columns

Purpose

To optimize the width of all the displayed columns in the Message Viewer or in the dSPACE Log.

Related topics

References

dSPACE Log	520
Message Viewer	493

Fit Column Width

Purpose

To resize the selected column to be just a bit larger than the largest column value.

Related topics

References

dSPACE Log	520
Message Viewer	493

Lock Scrolling

Purpose

To disable the automatic vertical scrolling mechanism in the Message Viewer or the dSPACE Log.

Description

By default, the Message Viewer and the dSPACE Log automatically scrolls to the latest entry that is displayed. Locking it enhances ConfigurationDesk's performance.

Related topics

References

dSPACE Log	520
Message Viewer	493

Message Viewer

Purpose

To show or hide the Message Viewer.

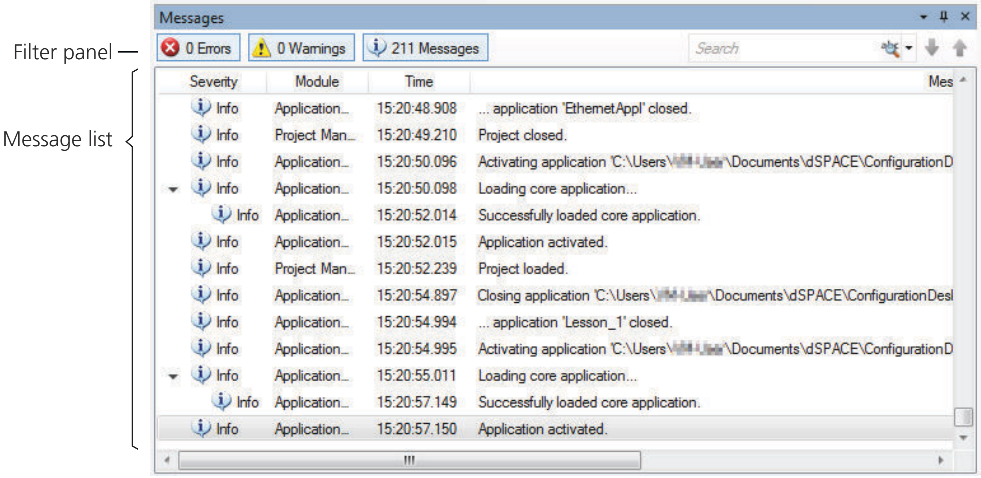
Result

The Message Viewer opens or closes.

Description

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

The Message Viewer looks like this:



	Severity	Module	Time	Mes
	Info	Application...	15:20:48.908	... application 'EthernetAppl' closed.
	Info	Project Man...	15:20:49.210	Project closed.
	Info	Application...	15:20:50.096	Activating application 'C:\Users\[redacted]\Documents\dSPACE\ConfigurationD
▾	Info	Application...	15:20:50.098	Loading core application...
	Info	Application...	15:20:52.014	Successfully loaded core application.
	Info	Application...	15:20:52.015	Application activated.
	Info	Project Man...	15:20:52.239	Project loaded.
	Info	Application...	15:20:54.897	Closing application 'C:\Users\[redacted]\Documents\dSPACE\ConfigurationDesl
	Info	Application...	15:20:54.994	... application 'Lesson_1' closed.
	Info	Application...	15:20:54.995	Activating application 'C:\Users\[redacted]\Documents\dSPACE\ConfigurationD
▾	Info	Application...	15:20:55.011	Loading core application...
	Info	Application...	15:20:57.149	Successfully loaded core application.
	Info	Application...	15:20:57.150	Application activated.

For descriptions of context menu commands, refer to Message Handling on page 489.

Filter panel

If the filter panel is active, it lets you filter the message list and search for text in the message list. Refer to Show Filter Panel on page 495.

Message list

The message list provides the following information for each message:

Information	Description
Severity	The severity level of the message, indicated by one of the following symbols: ▪ ⊗ Errors ▪ ⚠ Warnings ▪ ⓘ Other messages, i.e., infos, advice, and questions
Module[1)]	The module that issued the message.

Information	Description
Time[1]	The time the message was issued.
Message	The content of the message.
Main Module Number[1] (Main Module)	The main module that issued the message.
Submodule Number[1] (Submodule)	The submodule that issued the message.
Message Code[1]	The code of the message.

[1] You can specify whether this information is displayed via the Show Columns command.

A message can be identified by the combination of message code, main module number and submodule number.

Reset Columns

Purpose

To reset the display of message list columns to the default.

Related topics

References

dSPACE Log..520
Message Viewer...493
Show Columns..494

Show Columns

Purpose

To add/remove, for example, the following columns to/from the message list of the Message Viewer/dSPACE Log:

- Module
- Time
- Main Module Number
- Submodule Number
- Message Code

For more information on the columns, refer to Message Viewer on page 493 and dSPACE Log on page 520.

Related topics

References

dSPACE Log..520
Message Viewer..493
Reset Columns..494

Show Details

Purpose

To show/hide details of a selected message.

Description

If active, the dSPACE Log pane is split horizontally. The bottom pane displays properties of the message selected in the upper pane.

Related topics

References

dSPACE Log..520

Show Filter Panel

Purpose

To show or hide the filter panel of the Message Viewer/dSPACE Log.

Filter panel

The filter panel lets you filter the message list and search for text in the message list.

Edit Product Filter (button - only in dSPACE Log) Lets you specify a list of products for filtering the message list.

To apply the product filter, click the Enable/Disable Product Filter button next to the ▽ button.

Enable/Disable Product Filter (Products button - only in dSPACE Log) Lets you enable/disable the product filter.

Show/Hide Errors (⊗ Errors button) Lets you display or hide errors.

Show/Hide Warnings (⚠ Warnings button) Lets you display or hide warnings.

Show/Hide other Messages (ℹ Messages button) Lets you display or hide other messages, i.e., infos, advice, and questions.

Search Lets you enter a text string for searching the message list.

You can use the following wildcards in the text string:

- **?** (wildcard for one character)
- ***** (wildcard for any number of characters)

To mask a wildcard, enter the \ character before the wildcard.

To select the next occurrence of the search string, click ⬇ next to the edit field, or press the `Enter` key while the search field has the focus.

To select the previous occurrence of the search string, click ⬆ next to the edit field.

Related topics

References

dSPACE Log	520
Message Viewer	493

Show Message

Purpose

To show a selected message in a separate dialog.

Tip

You can also double-click a message to show it in a separate dialog.

Related topics

References

dSPACE Log	520
Message Viewer	493
Show Details	495

Tree View

Purpose

To toggle between flat and hierarchy views of the messages.

Related topics

References

Window Handling

Objective

ConfigurationDesk's window handling provides various commands and dialogs, which are accessible via the **View** ribbon and the context menus of the ConfigurationDesk components.

Where to go from here

Information in this section

Allow Docking

Purpose

To allow the pane to be docked.

Result

When moved within ConfigurationDesk's main window, the pane is docked if ConfigurationDesk finds a suitable place for it.

Tip

It might be useful to not allow docking if you want to keep a floating pane constantly open and available without docking it by accident.

Description

When docked, the pane has no window frame but a header bar for controlling its state, position, and size.

Related topics

HowTos

How to Move Panes to Different Positions (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Dock / Docked...505
Float / Floating...507
Float in Working Area...507

Apply Original Order

Purpose

To restore the original order of the entries in all columns.

Related topics

Basics

Introduction to Tables (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Clear Sorting	504
Sort Ascending	510
Sort Descending	511

Arrange Icons

Purpose

To arrange all minimized windows in ConfigurationDesk's working area.

Result

All minimized windows are arranged horizontally at the bottom of ConfigurationDesk's working area.

Auto Hide

Purpose

To automatically hide docked panes that you are currently not using.

Description

The Auto Hide command is enabled only for panes in docked mode.

Related topics

Basics

Customizing View Sets (ConfigurationDesk Real-Time Implementation Guide 📖)

Best Fit

Purpose

To optimize the width of the selected column.

Result

The width of the selected column is optimized.

Description

The width of the selected column is resized to be just a bit larger than the largest column value.

Related topics

Basics

Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide 🕮)

Best Fit All Columns

Purpose

To optimize the width of all the displayed columns.

Result

The width of all the displayed columns is optimized according to their content.

Related topics

Basics

Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide 🕮)

Cascade

Purpose

To cascade all currently open windows in ConfigurationDesk´s working area.

Result

The windows are cascaded.

Description

The windows are cascaded diagonally, one on top of the other. You can move them and pull them to any size you desire.

Clear Sorting

Purpose

To restore the original order of the entries in a selected column.

Related topics

Basics

Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Apply Original Order..502
Sort Ascending..510
Sort Descending..511

Close / Close Window

Purpose

To close a pane in ConfigurationDesk.

Close All

Purpose

To close all windows in ConfigurationDesk's working area.

Close All But This

Purpose

To close all windows except for the selected one in the working area.

Result

All windows except for the selected one are closed.

Close Window

Purpose

To close the window currently displayed in the working area.

Result

The window currently displayed in the working area is closed.

Column Chooser

Purpose

To open a dialog for customizing the columns of the current table.

Description

With the column chooser, you can drag a column from the opened dialog to the current table. To remove a column, you can drag it to the column chooser dialog.

Related topics

Basics

Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide 🕮)

Dock / Docked

Purpose

To place a floating window at a certain location in the user interface as a pane and connect it to its neighboring panes.

Description

If you dock a floating window, it is moved to a standard location in the user interface as a pane and docked to its neighboring panes.

The **Docked** command from the title bar's context menu is available for a pane that is in floating state.

If you drag a pane and drop it on a docking sticker, it is automatically docked at the selected position.

> **Note**
>
> For some panes, you must select the **Allow Docking** option to enable docking at all.

Related topics

HowTos

How to Move Panes to Different Positions (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Allow Docking 501
Float / Floating 507
Float in Working Area 507

Fix left / Fix right

Purpose

To fix a column to the left or right side of a table.

Result

The selected column is fixed to the left or right side of the table.

Description

Fixed columns do not move if you horizontally scroll through columns.

Related topics

Basics

Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Unfix 513

Float / Floating

Purpose

To make a pane moveable on the entire screen.

Description

- The **Float** command on the **View** ribbon or on window tabs is available only for a pane that is floating in the working area. If you float the component, the window type changes, and you can drag it to any place on your screen, even outside of the ConfigurationDesk user interface. The window is not docked to other windows.
- The **Floating** command from a title bar's context menu is available for a pane that is in docked state.

In the **ConfigurationDesk Options** dialog, you can configure that panes can be changed to the floating state by dragging their title bar. Refer to Views Page on page 487.

Related topics

HowTos

How to Move Panes to Different Positions (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Float in Working Area

Purpose

To move a pane to the working area.

Result

The pane is moved to the working area as a tabbed window. If you have used the **Cascade** command, the pane is moved to the working area as a floating window.

Related topics

HowTos

How to Move Panes to Different Positions (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Minimize Window

Purpose

To minimize the window currently displayed in the working area.

Result

The window currently displayed in the working area is minimized. If multiple windows are open in the working area, the other windows are cascaded.

More Windows

Purpose

To arrange windows in ConfigurationDesk's working area as you require them.

Result

Windows are arranged in ConfigurationDesk's working area according to your settings.

Description

The Arrange Windows dialog opens for you to arrange all the windows available in ConfigurationDesk's working area.

Select Window Lets you select the windows whose settings you want to change.

Activate Lets you activate the selected window.

OK Lets you close the dialog.

Save Lets you save a modified script file that is currently open in the Source Code Editor in the working area.

Close Window Lets you close the selected window.

Cascade Lets you cascade the selected windows in ConfigurationDesk's working area.

Tile Horizontally Lets you arrange the selected windows in ConfigurationDesk's working area horizontally.

Tile Vertically Lets you arrange the selected windows in ConfigurationDesk's working area vertically.

Minimize Lets you minimize the selected windows.

Reset View Set

Purpose

To reset the currently active view set to its default state.

Description

When you save a view set via **Save View Set**, the current state of the view set is saved as the new default state.

You can reset all the view set settings to the state of the first installation via the Views Page on page 487.

Related topics

Basics

Customizing View Sets (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Customize Quick Access Toolbar/Customize/More Commands...518
Save View Set...510
Switch View Sets...512
Views Page...487

Restore Window

Purpose

To restore the position of floating windows in the working area.

Result

The position of windows that you floated in the working area is restored. If you did not float windows in the working area in the current session, the windows are cascaded.

Save View Set

Purpose

To save the current settings of the active view set.

Description

The current state of the view set is saved as the new default state. The next time you use **Reset View Set** the view set is reset to the default state.

You can reset all the view set settings to the state of the first installation via the Views Page on page 487.

Related topics

Basics

Customizing View Sets (ConfigurationDesk Real-Time Implementation Guide)

References

Customize Quick Access Toolbar/Customize/More Commands.. 518
Reset View Set.. 509
Switch View Sets... 512

Sort Ascending

Purpose

To sort the entries of a selected column alphabetically in ascending order.

Description

You can sort a column by using this command from the column header's context menu or just by clicking the column header.

Related topics

Basics

Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Apply Original Order.. 502
Sort Descending.. 511

Sort Descending

Purpose

To sort the entries of a selected column alphabetically in descending order.

Description

You can sort a column by using this command from the column header's context menu or just by clicking the column header.

Related topics

Basics

Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide 🕮)

References

Apply Original Order.. 502
Sort Ascending.. 510

Switch Controlbars

Purpose

To show or activate panes in the current view set, for example, the Message Viewer.

Result

The panes are either shown or activated.

Description

To hide a pane, you can use the Switch Controlbars command again or use the Close command or symbol in the pane's title bar.

Related topics

Basics

Customizing View Sets (ConfigurationDesk Real-Time Implementation Guide)

Switch View Sets

Purpose

To switch between ConfigurationDesk's view sets.

Result

The selected view set is active.

Related topics

Basics

Overview of the User Interface of ConfigurationDesk (ConfigurationDesk Real-Time Implementation Guide)

Switch Windows

Purpose

To activate another open window.

Result

The selected window is opened in the working area.

Tile Horizontally

Purpose

To arrange the currently open windows in ConfigurationDesk horizontally.

Result

The windows are arranged horizontally.

Description

The windows are arranged so that each window has the same size, initially without overlaps. You can move the child windows, pull them to any size you desire, and overlap them.

Tile Vertically

Purpose

To arrange the currently open windows in ConfigurationDesk vertically.

Result

The windows are arranged vertically.

Description

The windows are arranged so that each window has the same size, initially without overlaps. You can move the child windows, pull them to any size you desire, and overlap them.

Unfix

Purpose

To unfix a column that was previously fixed to the left or right side of a table.

Result

The fixed column is unfixed.

Related topics

Basics

Customizing Table Rows and Columns (ConfigurationDesk Real-Time Implementation Guide 📖)

References

Fix left / Fix right..506

Windows

Purpose

To open view-set-specific panes for the current view set.

Result

The selected pane opens.

Description

On the Home ribbon, you can select a pane from the Navigation – Windows list.

The list contains the panes that are open in the current view set by default, plus additional panes that are associated with the view set's purpose. Panes that are already open in the current view set are grayed out and cannot be selected.

Related topics

Basics

Customizing View Sets (ConfigurationDesk Real-Time Implementation Guide 🕮)

Workbook Mode

Purpose

To enable or disable the Workbook mode for ConfigurationDesk's working area.

Result

If there is a checkmark to the left, the Workbook mode is enabled in ConfigurationDesk's working area.

Description

In the Workbook mode, each separate window in the working area can be activated by a tab at the bottom or the top of the working area. The position of the workbook tabs can be specified in the ConfigurationDesk Options dialog, refer to Views Page on page 487. You can sort the tabs by dragging a tab to the desired position. A tab provides a specific context menu.

If there is no checkmark, the Workbook mode is deactivated.

Basic Interface

Objective

ConfigurationDesk's basic interface provides various commands and dialogs that are accessible via ribbon, symbols, and the context menus.

Where to go from here

Information in this section

Add to Quick Access Toolbar

Purpose

To add the selected command to the Quick Access Toolbar.

Result

The command is added to the Quick Access Toolbar.

Related topics

HowTos

How to Customize the Quick Access Toolbar (ConfigurationDesk Real-Time Implementation Guide 🕮)

Customize (User Functions)

Purpose

To add Python scripts and external functions to ConfigurationDesk's User Functions ribbon group.

Result

The user function ⧉ is added to the User Functions ribbon group.

Description

Adding external functions gives you a quick and easy access to any function you might need while working with ConfigurationDesk.

User functions dialog

User Functions Menu Items Lets you select which of the currently available user functions you want to edit.

Description Lets you enter a description for the user function.

Command Lets you select the path and folder to open the program file from.

Arguments Lets you browse for a file or folder argument to pass to the selected user function.

Initial directory Lets you browse for the initial folder to execute the program in.

Image Lets you load, edit and clear an image for the selected user function. You can use images in bitmap or PNG format (16 × 16 pixels) to be used as an icon in the ribbon.

Show window (Applies to console applications) Indicates whether the window of the running process is displayed.

Related topics

References

Customize Quick Access Toolbar/Customize/More Commands

Purpose

To customize the Quick Access Toolbar and view sets.

Result

The Quick Access Toolbar and the view sets are customized according to the settings in the **Customize** dialog.

Description

ConfigurationDesk opens the **Customize** dialog, which lets you customize the Quick Access Toolbar and view sets.

Tabbed pages

The **Customize** dialog consists of the following pages:

Purpose	Page
To customize the Quick Access Toolbar.	Quick Access Toolbar page
To create and manage view sets.	View Sets page

Quick Access Toolbar page

To customize the Quick Access Toolbar.

Choose commands from Lists all the available ribbons. Lets you select a ribbon to display its commands.

Add >> Lets you add the selected command to the Quick Access Toolbar.

<< Remove Lets you remove the selected command from the Quick Access Toolbar.

Reset to Factory Settings Lets you reset the Quick Access Toolbar to the default.

Show Quick Access Toolbar below the ribbon Lets you specify whether to display the Quick Access Toolbar above or below the ribbon.

Up Lets you move the selected command toward the top of the list to specify the order of commands on the Quick Access Toolbar.

Down Lets you move the selected command toward the bottom of the list to specify the order of commands on the Quick Access Toolbar.

View Sets page

To create and manage view sets.

A view set is a named configuration of the panes of the application. The configuration includes the geometry, visibility, and docking or floating state of panes.

All the modifications you make to the screen arrangement are automatically saved to the currently active view set.

- To discard all the modifications that you made in the screen arrangement, you can reset the active view set to its default state via Reset View Set on page 509.
- You can save the state of the current view set as its default state via Save View Set on page 510.
- If you want to reset all the view set settings to the state of the first installation, you can do so via the **ConfigurationDesk Options** dialog. Refer to Views Page on page 487.

View sets Lists all the available view sets and their icons.

New View Set Lets you create a new view set. Opens the **New View Set** dialog for you to enter a unique name for the new view set. The panes and the screen arrangement of the new view set are based on the currently active view set.

The new view set is added to the navigation bar.

Rename View Set Lets you rename the selected view set. Opens the **Rename View Set** dialog for you to enter a unique name.

Delete View Set Lets you delete the selected view set. The last remaining view set cannot be deleted.

Up Lets you move the selected view set toward the top of the list. This changes the order of view sets on the navigation bar.

Down Lets you move the selected view set toward the bottom of the list. This changes the order of view sets on the navigation bar.

Load Icon Lets you open a standard dialog to select a BMP or PNG image as the icon representing the view set on toolbars and in menus. The image width and height must not exceed 16 pixels.

Edit Icon Lets you open the **Button Editor** dialog to create an individual icon representing the currently selected view set on toolbars and in menus.

Reset Icon Lets you clear the user-selected icon and display the standard icon for view sets instead.

Reset to Factory Settings Lets you reset all view set settings to the state of the first installation.

Note

All the user-defined view sets are deleted.

Import Opens the **Import View Sets** dialog that lets you select a VSET file containing view sets to be imported.

In the dialog, you can specify whether naming conflicts should be resolved:

- By overriding existing view sets, or
- By renaming the imported ones.

> **Note**
>
> To import a view set, the exporting and the importing application must have the same configurations (such as the available licenses).

Export Lets you export all the view sets to a view set (VSET) file. A VSET file also contains the icons assigned to the view sets.

Related topics

Basics

Customizing View Sets (ConfigurationDesk Real-Time Implementation Guide 🕮)

HowTos

How to Customize the Quick Access Toolbar (ConfigurationDesk Real-Time Implementation Guide 🕮)

dSPACE Help

Purpose

To open the user documentation of ConfigurationDesk.

Result

The user documentation of ConfigurationDesk opens.

Related topics

References

Using dSPACE Help... 531

dSPACE Log

Purpose

To open the dSPACE Log in ConfigurationDesk's working area.

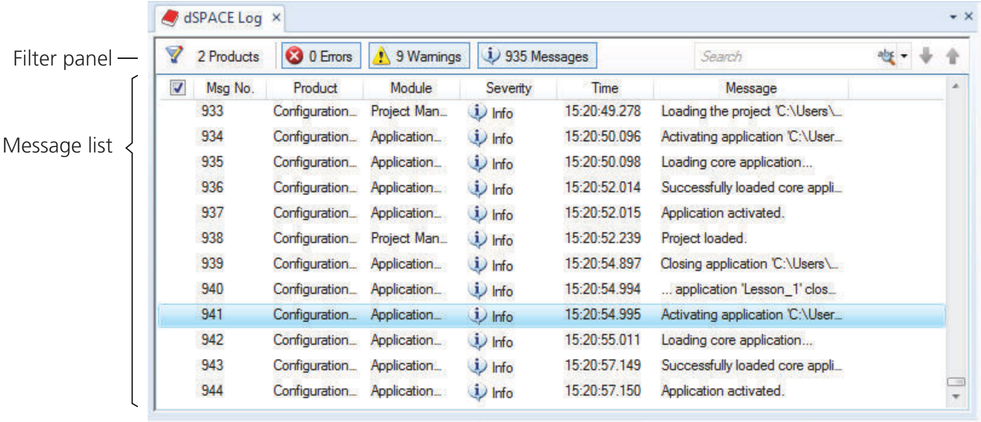
Result

The dSPACE Log is shown in the working area.

Description

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

The dSPACE Log Viewer looks like this:



Filter panel

Message list

dSPACE Log ×

2 Products | ✖ 0 Errors | ⚠ 9 Warnings | ⓘ 935 Messages | Search

☑	Msg No.	Product	Module	Severity	Time	Message
	933	Configuration…	Project Man…	ⓘ Info	15:20:49.278	Loading the project 'C:\Users\…
	934	Configuration…	Application…	ⓘ Info	15:20:50.096	Activating application 'C:\User…
	935	Configuration…	Application…	ⓘ Info	15:20:50.098	Loading core application...
	936	Configuration…	Application…	ⓘ Info	15:20:52.014	Successfully loaded core appli…
	937	Configuration…	Application…	ⓘ Info	15:20:52.015	Application activated.
	938	Configuration…	Project Man…	ⓘ Info	15:20:52.239	Project loaded.
	939	Configuration…	Application…	ⓘ Info	15:20:54.897	Closing application 'C:\Users\…
	940	Configuration…	Application…	ⓘ Info	15:20:54.994	... application 'Lesson_1' clos…
	941	Configuration…	Application…	ⓘ Info	15:20:54.995	Activating application 'C:\User…
	942	Configuration…	Application…	ⓘ Info	15:20:55.011	Loading core application...
	943	Configuration…	Application…	ⓘ Info	15:20:57.149	Successfully loaded core appli…
	944	Configuration…	Application…	ⓘ Info	15:20:57.150	Application activated.

For descriptions of context menu commands, refer to Message Handling on page 489.

Filter panel

If the filter panel is active, it lets you filter the message list and search for text in the message list. Refer to Show Filter Panel on page 495.

Message list

The message list provides the following information for each message:

Information	Description
Show/Hide Messages of Log Sessions	Select this option to expand the messages of all sessions. Clear it to collapse the messages of all sessions. You can expand or collapse the messages of a single session by clicking the corresponding icons (▸/▾) in this column.
Message Number[1)] (Msg No)	The consecutive number of the message in the dSPACE Log.
Product[1)]	The product that issued the message.
Process ID[1)] (PID)	The ID of the process that issued the message.
Module[1)]	The module that issued the message.
Severity[1)]	The severity level of the message, indicated by one of the following symbols: ▪ ✖ Error ▪ ⚠ Warning ▪ ⚑ Session start

Information	Description
	- ⓘ Other messages, i.e., infos, advice, and questions
Time[1]	The time the message was issued.
Message	The content of the message.
Main Module Number[1] (Main Module)	The main module that issued the message.
Submodule Number[1] (Submodule)	The submodule that issued the message.
Message Code[1]	The code of the message.
Thread ID[1] (TID)	The ID of the thread that issued the message.

[1] You can specify whether this information is displayed via the **Show Columns** command.

A message can be identified by the combination of message code, main module number and submodule number.

Execute (User Function)

Purpose

To carry out any one of the user functions you added via the **Customize** command.

Result

The result of these user functions depends on the settings you created.

Description

The defined user functions are available via the **Automation** ribbon. The first nine user functions are also available via icons.

Related topics

References

Exit

Purpose

To prompt to save the project and to exit the current ConfigurationDesk session.

Description

If you made any changes to any open files in ConfigurationDesk, you are prompted to save them before the session is quit.

Export View

Purpose

To access different commands that let you export or print data from different panes for documentation purposes.

Result

Several commands that let you export or print data are available in a submenu.

Description

The commands available in the submenu are:

- Print on page 526
- Export View to XML/CSV on page 523

Related topics

Basics

Exporting Data for Documentation Purposes (ConfigurationDesk Real-Time Implementation Guide 📖)

Export View to XML/CSV

Purpose

To export data from various panes to an XML or CSV file.

Result

ConfigurationDesk opens a standard dialog for you to export the data.

Description

The data in the XML or CSV file can be used for debugging purposes or to compare data from different projects.

Related topics

Basics

Exporting Data for Documentation Purposes (ConfigurationDesk Real-Time Implementation Guide 📖)

Help (Backstage View)

Purpose

To provide access to help commands.

Description

You have access to commands such as:

- dSPACE Help on page 520
- Using dSPACE Help on page 531
- New Features and Migration on page 525
- PDF Files on page 526
- Keyboard Help on page 524

Keyboard Help

Purpose

To display a list of available shortcut keys.

Result

ConfigurationDesk opens the Active Keyboard Shortcuts dialog that lists the available shortcut keys.

Active Keyboard Shortcuts dialog

ConfigurationDesk displays a list with the shortcut keys that are globally available.

Copy List Lets you copy the list of keyboard shortcuts to the clipboard. You can paste them, for example, to a text document and print them.

Licenses

Purpose

To open the Licenses dialog which displays accessible and used licenses.

Result

ConfigurationDesk opens the Licenses dialog which displays license information (accessible and used licenses).

Licenses dialog

License Name Lists all licenses which are available for the running ConfigurationDesk version. For more details on the available licenses, refer to Overview of Licenses (ConfigurationDesk Real-Time Implementation Guide 🕮).

Short Name Lists the short names of the licenses.

Status Lists the status of the licenses.

- A license indicated as accessible can be used for your ConfigurationDesk projects.
- A license indicated as used is currently used by your active ConfigurationDesk application. For function block licenses and the SCALEXIO FIU licenses, the number of instances is displayed.

Minimize the Ribbon

Purpose

To toggle between a minimized and expanded ribbon.

Result

If the ribbon is minimized, only the ribbon tabs are shown on the user interface.

Description

To access a command when the ribbon is minimized, click a ribbon tab. The commands remain on the user interface until you execute a command from the ribbon.

New Features and Migration

Purpose

To display new features and required migration steps for all the products in the current dSPACE Release.

Result

dSPACE Help opens with New Features and Migration 🕮 displayed. Navigate to the specific product information to read about the new features of a specific product. If there are migration steps required, the necessary steps are described.

PDF Files

Purpose

To open a folder containing documentation PDF files of the current dSPACE Release.

Print

Purpose

To print the contents of various panes.

Result

ConfigurationDesk opens a Preview dialog for you to prepare and execute the print process.

Redo

Purpose

To redo your most recent actions or changes in ConfigurationDesk.

Result

Any change or action that was undone via the Undo command is performed once again.

Description

You can redo actions or changes that you have carried out in ConfigurationDesk, for example:

- Changes to the signal chain such as mapping lines or setting block properties
- Changes to properties and tables such as options in the Build Configuration table
- Actions in browsers such as adding or removing devices in the **External Device Browser**
- Actions in the **Working View Manager**

If you carry the command out once, one change or action is redone; if you carry it out twice, two are redone, and so on.

Note

Redo is not supported for:

- Changes affecting the file system, for example:
  - Creating new projects or ConfigurationDesk applications
  - Renaming project files or applications
  - Running the build process
- Changes related to the **Platform Manager**, for example:
  - Registering or removing SCALEXIO systems or MicroAutoBox III systems
  - Configuring loads or names in the Platform Manager
  - Downloading RTAs
- Changes only affecting the display of information, for example:
  - Highlighting blocks in the signal chain
  - Setting display filters
  - Arranging panes
- Changes in the **Message Viewer** such as clearing or sorting messages

Related topics

References

Undo.. 529

Remove from Quick Access Toolbar

Purpose

To remove the selected command from the Quick Access Toolbar.

Result

The command is removed from the Quick Access Toolbar.

Show Quick Access Toolbar Above / Below the Ribbon

Purpose

To show the Quick Access Toolbar above or below the ribbon.

Result

The Quick Access Toolbar is shown above or below the ribbon.

Start Page

Purpose

To display the Start page.

Result

The Start page is displayed in the working area.

Start Page

The Start page allows you, for example, to open an existing project or create a new one. The Start page also provides quick access to the product documentation.

New Project + Application / New Application Opens the File – New backstage view, where you can create a new project and ConfigurationDesk application or add an application to a currently open project.

For more information, refer to:

- Create New Project and Application on page 39
- Create Application on page 34

Open Project + Application Lets you open an existing project.

For details, refer to Open Project/Open Project and Application on page 53.

Recent Lets you open one of the most recently opened projects and applications.

Basic Documentation / Advanced Documentation / New Features and Migration Lets you display different parts of the ConfigurationDesk documentation.

Status Bar

Purpose

To show or hide ConfigurationDesk's status bar at the bottom of ConfigurationDesk's main window.

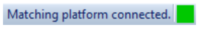
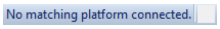
Result

A checkmark next to the command on the View ribbon indicates that the status bar is shown at the bottom of the main window. If there is no checkmark, it is hidden.

Description

The status bar visualizes the current application state of the active ConfigurationDesk application.

Each active ConfigurationDesk application has one of two application states, depending on the hardware topology and registered platforms:

State	Status Bar Visualization	Description
Matching platform connected	 Matching platform connected. 🟩 (green)	The hardware topology of the active application matches the hardware topology of an accessible hardware system displayed in the Platform Manager.
No matching platform connected	 No matching platform connected. □ (white)	The hardware topology of the active application does not match the hardware topology of any accessible hardware system displayed in the Platform Manager or no registered platform is available.

For more details, refer to Basics on Connecting a ConfigurationDesk Application to a Hardware System (ConfigurationDesk Real-Time Implementation Guide).

Related topics

Basics

Basics on Connecting a ConfigurationDesk Application to a Hardware System (ConfigurationDesk Real-Time Implementation Guide)

Undo

Purpose

To undo your most recent actions or changes in ConfigurationDesk.

Result

The previously performed user actions or changes are reversed.

Description

You can undo actions or changes that you have carried out in ConfigurationDesk, for example:

- Changes to the signal chain such as mapping lines or setting block properties
- Changes to properties and tables such as options in the Build Configuration table
- Actions in browsers such as adding or removing devices in the External Device Browser
- Actions in the Working View Manager

If you carry the command out once, one change or action is undone; if you carry it out twice, two are undone, and so on.

Note

Undo is not supported for:

- Changes affecting the file system, for example:
  - Creating new projects or ConfigurationDesk applications
  - Renaming project files or applications
  - Running the build process
- Changes related to the **Platform Manager**, for example:
  - Registering or removing SCALEXIO systems or MicroAutoBox III systems
  - Configuring loads or names in the Platform Manager
  - Downloading RTAs
- Changes only affecting the display of information, for example:
  - Highlighting blocks in the signal chain
  - Setting display filters
  - Arranging panes
- Changes in the **Message Viewer** such as clearing or sorting messages

Related topics

References

Redo.. 526

User Functions Output

Purpose

To show the User Functions Output pane.

Result

The User Functions Output pane is displayed.

Description

The User Functions Output pane provides access to the output of user function tools added to the user interface. The output of Python scripts is displayed in the Interpreter pane.

Related commands

The User Functions Output pane provides the following commands:

Copy To copy the entries in the user functions output viewer to the Clipboard.

Select All To select all the entries in the User Functions Output pane.

Clear All To clear all the entries in the User Functions Output pane.

Font To specify the font of the displayed output.

Related topics

References

Customize (User Functions).. 517
Execute (User Function)... 522

Using dSPACE Help

Purpose

To get information on working with dSPACE Help.

Result

dSPACE Help opens. It provides information on general handling and instructions on using dSPACE Help.

ConfigurationDesk Glossary

Introduction

The glossary briefly explains the most important expressions and naming conventions used in the ConfigurationDesk documentation.

Where to go from here

Information in this section

A

Application There are two types of applications in ConfigurationDesk:

- A part of a ConfigurationDesk project: ConfigurationDesk application.
- An application that can be executed on dSPACE real-time hardware: real-time application.

Application process A component of a processing unit application. An application process contains one or more tasks.

Application process component A component of an application process. The following application process components are available in the **Components** subfolder of an application process:

- Behavior models that are assigned to the application process, including their predefined tasks, runnable functions, and events.
- Function blocks that are assigned to the application process.

AutomationDesk A dSPACE software product for creating and managing any kind of automation tasks. Within the dSPACE tool chain, it is mainly used for automating tests on dSPACE hardware.

AUTOSAR system description file An AUTOSAR XML (ARXML) file that describes a system according to AUTOSAR. A system is a combination of a hardware topology, a software architecture, a network communication, and information on the mappings between these elements. The described network communication usually consists of more than one bus system (e.g., CAN, LIN, FlexRay).

B

Basic PDU A general term used in the documentation to address all the PDUs the Bus Manager supports, except for container IPDUs, multiplexed IPDUs, and secured IPDUs. Basic PDUs are represented by the ▶ or ◀ symbol in tables and browsers. The Bus Manager provides the same functionalities for all basic PDUs, such as ISignal IPDUs or NMPDUs.

Behavior model A model that contains the control algorithm for a controller (function prototyping system) or the algorithm of the controlled system (hardware-in-the-loop system). It does not contain I/O functionality nor access to the hardware. Behavior models can be modeled, for example, in MATLAB/Simulink by using Simulink Blocksets and Toolboxes from the MathWorks®.

You can add Simulink behavior models to a ConfigurationDesk application. You can also add code container files containing a behavior model such as Functional Mock-up Units, or Simulink implementation containers to a ConfigurationDesk application.

Bidirectional signal port A signal port that is independent of a data direction or current flow. This port is used, for example, to implement bus communication.

BSC file A bus simulation container file that is generated with the Bus Manager and contains the configured bus communication of one application process.

Build Configuration table A pane that lets you create build configuration sets and configure build settings, for example, build options, or the build and download behavior.

Build Log Viewer A pane that displays messages and warnings during the build process.

Build process A process that generates an executable real-time application based on your ConfigurationDesk application that can be run on a SCALEXIO system or MicroAutoBox III system. The build process can be controlled and configured via the Build Log Viewer. If the build process is successfully finished, the build result files (build results) are added to the ConfigurationDesk application.

Build results The files that are created during the build process. Build results are named after the ConfigurationDesk application and the application process from which they originate. You can access the build results in the Project Manager.

Bus access The representation of a run-time communication cluster. By assigning one or more bus access requests to a bus access, you specify which communication clusters form one run-time communication cluster.

In ConfigurationDesk, you can use a bus function block (CAN, LIN) to implement a bus access. The hardware resource assignment of the bus function block specifies the bus channel that is used for the bus communication.

Bus access request The representation of a request regarding the bus access. There are two sources for bus access requests:

- At least one element of a communication cluster is assigned to the Simulated ECUs, Inspection, or Manipulation part of a bus configuration. The related bus access requests contain the requirements for the bus channels that are to be used for the cluster's bus communication.

- A frame gateway is added to the **Gateways** part of a bus configuration. Each frame gateway provides two bus access requests that are required to specify the bus channels for exchanging bus communication.

Bus access requests are automatically included in BSC files. To build a real-time application, each bus access request must be assigned to a bus access.

Bus Access Requests table A pane that lets you access bus access requests of a ConfigurationDesk application and assign them to bus accesses.

Bus configuration A Bus Manager element that implements bus communication in a ConfigurationDesk application and lets you configure it for simulation, inspection, and/or manipulation purposes. The required bus communication elements must be specified in a communication matrix and assigned to the bus configuration. Additionally, a bus configuration lets you specify gateways for exchanging bus communication between communication clusters. A bus configuration can be accessed via specific tables and its related Bus Configuration function block.

Bus Configuration Function Ports table A pane that lets you access and configure function ports of bus configurations.

Bus Configurations table A pane that lets you access and configure bus configurations of a ConfigurationDesk application.

Bus Inspection Features table A pane that lets you access and configure bus configuration features of a ConfigurationDesk application for inspection purposes.

Bus Manager

- Bus Manager in ConfigurationDesk

  A ConfigurationDesk component that lets you configure bus communication and implement it in real-time applications or generate bus simulation containers.
- Bus Manager (stand-alone)

  A dSPACE software product based on ConfigurationDesk that lets you configure bus communication and generate bus simulation containers.

Bus Manipulation Features table A pane that lets you access and configure bus configuration features of a ConfigurationDesk application for manipulation purposes.

Bus simulation container A container that contains bus communication configured with the Bus Manager. Bus simulation container (BSC) files can be used in the VEOS Player and in ConfigurationDesk. In the VEOS Player, they let you implement the bus communication in an offline simulation application.

In ConfigurationDesk, they let you implement the bus communication in a real-time application independently from the Bus Manager.

Bus Simulation Features table A pane that lets you access and configure bus configuration features of a ConfigurationDesk application for simulation purposes.

Buses Browser A pane that lets you display and manage the communication matrices of a ConfigurationDesk application. For example, you can access communication matrix elements and assign them to bus configurations. This pane is available only if you work with the Bus Manager.

C

Cable harness A bundle of cables that provides the connection between the I/O connectors of the real-time hardware and the external devices, such as the ECUs to be tested. In ConfigurationDesk, it is represented by an external cable harness component.

CAFX file A ConfigurationDesk application fragment file that contains signal chain elements that were exported from a user-defined working view or the Temporary working view of a ConfigurationDesk application. This includes the elements' configuration and the mapping lines between them.

CDL file A ConfigurationDesk application file that contains links to all the documents related to an application.

Channel multiplication A feature that allows you to enhance the max. current or max. voltage of a single hardware channel by combining several channels. ConfigurationDesk uses a user-defined value to calculate the number of hardware channels needed. Depending on the function block type, channel multiplication is provided either for current enhancement (two or more channels are connected in parallel) or for voltage enhancement (two or more channels are connected in series).

Channel request A channel assignment required by a function block. ConfigurationDesk determines the type(s) and number of channels required for a function block according to the assigned channel set, the function block features, the block configuration and the required physical ranges. ConfigurationDesk provides a set of suitable and available hardware resources for each channel request. This set is produced according to the hardware topology added to the active ConfigurationDesk application. You have to assign each channel request to a specific channel of the hardware topology.

Channel set A number of channels of the same channel type located on the same I/O board or I/O unit. Channels in a channel set can be combined, for example, to provide a signal with channel multiplication.

Channel type A term to indicate all the hardware resources (channels) in the hardware system that provide exactly the same characteristics. Examples for channel type names: Flexible In 1, Digital Out 3, Analog In 1. An I/O board in a hardware system can have channel sets of several channel types. Channel sets of one channel type can be available on different I/O boards.

Cluster Communication cluster.

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

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

or

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

Communication cluster A communication network of network nodes that are connected to the same physical channels and share the same bus protocol and address range.

Communication matrix A file that defines the communication of a bus network. It can describe the bus communication of one communication cluster or a bus network consisting of different bus systems and clusters. Files of various file formats can be used as a communication matrix: For example, AUTOSAR system description files, DBC files, LDF files, and FIBEX files.

Communication package A package that bundles Data Inport blocks which are connected to Data Outport blocks. Hence, it also bundles the signals that are received by these blocks. If Data Inport blocks are executed within the same task and belong to the same communication package, their data inports are read simultaneously. If Data Outport blocks that are connected to the Data Inport blocks are executed in the same task, their output signals are sent simultaneously in one data package. Thus, communication packages guarantee simultaneous signal updates within a running task (data snapshot).

Configuration port A port that lets you create the signal chain for the bus communication implemented in a Simulink behavior model. The following configuration ports are available:

- The configuration port of a Configuration Port block.
- The Configuration port of a CAN, LIN, or FlexRay function block.

To create the signal chain for bus communication, the configuration port of a Configuration Port block must be mapped to the Configuration port of a CAN, LIN, or FlexRay function block.

Configuration Port block A model port block that is created in ConfigurationDesk during model analysis for each of the following blocks found in the Simulink behavior model:

- RTICANMM ControllerSetup block
- RTILINMM ControllerSetup block
- FLEXRAYCONFIG UPDATE block

Configuration Port blocks are also created for bus simulation containers. A Configuration Port block provides a configuration port that must be mapped to the Configuration port of a **CAN**, **LIN**, or **FlexRay** function block to create the signal chain for bus communication.

ConfigurationDesk application A part of a ConfigurationDesk project that represents a specific implementation. You can work with only one application at a time, and that application must be activated.

An application can contain:

- Device topology
- Hardware topology
- Model topology
- Communication matrices
- External cable harness
- Build results (after a successful build process has finished)

You can also add folders with application-specific files to an application.

ConfigurationDesk model interface The part of the model interface that is available in ConfigurationDesk. This specific term is used to explicitly distinguish between the model interface in ConfigurationDesk and the model interface in Simulink.

Conflict A result of conflicting configuration settings that is displayed in the Conflicts Viewer. ConfigurationDesk allows flexible configuration without strict constraints. This lets you work more freely, but it can lead to conflicting configuration settings. ConfigurationDesk automatically detects conflicts and provides the **Conflicts Viewer** to display and help resolve them. Before you build a real-time application, you have to resolve at least the most severe conflicts (e.g., errors that abort the build process) to get proper build results.

Conflicts Viewer A pane that displays the configuration conflicts that exist in the active ConfigurationDesk application. You can resolve most of the conflicts directly in the **Conflicts Viewer**.

Container IPDU A term according to AUTOSAR. An IPDU that contains one or more other IPDUs (i.e., contained IPDUs). When a container IPDU is mapped to a frame, all its contained IPDUs are included in that frame as well.

ControlDesk A dSPACE software product for managing, instrumenting and executing experiments for ECU development. ControlDesk also supports calibration, measurement and diagnostics access to ECUs via standardized protocols such as CCP, XCP, and ODX.

CTLGZ file A ZIP file that contains a V-ECU implementation. CTLGZ files are exported by TargetLink or SystemDesk. You can add a V-ECU implementation based on a CTLGZ file to the model topology just like adding a Simulink model based on an SLX file.

Cycle time restriction A value of a runnable function that indicates the sample time the runnable function requires to achieve correct results. The cycle time restriction is indicated by the **Period** property of the runnable function in the Properties Browser.

D

Data inport A port that supplies data from ConfigurationDesk's function outports to the behavior model.

In a multimodel application, data inports also can be used to provide data from a data outport associated to another behavior model (model communication).

Data outport A port that supplies data from behavior model signals to ConfigurationDesk's function inports.

In a multimodel application, data outports also can be used to supply data to a data inport associated to another behavior model (model communication).

DBC file A Data Base Container file that describes CAN or LIN bus systems. Because the DBC file format was primarily developed to describe CAN networks, it does not support definitions of LIN masters and schedules.

Device block A graphical representation of devices from the device topology in the signal chain. It can be mapped to function blocks via device ports.

Device connector A structural element that lets you group device pins in a hierarchy in the External Device Connectors table to represent the structure of the real connector of your external device.

Device pin A representation of a connector pin of your external device. Device ports are assigned to device pins. ConfigurationDesk can use the device pin assignment together with the hardware resource assignment and the device port mapping to calculate the external cable harness.

Device port An element of a device topology that represents the signal of an external device in ConfigurationDesk.

Device port group A structural element of a device topology that can contain device ports and other device port groups.

Device topology A component of a ConfigurationDesk application that represents external devices in ConfigurationDesk. You can create a device topology from scratch or easily extend an existing device topology. You can also merge device topologies to extend one. To edit or create device topologies independently of ConfigurationDesk, you can export and import DTFX and XLSX files.

Documents folder A standard folder for user-specific documents.

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

DSA file A dSPACE archive file that contains a ConfigurationDesk application and all the files belonging to it as one unit. It can later be imported to another ConfigurationDesk project.

dSPACE Help The dSPACE online help that contains all the relevant user documentation for dSPACE products. Via the **F1** key or the **Help** button in the dSPACE software you get context-sensitive help on the currently active context.

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

DTFX file A device topology export file that contains information on the interface to the external devices, such as the ECUs to be tested. The information includes details of the available device ports, their characteristics, and the assigned pins.

E

ECHX file An external cable harness file that contains the wiring information for the external cable harness. The external cable harness is the connection between the I/O connectors of the real-time hardware and the devices to be tested, for example, ECUs.

ECU Abbreviation of *electronic control unit*.

An embedded computer system that consists of at least one CPU and associated peripherals. An ECU contains communication controllers and communication connectors, and usually communicates with other ECUs of a bus network. An ECU can be member of multiple bus systems and communication clusters.

ECU application An application that is executed on an ECU. In ECU interfacing scenarios, parts of the ECU application can be accessed (e.g., by a real-time application) for development and testing purposes.

ECU function A function of an ECU application that is executed on the ECU. In ECU interfacing scenarios, an ECU function can be accessed by functions that are part of a real-time application, for example.

ECU Interface Manager A dSPACE software product for preparing ECU applications for ECU interfacing. The ECU Interface Manager can generate ECU interface container (EIC) files to be used in ConfigurationDesk.

ECU interfacing A generic term for methods and tools to read and/or write individual ECU functions and variables of an ECU application. In ECU interfacing scenarios, you can access ECU functions and variables for development and testing purposes while the ECU application is executed on the ECU. For example, you can perform ECU interfacing with SCALEXIO systems or MicroAutoBox III systems to access individual ECU functions by a real-time application.

EIC file An ECU interface container file that is generated with the ECU Interface Manager and describes an ECU application that is configured for ECU interfacing. You can import EIC files to ConfigurationDesk to perform ECU interfacing with SCALEXIO systems or MicroAutoBox III systems.

Electrical interface unit A segment of a function block that provides the interface to the external devices and to the real-time hardware (via hardware resource assignment). Each electrical interface unit of a function block usually needs a channel set to be assigned to it.

Event A component of a ConfigurationDesk application that triggers the execution of a task. The following event types are available:

- Timer event
- I/O event
- Software event

Event port An element of a function block. The event port can be mapped to a runnable function port for modeling an asynchronous task.

Executable application The generic term for real-time applications and offline simulation applications. In ConfigurationDesk, an executable application is always a real-time application since ConfigurationDesk does not support offline simulation applications.

Executable application component A component of an executable application. The following components can be part of an executable application:

- Imported behavior models including predefined tasks, runnable functions, and events. You can assign these behavior models to application processes via drag & drop or by selecting the **Assign Model** command from the context menu of the relevant application process.
- Function blocks added to your ConfigurationDesk application including associated I/O events. Function blocks are assigned to application processes via their model port mapping.

Executable Application table A pane that lets you model executable applications (i.e., real-time applications) and the tasks used in them.

EXPSWCFG file An experiment software configuration file that contains configuration data for automotive fieldbus communication. It is created during the build process and contains the data in XML format.

External cable harness A component of a ConfigurationDesk application that contains the wiring information for the external cable harness (also known as cable harness). It contains only the logical connections and no additional information such as cable length, cable diameters, dimensions or the arrangement of connection points, etc. It can be calculated by ConfigurationDesk or imported from a file so that you can use an existing cable harness and do not have to build a new one. The wiring information can be exported to an ECHX file or XLSX file.

External device A device that is connected to the dSPACE hardware, such as an ECU or external load. The external device topology is the basis for using external devices in the signal chain of a ConfigurationDesk application.

External Device Browser A pane that lets you display and manage the device topology of your active ConfigurationDesk application.

External Device Configuration table A pane that lets you access and configure the most important properties of device topology elements via table.

External Device Connectors table A pane that lets you specify the representation of the physical connectors of your external device including the device pin assignment.

F

FIBEX file An XML file according the ASAM MCD-2 NET standard (also known as Field Bus Exchange Format) defined by ASAM. The file can describe more than one bus system (e.g., CAN, LIN, FlexRay). It is used for data exchange between different tools that work with message-oriented bus communication.

Find Results Viewer A pane that displays the results of searches you performed via the Find command.

FMU file A Functional Mock-up Unit file that describes and implements the functionality of a model. It is an archive file with the file name extension FMU. The FMU file contains:

- The functionality defined as a set of C functions provided either in source or in binary form.
- The model description file (`modelDescription.xml`) with the description of the interface data.
- Additional resources needed for simulation.

You can add an FMU file to the model topology just like adding a Simulink model based on an SLX file.

Frame A piece of information of a bus communication. It contains an arbitrary number of non-overlapping PDUs and the data length code (DLC). CAN frames and LIN frames can contain only one PDU. To exchange a frame via bus channels, a frame triggering is needed.

Frame triggering An instance of a frame that is exchanged via a bus channel. It includes transmission information of the frame (e.g., timings, ID, sender, receiver). The requirements regarding the frame triggerings depend on the bus system (CAN, LIN, FlexRay).

Function block A graphical representation in the signal chain that is instantiated from a function block type. A function block provides the I/O functionality and the connection to the real-time hardware. It serves as a container for functions, for electrical interface units and their logical signals. The function block's ports (function ports and/or signal ports), provide the interfaces to the neighboring blocks in the signal chain.

Function block type A software plug-in that provides a specific I/O functionality. Every function block type has unique features which are different from other function block types.

To use a function block type in your ConfigurationDesk application, you have to create an instance of it. This instance is called a function block. Instances of function block types can be used multiple times in a ConfigurationDesk application. The types and their instantiated function blocks are displayed in the function library of the Function Browser.

Function Browser A pane that displays the function library in a hierarchical tree structure. Function block types are grouped in function classes. Instantiated function blocks are added below the corresponding function block type.

Function inport A function port that inputs the values from the behavior model to the function block to be processed by the function.

Function library A collection of function block types that allows access to the I/O functionality in ConfigurationDesk. The I/O functionality is based on function block types. The function library provides a structured tree view on the available function block types. It is displayed in the Function Browser.

Function outport A function port that outputs the value of a function to be used in the behavior model.

Function port An element of a function block that provides the interface to the behavior model via model port blocks.

Functional Mock-up Unit An archive file that describes and implements the functionality of a model based on the Functional Mock-up Interface (FMI) standard.

G

Global working view The default working view that always contains all signal chain elements.

H

Hardware resource A hardware element (normally a channel on an I/O board or I/O unit) which is required to execute a function block. A hardware resource can be localized unambiguously in a hardware system. Every hardware resource has specific characteristics. A function block therefore needs a hardware resource that matches the requirements of its functionality. This means that not every function block can be executed on every hardware resource. There could be limitations on a function block's features and/or the physical ranges.

Hardware resource assignment An action that assigns the electrical interface unit of a function block to one or more hardware resources. Function blocks can be assigned to any hardware resource which is suitable for the functionality and available in the hardware topology of your ConfigurationDesk application.

Hardware Resource Browser A pane that lets you display and manage all the hardware components of the hardware topology that is contained in your active ConfigurationDesk application in a hierarchical structure.

Hardware topology A component of a ConfigurationDesk application that contains information on a specific hardware system which can be used with ConfigurationDesk. It provides information on the components of the system, such as channel type and slot numbers. It can be scanned automatically from a registered platform, created in ConfigurationDesk's Hardware Resource Browser from scratch, or imported from an HTFX file.

HTFX file A file containing the hardware topology after an explicit export. It provides information on the components of the system and also on the channel properties, such as board and channel types and slot numbers.

I

I/O event An asynchronous event triggered by I/O functions. You can use I/O events to trigger tasks in your application process asynchronously. You can assign the events to the tasks via drag & drop, via the Properties Browser if you have selected a task, or via the Assign Event command from the context menu of the relevant task.

Interface model A temporary Simulink model that contains blocks from the Model Interface Blockset. ConfigurationDesk initiates the creation of an interface model in Simulink. You can copy the blocks with their identities from the interface model and paste them into an existing Simulink behavior model.

Interpreter A pane that lets you run Python scripts and execute line-based commands.

Inverse model port block A model port block that has the same configuration (same name, same port groups, and port names) but the inverse data direction as the original model port block from which it was created.

IOCNET Abbreviation of I/O carrier network.

A dSPACE proprietary protocol for internal communication in a SCALEXIO system between the real-time processors and I/O units. The IOCNET lets you connect more than 100 I/O nodes and place the parts of your SCALEXIO system long distances apart.

IPDU Abbreviation of interaction layer protocol data unit.

A term according to AUTOSAR. An IPDU contains the communication data that is routed from the interaction layer to a lower communication layer and vice versa. An IPDU can be implemented, for example, as an ISignal IPDU, multiplexed IPDU, or container IPDU.

ISignal A term according to AUTOSAR. A signal of the interaction layer that contains communication data as a coded signal value. To transmit the communication data on a bus, ISignals are instantiated and included in ISignal IPDUs.

ISignal IPDU A term according to AUTOSAR. An IPDU whose communication data is arranged in ISignals. ISignal IPDUs allow the exchange of ISignals between different network nodes.

L

LDF file A LIN description file that describes networks of the LIN bus system according to the LIN standard.

LIN master A member of a LIN communication cluster that is responsible for the timing of LIN bus communication. A LIN master provides one LIN master task and one LIN slave task. The LIN master task transmits frame headers on the bus, and provides LIN schedule tables and LIN collision resolver tables. The LIN slave task transmits frame responses on the bus. A LIN cluster must contain exactly one LIN master.

LIN schedule table A table defined for a LIN master that contains the transmission sequence of frame headers on a LIN bus. For each LIN master, several LIN schedule tables can be defined.

LIN slave A member of a LIN communication cluster that provides only a LIN slave task. The LIN slave task transmits frame responses on the bus when they are triggered by a frame header. The frame header is sent by a LIN master. A LIN cluster can contain several LIN slaves.

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

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

Logical signal An element of a function block that combines all the signal ports which belong together to provide the functionality of the signal. Each logical signal causes one or more channel requests. Channel requests are available after you have assigned a channel set to the logical signal.

Logical signal chain A term that describes the logical path of a signal between an external device and the behavior model. The main elements of the logical signal chain are represented by different graphical blocks (device blocks, function blocks and model port blocks). Every block has ports to provide the mapping to neighboring blocks.

In the documentation, usually the short form 'signal chain' is used instead.

M

MAP file A file that maps symbolic names to physical addresses.

Mapping line A graphical representation of a connection between two ports in the signal chain. You can draw mapping lines in a working view.

MCD file A model communication description file that is used to implement a multimodel application. It lets you add several behavior models that were separated from an overall model to the model topology.

The MCD file contains information on the separated models and information on the connections between them. The file is generated with the Model Separation Setup Block in MATLAB/Simulink. The block resides in the Model Interface Blockset (dsmpblib) from dSPACE.

MDL file A Simulink model file that contains the behavior model. You can add an MDL file to your ConfigurationDesk application.

As of MATLAB® R2012a, the file name extension for the Simulink model file has been changed from MDL to SLX by The MathWorks®.

Message Viewer A pane that displays a history of all error and warning messages that occur during work with ConfigurationDesk.

Model analysis A process that analyzes the model to determine the interface of a behavior model. You can select one of the following commands:

- Analyze Simulink Model (Model Interface Only)

  Analyzes the interface of a behavior model. The model topology of your active ConfigurationDesk application is updated with the properties of the analyzed behavior model.
- Analyze Simulink Model (Including Task Information)

  Analyzes the model interface and the elements of the behavior model that are relevant for the task configuration. The task configuration in ConfigurationDesk is then updated accordingly.

Model Browser A pane that lets you display and access the model topology of an active ConfigurationDesk application. The Model Browser provides access to all the model port blocks available in the behavior models which are linked to a ConfigurationDesk application. The model elements are displayed in a hierarchy, starting with the model roots. Below the model root, all the subsystems containing model port blocks are displayed as well as the associated model port blocks.

Model communication The exchange of signal data between the models within a multimodel application. To set up model communication, you must use a mapping line to connect a data outport (sending model) to a data inport (receiving model). The best way to set up model communication is using the Model Communication Browser.

Model Communication Browser A pane that lets you open and browse working views like the **Signal Chain Browser**, but shows only the Data Outport and Data Inport blocks and the mapping lines between them.

Model Communication Package table A pane that lets you create and configure model communication packages which are used for model communication in multimodel applications.

Model implementation An implementation of a behavior model. It can consist of source code files, precompiled objects or libraries, variable description files and a description of the model's interface. Specific model implementation types are, for example, model implementation containers, such as Functional Mock-up Units or Simulink implementation containers.

Model implementation container A file archive that contains a model implementation. Examples are FMUs, SIC files, and VECU files.

Model interface An interface that connects ConfigurationDesk with a behavior model. This interface is part of the signal chain and is implemented via model port blocks. The model port blocks in ConfigurationDesk can provide the interface to:

- Model port blocks (from the Model Interface Package for Simulink) in a Simulink behavior model. In this case, the model interface is also called ConfigurationDesk model interface to distinguish it from the Simulink model interface available in the Simulink behavior model.
- Different types of model implementations based on code container files, e.g., Simulink implementation containers, Functional Mock-up Units, and V-ECU implementations.

Model Interface Package for Simulink A dSPACE software product that lets you specify the interface of a behavior model that you can directly use in ConfigurationDesk. You can also create a code container file (SIC file) that contains the model code of a Simulink behavior model. The SIC file can be used in ConfigurationDesk and VEOS Player.

Model port An element of a model port block. Model ports provide the interface to the function ports and to other model ports (in multimodel applications).

These are the types of model ports:

- Data inport
- Data outport
- Runnable function port
- Configuration port

Model port block A graphical representation of the ConfigurationDesk model interface in the signal chain. Model port blocks contain model ports that can be mapped to function blocks to provide the data flow between the function blocks in ConfigurationDesk and the behavior model. The model ports can also be mapped to the model ports of other model port blocks with data inports or data outports to set up model communication. Model port blocks are available in different types and can provide different port types:

- Data port blocks with data inports and data outports
- Runnable Function blocks with runnable function ports
- Configuration Port blocks with configuration ports. Configuration Port blocks are created during model analysis for each of the following blocks found in the Simulink behavior model:
  - RTICANMM ControllerSetup block
  - RTILINMM ControllerSetup block
  - FLEXRAYCONFIG UPDATE block

  Configuration Port blocks are also created for bus simulation containers.

Model Separation Setup Block A block that is contained in the Model Interface Package for Simulink. It is used to separate individual models from an overall model in MATLAB/Simulink. Additionally, model separation generates a model communication description file (MCD file) that contains information on the separated models and their connections. You can use this MCD file in ConfigurationDesk.

Model topology A component of a ConfigurationDesk application that contains information on the subsystems and the associated model port blocks of all the behavior models that have been added to a ConfigurationDesk application.

Model-Function Mapping Browser A pane that lets you create and update signal chains for Simulink behavior models. It directly connects them to I/O functionality in ConfigurationDesk.

MTFX file A file containing a model topology when explicitly exported. The file contains information on the interface to the behavior model, such as the implemented model port blocks including their subsystems and where they are used in the model.

Multicore real-time application A real-time application that is executed on several cores of one PU of the real-time hardware.

Multimodel application A real-time application that executes several behavior models in parallel on dSPACE real-time hardware (SCALEXIO or MicroAutoBox III).

Multiplexed IPDU A term according to AUTOSAR. An IPDU that consists of one dynamic part, a selector field, and one optional static part. Multiplexing is used to transport varying ISignal IPDUs via the same bytes of a multiplexed IPDU.

- The dynamic part is one ISignal IPDU that is selected for transmission at run time. Several ISignal IPDUs can be specified as dynamic part alternatives. One of these alternatives is selected for transmission.

- The selector field value indicates which ISignal IPDU is transmitted in the dynamic part during run time. For each selector field value, there is one corresponding ISignal IPDU of the dynamic part alternatives. The selector field value is evaluated by the receiver of the multiplexed IPDU.
- The static part is one ISignal IPDU that is always transmitted.

Multi-PU application Abbreviation of multi-processing-unit application. A multi-PU application is a real-time application that is partitioned into several processing unit applications. Each processing unit application is executed on a separate PU of the real-time hardware. The processing units are connected via IOCNET and can be accessed from the same host PC.

N

Navigation bar An element of ConfigurationDesk's user interface that lets you switch between view sets.

Network node A term that describes the bus communication of an ECU for only one communication cluster.

O

Offline simulation A purely PC-based simulation scenario without a connection to a physical system, i.e., neither simulator hardware nor ECU hardware prototypes are needed. Offline simulations are independent from real time and can run on VEOS.

Offline simulation application An application that runs on VEOS to perform offline simulation. An offline simulation application can be built with the VEOS Player and the resulting OSA file can be downloaded to VEOS.

OSA file An offline simulation application file that is built with the VEOS Player and can be downloaded to VEOS to perform offline simulation.

P

Parent port A port that you can use to map multiple function ports and model ports. All child ports with the same name are mapped. ConfigurationDesk enforces the mapping rules and allows only mapping lines which agree with them.

PDU Abbreviation of protocol data unit.

A term according to AUTOSAR. A PDU transports data (e.g., control information or communication data) via one or multiple network layers according to the AUTOSAR layered architecture. Depending on the involved layers and the function of a PDU, various PDU types can be distinguished, e.g., ISignal IPDUs, multiplexed IPDUs, and NMPDUs.

Physical signal chain A term that describes the electrical wiring of external devices (ECU and loads) to the I/O boards of the real-time hardware. The physical signal chain includes the external cable harness, the pinouts of the connectors and the internal cable harness.

Pins and External Wiring table A pane that lets you access the external wiring information

Platform A dSPACE real-time hardware system that can be registered and displayed in the Platform Manager.

Platform Manager A pane that lets you handle registered hardware platforms. You can download, start, and stop real-time applications via the Platform Manager. You can also update the firmware of your SCALEXIO system or MicroAutoBox III system.

Preconfigured application process An application process that was created via the Create preconfigured application process command. If you use the command, ConfigurationDesk creates new tasks for each runnable function provided by the model which is not assigned to a predefined task. ConfigurationDesk assigns the corresponding runnable function and (for periodic tasks) timer events to the new tasks. The tasks are preconfigured (e.g., DAQ raster name, period).

Processing Resource Assignment table A pane that lets you configure and inspect the processing resources in an executable application. This table is useful especially for multi-processing-unit applications.

Processing unit application A component of an executable application. A processing unit application contains one or more application processes.

Project A container for ConfigurationDesk applications and all project-specific documents. You must define a project or open an existing one to work with ConfigurationDesk. Projects are stored in a project root folder.

Project Manager A pane that provides access to ConfigurationDesk projects and applications and all the files they contain.

Project root folder A folder on your file system to which ConfigurationDesk saves all project-relevant data, such as the applications and documents of a project. Several projects can use the same project root folder. ConfigurationDesk uses the Documents folder as the default project root folder. You can specify further project root folders. Each can be made the default project root folder.

Properties Browser A pane that lets you access the properties of selected elements.

PU Abbreviation of processing unit.

A hardware assembly that consists of a motherboard or a dSPACE processor board, possibly additional interface hardware for connecting I/O boards, and an enclosure, i.e., a SCALEXIO Real-Time PC.

R

Real-time application An application that can be executed in real time on dSPACE real-time hardware. The real-time application is the result of a build process. It can be downloaded to real-time hardware via an RTA file. There are different types of real-time applications:

- Single-core real-time application.
- Multicore real-time application.
- Multi-PU application.

Restbus simulation A simulation method to test real ECUs by connecting them to a simulator that simulates the other ECUs in the communication clusters.

RTA file A real-time application file. An RTA file is an executable object file for processor boards. It is created during the build process. After the build process it can be downloaded to the real-time hardware.

Runnable function A function that is called by a task to compute results. A model implementation provides a runnable function for its base rate task. This runnable function can be executed in a task that is triggered by a timer event. In addition, a Simulink behavior model provides a runnable function for each Hardware-Triggered Runnable Function block contained in the Simulink behavior model. This runnable function is suitable for being executed in an asynchronous task.

Runnable Function block A type of model port block. A Runnable Function block provides a runnable function port that can be mapped to an event port of a function block for modeling an asynchronous task.

Runnable function port An element of a Runnable Function block. The runnable function port can be mapped to an event port of a function block for modeling an asynchronous task.

RX Communication data that is received by a bus member.

S

SCALEXIO system A dSPACE hardware-in-the-loop (HIL) system consisting of one or more real-time industry PCs (PUs), I/O boards, and I/O units. They communicate with each other via the IOCNET. The system simulates the environment to test an ECU. It provides the sensor signals for the ECU, measures the signals of the ECU, and provides the power (battery voltage) for the ECU and a bus interface for restbus simulation.

SDF file A system description file that contains information on the CPU name(s), the corresponding object file(s) to be downloaded and the corresponding variable description file(s). It is created during the build process.

Secured IPDU A term according to AUTOSAR. An IPDU that secures the payload of another PDU (i.e., authentic IPDU) for secure onboard communication (SecOC). The secured IPDU contains the authentication information that is used to secure the authentic IPDU's payload. If the secured IPDU is configured as a cryptographic IPDU, the secured IPDU and the authentic IPDU are mapped to different frames. If the secured IPDU is not configured as a cryptographic IPDU, the authentic IPDU is directly included in the secured IPDU.

SIC file A Simulink implementation container file that contains the model code of a Simulink behavior model. The SIC file can be used in ConfigurationDesk and in VEOS Player.

Signal chain A term used in the documentation as a short form for logical signal chain. Do not confuse it with the physical signal chain.

Signal Chain Browser A pane that lets you open and browse working views such as the Global working view or user-defined working views.

Signal inport A signal port that represents an electrical connection point of a function block which provides signal measurement (= input) functionality.

Signal outport A signal port that represents an electrical connection point of a function block which provides signal generation (= output) functionality.

Signal port An element of a function block that provides the interface to external devices (e.g., ECUs) via device blocks. It represents an electrical connection point of a function block.

Signal reference port A signal port that represents a connection point for the reference potential of inports, outports and bidirectional ports. For example: With differential signals, this is a reference signal, with single-ended signals, it is the ground signal (GND).

Simulink implementation container A container that contains the model code of a Simulink behavior model. A Simulink implementation container is generated from a Simulink behavior model by using the Model Interface Package for Simulink. The file name extension of a Simulink implementation container is SIC.

Simulink model interface The part of the model interface that is available in the connected Simulink behavior model.

Single-core real-time application An executable application that is executed on only one core of the real-time hardware.

Single-PU system Abbreviation of single-processing-unit system.

A system consisting of exactly one PU and the directly connected I/O units and I/O routers.

SLX file A Simulink model file that contains the behavior model. You can add an SLX file to your ConfigurationDesk application.

As of MATLAB® R2012a, the file name extension for the Simulink model file has been changed from MDL to SLX by The MathWorks®.

Software event An event that is activated from within a task to trigger another subordinate task. Consider the following example: A multi-tasking Simulink behavior model has a base rate task with a sample time = 1 ms and a periodic task with a sample time = 4 ms. In this case, the periodic task is triggered on every fourth execution of the base rate task via a software event. Software events are available in ConfigurationDesk after model analysis.

Source Code Editor A Python editor that lets you open and edit Python scripts that you open from or create in a ConfigurationDesk project in a window in the working area. You cannot run a Python script in a **Source Code Editor** window. To run a Python script you can use the **Run Script** command in the Interpreter or on the **Automation** ribbon or the **Run** context menu command in the Project Manager.

Structured data port A hierarchically structured port of a Data Inport block or a Data Outport block. Each structured data port consists of more structured and/or unstructured data ports. The structured data ports can consist of signals with different data types (single, double, int8, int16, int32, int64, uint8, uint16, uint32, uint64, Boolean).

SystemDesk A dSPACE software product for development of distributed automotive electrics/electronics systems according to the AUTOSAR approach.

SystemDesk is able to provide a V-ECU implementation container (as a VECU file) to be used in ConfigurationDesk.

T

Table A type of pane that offers access to a specific subset of elements and properties of the active ConfigurationDesk application in rows and columns.

TargetLink A dSPACE software product for production code generation. It lets you generate highly efficient C code for microcontrollers in electronic control units (ECUs). It also helps you implement control systems that have been modeled graphically in a Simulink/Stateflow diagram on a production ECU.

TargetLink is able to provide a V-ECU implementation container (as a VECU file) or a Simulink implementation container (SIC file) to be used in ConfigurationDesk.

Task A piece of code whose execution is controlled by a real-time operating system (RTOS). A task is usually triggered by an event, and executes one or more runnable functions. In a ConfigurationDesk application, there are predefined tasks that are provided by executable application components. In addition, you can create user-defined tasks that are triggered, for example, by I/O events. Regardless of the type of task, you can configure the tasks by specifying the priority, the DAQ raster name, etc.

Task Configuration table A pane that lets you configure the tasks of an executable application.

Temporary working view A working view that can be used for drafting a signal chain segment, like a notepad.

Timer event A periodic event with a sample rate and an optional offset.

Topology A hierarchy that serves as a repository for creating a signal chain. All the elements of a topology can be used in the signal chain, but not every element needs to be used. You can export each topology from and import it to a ConfigurationDesk application. Therefore a topology can be used in several applications.

A ConfigurationDesk application can contain the following topologies:

- Device topology
- Hardware topology
- Model topology

TRC file A variable description file that contains all variables (signals and parameters) which can be accessed via the experiment software. It is created during the build process.

TX Communication data that is transmitted by a bus member.

U

User function An external function or program that is added to the Automation – User Functions ribbon group for quick and easy access during work with ConfigurationDesk.

V

VECU file A ZIP file that contains a V-ECU implementation. A VECU file can contain data packages for different platforms. VECU files are exported by TargetLink or SystemDesk. You can add a V-ECU implementation based on a VECU file to the model topology in the same way as adding a Simulink model based on an SLX file.

VEOS The dSPACE software product for performing offline simulation. VEOS is a PC-based simulation platform which allows offline simulation independently from real time.

VEOS Player A software running on the host PC for building offline simulation applications. Offline simulation applications can be downloaded to VEOS to perform offline simulation. ConfigurationDesk lets you generate a bus simulation container (BSC file) via the Bus Manager. You can then import the BSC file into the VEOS Player.

View set A specific arrangement of some of ConfigurationDesk's panes. You can switch between view sets by using the navigation bar. ConfigurationDesk provides preconfigured view sets for specific purposes. You can customize existing view sets and create user-defined view sets.

VSET file A file that contains all view sets and their settings from the current ConfigurationDesk installation. A VSET file can be exported and imported via the View Sets page of the Customize dialog.

W

Working area The central area of ConfigurationDesk's user interface.

Working view A view of the signal chain elements (blocks, ports, mappings, etc.) used in the active ConfigurationDesk application. A working view can be opened in the Signal Chain Browser or the Model Communication Browser. ConfigurationDesk provides two default working views: The Global working view and the Temporary working view. In the Working View Manager, you can also create user-defined working views that let you focus on specific signal chain segments according to your requirements.

Working View Manager A pane that lets you manage the working views of the active ConfigurationDesk application. You can use the Working View Manager for creating, renaming, and deleting working views, and also to open a working view in the Signal Chain Browser or the Model Communication Browser.

X

XLSX file A Microsoft Excel™ file format that is used for the following purposes:

- Creating or configuring a device topology outside of ConfigurationDesk.
- Exporting the wiring information for the external cable harness.
- Exporting the configuration data of the currently active ConfigurationDesk application for documentation purposes.