



**E3**  
series

**New Features**  
**2022-23.00**



### **Technical Inquiries**

Please contact our support department!

E-Mail: [e3-support@de.zuken.com](mailto:e3-support@de.zuken.com)

### **Note:**

Zuken is not responsible for any errors which may appear in this documentation. Liability, due to direct and indirect losses resulting from the delivery or use of this documentation, is excluded to the extent permitted by law.

This documentation contains copyrighted information.

All rights, especially those pertaining to the duplication and distribution as well as the translation, are reserved. This documentation, whether wholly or in part, may not be reproduced in any form (photocopy, microfilm, etc.), or processed, duplicated or distributed using an electronic system without Zuken's prior written consent.

### **Contact**

Zuken E3 GmbH  
Lämmerweg 55  
D-89079 Ulm/Einsingen

Phone: +49 7305/9309-0

Fax: +49 7305/9309-99

Web: <http://www.zuken.com>

E-Mail: [e3-info@de.zuken.com](mailto:e3-info@de.zuken.com)

# Contents

<b>E<sup>3</sup>.series Releasenotes</b> .....	<b>1</b>
Update Information of Version 2022.....	2
Windows Support .....	2
Installation .....	2
Databases .....	2
Projects 2	
Multi-User Installation .....	3
ORACLE 3	
Microsoft SQL Server .....	3
Using an Existing Work Environment.....	3
New Features in 2022 Build 23.00 .....	4
Overview - New Features in COM-Interface .....	7
COM-Interface - Enhancements and Changed Behavior of Interfaces and Functions .....	9
Enhancement to <i>e3CavityPart - Create</i> .....	9
Enhancement to <i>e3Connection - CreateConnectionBetweenPoints</i> .....	9
Enhancement to <i>e3DbeApplication - NewModel</i> .....	10
Enhancements to <i>e3Device - CreateInlineConnectorsEx, CreateInlineConnectorsOnConnectionLine, PlaceOnLineSlot, PlugWith</i> and <i>PlugWithMatingPins</i> .....	10
Enhancement to <i>e3Job - GetBomPartList</i> .....	11
Enhancements to <i>e3FunctionalPort - SetPortType</i> and <i>SetUserDefined</i> .....	11
Enhancements to <i>e3Pin - GetEndCavityPartIds, GetTypeId, IsPanelPathLocked, LockPanelPath, PlugWithMatingPin, SetColour, SetColourByDescription, SetColourDescription, SetCrossSection, SetCrossSectionByDescription, SetLength</i> and <i>SetOuterDiameterByDescription</i> .....	11
Customer Request: Enhancement to Trigger <i>AfterWireDisconnected</i> .....	13
Customer Request: Enhancements to <i>e3Device - InsertTerminalPlan, GetTerminalPlanSettings, SetTerminalPlanSettings</i> and <i>e3Job - GetTerminalPlanSettings</i> and <i>SetTerminalPlanSettings</i> .....	13
Customer Request: Enhancement to <i>e3Symbol - PlaceInteractively</i> .....	14
Customer Request: Enhancement to Several Functions in <i>e3CavityPart</i> and <i>e3Pin</i> for Defining Assemblies with Conductor Connections in <i>E<sup>3</sup>.series</i> .....	14
Customer Request: Change Behavior of <i>e3StructureNode - MoveTo</i> .....	15
COM-Interface - New Interfaces and Functions .....	16
New Function <i>e3Connection - CreateConnection</i> .....	16
New Function <i>e3DbeNode - IsBusbarPin</i> .....	17
New Functions <i>e3Device - GetAssignedBusbarPins, GetPhysicalLength</i> and <i>IsBusbar</i> .....	17
New Function <i>e3NetSegment - IsBusbar</i> .....	18
New Functions <i>e3Pin - AssignBusbar, DeAssignBusbar</i> and <i>IsBusbar</i> .....	18
New Function <i>e3Tree - GetSelectedBusbarIds</i> .....	19
New Functions <i>e3Job - GetAllBusbarConnectionIds, GetBusbarIds, GetSelectedBusbarIds</i> and <i>GetTreeSelectedBusbarIds</i> .....	20
Customer Request: New Function <i>e3Job - GetTextTypes</i> .....	21
Customer Request: New Function <i>e3Job - UpdateComponentVersion</i> .....	21
Customer Request: New Function <i>e3Outline - IsThreadedHole</i> .....	22
Customer Request: New Function <i>e3Pin - GetWireHoseTubeStyle</i> .....	22
Customer Request: New Functions <i>e3DbeGraph - GetInsideGraphIds, e3DbeSymbol - GetNodeIDs</i> and <i>e3DbeText - GetPosition</i> .....	23

Customer Request: New Functions *e3Device*, *e3Component* and *e3Symbol- GetStateId, GetStateIds* and *SetStateId*..... 24

Customer Request: New Functions *e3Device - IsPreventedAgainstPhysicalChangesOfCores* and *e3Pin - IsCoreEndLockedPermanent* ..... 26

Customer Request: New Function *e3Group - GetAnyIds* ..... 26

Customer Request: New Functions *GetGID, GetGUID, GetId, SetGID, SetGUID* and *SetId* in Different Interfaces..... 27

Customer Request: New Functions *e3Job - GetInvertDisplayColour* and *SetInvertDisplayColour* ..... 28

Customer Request: New Trigger *AfterModifyBusbarEnds*..... 29

Customer Request: New Interface *e3DbeNode* to Work with Nodes..... 29

Customer Request: New Interface *e3State* to Work with States in eCheck ..... 29

General Changes ..... 31

    E<sup>3</sup>.series is compatible with Windows 11 Pro and will only be released as 64-bit version ..... 31

    Setting for Inverting Displayed Colors ..... 31

    New Icons for Commands and Updated Icons for Objects ..... 32

    Tables Available in the Modules *E<sup>3</sup>.view, E<sup>3</sup>.viewPlus* and *E<sup>3</sup>.redliner* ..... 33

    New Licensing Mechanism ..... 33

Database Editor ..... 35

    Define Assemblies with Conductor Connections ..... 35

    Create and Use Blocks with Connectors with Inserts ..... 37

Importing and Exporting Data ..... 38

    Importing Data ..... 38

    Setting for Merging Attributes ..... 38

Multiuser ..... 41

    Enhanced Log Files ..... 41

    Password Encryption for Multiuser Server ..... 42

Placement / Purge ..... 44

    Setting for Overlapping of Automatically Placed Bundle Symbols ..... 44

Project Handling ..... 46

    Define Drill Holes as Threaded Holes ..... 46

    E<sup>3</sup>.eCheck: Check Fuse Melting Time and Wire Ignition Time and Starting E<sup>3</sup>.series without an eCheck License 49

    Enhanced Behavior for Updating Components ..... 51

    Do Not Show Deleted Objects in Graphical Comparison of Project States ..... 51

    New Text Type for Segments ..... 54

    Combine Text Types ..... 55

Panel (also: 3D) ..... 57

    3D Display of Individual Panel Sheets ..... 57

    Setting for Automatic Connections in Panels..... 57

    Mark Cable Duct Inlets and Outlets in the 3D View ..... 59

    Correction Factor for Space Requirement of Cables in Cable Ducts..... 61

Tables 63

    Display Subcircuits in the Component Table ..... 63

Connection/Busses, Signals/Logic Lines, Supply ..... 66

    Setting for Plugging Pins with Same Name to One Another ..... 66

    Display Jumped Terminals on Panel Sheets ..... 67

    New Object "Busbar" ..... 69

    Attribute Text Templates for Busbars ..... 69

Connect devices with busbars in the panel .....	70
How busbars are displayed .....	71
Settings for Exporting Busbars .....	72
Settings for Busbar Connection Lines .....	73
New Attribute Owner.....	74
New Connection Type.....	75
Busbar Signals .....	75
Create or Change Busbars in the Database .....	75
Create and delete busbar connections .....	76
Assign busbar to busbar connections.....	77
Place and connect busbars.....	78
Rename Signal Names via the Signal Tree .....	81

# **E<sup>3</sup>.series Releasenotes**

Date: 7/20/2022

This manual describes the following functions of version:

2022 **Build** 23.00

## **!! Please Note when Switching to New Version !!**

Projects and databases are converted when opening them with this new **E<sup>3</sup>.series** Release. Please consider that these projects cannot be edited with an older version afterward. That's why we recommend to backup projects before.

### **Note:**

- o The latest version of the license server must be installed when using a FlexNet license server.
- o Starting with Microsoft Office Version 2016, the TrueType font **,Arial Unicode MS'** is no longer included in the standard delivery of Microsoft Office due to financial reasons.

This font has been and is still in use for **E<sup>3</sup>.series**, **E<sup>3</sup>** template projects and **E<sup>3</sup>** example projects. You have most likely used this font to documents for **E<sup>3</sup>** projects. Presently, Zuken is not aware of any comparable functionality that is free-of-charge for **,Arial Unicode MS'**.

To ensure that your new and older projects and the documents generated from them are displayed correctly on devices without older MS Office versions, we recommend you purchase and install the original font **,Arial Unicode MS Regular'** on your devices. Furthermore, we recommend you change your templates to another font style so that new projects no longer use this font.

- **New Features**

**See also:**

## Update Information of Version 2022

- o [Windows Support](#)
- o [Installation](#)
- o [Databases](#)
- o [Projects](#)
- o [Multi-User Installation](#)
- o [ORACLE](#)
- o [Microsoft SQL Server](#)
- o [Using an Existing Work Environment](#)

### Windows Support

The following Windows systems are supported with **E3.series** 2022:

- Windows 10
- Windows 11 Pro
- Windows Server 2008/2012/2016/2019

For more information please refer to the installation description.

### Installation

Before installing, we recommend you make a backup of your current installation (at least the data and databases).

The installation of **E3.series** 2022 is started using

**SETUP.EXE**

which can be found on the DVD's main directory.

The Version can be installed in addition to an existing **E3.series** Version.

The Default Directory is: `\Program Files\Zuken\E3.series_<version>`

Therefore, this version can also be used alongside existing versions of **E3.series**. However, note that you cannot use your existing database(s), or the existing projects, in both versions, as they are updated when opened in **E3.series** 2022, so copies should be used for **E3.series** 2022.

### Databases

With this new version, a new default database is installed. Existing databases are not overwritten.

The new database contains new components and symbols.

### Projects

Existing projects are automatically converted upon opening with the new version. So long as the file has not been saved in the new version, it can still be opened with an older version. After saving a project with **E3.series** Version 2022, a project can no longer be opened with an earlier version.

## Multi-User Installation

Existing multi-user projects from Version 2021 can directly be opened with Version 2022. They are automatically converted upon opening. However, this cannot be undone, and the projects can no longer be opened with Version 2021.

That's why we recommend installing Version 2022 multi-user projects in parallel to the existing version, so that it's also possible to still access these project with **E3.series** Version 2021.

To transfer the projects to Version 2022, these projects must be saved in Version 2021 as Single-User projects. They can then be stored in the multi-user environment of Version 2022.

In order to run the Multi-User databases in parallel

- a new SQL Server process must be installed for Microsoft SQL Server or another server must be installed
- a new database user account for ORACLE must be used

The E3 Server process must then access the new database or the new database user account.

## ORACLE

When setting up the **E3**-MU-Server, a new name must be specified for the Multi-User Administrator.

Thus a new database user account is created and both **E3.series** versions can work simultaneously in the multi-user environment.

## Microsoft SQL Server

First, a new instance of the SQL Server processes must be installed.

Enter a new instance name, which must be used with the set-up installation of the **E3** MU-Server.

The rest of the installation continues as described in the separate **MuSetup** installation description.

## Using an Existing Work Environment

Due to the new software component for the user interface, adjustments to the work environment (arrangement of windows, structure of toolbars, own programs added, hidden commands, assignment of keys to commands, ...) cannot be adopted by versions before 2020.

These assignments must be recreated and saved in the working environment.



## New Features in 2022 Build 23.00

The following functionality is new in Version 2022 Build 23.00:

- [Overview - New Features in COM-Interface](#)
  - [COM-Interface - Enhancements and Changed Behavior of Interfaces and Functions](#)
    - **Enhancement to e3CavityPart - Create**
    - **Enhancement to e3Connection - CreateConnectionBetweenPoints**
    - **Enhancement to e3DbeApplication - NewModel**
    - **Enhancements to e3Device - CreateInlineConnectorsEx, CreateInlineConnectorsOnConnectionLine, PlaceOnLineSlot, PlugWith and PlugWithMatingPins**
    - **Enhancement to e3Job - GetBomPartList**
    - **Enhancements to e3FunctionalPort - SetPortType and SetUserDefined**
    - **Enhancements to e3Pin - GetEndCavityPartIds, GetTypeId, IsPanelPathLocked, LockPanelPath, PlugWithMatingPin, SetColour, SetColourByDescription, SetColourDescription, SetCrossSection, SetCrossSectionByDescription, SetLength and SetOuterDiameterByDescription**
    - **Customer Request: Enhancement to Trigger AfterWireDisconnected**
    - **Customer Request: Enhancements to e3Device - InsertTerminalPlan, GetTerminalPlanSettings, SetTerminalPlanSettings and e3Job - GetTerminalPlanSettings and SetTerminalPlanSettings**
    - **Customer Request: Enhancement to e3Symbol - PlaceInteractively**
    - **Customer Request: Enhancement to Several Functions in e3CavityPart and e3Pin for Defining Assemblies with Conductor Connections in E3.series**
    - **Customer Request: Change Behavior of e3StructureNode - MoveTo**
  - [COM-Interface - New Interfaces and Functions](#)
    - **New Function e3Connection - CreateConnection**
    - **New Function e3DbeNode - IsBusbarPin**
    - **New Functions e3Device - GetAssignedBusbarPins, GetPhysicalLength and IsBusbar**
    - **New Function e3NetSegment - IsBusbar**
    - **New Functions e3Pin - AssignBusbar, DeAssignBusbar and IsBusbar**
    - **New Function e3Tree - GetSelectedBusbarIds**
    - **New Functions e3Job - GetAllBusbarConnectionIds, GetBusbarIds, GetSelectedBusbarIds and GetTreeSelectedBusbarIds**
    - **Customer Request: New Function e3Job - GetTextTypes**
    - **Customer Request: New Function e3Job - UpdateComponentVersion**
    - **Customer Request: New Function e3Outline - IsThreadedHole**
    - **Customer Request: New Function e3Pin - GetWireHoseTubeStyle**
    - **Customer Request: New Functions e3DbeGraph - GetInsideGraphIds, e3DbeSymbol - GetNodeIDs and e3DbeText - GetPosition**
    - **Customer Request: New Functions e3Device, e3Component and e3Symbol- GetStateId, GetStateIds and SetStateId**

- **Customer Request: New Functions e3Device - IsPreventedAgainstPhysicalChangesOfCores and e3Pin - IsCoreEndLockedPermanent**
  - **Customer Request: New Function e3Group - GetAnyIds**
  - **Customer Request: New Functions GetGID, GetGUID, GetId, SetGID, SetGUID and SetId in Different Interfaces**
  - **Customer Request: New Functions e3Job - GetInvertDisplayColour and SetInvertDisplayColour**
  - **Customer Request: New Trigger AfterModifyBusbarEnds**
  - **Customer Request: New Interface e3DbNode to Work with Nodes**
  - **Customer Request: New Interface e3State to Work with States in eCheck**
- 
- **General Changes**
    - **E3.series is compatible with Windows 11 Pro and will only be released as 64-bit version**
    - **Setting for Inverting Displayed Colors**
    - **New Icons for Commands and Updated Icons for Objects**
    - **Tables Available in the Modules E3.view, E3.viewPlus and E3.redliner**
    - **New Licensing Mechanism**
- 
- **Database Editor**
    - **Define Assemblies with Conductor Connections**
    - **Create and Use Blocks with Connectors with Inserts**
- 
- **Importing and Exporting Data**
    - **Setting for Merging Attributes**
- 
- **Multiusers**
    - **Enhanced Log Files**
    - **Password Encryption for Multiusers Server**
- 
- **Placement / Purge**
    - **Setting for Overlapping of Automatically Placed Bundle Symbols**
- 
- **Project Handling**
    - **Define Drill Holes as Threaded Holes**

- **E3.eCheck: Check Fuse Melting Time and Wire Ignition Time and Starting E3.series without an eCheck License**
- **Enhanced Behavior for Updating Components**
- **Do Not Show Deleted Objects in Graphical Comparison of Project States**
- **New Text Type for Segments**
- **Combine Text Types**
  
- **Panel (also: 3D)**
  - **3D Display of Individual Panel Sheets**
  - **Setting for Automatic Connections in Panels**
  - **Mark Cable Duct Inlets and Outlets in the 3D View**
  - **Correction Factor for Space Requirement of Cables in Cable Ducts**
  
- **Tables**
  - **Display Subcircuits in the Component Table**
  
- **Connection/Busses, Signals/Logic Lines, Supply**
  - **Setting for Plugging Pins with Same Name to One Another**
  - **Display Jumped Terminals on Panel Sheets**
  - **New Object "Busbar"**
  - **Rename Signal Names via the Signal Tree**

## Overview - New Features in COM-Interface

The following new features are available in the COM Interface and described in the individual chapters:

### COM-Interface - Enhancements and Changed Behavior of Interfaces and Functions

- **Enhancement to e3CavityPart - Create**
- **Enhancement to e3Connection - CreateConnectionBetweenPoints**
- **Enhancement to e3DbeApplication - NewModel**
- **Enhancements to e3Device - CreateInlineConnectorsEx, CreateInlineConnectorsOnConnectionLine, PlaceOnLineSlot, PlugWith and PlugWithMatingPins**
- **Enhancement to e3Job - GetBomPartList**
- **Enhancements to e3FunctionalPort - SetPortType and SetUserDefined**
- **Enhancements to e3Pin - GetEndCavityPartIds, GetTypeId, IsPanelPathLocked, LockPanelPath, PlugWithMatingPin, SetColour, SetColourByDescription, SetColourDescription, SetCrossSection, SetCrossSectionByDescription, SetLength and SetOuterDiameterByDescription**
- **Customer Request: Enhancement to Trigger AfterWireDisconnected**
- **Customer Request: Enhancements to e3Device - InsertTerminalPlan, GetTerminalPlanSettings, SetTerminalPlanSettings and e3Job - GetTerminalPlanSettings and SetTerminalPlanSettings**
- **Customer Request: Enhancement to e3Symbol - PlaceInteractively**
- **Customer Request: Enhancement to Several Functions in e3CavityPart and e3Pin for Defining Assemblies with Conductor Connections in E3.series**
- **Customer Request: Change Behavior of e3StructureNode - MoveTo**

### COM-Interface - New Interfaces and Functions

- **New Function e3Connection - CreateConnection**
- **New Function e3DbeNode - IsBusbarPin**
- **New Functions e3Device - GetAssignedBusbarPins, GetPhysicalLength and IsBusbar**
- **New Function e3NetSegment - IsBusbar**
- **New Functions e3Pin - AssignBusbar, DeAssignBusbar and IsBusbar**
- **New Function e3Tree - GetSelectedBusbarIds**
- **New Functions e3Job - GetAllBusbarConnectionIds, GetBusbarIds, GetSelectedBusbarIds and GetTreeSelectedBusbarIds**
- **Customer Request: New Function e3Job - GetTextTypes**
- **Customer Request: New Function e3Job - UpdateComponentVersion**
- **Customer Request: New Function e3Outline - IsThreadedHole**
- **Customer Request: New Function e3Pin - GetWireHoseTubeStyle**
- **Customer Request: New Functions e3DbeGraph - GetInsideGraphIds, e3DbeSymbol - GetNodeIDs and e3DbeText - GetPosition**
- **Customer Request: New Functions e3Device, e3Component and e3Symbol- GetStateId, GetStateIds and SetStateId**
- **Customer Request: New Functions e3Device - IsPreventedAgainstPhysicalChangesOfCores and e3Pin - IsCoreEndLockedPermanent**
- **Customer Request: New Function e3Group - GetAnyIds**
- **Customer Request: New Functions GetGID, GetGUID, GetId, SetGID, SetGUID and SetId in Different Interfaces**
- **Customer Request: New Functions e3Job - GetInvertDisplayColour and SetInvertDisplayColour**

- **Customer Request: New Trigger AfterModifyBusbarEnds**
- **Customer Request: New Interface e3DbNode to Work with Nodes**
- **Customer Request: New Interface e3State to Work with States in eCheck**

## COM-Interface - Enhancements and Changed Behavior of Interfaces and Functions

This chapter provides an overview of which interfaces or functions have been enhanced in **E3.series** 2022 or had their behavior changed.

All enhancements that could be assigned to a customer request, which was documented by the **E3.series** Support in the form of a work item, are listed separately at the end of the chapter.

For detailed documentation, refer to the Help integrated in **E3.series**:

- **Enhancement to e3CavityPart - Create**
- **Enhancement to e3Connection - CreateConnectionBetweenPoints**
- **Enhancement to e3DbeApplication - NewModel**
- **Enhancements to e3Device - CreateInlineConnectorsEx, CreateInlineConnectorsOnConnectionLine, PlaceOnLineSlot, PlugWith and PlugWithMatingPins**
- **Enhancement to e3Job - GetBomPartList**
- **Enhancements to e3FunctionalPort - SetPortType and SetUserDefined**
- **Enhancements to e3Pin - GetEndCavityPartIds, GetTypeId, IsPanelPathLocked, LockPanelPath, PlugWithMatingPin, SetColour, SetColourByDescription, SetColourDescription, SetCrossSection, SetCrossSectionByDescription, SetLength and SetOuterDiameterByDescription**
- **Customer Request: Enhancement to Trigger AfterWireDisconnected**
- **Customer Request: Enhancements to e3Device - InsertTerminalPlan, GetTerminalPlanSettings, SetTerminalPlanSettings and e3Job - GetTerminalPlanSettings and SetTerminalPlanSettings**
- **Customer Request: Enhancement to e3Symbol - PlaceInteractively**
- **Customer Request: Enhancement to Several Functions in e3CavityPart and e3Pin for Defining Assemblies with Conductor Connections in E3.series**
- **Customer Request: Change Behavior of e3StructureNode - MoveTo**

### Enhancement to e3CavityPart - Create

The return values of the following function have been supplemented in the **e3CavityPart** interface:

**return = e3CavityPart.Create ( pinid, name, type )**

- o The return value -7 is new and means that an error occurred when calling the function. The return value for **pinid** indicates that it is a pin of a busbar. Connector pin terminals cannot be defined for pins on busbars.

### Enhancement to e3Connection - CreateConnectionBetweenPoints

The return values of the following function have been supplemented in the **e3Connection** interface:

**return = e3Connection.CreateInlineConnectorsEx ( shti, x1, y1, x2, y2, flags)**

- o The return value -11 is new and means that an error occurred when calling the function. The return value indicates that the busbar cannot be connected to another object that is not a type of **busbar**.
- o The return value -12 is new and means that an error occurred when calling the function. The return value indicates that the two busbars cannot be connected to each other.

### Enhancement to *e3DbeApplication* - *NewModel*

The behavior of the following function has been enhanced in the *e3DbeApplication* interface:

**return = e3DbeApplication.NewModel ( name, baseName, flags )**

- o The list of available values for the parameter **flags** has been extended to define the model type of the model to be created.  
The following values are possible:
  - 0**: The model is created with the model type *Device*.
  - 1**: The model is created with the model type *Mount*.
  - 2**: The model is created with the model type *Cable Duct*.
  - 4**: The model is created with the model type *Busbar*.
- o The return value -1 is new and means that an error occurred when calling the function.  
The return value indicates that the value for **flags** is invalid.

### Enhancements to *e3Device* - *CreateInlineConnectorsEx*, *CreateInlineConnectorsOnConnectionLine*, *PlaceOnLineSlot*, *PlugWith* and *PlugWithMatingPins*

The return values of the following functions have been supplemented in the *e3Device* interface:

**return = e3Device.CreateInlineConnectorsEx ( newCoreIDs, newDeviceIDs, flags, fromPinIDs, toPinIDs, viewNumbers, fBSheetIDs, compName, compVersion, newSymbolIDs )**

- o The return value -8 is new and means that an error occurred when calling the function.  
The return value indicates that one of the specified pins is the pin of a busbar.

**return = e3Device.CreateInlineConnectorsOnConnectionLine ( newCoreIDs, newDeviceIDs, flags, LineIDs, compName, compVersion, newSymbolIDs )**

- o The return value -14 is new and means that an error occurred when calling the function.  
The return value indicates that one of the specified connection lines is a connection line of a busbar.

**return = e3Device.PlaceOnLineSlot ( slotid, x, rotation, combined, collisionids )**

- o The return value -10 is new for the collision description.  
The return value indicates that the device cannot be placed because it collides with a pin involved in another electrical connection.
- o The return value -11 is new for the collision description.  
The return value indicates that the device cannot be placed because pins that would have to be electrically connected during placement have different user-defined signals.

**return = e3Device.PlugWith ( id )**

- o The return value -15 is new and means that an error occurred when calling the function.  
The return value indicates that one of the pins is the pin of a busbar.
- o The return value -16 is new and means that an error occurred when calling the function.  
The return value indicates that the setting *Deny plugging pins with different pin names* is active and two pins with different names cannot be connected.

**return = e3Device.PlugWithMatingPins ( deviceIds )**

- o The return value -15 is new and means that an error occurred when calling the function. The return value indicates that one of the pins is the pin of a busbar.

**Enhancement to e3Job - GetBomPartList**

The behavior of the following function has been enhanced in the **e3Job** interface:

**return = e3Job.GetBomPartList ( consumer, outputFormatVersion, flags, keyAttribut, quantityAttribut, lengthAttribut, additionalAttributes, result )**

- o The parameter **flags** supports the new value **0x0080**.  
The value can be used in order to determine all busbars in the project.
- o In the output file generated by GetBomPartList there is a new value **0x00000004** displayed in the column **BomType2**.  
The value indicates that the object is a busbar.
- o The parameter **outputFormatVersion** can be called with the new value **4.1**.  
The following changes have been made to the output format version:
  - o The values for conductor/wire lengths are output exactly in the format in which they are stored.  
This means that integer values specified as a decimal number in the format **n.0** are also output exactly in this format.
  - o A decimal point (.) is used as a separator for comma numbers.

**Enhancements to e3FunctionalPort - SetPortType and SetUserDefined**

The behavior of the following functions has been enhanced in the **e3FunctionalPort** interface:

**return = e3FunctionalPort.SetPortType ( type )**

- o The return value -4 is new and means that an error occurred when calling the function. The return value indicates that the port type of the functional port cannot be changed because the pin of the functional port is assigned to a busbar.

**return = e3FunctionalPort.SetUserDefined ( type )**

- o The return value -3 is new and means that an error occurred when calling the function. The return value indicates that the property **User-defined connection** of the functional port cannot be changed because the pin of the functional port is assigned to a busbar.

**Enhancements to e3Pin - GetEndCavityPartIds, GetTypeId, IsPanelPathLocked, LockPanelPath, PlugWithMatingPin, SetColour, SetColourByDescription, SetColourDescription, SetCrossSection, SetCrossSectionByDescription, SetLength and SetOuterDiameterByDescription**

The behavior of the following function has been enhanced in the **e3Pin** interface:

**return = e3Pin.GetEndCavityPartIds ( which, cavities, types )**



- o The return value -3 is new and means that an error occurred when calling the function. The return value indicates that the selected object is a busbar and therefore has no cavity part.

The behavior of the following function has been enhanced in the **e3Pin** interface:

**return = e3Pin.GetTypeId ( )**

- o The return value 18 is new and means the function was successfully called. The return value indicates that the pin is of the type **busbar**.

**return = e3Pin.IsPanelPathLocked ( )**

- o The return value -2 is new and means that an error occurred when calling the function. The return value indicates that the function is not compatible with objects of the type **busbar** and the wire belongs to a busbar.

**return = e3Pin.LockPanelPath ( lock )**

- o The return value -2 is new and means that an error occurred when calling the function. The return value indicates that the function is not compatible with objects of the type **busbar** and the wire belongs to a busbar.

**return = e3Pin.PlugWithMatingPin ( )**

- o The return value -10 is new and means that an error occurred when calling the function. The return value indicates that one of the pins is the pin of a busbar and therefore the pins are not compatible.

**return = e3Pin.SetColour ( color )**

- o The return value -4 is new and means that an error occurred when calling the function. The return value indicates that the selected object is a busbar and the color cannot be changed.

**return = e3Pin.SetColourByDescription ( color )**

- o The return value -4 is new and means that an error occurred when calling the function. The return value indicates that the selected object is a busbar and the color based on the description cannot be changed.

**return = e3Pin.SetColourDescription ( color )**

- o The return value -4 is new and means that an error occurred when calling the function. The return value indicates that the selected object is a busbar and the description of the color cannot be changed.

**return = e3Pin.SetCrossSection ( crossec )**

- o The return value -5 is new and means that an error occurred when calling the function. The return value indicates that the selected object is a busbar and the cross-section cannot be changed.

**return = e3Pin.SetCrossSectionByDescription ( description )**

- o The return value -5 is new and means that an error occurred when calling the function. The return value indicates that the selected object is a busbar and the cross-section based on the description cannot be changed.

**return = e3Pin.SetLength ( length )**

- o The return value -4 is new and means that an error occurred when calling the function. The return value indicates that the selected object is a busbar and the length of the busbar cannot be changed.

**return = e3Pin.SetOuterDiameterByDescription ( description )**

- o The return value -6 is new and means that an error occurred when calling the function. The return value indicates that the selected object is a busbar and the outer diameter cannot be changed.

### **Customer Request: Enhancement to Trigger *AfterWireDisconnected***

The following arguments have been added to the trigger ***AfterWireDisconnected*** to indicate when cables, wires or conductors are generated from a connection:

- o **wireNumber**: The argument specifies the identifier of a cable, a conductor or a wire.
- o **endPtr1**: The argument specifies the identifier of the first end of a conductor or a wire.
- o **endPtr2**: The argument specifies the identifier of the second end of a conductor or a wire.

**Reference:** Designer-28497

### **Customer Request: Enhancements to *e3Device - InsertTerminalPlan, GetTerminalPlanSettings, SetTerminalPlanSettings* and *e3Job - GetTerminalPlanSettings* and *SetTerminalPlanSettings***

The parameter ***ShowAllEquivalentPins*** has been added to the following functions in the ***e3Device*** interface:

**return = e3Device.InsertTerminalPlan ( parameters )**

**return = e3Device.GetTerminalPlanSettings ( parameters )**

**return = e3Device.SetTerminalPlanSettings ( parameters )**

The parameter ***ShowAllEquivalentPins*** has been added to the following functions in the ***e3Job*** interface:

**return = e3Job.GetTerminalPlanSettings ( parameters )**

**return = e3Job.SetTerminalPlanSettings ( parameters )**

The new parameter ***ShowAllEquivalentPins*** can be used to set whether equivalent pins are also displayed in terminal plans.

- o If the parameter has the value **True** or **1**, equivalent pins are displayed.  
If the parameter has the value **False** or **0**, equivalent pins are not displayed.

**Reference:** Designer-37909

### **Customer Request: Enhancement to e3Symbol - PlaceInteractively**

The behavior of the following function has been enhanced in the **e3SymbolDevice** interface:

**return = e3Symbol.PlaceInteractively ( )**

- o It is now possible to move placed symbols to another position interactively with **PlaceInteractively ( )**.  
To do so, call up the function with the selected symbol and click on the position to which the symbol is to be moved.

**Reference:** Designer-39943

### **Customer Request: Enhancement to Several Functions in e3CavityPart and e3Pin for Defining Assemblies with Conductor Connections in E3.series**

The return values of the following functions have been supplemented in the **e3CavityPart** interface:

**return = e3CavityPart.Create ( pinid, name, type )**

- o The return value -6 is new and means that an error occurred when calling the function. The return value indicates that unauthorized changes would be made by the function to a conductor/wire of the pin.

**return = e3CavityPart.SetDisableAutomaticSelection ( DisableAutomaticSelection )**

- o The return value -5 is new and means that an error occurred when calling the function. The return value indicates that unauthorized changes would be made by the function to a conductor/wire of the pin.

**return = e3CavityPart.SetValue ( value )**

- o The return value -6 is new and means that an error occurred when calling the function. The return value indicates that unauthorized changes would be made by the function to a conductor/wire of the pin.

The return values of the following functions have been supplemented in the **e3Pin** interface:

**return = e3Pin.LockCoreEnd ( which, lock )**

- o The return value -2 is new and means that an error occurred when calling the function. The return value indicates that the conductor/wire is permanently locked.

**return = e3Pin.SetEndPinId ( which, pini )**

- o The return value -9 is new and means that an error occurred when calling the function. The return value indicates that the conductor/wire is locked.
- o The return value -10 is new and means that an error occurred when calling the function. The return value indicates that unauthorized changes would be made by the function to a conductor/wire of the pin.

**References:** Designer-08203 and Designer-38679

### **Customer Request: Change Behavior of e3StructureNode - MoveTo**

The behavior of the following function has been changed in the **e3StructureNode** interface:

**return = e3StructureNode.MoveTo ( parentId, afterId )**

- o The parameters of the functions have been renamed from **destID** to **parentId** and from **before** to **afterId**.  
**parentId** corresponds to the structure node into which the selected structure node is moved.  
**afterId** determines the position within **parentId** where the selected structure node will be moved to.
- o The values of the parameter **afterId** have a new meaning:  
If **afterId** has the value 0, the structure node is moved to the last position of **parentId**.  
If **afterId** has the value 1, the structure node is moved to the first position of **parentId**.  
All other values are interpreted as item number of **parentId**. The structure node is moved behind the corresponding position of the item number.
- o The return value -5 is new and means that an error occurred when calling the function. The return value indicates that **parentId** or **afterId** has no valid value.

**Reference:** Designer-39513

## COM-Interface - New Interfaces and Functions

This chapter provides an overview of which interfaces or functions that are new in **E<sup>3</sup>.series** 2022.

All enhancements that could be assigned to a customer request, which was documented by **E<sup>3</sup>.series** support in the form of a work item, are listed separately at the end of the chapter.

For detailed documentation, refer to the Help integrated in **E<sup>3</sup>.series**:

- **New Function e3Connection - CreateConnection**
- **New Function e3DbeNode - IsBusbarPin**
- **New Functions e3Device - GetAssignedBusbarPins, GetPhysicalLength and IsBusbar**
- **New Function e3NetSegment - IsBusbar**
- **New Functions e3Pin - AssignBusbar, DeAssignBusbar and IsBusbar**
- **New Function e3Tree - GetSelectedBusbarIds**
- **New Functions e3Job - GetAllBusbarConnectionIds, GetBusbarIds, GetSelectedBusbarIds and GetTreeSelectedBusbarIds**
- **Customer Request: New Function e3Job - GetTextTypes**
- **Customer Request: New Function e3Job - UpdateComponentVersion**
- **Customer Request: New Function e3Outline - IsThreadedHole**
- **Customer Request: New Function e3Pin - GetWireHoseTubeStyle**
- **Customer Request: New Functions e3DbeGraph - GetInsideGraphIds, e3DbeSymbol - GetNodeIDs and e3DbeText - GetPosition**
- **Customer Request: New Functions e3Device, e3Component and e3Symbol- GetStateId, GetStateIds and SetStateId**
- **Customer Request: New Functions e3Device - IsPreventedAgainstPhysicalChangesOfCores and e3Pin - IsCoreEndLockedPermanent**
- **Customer Request: New Function e3Group - GetAnyIds**
- **Customer Request: New Functions GetGID, GetGUID, GetId, SetGID, SetGUID and SetId in Different Interfaces**
- **Customer Request: New Functions e3Job - GetInvertDisplayColour and SetInvertDisplayColour**
- **Customer Request: New Trigger AfterModifyBusbarEnds**
- **Customer Request: New Interface e3DbeNode to Work with Nodes**
- **Customer Request: New Interface e3State to Work with States in eCheck**

### New Function *e3Connection - CreateConnection*

The following function has been added in the **e3Connection** interface to create connections:

```
return = e3Connection.CreateConnection ( flags, shti, pnts, x, y, PointTypArr )
```

#### Parameter:

- o **flags**: Specifies the type of connection to be created.  
The following values are possible:  
**0x0000**: Creates a standard connection.  
**0x0001**: Creates a bus connection.  
**0x0002**: Creates a connection for a formboard connection.  
**0x0004**: Creates a busbar connection.
- o **shti**: Specifies the ID of a sheet.
- o **pnts**: Specifies with how many deviation points the connection is created.
- o **x**: Specifies the x-coordinates of the deviation points.
- o **y**: Specifies the y-coordinates of the deviation points.

- o **PointTypArr**: An optional parameter with which the function can be enhanced.  
If the parameter has the value **0**, the points of the connection form a straight line.  
If the parameter has the value **1**, the points of the connection form a curve.  
If the parameter is not specified, the points of the connection form a straight line.

**Return Values:**

- o **return > 0**: Successfully called the function - the number of connections created is output.
- o **return -1**: Failed calling the function - no project opened.
- o **return -2**: Failed calling the function - the value for the parameter **flags** is not supported.
- o **return -3**: Failed calling the function - the logic connection cannot be created because **E<sup>3</sup>** was not started with the **logic** module.
- o **return -4**: Failed calling the function - the value for the parameter **shti** is not a valid ID for a sheet in the project.
- o **return -5**: Failed calling the function - the selected sheet is locked.
- o **return -6**: Failed calling the function - the formboard connection cannot be created because no valid license for **E<sup>3</sup>.formboard** was found.
- o **return -7**: Failed calling the function - the connection cannot be created because no valid license for **E<sup>3</sup>.functionaldesign** was found.
- o **return -8**: Failed calling the function - the topology connection cannot be created because no valid license for **E<sup>3</sup>.topology** was found.
- o **return -9**: Failed calling the function - the value for the parameter **pnts** is greater than the length of the arrays **x** and **y**.
- o **return -10**: Failed calling the function - the value for the parameter **PointTypArr** is less than the value for the parameter **pnts**.
- o **return -11**: Failed calling the function - the connection cannot be created.

**New Function e3DbeNode - IsBusbarPin**

The following function has been added to the **e3DbeNode** interface to check whether the node is a busbar pin:

**return = e3DbeNode.IsBusbarPin ( )**

**Return Values:**

- o **return 1**: Successfully called the function - the node is a busbar pin.
- o **return 0**: Successfully called the function - the node is not a busbar pin.
- o **return -1**: Failed calling the function - no project opened.
- o **return -2**: Failed calling the function - no valid node ID.

**New Functions e3Device - GetAssignedBusbarPins, GetPhysicalLength and IsBusbar**

The following function has been added in the **e3Device** interface to query the IDs of pins assigned to the selected busbar:

**return = e3Device.GetAssignedBusbarPins ( flags )**

**Parameter:**

- o **flags**: An optional parameter with which the function can be enhanced. Presently, the parameter does not support any values.

**Return Values:**

- o **return >= 0**: Successfully called the function - specifies the number of pins assigned to the selected busbar.
- o **return -1**: Failed calling the function - no project opened or no device selected.
- o **return -2**: Failed calling the function - the selected device is not a busbar.

The following function has been added in the **e3Device** interface to query the length of the selected busbar:

**return = e3Device.GetPhysicalLength ( )**

**Return Values:**

- o **return > 0.0**: Successfully called the function - specifies the length of a device of the type **busbar**, **mounting rail** or **cable duct**.  
The unit of measurement for the length corresponds with the unit set in the project.
- o **return = 0.0**: Inconclusive - the device has no length or an error occurred.

The following function has been added in the **e3Device** interface to check whether a device is a busbar:

**return = e3Device.IsBusbar ( )**

**Return Values:**

- o **return 1**: Successfully called the function - the device is a busbar.
- o **return 0**: Successfully called the function - the device is not a busbar.
- o **return -1**: Failed calling the function - no device IDs found or no project opened.

**New Function e3NetSegment - IsBusbar**

The following function has been added in the **e3NetSegment** interface to check whether a net segment is a busbar:

**return = e3NetSegment.IsBusbar ( )**

**Return Values:**

- o **return 1**: Successfully called the function - the net segment is a busbar.
- o **return 0**: Successfully called the function - the net segment is not a busbar.
- o **return -1**: Failed calling the function - no net segment IDs found or no project opened.

**New Functions e3Pin - AssignBusbar, DeAssignBusbar and IsBusbar**

The following function has been added in the **e3Pin** interface to assign busbars to a net:

**return = e3Pin.AssignBusbar ( ids, flags )**

**Parameter:**

- o **ids**: Specifies IDs of objects used in a net.
- o **flags**: An optional parameter with which the function can be enhanced. Presently, the parameter does not support any values.

**Return Values:**

- o **return 1**: Successfully called the function - **ids** was assigned to the busbar.
- o **return -1**: Failed calling the function - no valid ID of an **e3Pin** object found or no project opened.
- o **return -2**: Failed calling the function - selected **e3Pin** object is not a busbar.
- o **return -3**: Failed calling the function - busbar is locked.
- o **return -4**: Failed calling the function - no valid object selected.
- o **return -5**: Failed calling the function - selected object is locked.
- o **return -6**: Failed calling the function - selected object cannot be assigned to the busbar.

The following function has been added in the **e3Pin** interface to unassign busbars from a net:

**return = e3Pin.DeAssignBusbar ( flags )**

**Parameter:**

- o **flags**: An optional parameter with which the function can be enhanced. Presently, the parameter does not support any values.

**Return Values:**

- o **return 1**: Successfully called the function - assignment of **ids** to the busbar has been removed.
- o **return 0**: Successfully called the function - **ids** is not assigned to any busbars.
- o **return -1**: Failed calling the function - no valid ID of an **e3Pin** object found or no project opened.
- o **return -2**: Failed calling the function - selected **e3Pin** object is not a busbar.
- o **return -3**: Failed calling the function - busbar is locked.
- o **return -4**: Failed calling the function - busbar is placed on a panel sheet and electrically connected.

The following function has been added in the **e3Pin** interface to check whether a pin is a busbar:

**return = e3Pin.IsBusbar ( )**

**Return Values:**

- o **return 1**: Successfully called the function - the pin is a busbar.
- o **return 0**: Successfully called the function - the pin is not a busbar.
- o **return -1**: Failed calling the function - no pin IDs found or no project opened.

**New Function e3Tree - GetSelectedBusbarIds**

The following function has been added in the **e3Tree** interface to query the IDs of busbars that are selected in the tree view of the project:

**return = e3Tree.GetSelectedBusbarIds ( ids )**

**Parameter:**



- o **ids**: Specifies all IDs of busbars, which are selected on sheets in the project.

**Return Values:**

- o **return > 0**: Successfully called the function - the number of busbar IDs, which are selected on sheets in the project, is output.
- o **return -1**: Failed calling the function - no busbars selected or an error occurred.

**New Functions *e3Job* - *GetAllBusbarConnectionIds*, *GetBusbarIds*, *GetSelectedBusbarIds* and *GetTreeSelectedBusbarIds***

The following function has been added in the **e3Job** interface to query the IDs of busbar connections:

**return = e3Job.GetAllBusbarConnectionIds ( ids, flags )**

**Parameter:**

- o **ids**: Specifies all IDs of connection lines that belong to the busbar connection.
- o **flags**: An optional parameter with which the function can be enhanced.  
If the parameter has the value **0x0001**, logical connection lines are not output without an electrical connection.

**Return Values:**

- o **return > 0**: Successfully called the function - the number of busbar connections is output.
- o **return 0**: Successfully called the function - no busbar connections were found in the project.
- o **return -1**: Failed calling the function - no project opened or no **e3Job** object selected.

The following function has been added in the **e3Job** interface to query busbar IDs:

**return = e3Job.GetBusbarIds ( ids, flags )**

**Parameter:**

- o **ids**: Specifies all busbar IDs.
- o **flags**: An optional parameter with which the function can be enhanced.  
Presently, the parameter does not support any values.

**Return Values:**

- o **return > 0**: Successfully called the function - the number of busbars is output.
- o **return -1**: Failed calling the function - no project opened or no **e3Job** object selected.

The following function has been added in the **e3Job** interface to query busbar IDs that are selected on sheets in the project:

**return = e3Job.GetSelectedBusbarIds ( ids )**

**Parameter:**

- o **ids**: Specifies all IDs of busbars that are selected on sheets in the project.

**Return Values:**

- o **return > 0**: Successfully called the function - the number of busbar IDs, which are selected on sheets in the project, is output.
- o **return -1**: Failed calling the function - no busbars selected or an error occurred.

The following function has been added in the **e3Job** interface to query busbar IDs that are selected in the project tree view:

**return = e3Job.GetTreeSelectedBusbarIds ( ids )**

**Parameter:**

- o **ids**: Specifies all IDs of busbars that are selected in the project tree view.

**Return Values:**

- o **return > 0**: Successfully called the function - the number of busbar IDs, which are selected in the project tree view, is output.
- o **return -1**: Failed calling the function - no busbars selected or an error occurred.

**Customer Request: New Function e3Job - GetTextTypes**

The following function has been added in the **e3Job** interface to query the definition of text types:

**return = e3Job.GetTextTypes ( textTypeDefinitions )**

**Parameter:**

- o **textTypeDefinitions**: Specifies a dictionary with text type definitions.

**Return Values:**

- o **return > 0**: Successfully called the function - the number of text type definitions is output.
- o **return 0**: Failed calling the function - an error occurred.

**Reference:** Designer-38671

**Customer Request: New Function e3Job - UpdateComponentVersion**

The following function has been added in the **e3Job** interface to update the components of devices that are used in the project:

**return = e3Job.UpdateComponentVersion ( name, version, withSymbol )**

**Parameter:**

- o **name**: Specifies a component name.
- o **version**: Specifies the version of a component.  
The default value is **1**.

- o **withSymbol** : Specifies whether the symbol should also be updated if the number of pins or nodes differ compared to the current version.  
If the parameter has the value **True**, symbols are also updated.  
If the parameter has the value **False**, symbols are not updated.  
The default value is **True**.

**Return Values:**

- o **return 1**: Successfully called the function - the component was updated.
- o **return 0**: Failed calling the function - an error occurred.
- o **return -1**: Failed calling the function - the component was not found in the database.
- o **return -2**: Failed calling the function - the component is not used in the project.
- o **return -3**: Failed calling the function - the version of the component is not used in the project.

**Reference:** Designer-36766

**Customer Request: New Function e3Outline - IsThreadedHole**

The following function has been added in the **e3Outline** interface to query whether the contour of a device is defined as a threaded hole:

**return = e3Outline.IsThreadedHole ( )**

**Return Values:**

- o **return 1**: Successfully called the function - the contour is defined as a threaded hole.
- o **return 0**: Successfully called the function - the contour is not defined as a threaded hole.
- o **return -1**: Failed calling the function - no project opened.
- o **return -2**: Failed calling the function - no contour selected.
- o **return -3**: Failed calling the function - the selected object is not a contour.
- o **return -4**: Failed calling the function - the selected object is not a drill hole.

**Reference:** Designer-27296

**Customer Request: New Function e3Pin - GetWireHoseTubeStyle**

The following function has been added in the **e3Pin** interface to query the standard properties of wires, hoses and tubes:

**return = e3Pin.GetWireHoseTubeStyle ( colour\_code, descr\_type, descr, colour, line\_type, Rvalue, Gvalue, Bvalue, flags )**

**Parameter:**

- o **colour\_code**: Specifies the index, with which the color is identified in the database editor.
- o **descr\_type**: Specifies the description type defined in the database editor:
  - o **1**: The description type is the type **Wire colour**.
  - o **2**: The description type is the type **Material**.
- o **descr**: Specifies the description defined in the database editor.
- o **colour**: Specifies the color, which is used for the index.

- o **line\_type**: Specifies the line type defined in the database editor.
- o **Rvalue**: Specifies the red value of the color, which is displayed with **colour**.
- o **Gvalue**: Specifies the green value of the color, which is displayed with **colour**.
- o **Bvalue**: Specifies the blue value of the color, which is displayed with **colour**.
- o **flags**: An optional parameter with which the function can be enhanced.  
If the parameter has the value **0x0000**, all parameters are interpreted as listed in the description above.  
If the parameter has the value **0x0001**, the following rules are valid for interpreting the parameter:
  - o If **colour** has the value -1, the color has the value **Automatic**.
  - o If **colour** has the value -2, no value is set for the color.
  - o If **colour** has the value -1 or -2, the following also applies:
    - o If **line\_type** has a value less than 5, **Rvalue**, **Gvalue** and **Bvalue** are interpreted as RGB values of the color.
    - o If **line\_type** has the value 5 or greater, the value stands for exactly the line defined in the `FONTSDAT.DAT` file.

### Return Values:

- o **return 1**: Successfully called the function - information for color or material is output.
- o **return -1**: Failed calling the function - no pin ID selected or no project opened.
- o **return -2**: Failed calling the function - no valid ID selected.
- o **return -3**: Failed calling the function - the value for **flags** is invalid.
- o **return -4**: Failed calling the function - an error occurred.

**Reference:** Designer-38764

### Customer Request: New Functions *e3DbeGraph - GetInsideGraphIds*, *e3DbeSymbol - GetNodeIDs* and *e3DbeText - GetPosition*

The following function has been added in the **e3DbeGraph** interface to query graphic IDs located in the selected graphic:

**return = e3DbeGraph.GetInsideGraphIds ( ids, flags )**

### Parameter:

- o **ids**: Specifies an array with graphic IDs.
- o **flags**: An optional parameter with which the function can be enhanced.  
The parameter must have the value **0x0000** in the current version if specified.  
Other values are currently not supported.

### Return Values:

- o **return >= 0**: Successfully called the function - the number of IDs of graphics, which are available in **ids**, is output.
- o **return -1**: Failed calling the function - no project opened in **E3.series** database editor.
- o **return -2**: Failed calling the function - no graphic selected.
- o **return -3**: Failed calling the function - the value for **flags** is invalid.
- o **return -4**: Failed calling the function - graphic is not available.
- o **return -5**: Failed calling the function - graphic is placed on an invalid sheet.
- o **return -6**: Failed calling the function - graphic is not completely closed or has an invalid type.
- o **return -7**: Failed calling the function - an error occurred.

The following function has been added in the **e3DbeSymbol** interface to query node IDs on symbol sheets:

**return = e3DbeSymbol.GetNodeIDs ( ids, flags )**

**Parameter:**

- o **ids**: Specifies an array with node IDs.
- o **flags**: An optional parameter with which the function can be enhanced. The parameter must have the value **0x0000** in the current version if specified. Other values are currently not supported.

**Return Values:**

- o **return >= 0**: Successfully called the function - the number of node IDs, which are available in **ids**, is output.
- o **return -1**: Failed calling the function - no project opened in the **E3.series** database editor or no symbol selected.
- o **return -2**: Failed calling the function - the value for **flags** is invalid.

The following function has been added in the **e3DbeText** interface to query the position of a text object:

**return = e3DbeText.GetPosition ( x, y )**

**Parameter:**

- o **x**: Specifies the x-coordinate of the text object.
- o **y**: Specifies the y-coordinate of the text object.

**Return Values:**

- o **return 1**: Successfully called the function - the coordinates of the text are output.
- o **return -1**: Failed calling the function - no project opened in **E3.series**.
- o **return -2**: Failed calling the function - no text selected.

**Reference:** Designer-38137

**Customer Request: New Functions e3Device, e3Component and e3Symbol-GetStateId, GetStateIds and SetStateId**

The following functions have been added in the **e3Device** and **e3Symbol** interfaces to query and set the active state of devices and placed symbols:

**return = e3Device.GetStateId ( )** and  
**return = e3Symbol.GetStateId ( )**

**Return Values:**

- o **return > 0**: Successfully called the function - the state of the device or symbol is output.
- o **return 0**: Inconclusive - the device or symbol has no state or an error occurred.

**return = e3Device.SetStateId ( id )**

**Parameter:**

- o **id:** Specifies the ID of a state, with which the active state of the device or symbol should be changed.

**Return Values:**

- o **return 1:** Successfully called the function - the ID of the desired state was set as the active state of the device.
- o **return -1:** Failed calling the function - no device selected or no project opened.
- o **return -2:** Failed calling the function - the selected device is not assigned a component.
- o **return -3:** Failed calling the function - **id** is not a valid ID of a state in **e3Component**.
- o **return -4:** Failed calling the function - symbols of the device are placed on a locked sheet.
- o **return -5:** Failed calling the function - the device is a device view.

**return = e3Symbol.SetStateId ( id )**

**Parameter:**

- o **id:** Specifies the ID of a state, with which the active state of the symbol should be changed.

**Return Values:**

- o **return 1:** Successfully called the function - the ID of the desired state was set as the active state of the symbol.
- o **return -1:** Failed calling the function - no symbol selected or no project opened.
- o **return -2:** Failed calling the function - the selected object is not a valid symbol or a symbol of the wrong type.
- o **return -3:** Failed calling the function - **id** is not a valid ID of a state in **e3Component**.
- o **return -4:** Failed calling the function - the symbol is placed on a locked sheet.
- o **return -5:** Failed calling the function - assigned device of the symbol is a device view.
- o **return -6:** Failed calling the function - the symbol is placed on a schematic sheet.
- o **return -7:** Failed calling the function - the symbol is assigned an inactive variant/option.

The following functions have been added in the **e3Component** and **e3Symbol** interfaces to query all IDs of states, which are defined for the component or symbol:

**return = e3Component.GetStateIds ( ids )** and  
**return = e3Symbol.GetStateIds ( ids )**

**Parameter:**

- o **ids:** Specifies an array with IDs for states, which are defined for the component or symbol.

**Return Values:**

- o **return > 0:** Successfully called the function - e number of IDs of states, which are available in **ids**, is output.
- o **return 0:** Failed calling the function - no ID of states found or an error occurred.

**Reference:** Designer-24309

**Customer Request: New Functions *e3Device* - *IsPreventedAgainstPhysicalChangesOfCores* and *e3Pin* - *IsCoreEndLockedPermanent***

The following function has been added in the ***e3Device*** interface to check whether changes can be made to the conductors/wires of an assembly:

**return = e3Device.IsPreventedAgainstPhysicalChangesOfCores ( )**

**Return Values:**

- o **return 1:** Successfully called the function - no changes can be made to the conductors/wires of the assembly.
- o **return 0:** Successfully called the function - changes can be made to the conductors/wires of the assembly.
- o **return -1:** Failed calling the function - no project opened or no device selected.
- o **return -2:** Failed calling the function - component has no valid end types defined in the database for conductor connections.

The following function has been added in the ***e3Pin*** interface to check which ends of conductors are connected in the conductor connection of an assembly and are consequently blocked for changes:

**return = e3Pin.IsCoreEndLockedPermanent ( )**

**Return Values:**

- o **return 3:** Successfully called the function - both ends of the selected wire are connected.
- o **return 2:** Successfully called the function - end 2 of the selected wire is connected.
- o **return 1:** Successfully called the function - end 1 of the selected wire is connected.
- o **return 0:** Successfully called the function - neither end of selected wire is connected.
- o **return -1:** Failed calling the function - no project opened.
- o **return -2:** Failed calling the function - no object of the type ***Pin*** selected.
- o **return -3:** Failed calling the function - the selected object of the type ***Pin*** is not a wire.

**References:** Designer-08203 and Designer-38679

**Customer Request: New Function *e3Group* - *GetAnyIds***

The following function has been added in the ***e3Group*** interface to specifically query all IDs of objects of a certain type:

**return = e3Group.SetStateId ( flags, anyIds )**

**Parameter:**

- o **flags:** Specifies the objects from which the IDs are queried. The following values are possible:
  - &H01:** Specifies the IDs of all placed symbols and fields.
  - &H02:** Specifies the IDs of all dimension objects.
  - &H04:** Specifies the IDs of all graphic objects.
  - &H08:** Specifies the IDs of all groups.
  - &H10:** Specifies the IDs of all lines.
  - &H20:** Specifies the IDs of all texts.

- &H40:** Specifies the IDs of all sheets.
- &H80:** Specifies the IDs of all conductors and wires.
- o **anyIds:** Specifies a dictionary, in which a list with the IDs of the corresponding objects in the project is created for each object type found.  
The following keys are possible in the dictionary:
  - 22:** The key represents conductors and wires.
  - 28:** The key represents sheets.
  - 30:** The key represents placed symbols.
  - 31:** The key represents texts.
  - 32:** The key represents lines.
  - 34:** The key represents graphic objects.
  - 110:** The key represents groups.
  - 151:** The key represents dimension objects.

#### Return Values:

- o **return > 0:** Successfully called the function - the ID of the desired state was set as the active state of the device.
- o **return -1:** Failed calling the function - no symbol selected or no project opened.
- o **return -2:** Failed calling the function - the selected object is not a valid symbol or a symbol of the wrong type.

**Reference:** Designer-36963

#### Customer Request: New Functions *GetGID*, *GetGUID*, *GetId*, *SetGID*, *SetGUID* and *SetId* in Different Interfaces

The following functions have been added to different interfaces to control the import and export of objects from and to **E<sup>3</sup>.series** via the programming interface:

- **return = Interface.GetGID ()**  
The function queries the **Global Identifier** of objects of the respective interfaces.  
With *Global Identifiers* objects in **E<sup>3</sup>.series** multiuser projects can be identified.
- **Return Values:**
  - o **return "<GID>":** Successfully called the function - specifies the *Global Identifier* of the selected object.
  - o **return "<Empty>":** Failed calling the function - the selected object belongs to a type, which is not compatible with the function of the selected interface.
- **return = Interface.GetGUID ()**  
The function queries the **Globally Unique Identifier** of objects of the respective interfaces.  
With *Globally Unique Identifiers* objects in different **E<sup>3</sup>.series** projects can be identified.
- **Return Values:**
  - o **return "<GUID>":** Successfully called the function - specifies the *Globally Unique Identifier* of the selected object.
  - o **return "<Empty>":** Failed calling the function - the selected object belongs to a type, which is not compatible with the function of the selected interface.
- **return = Interface.GetId ()**  
The function queries the **Identifier** of objects of the respective interfaces.  
With *Identifiers* objects in **E<sup>3</sup>.series** projects can be identified.
- **Return Values:**
  - o **return > 0:** Successfully called the function - specifies the *Identifier* of the selected object.



- o **return 0**: Failed calling the function - the selected object belongs to a type, which is not compatible with the function of the selected interface.
- **return = Interface.SetGID ( gid )**
- **Parameter:**
  - o **gid**: Specifies a string, which is set as the *Global Identifier* for the selected object.
- **Return Values:**
  - o **return "<GID>"**: Successfully called the function - specifies the *Global Identifier* of the selected object.
  - o **return "<Empty>"**: Failed calling the function - the selected object belongs to a type, which is not compatible with the function of the selected interface.
- **return = Interface.SetGUID ( guid )**
- **Parameter:**
  - o **guid**: Specifies a string, which is set as the *Globally Unique Identifier* for the selected object.
- **Return Values:**
  - o **return "<GUID>"**: Successfully called the function - specifies the *Globally Unique Identifier* of the selected object.
  - o **return "<Empty>"**: Failed calling the function - the selected object belongs to a type, which is not compatible with the function of the selected interface.
- **return = Interface.SetId ( id )**
- **Parameter:**
  - o **id**: Specifies a string, which is set as the *Identifier* for the selected object.
- **Return Values:**
  - o **return > 0**: Successfully called the function - specifies the *Identifier* of the selected object.
  - o **return 0**: Failed calling the function - the selected object belongs to a type, which is not compatible with the function of the selected interface.

**Reference:** Designer-35432

### Customer Request: New Functions **e3Job** - *GetInvertDisplayColour* and *SetInvertDisplayColour*

The following function has been added in the **e3Job** interface to query whether the colors in **E3.series** are inverted:

**return = e3Job.GetInvertDisplayColour ( )**

#### **Return Values:**

- o **return > 0**: Successfully called the function - the setting for inverting the colors is active.
- o **return 0**: Successfully called the function - the setting for inverting the colors is inactive.

The following function has been added in the **e3Job** interface to invert the colors, which are displayed in **E3.series**:

**return = e3Job.SetInvertDisplayColour ( value )**

**Parameter:**

- o **value:** Specifies whether the colors in **E3.series** should be inverted:
  - o **True:** Activates the setting.
  - o **False:** Deactivates the setting.

**Return Values:**

- o **return 1:** Successfully called the function - the setting for inverting the colors was previously inactive and is now active.
- o **return 0:** Inconclusive - the setting for inverting the colors was previously active and is now inactive or an error occurred.

**Reference:** Designer-37998

**Customer Request: New Trigger *AfterModifyBusbarEnds***

The following trigger has been added to work with busbars:

- ***AfterModifyBusbarEnds*** – the trigger is executed when a busbar is electrically connected, the electrical connection of a busbar is disconnected or a busbar is deleted that was previously electrically connected.

The triggers ***AfterWireConnected*** and ***AfterWireDisonnected*** are not executed when busbar connections are changed.

**Reference:** Designer-39351

**Customer Request: New Interface *e3DbeNode* to Work with Nodes**

The ***e3DbeNode*** interface has been added to work with nodes.

The following functions are available in the ***e3DbeNode*** interface:

- o **return = e3DbeNode.GetDirection ( flags )**
- o **return = e3DbeNode.GetId ( )**
- o **return = e3DbeNode.GetPosition ( x, y )**
- o **return = e3DbeNode.GetTextIds ( ids, texttype )**
- o **return = e3DbeNode.HasPassWires ( )**
- o **return = e3DbeNode.IsBusPin ( )**
- o **return = e3DbeNode.SetId ( id )**

**Reference:** Designer-38137

**Customer Request: New Interface *e3State* to Work with States in eCheck**

The ***e3State*** interface has been added to work with states of objects in eCheck.

The following functions are available in the ***e3State*** interface:

- o **return = e3State.GetAttributeIds ( ids, attnam )**

- o return = e3State.GetAttributeValue ( name )
- o return = e3State.GetGID ( )
- o return = e3State.GetGUID ( )
- o return = e3State.GetId ( )
- o return = e3State.GetName ( )
- o return = e3State.GetOwnerType ( )
- o return = e3State.GetStateType ( )
- o return = e3State.HasAttribute ( name )
- o return = e3State.SetGID ( gid )
- o return = e3State.SetGUID ( guid )
- o return = e3State.SetId ( id )

**Reference:** Designer-24309

## General Changes

- **E3.series is compatible with Windows 11 Pro and will only be released as 64-bit version**
- **Setting for Inverting Displayed Colors**
- **New Icons for Commands and Updated Icons for Objects**
- **Tables Available in the Modules E3.view, E3.viewPlus and E3.redliner**
- **New Licensing Mechanism**

**E3.series is compatible with Windows 11 Pro and will only be released as 64-bit version**

**E3.series** is also compatible with Windows 11 Pro starting with Build 23.00.

**E3.series** versions from Build 23.00 are only available as 64-bit versions.

**Note:** **E3.series** is also executable with other editions of Windows 11.  
Full compatibility with Windows 11 has been tested only with Windows 11 Pro.

Patches and service packs for previous versions are not affected by this change and will continue to be provided as 32-bit version.

Add-ons, which have been released as 32-bit version up till now, will also continue to be available in 32-bit.

### Setting for Inverting Displayed Colors

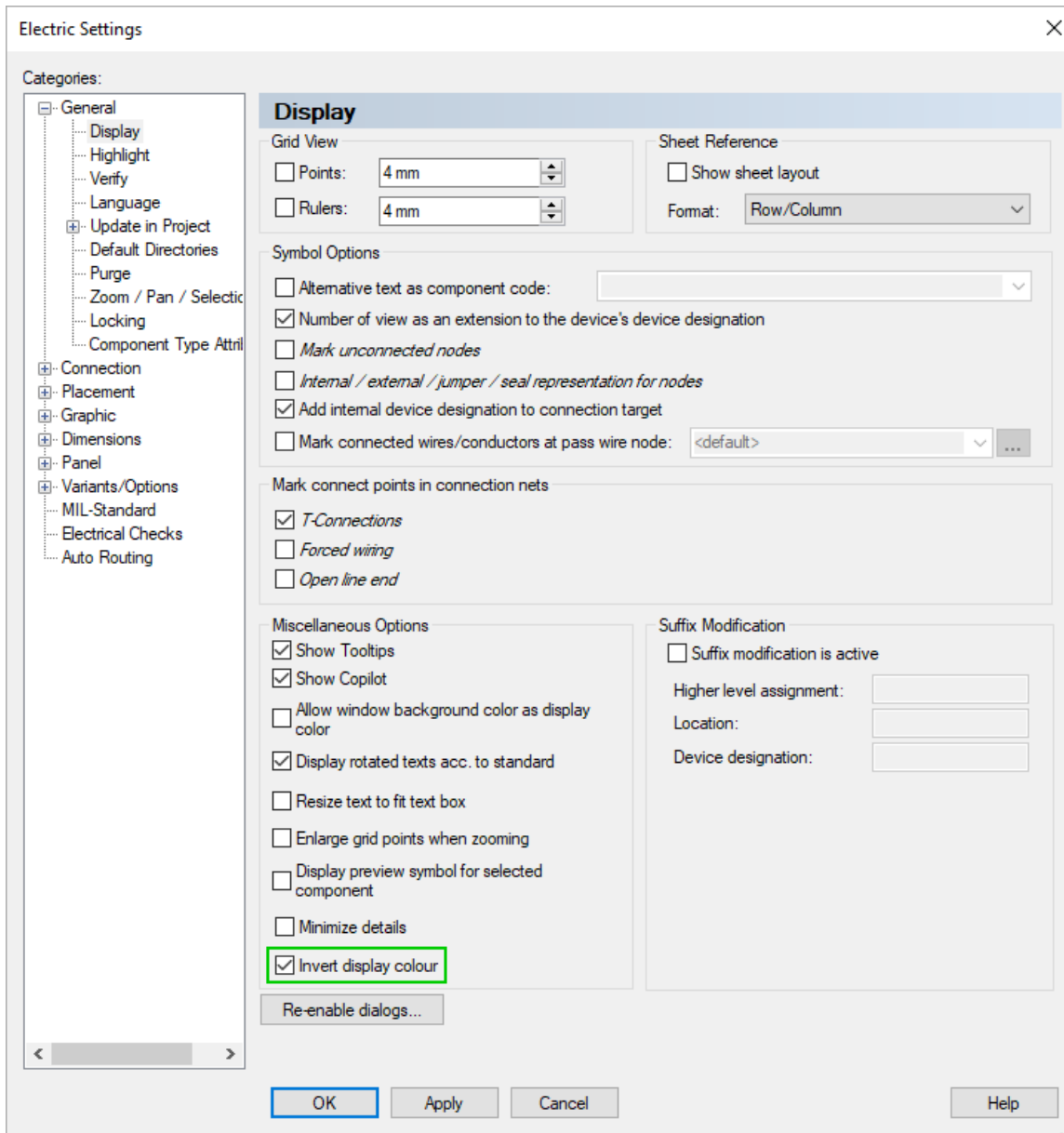
With one setting, all colors on sheets and in the preview window can be inverted.

This can improve screen ergonomics when working with **E3.series**.

To invert the color setting, select the main menu command **Tools** → **Settings...**  
The **Electric Settings** dialog opens.

**Note:** The setting can be activated and deactivated in the **Electric Settings** and **Fluid Settings** dialogs.

To do so, activate the setting **Invert display color** under **General** → **Display**. The setting is project-specific:



All colors on sheets and in the preview window are inverted.  
The color values are reversed by subtracting the RGB values from 255.

The colors of all other content, for example dialogs or images on sheets, are not reversed by this.

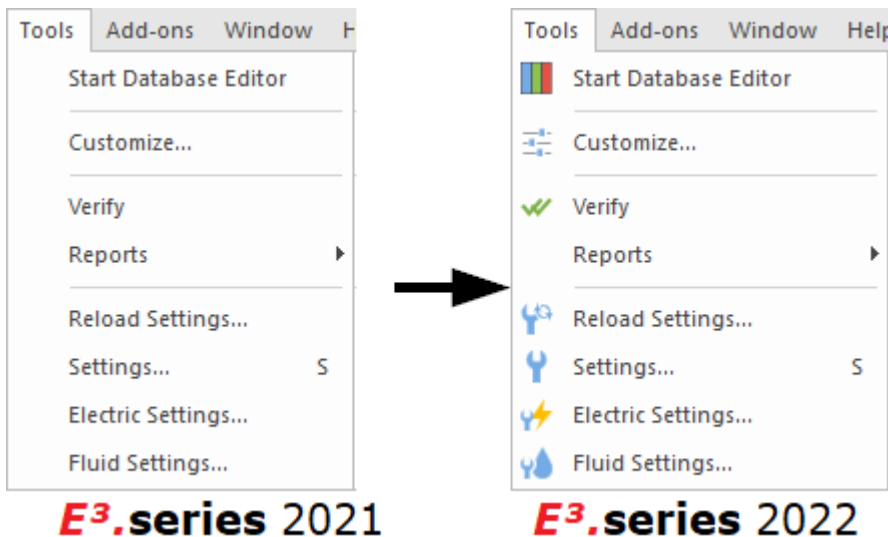
Exported content is not affected by the color reversal.

**Example:** All white areas will turn black as a result of the setting.

**Reference:** Designer-37998

## New Icons for Commands and Updated Icons for Objects

The **E<sup>3</sup>.series** user interface contains new icons that were previously only identified by text. These include, for example, the commands in the main menu window **Tools**:



This means, for example, that the commands take up less space in toolbars and can be found and used more quickly thanks to them being self-explanatory.

With the symbolism used for the new icons, working with **E3.series** becomes more intuitive and usability is improved.

The icons for objects in **E3.series** have been revised. This uses already known symbolism with a new graphic style.

As a result, the **E3.series** user interface has a modern look and fits better into the shape and color scheme of current computer programs.

### Tables Available in the Modules **E3.view**, **E3.viewPlus** and **E3.redliner**

Devices, sheets, pins, connections, cabling and terminals in tables can now be displayed in the modules **E3.view**, **E3.viewPlus** and **E3.redliner**.

This enables the user to have an even better overview of project with the modules used for the project view.

To open a table, select **View -> Device Table, Sheet Table, Pin Table, Connection Table, Cabling Table** or **Terminal Table** in the main menu. The selected table opens.

The tables can be configured, sorted and used for searching.

Exporting as a CSV file is not supported with the modules **E3.view**, **E3.viewPlus** and **E3.redliner**.

**References:** Designer-17932 and Designer-30545

### New Licensing Mechanism

As of **E3.series** Version 2022 23.00 a new license mechanism is used.

With the new licensing mechanism, all Zuken tools are managed through the same software. This makes the management of Zuken's licenses easier and more transparent.

To use **E3.series** with the new license mechanism of the **Floating Licenses** server, install the license server with the file `setup.exe` from the folder <Installation folder E3.series>\FLEXNET\Licence>.

The license server is installed. The default installation directory is `C:\zuken\license`.

Then start the installed license server with the file `lmttools.exe` and add the floating license for **E3.series** in the dialog under **Service/License File** → **Configuration using License File**.

To use the new license mechanism of the license server for **Nodelocked Licenses**, copy the license file named `license.dat` in the installation directory of **E3.series**.

#### Notes:

- License servers installed with **previous versions** of **E3.series** are not compatible with **E3.series Version 2022 23.00 and later**.  
The new license server must therefore be reinstalled even if you have installed a previous version of **E3.series**.
- Similarly, license servers installed with **E3.series 2022 23.00 or higher** are not compatible with **previous versions** of **E3.series**.
- The modified license mechanism from **E3.series** Version 2022 23.00 also allows simultaneous operation with an existing license server of a previous version.

The existing license server of the previous version may be used for a transition period of 8 months from the date of issuance of the new license for testing, demo purposes and migration processes.

After this period, the operation of the old license server will be discontinued.

If it is necessary to continue to run the older version of **E3.series** for a period of time after 8 months, contact your Zuken sales representative.

## Database Editor

- Define Assemblies with Conductor Connections
- Create and Use Blocks with Connectors with Inserts

### Define Assemblies with Conductor Connections

Conductor connections can be defined for new or existing assemblies. It is then possible to define which conductors are connected to which pins within the assembly.

This allows assemblies to be stored in the database, which can then be ordered as pre-assembled parts, for example.

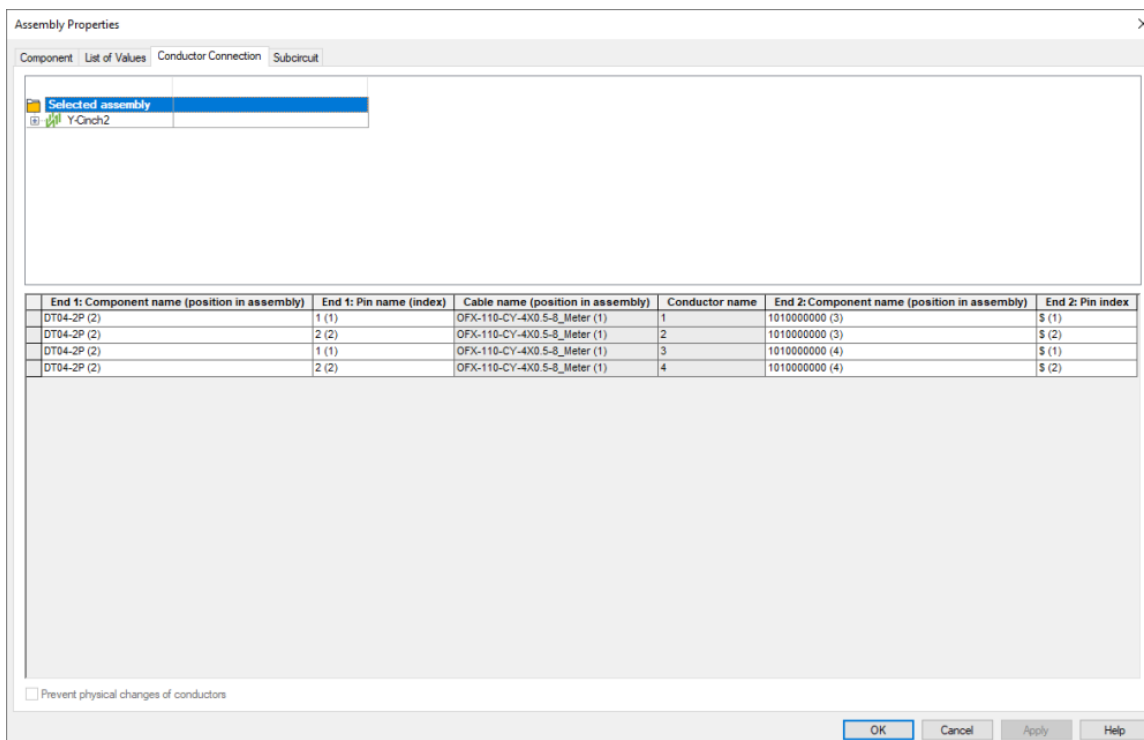
Conductor connections for a new assembly can be created in the database or edited in an existing assembly.

- To create a new assembly with conductor connections, right-click in the database editor component tree and select the context menu command **New Component**. The component wizard opens.

Enter all information regarding the assembly in the component wizard, select **Edit component graphically** in the last step and confirm the settings with **Finish**. The assembly is displayed on the component sheet and in the database editor project window.

- To add conductor connections to an existing assembly, right-click on the assembly to be edited in the component database and select the context menu command **Edit**. The assembly is displayed on the component sheet and in the database editor project window.

Next select the assembly on the component sheet or in the database editor project window and then the **Assembly Properties...** or **Properties...** in the displayed context menu. The **Assembly Properties** dialog opens:





In the upper part of the dialog all components of the assembly and all conductors of cables of the assembly are displayed in the tree view.

Depending on the selection in the tree view, either all conductors of the assembly, only the selected conductors or the conductors of the selected cables are displayed in the lower part of the dialog.

**Note:** Single conductors and conductors of cable bundle material cannot be assigned or defined as conductor connections in the database editor. Only cables are supported in the **Conductor Connection** tab.

The assignment of the selected conductors can be defined in the lower part of the dialog. When assigning the conductors, it is defined which conductor ends are connected to which component pins within the assembly.

**Note:** Only pins of connectors, standard components and terminals can be selected. The assignment does not check whether the pins are compatible in terms of the number of connection possibilities and conductor cross-section.

The following setting possibilities are available:

## Conductor Connection

<p><b>End 1: Component name (position in assembly) and End 2: Component name (position in assembly)</b></p>	<p>Defines between which components the conductor is connected. The position of the component in the assembly is displayed in brackets after the component name.</p>
<p><b>End 1: Pin name (index) and End 2: Pin name (index)</b></p>	<p>Defines on which available pins of the selected component the conductor is placed. The index of the pin is displayed in brackets after the pin name.</p>
<p><b>Cable name (position in assembly)</b></p>	<p>Displays the name of the cable used for the conductor connection. The position of the cable in the assembly is displayed in brackets after the cable name.</p>
<p><b>Conductor name</b></p>	<p>Indicates which name is defined for the conductor of the component.</p>
<p><b>Prevent physical changes of conductors</b></p>	<p>If the setting is active, the assembly can no longer be changed independently, e.g. connector pin terminals in the project.</p>

Some checks that are normally available for pins are not performed for conductor connections in assemblies. The conductor connections that have been defined in the database editor are automatically accepted as valid in the project. In addition, the following rules apply to conductor connections of assemblies in the project:

- As soon as a conductor is connected to a pin, which is part of an assembly, and there is no valid connector pin terminal for the assembly, no connector pin terminal is set.
- Conductor connections of assemblies are used for routing even if the physical properties do not match the rest of the connection.
- If only conductors from the same assembly are placed on the pins, no check takes place.
- If additional conductors/wires from the project are connected to a pin of an assembly, all conductors/wires on the pin are included in the check.

**Reference:** Designer-08203

### Create and Use Blocks with Connectors with Inserts

Blocks created in the database can use connectors with inserts as block connectors.

This makes the block functionality in **E<sup>3</sup>.series** even more extensive and flexible.

To create a connector with inserts as a block connector in the database, create a block in the database editor of **E<sup>3</sup>.series** and drag a connector with inserts into the workspace.

To use a block with a block connector in the form of a connector with inserts in the project, place a corresponding block created in the database in the project.

Note the following limitations of block connectors:

- Symbols of connector pin terminals of a **connector with inserts**, which are placed on a **block component**, are deleted in the following cases and must be placed again:
  - If the connector pin terminal of a connector with inserts is the type **connector** and it is exchanged with a cavity part of the type **connector with inserts**, or
  - the connector pin terminal is a connector with inserts and is exchanged with a cavity part of the type **connector**.
- Symbols of connector pin terminals of a **connector with inserts** (for example with the device designation **-X1**), which are placed on a **dynamic block**, are **not** deleted when exchanging with a connector with inserts, but rather receive a new device designation (for example **-X2**).  
The exchanged connector with inserts is then displayed in the device tree with its original device designation as unplaced in the structure of the connector with inserts.
- Connectors with inserts cannot be inserted on blocks using the **Update block** command.
- Connectors with inserts that have already been placed are not changed, added or deleted with the **Update block** command.

**Reference:** Designer-21972

## Importing and Exporting Data

### Importing Data

- **Setting for Merging Attributes**

#### Setting for Merging Attributes

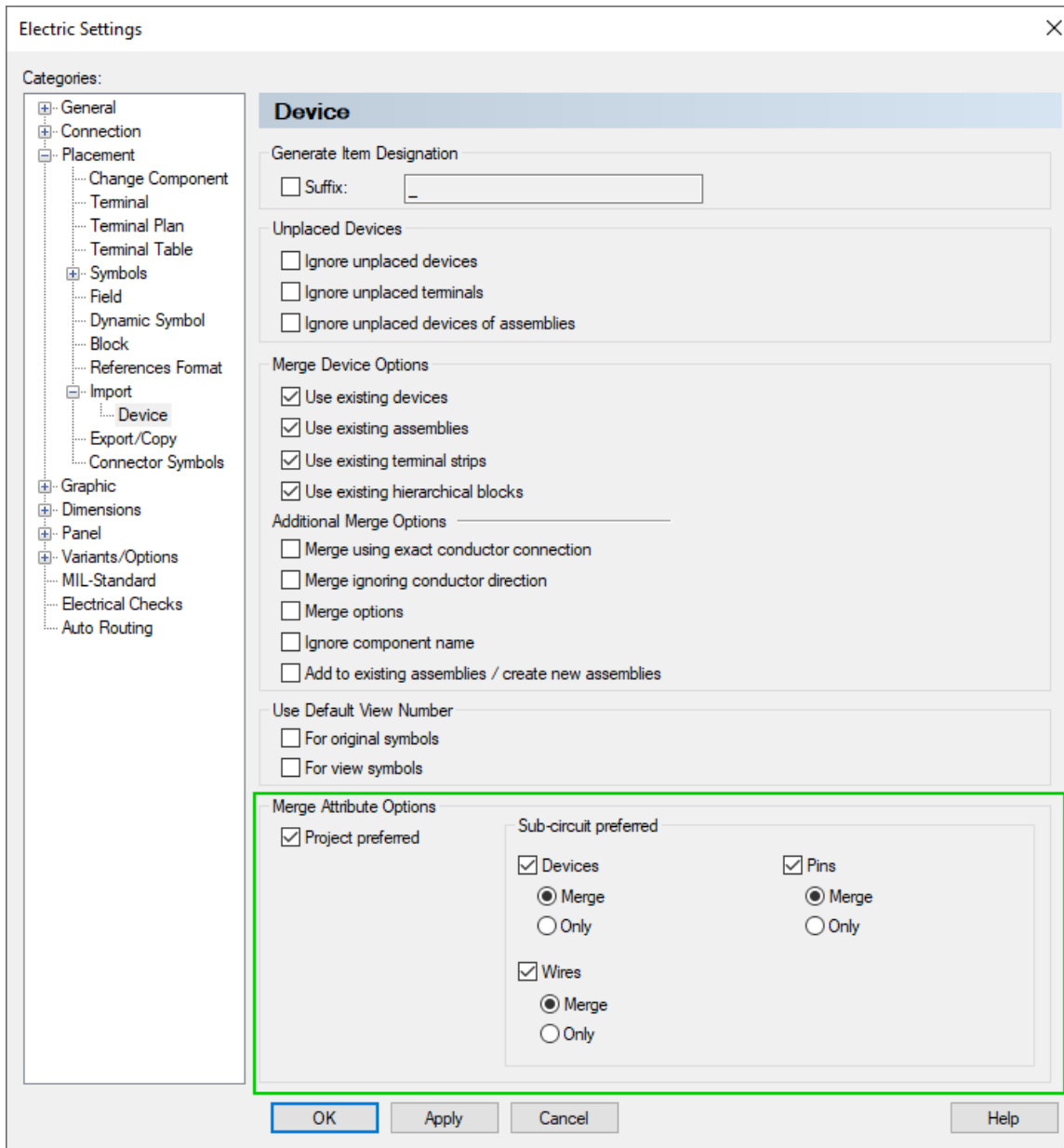
A setting can be used to control whether the attributes of imported devices are merged in the project.

This can be used to specifically control how the attributes of the project are combined or deleted when devices are imported.

To do so, select the main menu command **Tools** → **Settings...**  
The **Electric Settings** dialog opens.

**Note:** The function can be activated and deactivated in the **Electric Settings** and **Fluid Settings** dialogs.

Then set the behavior for merging attributes under **Placement** → **Import** → **Device**.  
Specify the behavior with the general setting **Project preferred** and/or specify the behavior for the available object types (devices, pins, wires). The setting is user-specific:



The settings have the following effect in the project:

### Options for Merging Attributes

This section specifies how attributes are merged in the project when importing subcircuits.

#### Project preferred

**Note:** The setting has the same effect as the setting **Merge attributes** from the previous version of **E<sup>3</sup>.series**.

When importing, the system checks whether the device attribute from the subcircuit is available on the device in the project. If not available, it is added.

This option affects:

- normal components

- cables/conductors
- connectors
- block connectors
- blocks (if merged as split blocks)

**Subcircuit preferred** When importing, the system checks whether the attribute from the subcircuit for the set object types is available in the project. If not present, it is added.

This option can be set separately for the following objects:

- **Devices**
- **Pins**
- **Wires**

**Note:** Object types, which are activated for merging attributes under **Subcircuit preferred**, are ignored by the setting **Project preferred**. For example, if **Pins** is activated under **Subcircuit preferred**, attributes of pins are only merged according to the criterion **Merge** or **Only** in the project.

One of the following methods for merging attributes can be set for each of the object types:

- **Merge:** All attribute of the imported objects are integrated into the project. Attributes that have already been created in the project are integrated into the project with the corresponding values of the object.
- **Only:** Integrates all attributes of the selected object in the project. Attributes that are present in the project but are not defined for the respective imported object types are deleted from the project.

**Reference:** Designer-38483

## Multuser

- **Enhanced Log Files**
- **Password Encryption for Multuser Server**

### Enhanced Log Files

Log files can also log information that is recorded in a so-called *Audit Log* or *Audit Trail*.

This makes it possible, for example, to track when which user opened a multuser project, when it was closed or a new project was created on the server.

The following actions are logged together with the information about the user and the timestamp:

- Multuser project opened
- Multuser project closed
- Multuser project created
- Multuser project copied
- Multuser project renamed

Messages regarding renaming projects are only stored in the multuser server log file. The log file of the renamed project does not store who created the project and when.

To include project-related actions in the log file, insert the element **<ProjectTracing>** with the value **1** in at least one of the following child elements (**<Eventlog>**, **<File>** or **<File<sub>n</sub>>**) under **<Log>** in the configuration file `configure.xml`.

#### Example:

```
[...]
<Log>
  <Eventlog>
    <Error>0</Error>
    <Information>0</Information>
    <InternalError>0</InternalError>
    <Success>0</Success>
    <Warning>0</Warning>
    <ProjectTracing>1</ProjectTracing>
  </Eventlog>
  <File>
    <Warning>1</Warning>
    <ProjectTracing>1</ProjectTracing>
    <Debug>
      <Normal>0</Normal>
      <Time>0</Time>
    </Debug>
  </File>
  <File1>
    <Error>0</Error>
    <ProjectTracing>1</ProjectTracing>
  </File1>
</Log>
[...]
```

When logging is active, the information concerning when which user performed which project-related action is written to the log file.

The logged messages are saved in the **<EventLog>** with the ID 57004.

**Example of format for saved message:**

```
[8192] [Project Trace ] [30.03.2022 12:03:12] [30196]: User 'JohnDoe' created new project 'Project1'.
```

**Reference:** Designer-17095

**Password Encryption for Multiuser Server**

Passwords for multiuser servers installed with the multiuser setup from **E3.series** 2022 are automatically encrypted. The password encryption can also be subsequently removed. Password encryption can be activated for existing multiuser servers that were set up with older versions of the multiuser setup.

This means that when working with multi-user servers from **E3.series**, operational security guidelines can be adhered to, for example.

Use the following steps to activate the password encryption on an existing multiuser server:

- Contact Zuken Support to receive the tool `CryptPassword.exe`.
- Start the tool and enter the old, unencrypted password. The tool generates an encrypted password.
- Create the parameter `Crypted` with a value of **1** in the configuration file `configure.xml`.
- Replace the old password with the new, encrypted password for the value of the parameter `ConnectDbms` in the configuration.

Use the alternate, masked spellings in the configuration file for the following special characters in the password:

Special Character	Notation in Configuration File
&	&amp;
<	&lt;
>	&gt;
"	&quot;
'	&apos;

- Make the following changes to the multiuser server registry under `Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Zuken\E3.series\<Version>\Multiuser:`
  - o Enter the encrypted password for the Authentication key in plain text.
  - o Create the key `Crypted` and assign it the value **1**.
- Create the key `Crypted` with the value **1** in the file `e3mu_client.reg` that was automatically created in the **E3.series** installation directory during multiuser setup.

The password encryption can be removed later using the following steps:

- Change the value of the parameter `Crypted` to **0** in the configuration file `configure.xml`.
- Make the following changes to the registry under `Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Zuken\E3.series\<Version>\Multiuser:`
  - o Enter the password for the Authentication key in plain text.
  - o Set the value **0** for the `Crypted` key.
- Set the value for the key `Crypted` to **0** in the file `e3mu_client.reg` that was automatically created in the **E3.series** installation directory during multiuser setup.

**Reference:** Designer-29355



## Placement / Purge

- **Setting for Overlapping of Automatically Placed Bundle Symbols**

### Setting for Overlapping of Automatically Placed Bundle Symbols

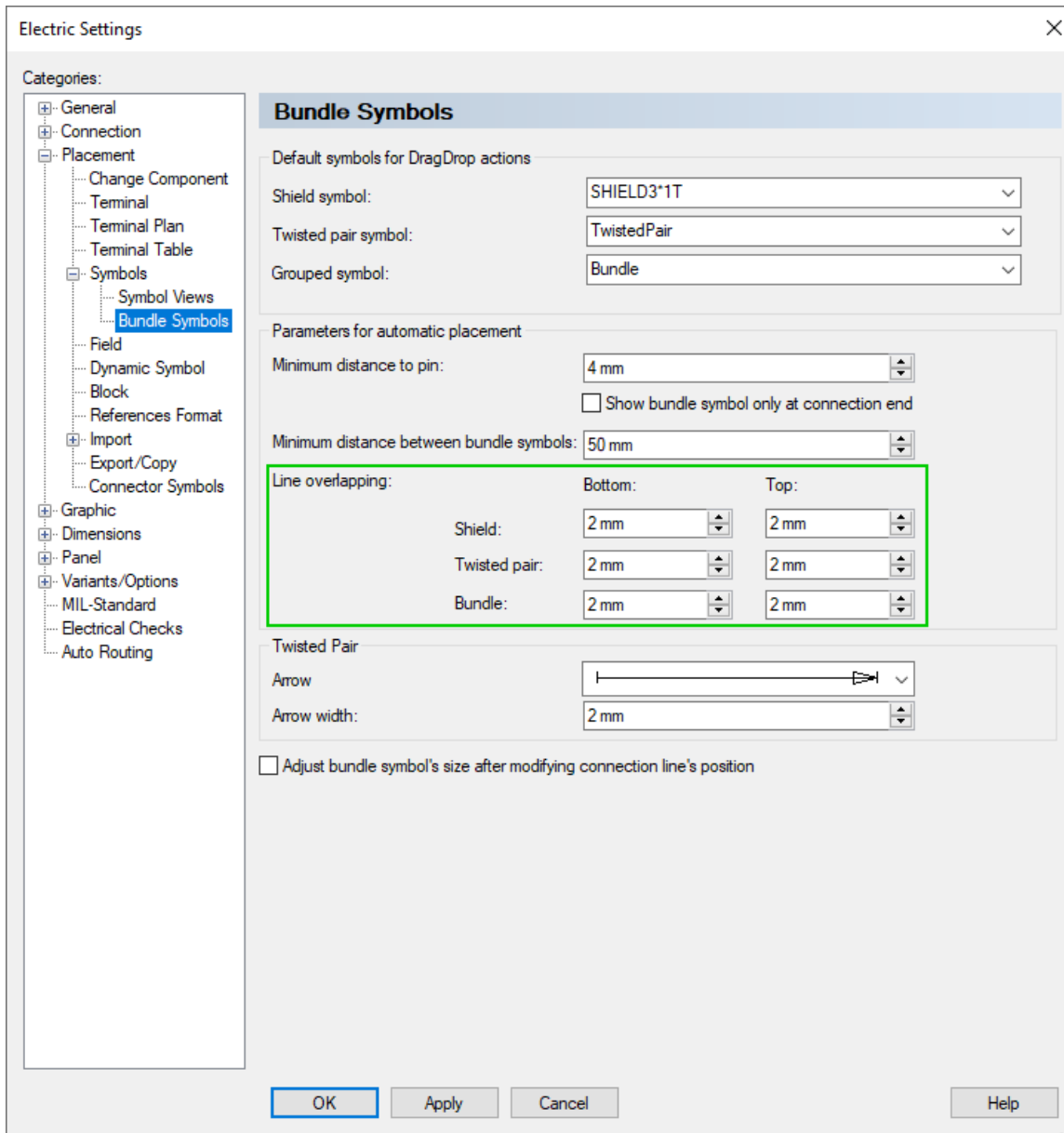
A new setting can be used to specify how far shields, twisted pairs and bundles overlap connection lines.

This allows, for example, the properties of automatically placed symbols to be controlled even better and plans to be developed in accordance with the IEEE 315 standard.

To set the behavior of overlapping symbols, select the command **Tools** → **Settings...** in the main menu.

The **Electric Settings** dialog opens.

Go to the sub-menu **Placement** → **Symbols** → **Bundle Symbols** and set for **Shield**, **Twisted pair** and **Bundle**, how far the bundles should overlap at the **bottom** and the **top**. The values can be between 0 and 20 millimeters or 0 and 0.787 inch. The setting is user-specific:



Automatically placed symbols overlap the connecting lines by the respective set value. When the symbols are placed rotated by 90°, the **bottom** determines the overlap to the right and the **top** determines the overlap to the left.

**Reference:** Designer-28568

## Project Handling

- **Define Drill Holes as Threaded Holes**
- **E3.eCheck: Check Fuse Melting Time and Wire Ignition Time and Starting E3.series without an eCheck License**
- **Enhanced Behavior for Updating Components**
- **Do Not Show Deleted Objects in Graphical Comparison of Project States**
- **New Text Type for Segments**
- **Combine Text Types**

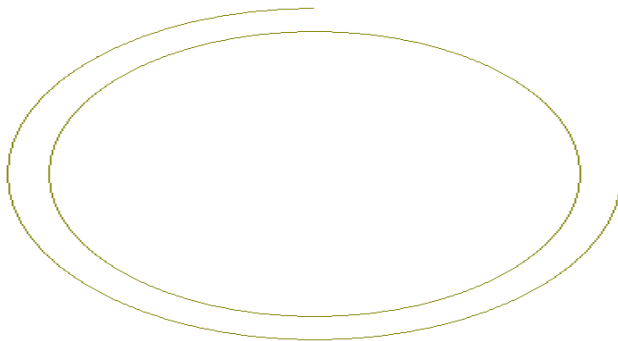
### Define Drill Holes as Threaded Holes

When defining drill holes, it can be specified whether the drill hole is created with a threaded hole or not.

This additional information also allows drilling machines to use the **E3.CutOut** drilling templates to distinguish between normal drill holes and threaded holes.

All threaded holes are displayed on the drilling template as a three-quarter circle with a complete circle in the center.

**Note:** The three-quarter circle has the **radius** defined for the drill hole. The distance between the three-quarter circle and the inner circle is always the same and does **not** allow any conclusion about the core diameter:

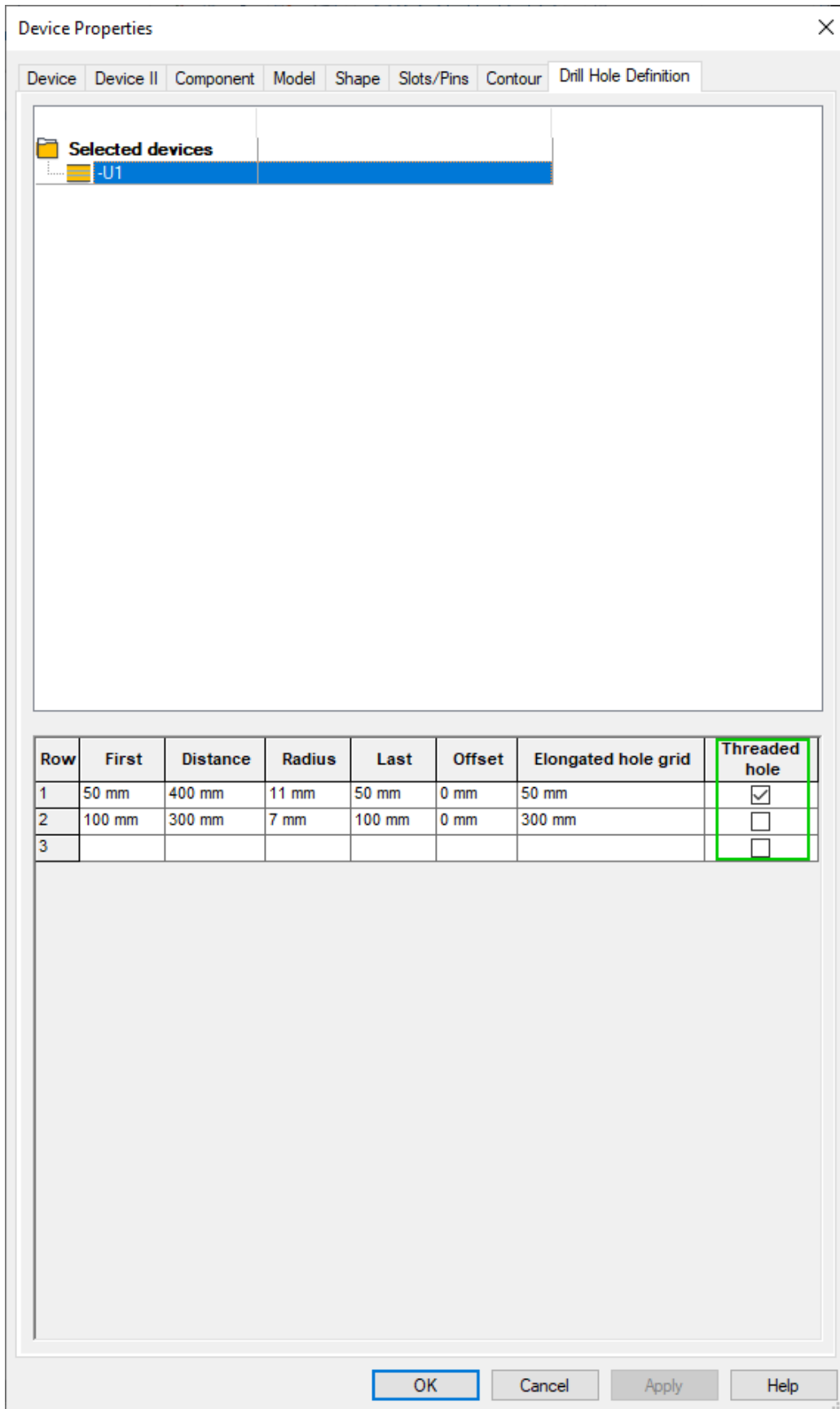


Threaded holes can be defined in two different ways:

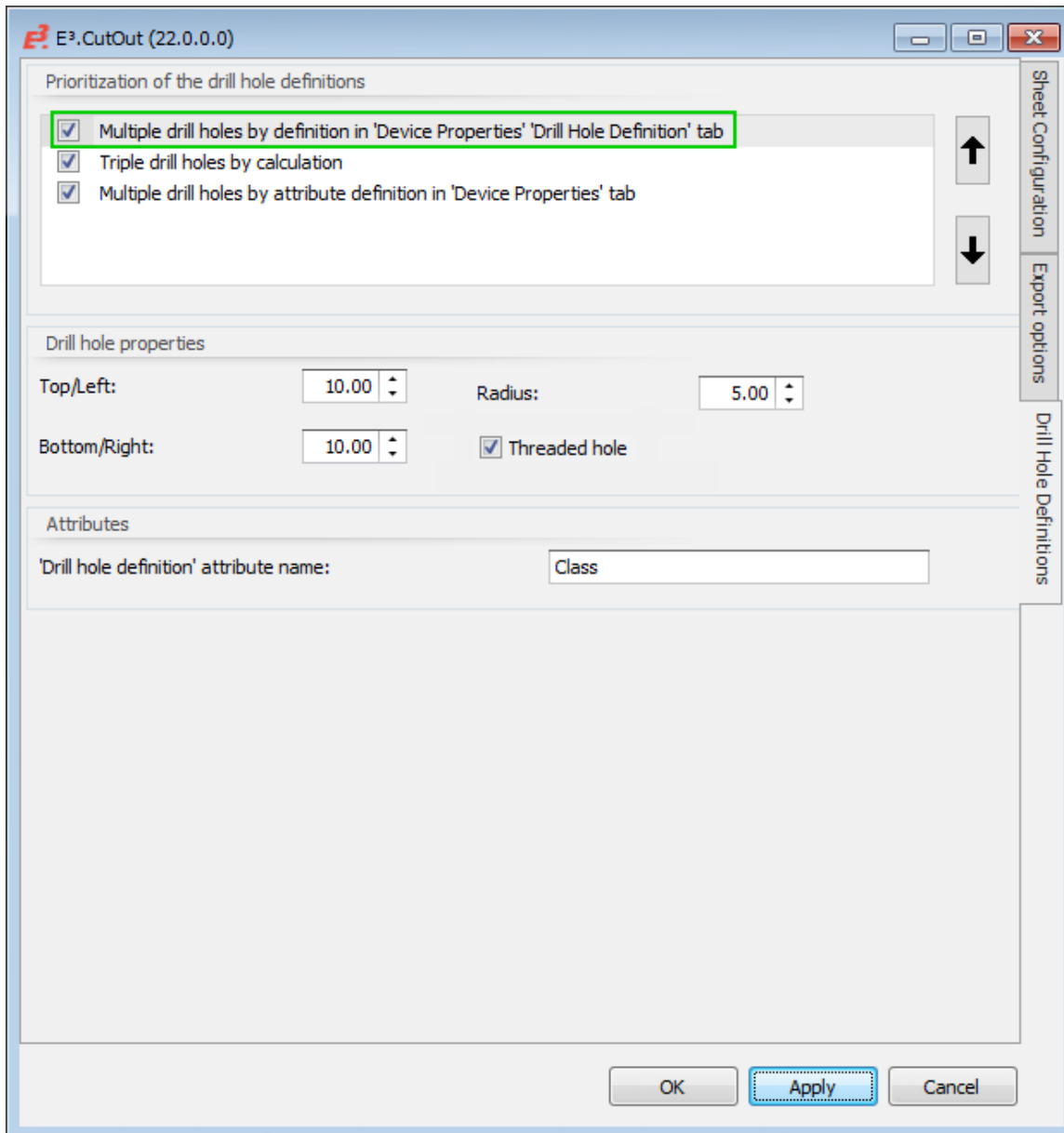
1. To define drill holes as threaded holes for the drill hole definition in the Device Properties, right-click on a cable duct or a mounting rail and select **Device Properties...** in the context menu.

The **Device Properties** dialog opens.

Switch to the **Drill Hole Definition** tab and activate the check box **Threaded hole** for all drill holes, which should be drilled as threaded holes:

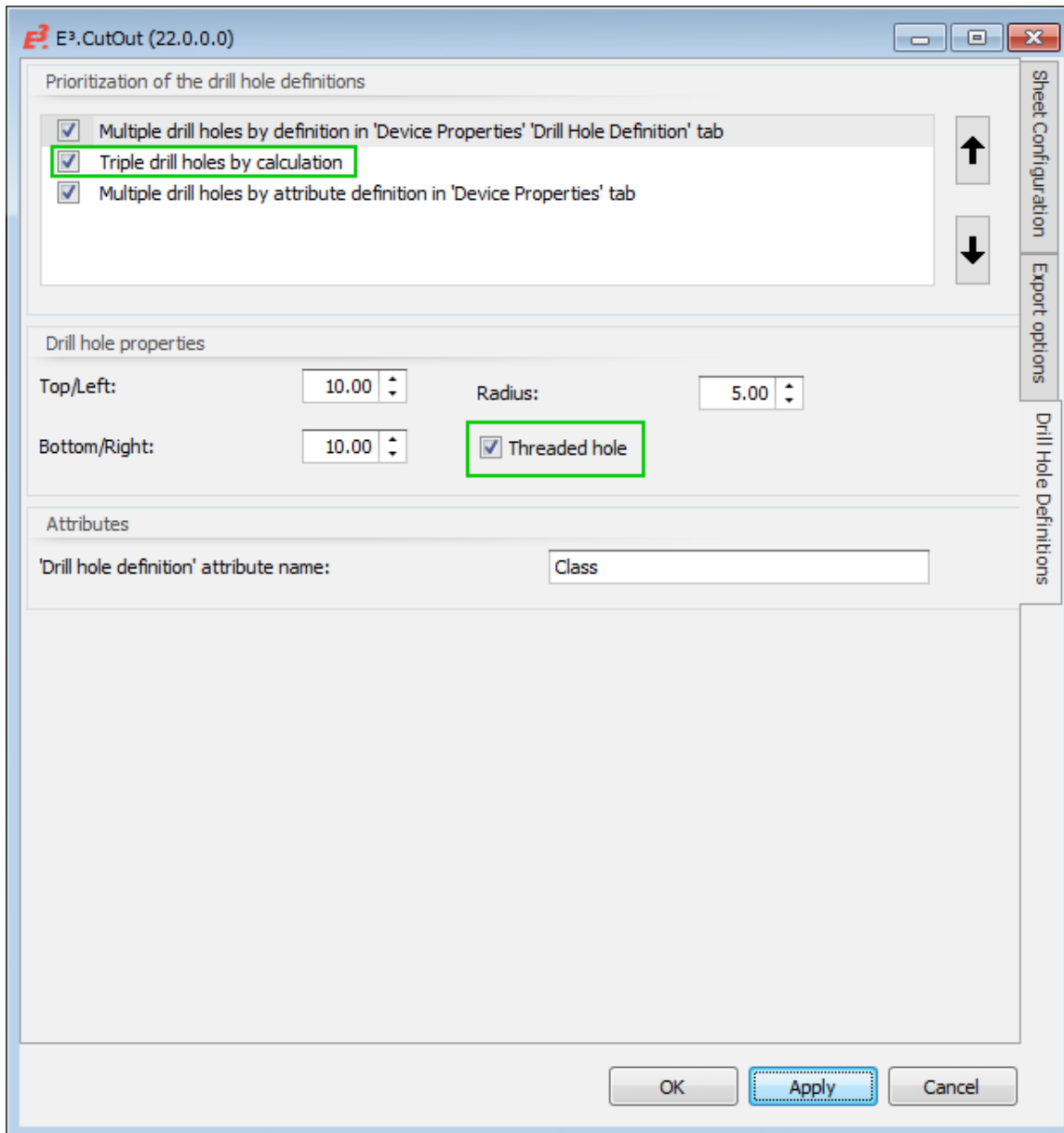


Activate the setting **Multiple drill holes by definition** in 'Device Properties' 'Drill Hole Definition' tab under **Drill Hole Definitions** in **E3.CutOut**:



All drill holes, which have the threaded hole check box activated in the **E<sup>3</sup>.series** Device Properties, are output as threaded holes on the **E<sup>3</sup>.CutOut** drill hole template.

2. To define drill holes in **E<sup>3</sup>.CutOut** as threaded holes with the setting **Triple drill holes by calculation**, activate the check box **Threaded hole** under **Drill Hole Definitions**:



**Reference:** Designer-27296

### **E<sup>3</sup>.eCheck: Check Fuse Melting Time and Wire Ignition Time and Starting E<sup>3</sup>.series without an eCheck License**

There is a new check in **E<sup>3</sup>.series** for determining the fuse melting time and wire ignition time in a circuit.

It checks the time when the fuse would blow, the wires between the fuses blow, and whether the time is before or after the fuse blows.

For the calculation it is determined how long the wire can be operated with the rated current of the fuses in the circuit.

Conductors before the first fuse and after the last load of a circuit are not considered by the check.

The calculation is performed using the formula described in the **Zuken Simulation and Verification Module** documentation under **Method for Calculating Wire Fire Point Time**. The document is part of the **E<sup>3</sup>.series** documentation.

The following prerequisites must be fulfilled to perform the **Check fuse melting time and wire ignition time**:

- The setting **Electrical Checks** → **Check fuse melting time and wire ignition time** must be activated.
- The functional units used for the type **Fuse** must have values for **Fuse operating time** in the database.  
The values can be changed in the component definition if required.  
If the attribute `.ECHECK_FUSE_RERATING_TEMPERATURE` is defined for the placed symbol, the values for the **Fuse temperature rerating** must also be created in the database and, if necessary, changed in the component definition.
- In the **Component Properties** the attribute **Wire kind** must be defined under **Body** for all wires that are used in the circuit.  
The various properties of the wire materials can be defined in the respective tabs in the Database Editor under **Format** → **Wire Materials...**
- The attribute **Rating (max. current fuse)** must be defined for all fuses in the circuit.

#### Notes:

- Only SI units are considered for checks with **E<sup>3</sup>.eCheck**.
- Values for **Fuse operating time** and **Fuse temperature rerating** can:
  - only be defined for **Fuse** type functional units and
  - not for states of fuses.
- Values for **Seconds [s]** under **Fuse operating time** and **Rerating factor [%]** under **Fuse temperature rerating** may only be 0 if **Fuse Load [%]** or **Temperature [°C]** is also 0.
- The fuse temperature rerating is only considered for the electrical check if:
  - the setting **Check fuse melting time and wire ignition time** is active,
  - a temperature is defined for the placed symbol with the attribute `.ECHECK_FUSE_RERATING_TEMPERATURE` and
  - the corresponding value is specified under **Temperature [°C]**.
- If the **Fuse temperature rerating** under **Temperature [°C]** does not specify a temperature that matches the value of the `.ECHECK_FUSE_RERATING_TEMPERATURE`, a temperature of 20 °C is assumed for the check.
- The number of values for **Fuse operating time** and **Fuse temperature rerating** can be supplemented with dummy values.  
To do this, insert the value 0 in both columns of the attribute to be supplemented.  
If you want to exclude an existing value from the check, set the value 0 in both columns.
- Values specified in the definition of the functional unit for **Fuse operating time** and **Fuse temperature rerating** can be changed in the component definition.

If the fuse blows after the cable blows, the following information is displayed in the output window under **Results**:

- The **Fuse operating time** with the device designation, symbol designation of the fuse and the sheet and position where the fuse is placed.
- The wire ignition time: This indicates after how many seconds the wire will blow at the given amperage and how long it would take the fuse to blow.
- If values were used for the **Fuse temperature rerating**, it also shows which temperatures and rerating factors were used.

The following components/functional units are displayed with error messages under **Results** in the output window:

- Conductors, for which the attribute **Wire kind** is not assigned.
- Conductors, for which no inner or outer material is specified with the defined wire material.
- Conductors, for which either the inner diameter is greater than the outer diameter or one of the two diameters is 0.
- Conductors with erroneous values for the **Thermal Capacity**, the **Gravity**, the **Inner Material** and the **Outer Material**.
- Conductors and fuses, in which the current is 0 amperes.
- Fuses missing the attribute **Rating (max. current fuse)**.
- Fuses missing values for the **Fuse operating time**.

With the new start up parameter `\noECheck E3.series` can be started without an eCheck license.

**Reference:** Designer-24251

### Enhanced Behavior for Updating Components

There are additional messages when updating components. In addition, adjustments have been made if, for example, a component version is used by a locked device.

The messages make it easier to understand the behavior of **E<sup>3</sup>.series** and, in the case of adjustments in connection with locked devices, improve operability.

The following rules apply when updating components:

- When updating a component that is available in several versions in the project, all selected versions will be updated to the current version. If a version of the component is used by a locked device, this version will not be updated.
- If the current version of the component is used by a locked device, no update is possible.
- Models are updated with their components. This is why updating components with the same model is only possible if all these components can be updated.

If other components need to be updated in addition to the selected components, a message is output:

- The selected, obsolete component is also used in the latest version elsewhere in the project. The latest version must then also be updated.
- The selected obsolete component uses a model that is also used by other components that are not selected for updating. All components that use the same model as the selected component must also be updated.
- The selected assembly uses an obsolete assembly element. The component of the assembly element must also be updated.

**Reference:** Designer-36766

### Do Not Show Deleted Objects in Graphical Comparison of Project States

When graphically comparing project states, it is possible to set whether and how deleted objects are displayed in the graphical comparison.



This improves the clarity of the graphical comparison. This becomes especially clear when numerous objects have been deleted or moved from one project state to the next.

To set whether and how deleted objects are displayed in the graphical comparison, **E3.series** must be started with the following parameters:

The parameters have the following meaning:

Parameters	Description
<b>/compareoldfile</b>	Displays the path, where the old <b>E3.series</b> project is located. Format: <b>[.e3s]</b>
<b>/comparenewfile</b>	Displays the path, where the new <b>E3.series</b> project is located. Format: <b>[.e3s]</b>
<b>/compareconfigfile</b>	Displays the path, where the configuration file is located. Format: <b>[.ini]</b>

Add the following configuration section in the configuration file and adjust the parameters according to your requirement:

## Parameters Entry

### Suppress display of deleted objects (Section [Result Options])

**In this section it is possible to set whether deleted objects are displayed in the graphical comparison and how deleted objects are highlighted in the comparison project. The highlighting can be combined with one another.**

#### BaseProject

The parameter specifies which project is used as the base project for the graphical comparison.

The following values are possible:

**OldFile:** The project specified under `/compareoldfile` is used as the base project.

**NewFile:** The project specified under `/comparenewfile` is used as the base project.

**ShowDeleted**

The parameter specifies whether deleted objects are displayed in the comparison file.

The following values are possible:

`true`: Deleted objects are displayed in the comparison file. The default value for `ShowDeleted` is `true`.

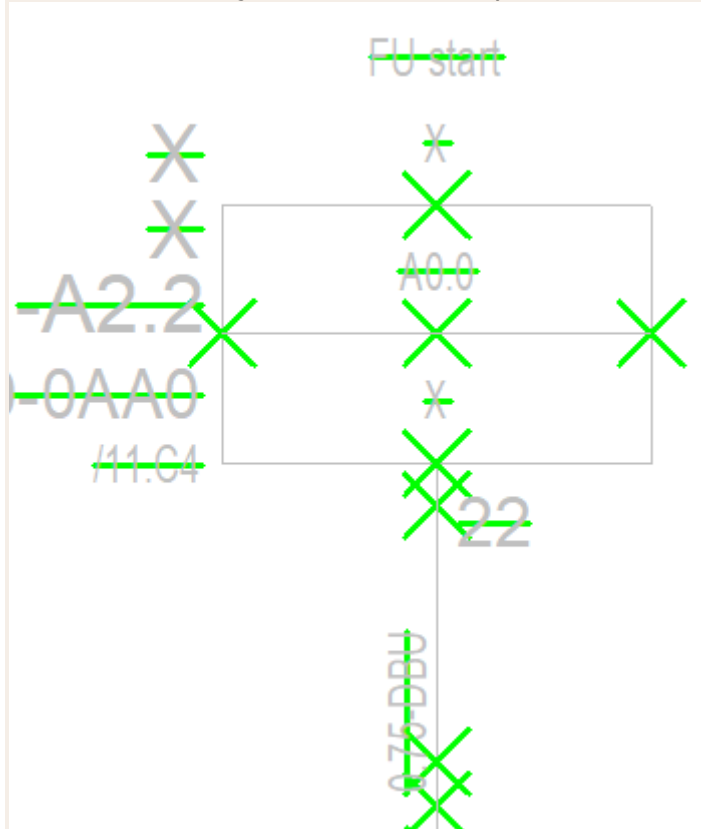
`false`: Deleted objects are not displayed in the comparison file. This parameter has an influence on the graphical comparison only if `BaseProject = NewFile` is defined in the configuration.

**MarkDeleted**

The parameter specifies whether deleted objects are highlighted in the comparison file.

The following values are possible:

`true`: Deleted objects are marked by an 'x' in the comparison file:



The default value for `MarkDeleted` is `true`.

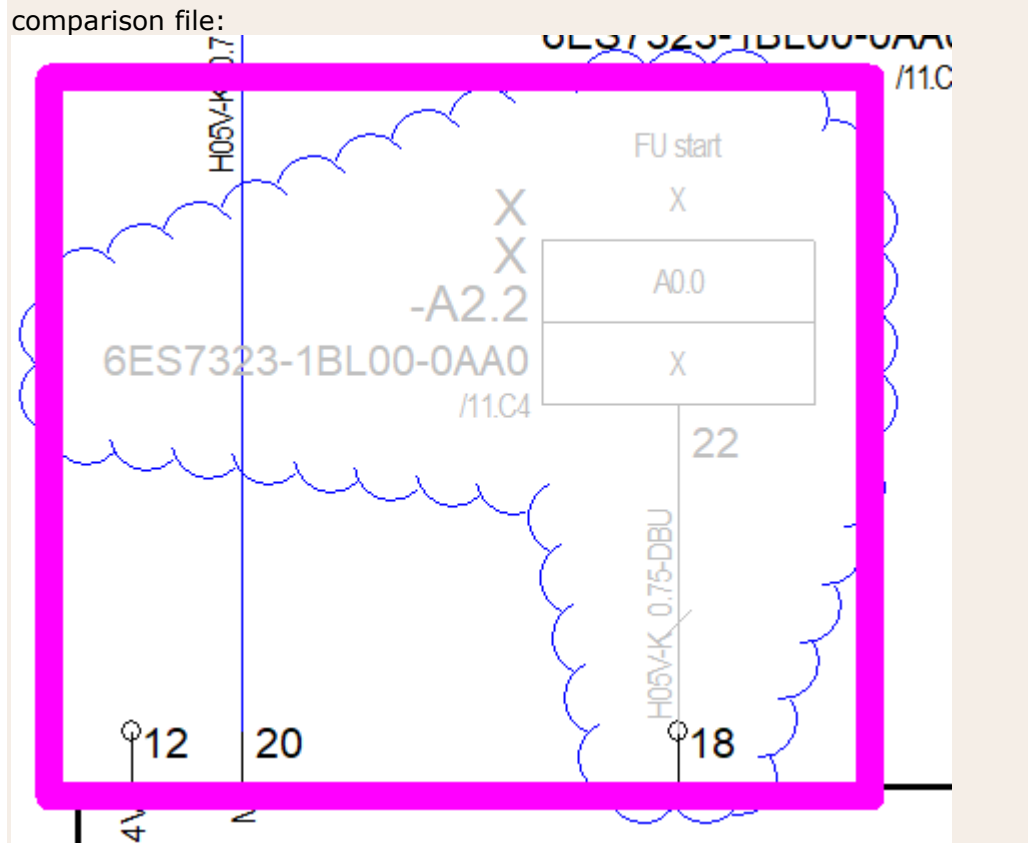
`false`: Deleted objects are not marked in the comparison file. This parameter has an influence on the graphical comparison only if `BaseProject = NewFile` is defined in the configuration.

**EncloseDeleted**

The parameter specifies whether deleted objects are highlighted with the graphic type **Cloud** in the comparison file.

The following values are possible:

`true`: Deleted objects are highlighted with the graphic type **Cloud** in the



The default value for `EncloseDeleted` is `true`.

`false`: Deleted objects are not highlighted with the graphic type **Cloud** in the comparison file. This parameter has an influence on the graphical comparison only if `BaseProject = NewFile` is defined in the configuration.



**Reference:** Designer-35298

### New Text Type for Segments

**E<sup>3</sup>.series** provides a new text type to display information about segments.

The new text type allows additional information to be associated with the project. The following text type has been added:

Text Type	Text Name	Max. Length	Owner	Description

<b>118</b> <b>5</b>	Segment outer diameter	255	Attribute text template	Text type for the outer diameter of segments
------------------------	------------------------	-----	-------------------------	--

**Reference:** Designer-39805

### Combine Text Types

Text types and fixed texts can be combined in a single text type.

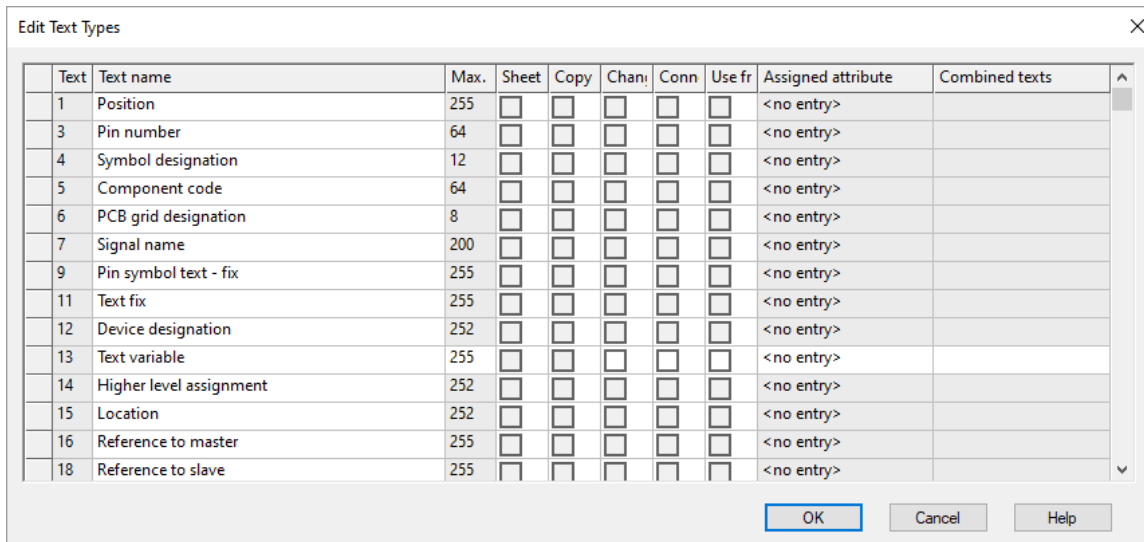
This allows multiple pieces of information about an object to be displayed by a single text type.

To combine a text type with other text types, select **Tools** → **Start Database Editor** in the main menu.

The Database Editor opens.

Open the list of all available text types under **Format** → **Text Types...**

The **Edit Text Types** dialog opens:



All text types that can show combined texts can be edited in the column **Combined texts**.

The text types must be specified in the format **<#Number>**.

**Example:** The combined text **<#7> - <#1037>** shows the text type **Signal name**, the separator **-** and the text type **Sheet reference**.

The following restrictions apply to combined texts:

- System-defined text types cannot be combined.
- Text types that display attributes cannot be combined.
- Text types with the owner **Sheet** can only be combined with text types from the same owner.
- Combined text cannot contain any other combined text.
- Text types with combined texts cannot have the properties **Copy from selected sheet**, **Changeable by script only**, **Connection target** and **Use from active pin terminal**.

When **Combined texts** are used on symbols, the following restrictions also apply:

- Texts referenced by combined texts must also be present on the symbol. If the text is missing on the symbol, a warning is issued. The corresponding combined text in the project remains empty.
- If combined texts contain a text type that must be bound to a pin point, the combined text must also be bound to a pin.

**Reference:** Designer-04221

## Panel (also: 3D)

- **3D Display of Individual Panel Sheets**
- **Setting for Automatic Connections in Panels**
- **Mark Cable Duct Inlets and Outlets in the 3D View**
- **Correction Factor for Space Requirement of Cables in Cable Ducts**


### 3D Display of Individual Panel Sheets

When editing control cabinets in **E<sup>3</sup>.series**, the 3D display of individual panel sheets can be activated.



Until now, all panel sheets were automatically activated for the 3D display.

This enhancement reduces the loading time when switching from 2D to 3D and improves the processing of designs on a specific panel sheet.


There are two ways to display a panel sheet in 3D:

1. Open the context menu of a panel sheet in the sheet tree and select **Open in 3D (Sheet)**.
2. Open a panel sheet and select  in the **Panel** toolbar.



The content of the panel sheet is displayed in 3D.

**Note:** Shared sheets and base sheets display the same contents for the 3D-display with  and .

There are two different ways to display the contents of all panel sheets in the project in 3D:

1. Open the context menu of a panel sheet in the sheet tree and select **Open in 3D (World)**.
2. Open any panel sheet and select  in the **Panel** toolbar.

The contents of all panel sheets are displayed in 3D.

**Note:** The loading process may take longer. If there is only one panel sheet in the project, the displayed content is identical with  and .

**Reference:** Designer-36735

### Setting for Automatic Connections in Panels

The settings for checks in panel have been enhanced.

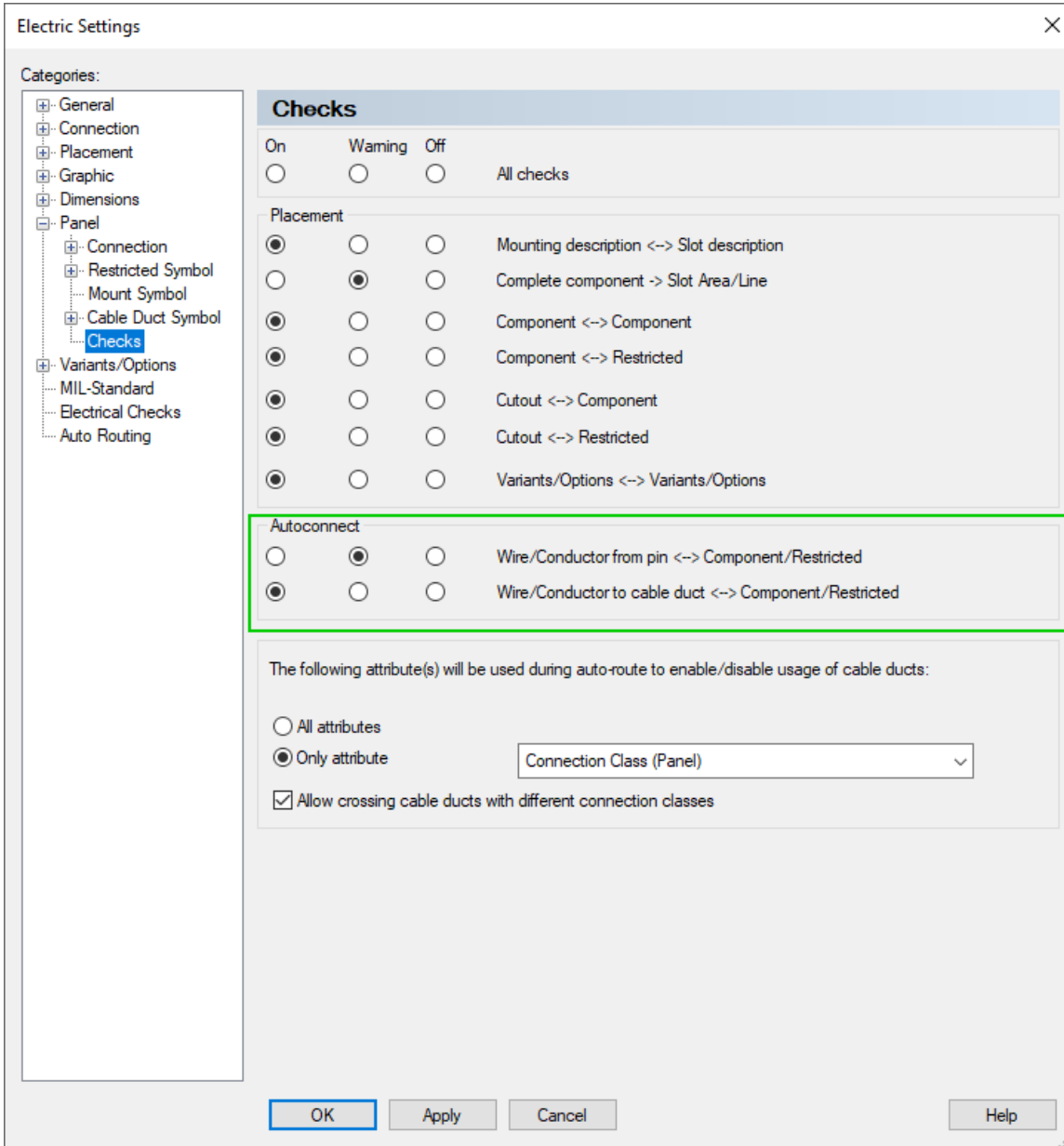
The enhancement allows better control of whether automatically routed connections can be made even if this results in collisions.

To specify the checks for automatic connections in the panel, select the main menu command **Tools** → **Settings...**

The **Electrical Settings** dialog opens.

**Note:** The setting can be activated and deactivated using the **Electrical Settings** and **Fluid Settings** dialogs.

Go to the sub-category **Panel** → **Checks** and specify the settings for **Wire/Conductor from pin <--> Component/Restricted** and **Wire/Conductor to cable duct <--> Component/Restricted** under **Autoconnect**. The setting is project-specific:



The settings have the following effects in the project:

## Autoconnect

The following checks are available for autoconnect in the panel:

- **Wire/Conductor from pin <--> Component/Restricted**

This setting allows you to control whether wires/conductors may be routed automatically if they pass through a component or restricted area on their way to or from a pin and cause a collision in the process.

If **On** is activated, wires/conductors cannot be routed during autoconnect if a collision with components or restricted areas is detected on the way to or from a pin.

If **Warning** is activated, wires/conductors can be routed during autoconnect even if a collision with components or restricted areas is detected on the way to or from a pin. A warning is displayed in this case.

If **Off** is activated, wires/conductors can be routed during autoconnect even if a collision with components or restricted areas is detected on the way to or from a pin. A warning is not displayed in this case.

- **Wire/Conductor to cable duct <--> Component/Restricted**

This setting allows you to control whether wires/conductors may be routed automatically if they pass through a component or restricted area on their way to or from a cable duct and cause a collision. This case occurs, for example, if one pin is on the front of a mounting plate and the other pin is on the rear of the mounting plate.

If **On** is activated, wires/conductors cannot be routed during autoconnect if a collision with components or restricted areas is detected on the way from or to a cable duct.

If **Warning** is activated, wires/conductors can be routed during autoconnect even if a collision with components or restricted areas is detected on the way from or to a cable duct. A warning is displayed in this case.

If **Off** is activated, wires/conductors can be routed during autoconnect even if a collision with components or restricted areas is detected on the way to or from a cable duct. A warning is not displayed.

**Reference:** Designer-37964

### Mark Cable Duct Inlets and Outlets in the 3D View

Cable duct inlets and outlets are marked in 3D views and in the PDF, STEP and XVL export formats.

This means that cable ducts with inlets or outlets can be immediately distinguished from those that do not have an inlet or outlet, even in the 3D views.

To display cable duct inlets and outlets in the 3D view of the project, open the context menu of a panel sheet with the defined cable duct inlets and outlets in the tree view and select **Open in 3D (Sheet)** or **Open in 3D (World)**.

Depending on the selection, either only the contents of the selected panel sheet or the contents of all panel sheets are displayed. Cable duct inlets and outlets are shown with an arrow in the 3D view.

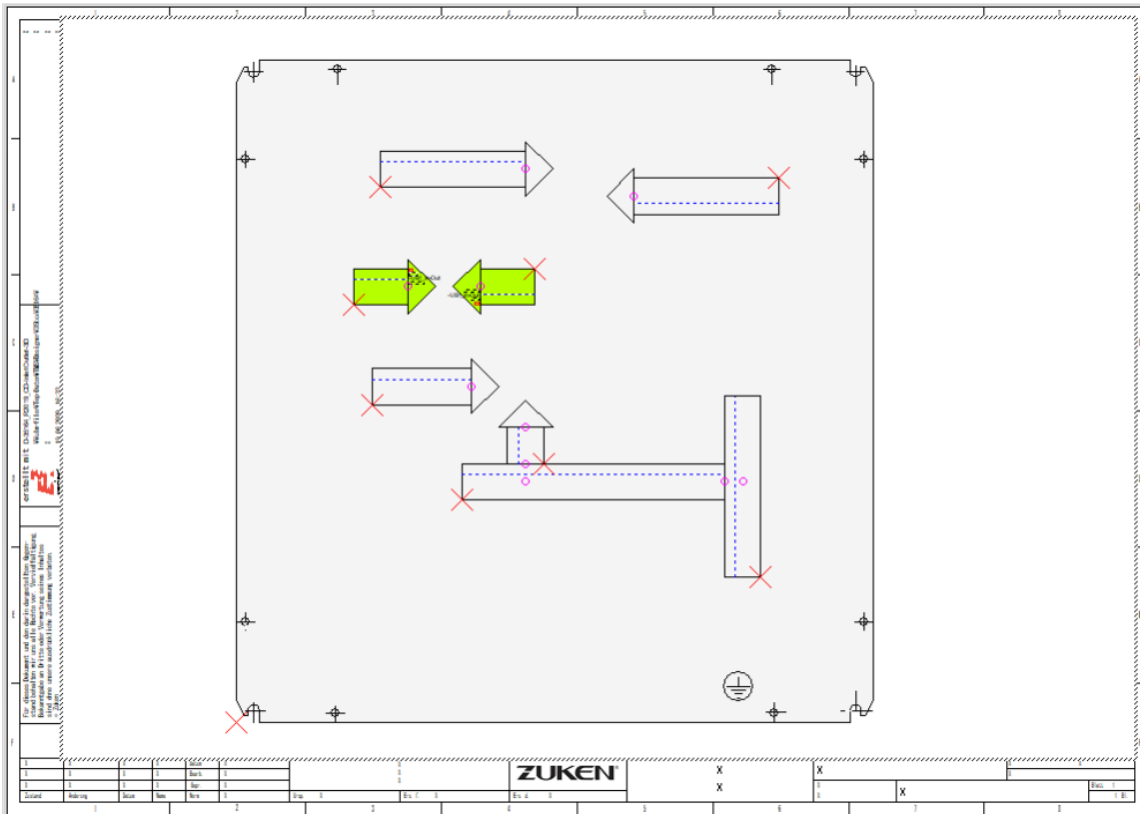


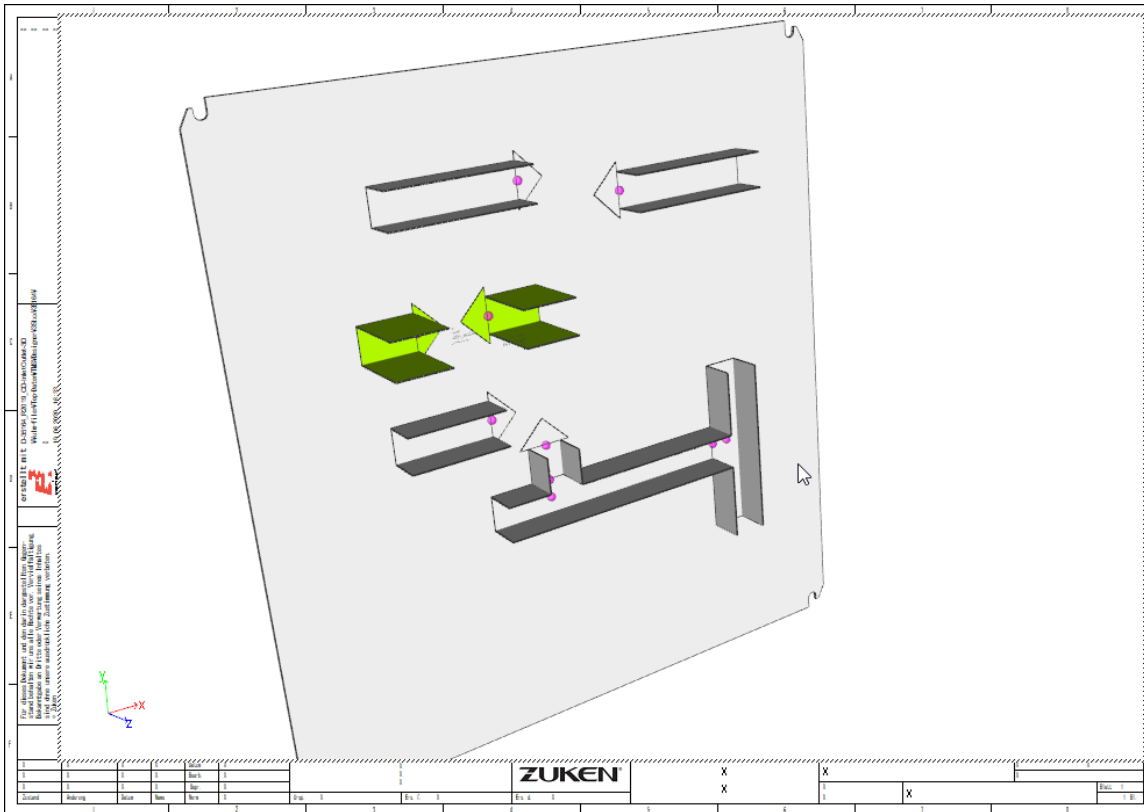
To display the cable duct inlets and outlets in an exported format in 3D, select the format **PDF...**, **STEP...** or **XVL...** under **File** → **Export** in the main menu:

- For exporting with **PDF...** the option **3D Export** must be selected in the **PDF Files** dialog.
- For exporting with **STEP...** the panel sheet to be exported must first be selected in the tree view.

Cable duct inlets and outlets are shown with an arrow in the exported file.

**Comparison of 2D and 3D Views:**





**Reference:** Designer-35164

### Correction Factor for Space Requirement of Cables in Cable Ducts

The calculated space requirement of cables in cable ducts can be multiplied by a correction factor.

The correction factor makes the value for the space requirement more precise because only the circumferences of the conductors/cables are used in the mathematical determination of the space requirement.

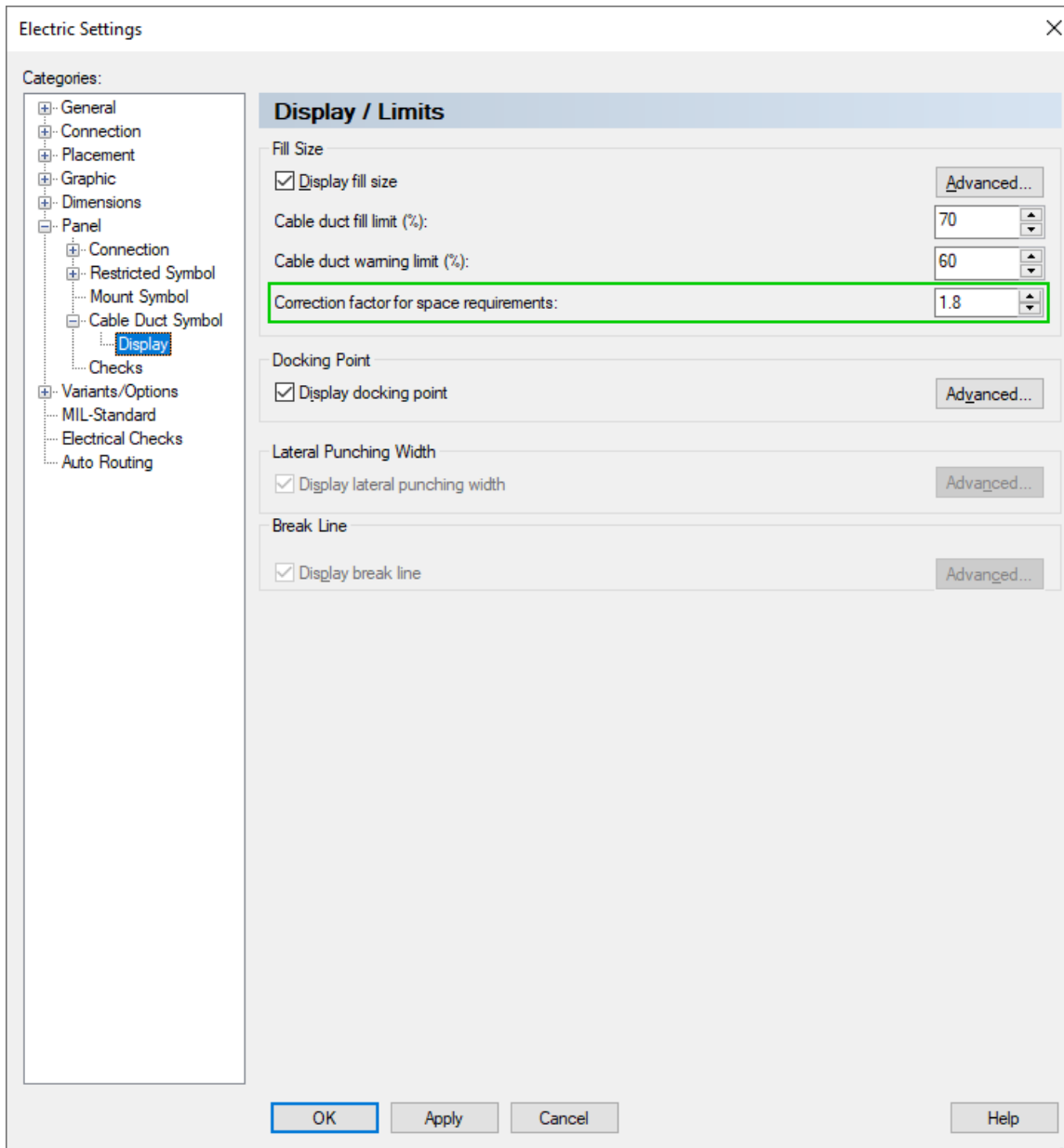
In practice, this ideal condition can hardly be achieved, since overlapping and crossings, for example, lead to increased space requirements.

To use a correction factor when calculating the space requirement, select the main menu command **Tools** → **Settings...**

The **Electrical Settings** dialog opens.

**Note:** The setting can be activated and deactivated using the **Electrical Settings** and **Fluid Settings** dialogs.

Go to the sub-category **Panel** → **Cable Duct Symbol** → **Display** and set a value for the **Correction factor for space requirements**. The setting is project-specific:



Values between 1.0 and 5.0 are permitted for the factor.  
The factor is multiplied by the calculated space requirement.  
With a value of 1.0, the space requirement does not increase accordingly.

The recommended value is 1.8, which in practice is a good approximation of the realistic space requirement. The value corresponds to the increase of the space requirement by 80%.

**Reference:** Designer-39801

## Tables

- **Display Subcircuits in the Component Table**

### Display Subcircuits in the Component Table

It is now possible to optionally display which components are created as subcircuits in the component table.

In addition, the attributes of all components show whether the attribute **subcircuit** is assigned. The value of the attribute is the name of the subcircuit.

This enhancement, for example, makes it even easier to search for subcircuits.

Subcircuits can also be placed directly on a sheet from the component table and used in the project.

To display the table, select **View** → **Database Window** → **Component Table** in the main menu.

The **Component Table** opens:

Component Table

Favorite  
Name:  Save Rename Delete

Search  
Device letter code:  Reset

Component name	Version	Version text	Device letter code	Table symbol	Default template	Preview Symbol	Topology symbol	Number of pins /	Length (of model)	Width (of model)	Depth (of model)
<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>
0-1355348-1	1	Zuken E3 Gm	X					18			
00.10048.3	1	Zuken E3 Gm	V					11			
00.6563	1	Zuken E3 Gm	V					7			
00.7127	1	Zuken E3 Gm	R					2			
00.7127.4	1	Zuken E3 Gm	R					2			
00.7667	1	Zuken E3 Gm	A					98			

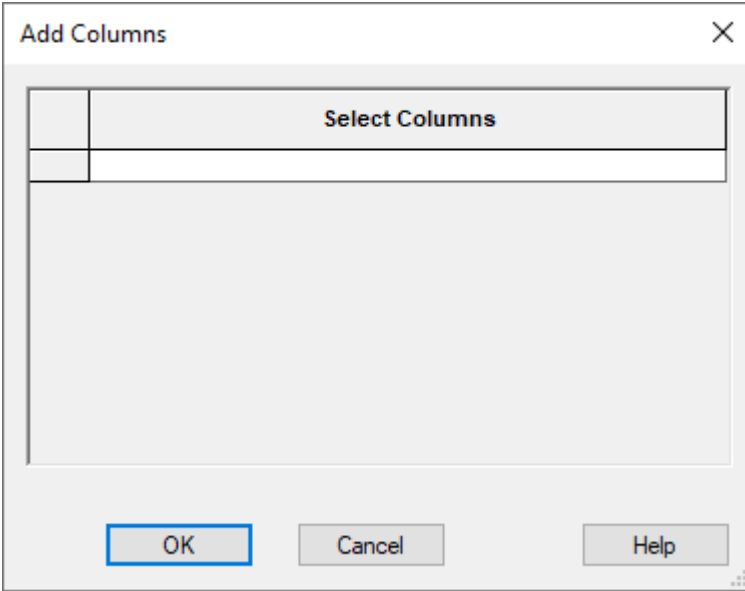
Attributes and preview

Name	Entry

Right-click on the column header and select the context menu command **Add Column**. The **Add Column** dialog opens:

Component	text	Device letter code	Table symbol	Default template	Preview Symbol	Topology symbol	Number of pins /	Length (of model)	Width (of model)	Depth (of model)
<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>
0-15		Gm X					18			
00.1		Gm V					11			
00.6	Sort Ascending	Gm V					7			
00.7	Sort Descending	Gm R					2			
00.7		Zuken E3 Gm R					2			
00.7667	1	Zuken E3 Gm A					98			

66



Click in the empty row under **Select Columns**.  
A menu with all available component attributes opens.

Select **Subcircuit** in the list.  
The additional column for the attribute is inserted.

If a component is created as a subcircuit in the database, the name of the subcircuit is displayed in the column.

In addition, the list of component attributes also shows whether the component is created as a subcircuit.

The display in the component attributes is independent of whether the column **Subcircuit** is displayed in the component table:

Component	Subcircuit	Version	Version text	Device letter	Table symbol	Default template	Preview Sy	Topology sy	Number of p	Length (of	Width (of m	Depth (of m
<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>	<All>
FLY		1	Zuken E3 Gm W						92			
FSH-BQ.3W2		1	Zuken E3 Gm FH						1			
FU-COMP_01	FU-COMP_01	1	Zuken E3 Gm A									
FU_1		1	Zuken E3 Gm F						2			
FU_1_01		1	Zuken E3 Gm F						2			
FU_3		1	Zuken E3 Gm F						6			
Fuse		1	Zuken E3 Gm						2			

Attributes and preview

Name	Entry
Article number	
Class	Sub-circuit
Component name	FU-COMP_01
Component type	27
Date	22.01.2007
Depth (of model)	
Description	Frequency converter
Device letter code	A
Length (of model)	
Main Class	Electric
Modification date	13.03.2009
Number of pins / conduc	
Subcircuit	FU-COMP_01.e3p
Supplier	Zuken E3 GmbH
Version	1

**Reference:** Designer-08225

## Connection/Busses, Signals/Logic Lines, Supply

- **Setting for Plugging Pins with Same Name to One Another**
- **Display Jumped Terminals on Panel Sheets**
- **New Object "Busbar"**
- **Rename Signal Names via the Signal Tree**

### Setting for Plugging Pins with Same Name to One Another

A setting can be used to ensure that pins can only be plugged together if they have identical names.

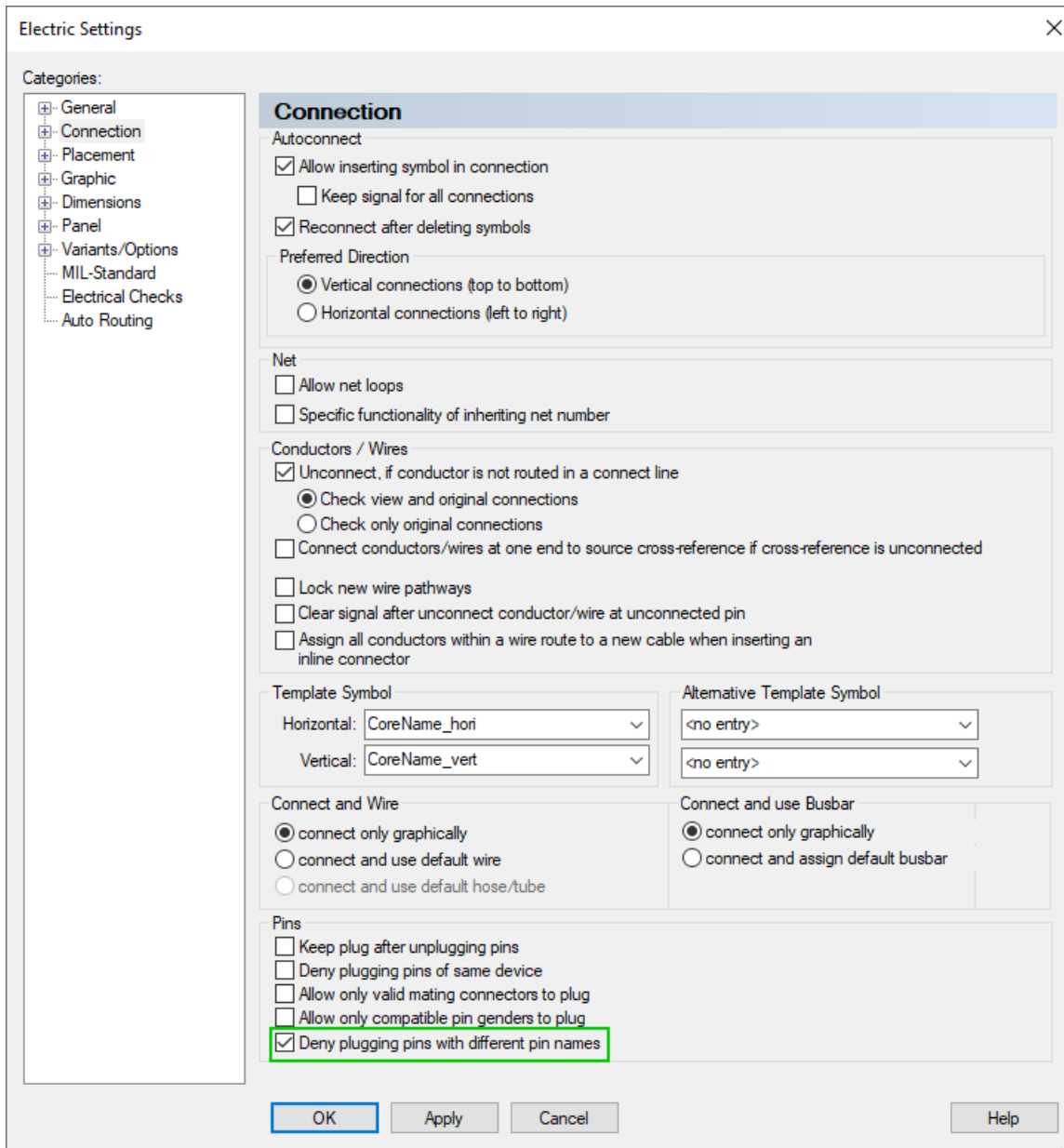
This can prevent pins with different names from being incorrectly plugged together.

To activate the setting for plugging pins with different names, select the main menu command **Tools** → **Settings...**

The **Electric Settings** dialog opens.

**Note:** The setting can be activated and deactivated using the **Electric Settings** and **Fluid Settings** dialogs.

Activate the setting **Deny plugging pins...** under **Connection** → **Pins**. The setting is user-specific and switched on by default:



If this setting is activated, only pins with the same name can be plugged together.

**Reference:** Designer-27313

## Display Jumped Terminals on Panel Sheets

Terminal pins, which are connected by a jumper, can be marked on panel sheets.

This enables a user to see whether jumpers are connected to all terminal pins or not on panel sheets.

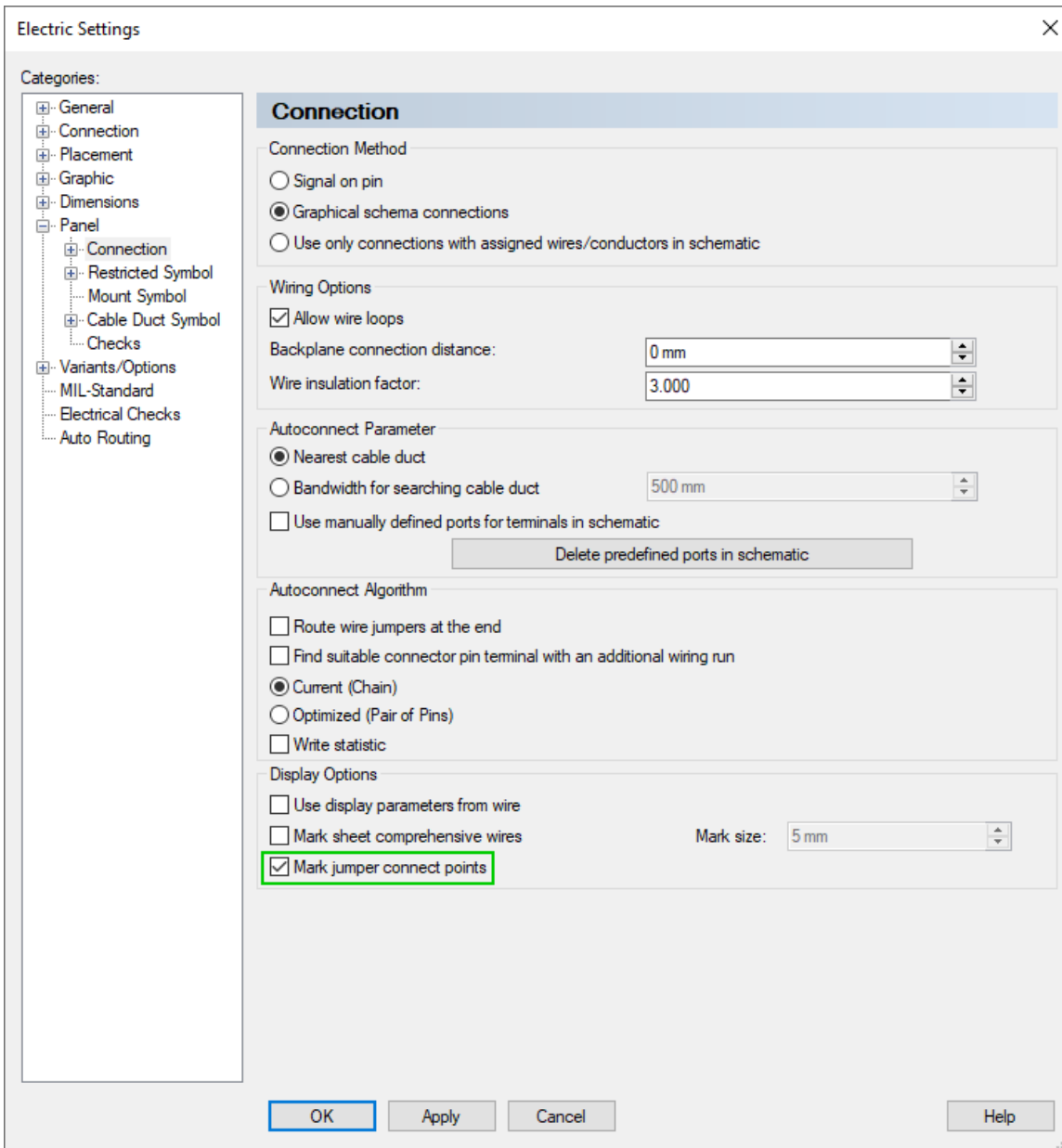
To set whether jumped terminals are marked on panel sheets, select the main menu command **Tools** → **Settings...**

The **Electric Settings** dialog opens.

**Note:** The setting can be activated and deactivated using the **Electric Settings** and **Fluid Settings** dialogs.



To do so, activate the setting **Mark jumper pins** under the sub-menu **Panel** → **Connection**.  
The setting is user-specific:



If the setting is active, all terminal pins connected by a jumper are highlighted on panel sheets. For pins not connected by the jumper, the connection line is "above" the pin without

any special marking:



The setting **Use display parameters from wire** is used for displaying the marking. If the setting **Use display parameters from wire** is not active, the connection line properties are used for displaying.

**Note:** If a user-defined line style between 5 and 24 is used from the `FONTSDAT.DAT` file, the line color cannot be changed.

The line color **Automatic** is used for line styles 5 to 25.

The setting affects the following export formats:

- DWG/DXF
- PDF
- DGN
- SVG
- CGM

**Reference:** Designer-37204

## New Object "Busbar"

In **E<sup>3</sup>.series** there is the new object type **busbar**.

**Note:** Busbar connections are not displayed in pin tables, terminal tables, connection tables and cabling tables.

Furthermore, only the names of busbar pins are displayed in terminal tables and terminal plans and not the connection targets.

The most important enhancements are displayed in the following sections:

## Attribute Text Templates for Busbars

For busbars and busbar connections, the attribute text templates of cables and conductors can be used.

Attribute text templates, which for example use the text type **Cable in connection**, display the item designation of the busbar. The text type **Conductor in connection** on the other hand displays the name of the busbar.

If attribute text templates display properties that are not available for busbars or busbar connections, nothing is displayed.

### Connect devices with busbars in the panel

When busbars are placed on devices, it is possible to see whether the busbar is electrically connected to the device.

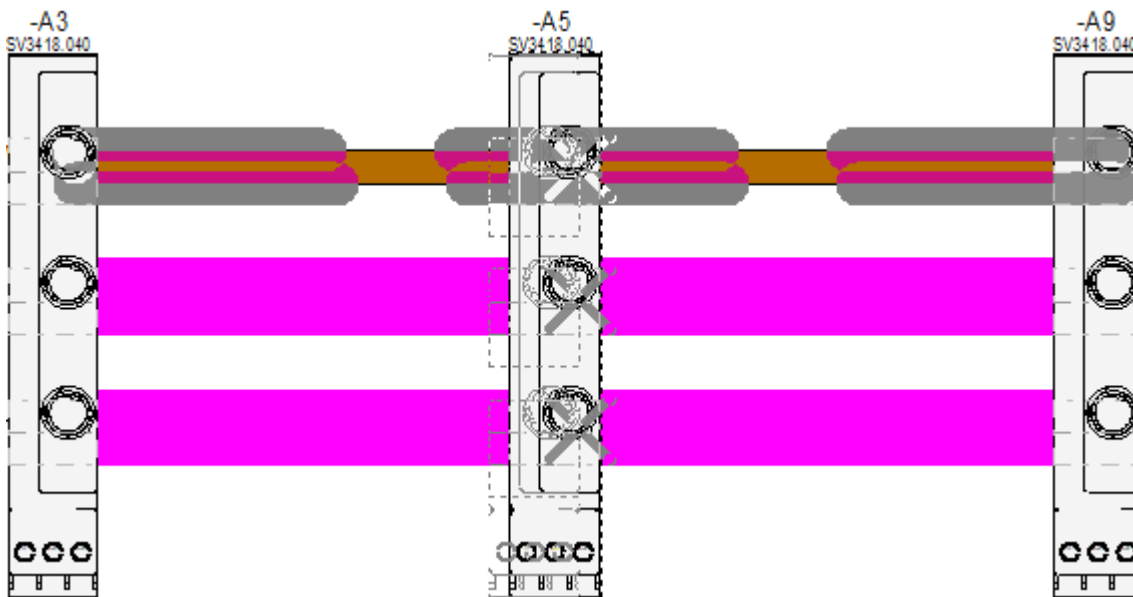
Suitable **busbar** type pins are highlighted with an X.

Busbar pins of devices, which are not connected when moving, are displayed as circles. The line thickness of the marking is set with the setting **Highlight - Width**.

If devices are plugged onto busbars, it is indicated whether the busbar is electrically connected and additionally also whether a physical connection is established.

Three parallel busbars are placed horizontally in the following illustrations.

The three devices **-A3**, **-A5** and **-A9** are busbar adapters, which can be connected with busbars:



In the example, the device **-A5** is moved. The three gray 'X's indicate that all three busbars are electrically connected to the pins of device **-A5**.

When an electrical connection is created, a message is displayed in the status bar:

Pin and busbar are electrically connected: '-A10:B->-WA4'

The gray outline of the upper busbar indicates that the slot of the upper busbar is plugged onto the upper slot of device **-A5** and thus physically connected.

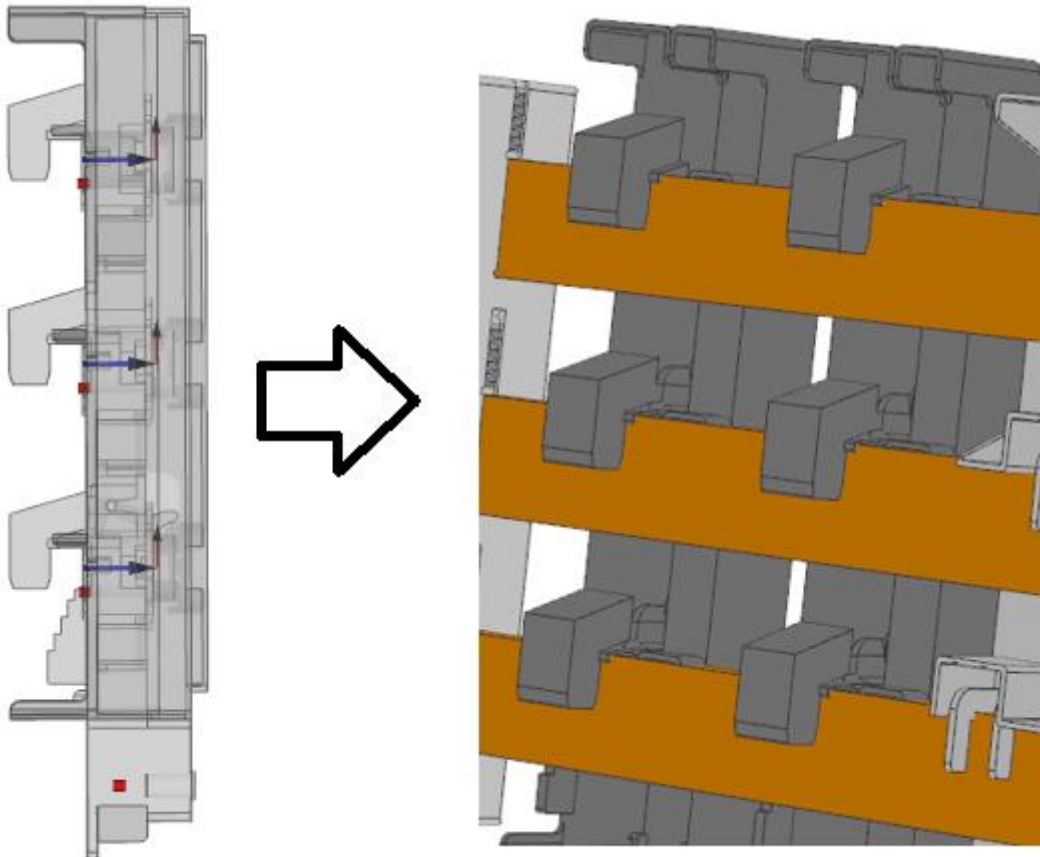
**Note:** Devices can only be plugged (mechanically) onto one busbar at a time. The alignment for the plugged connection takes place based on the pin which is connected to the busbar.

If the device can be plugged onto different busbar models when it is moved, the following prioritization decides which busbar the device is plugged onto:

1. The busbar on whose slot the device is already placed.
2. The busbar whose slot is highest in the direction of placement.

3. The busbar with the slot that has the least distance to the device pin in the placement plane.
4. The Y, X or Z position of a pin in the model that is found first. All pins are compared from top to bottom first based on their Y positions. If no pin is found that comes before all others as the only pin, all pins are compared from left to right based on their X position. If this check also fails to find a pin that comes before all others as the only pin, all pins are compared from bottom to top based on their Z position.


Make sure that the cutouts and device pins are defined in such a way that there is no collision when connecting devices to busbars and that contact can be made between the pin and the busbar:




### How busbars are displayed

Busbars are displayed as long, brown rectangles on sheets:



Busbars are displayed in tables and tree views with the following icon: 

Busbars, which are placed on busbar connections, are displayed with the following icon: 

Busbars, which are not placed on busbar connections, are displayed with the following icon: 

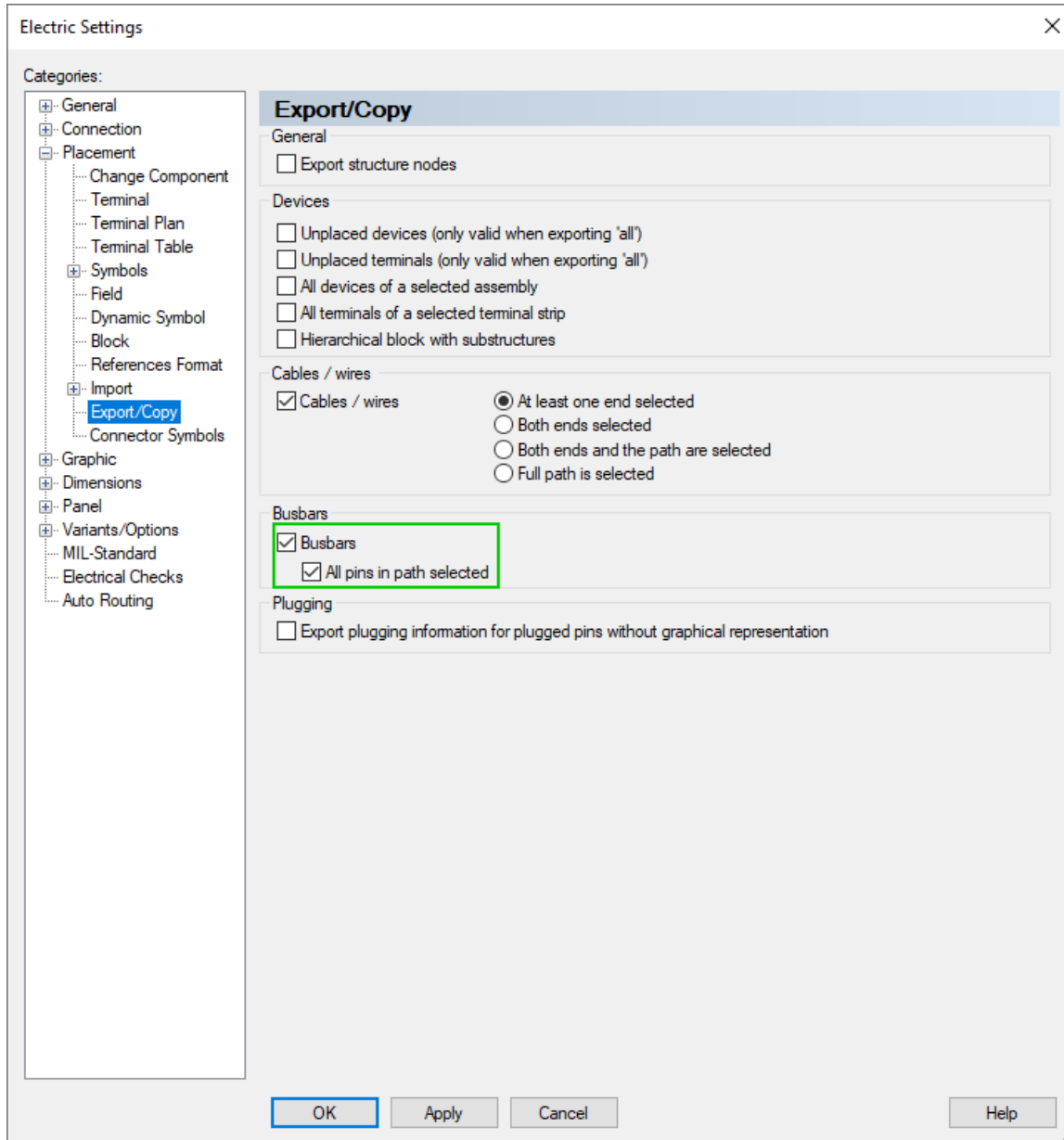
Device pins, which can be connected with busbars, are displayed with the following icon: 

## Settings for Exporting Busbars

There are two ways to export busbars:

1. To define the settings for exporting busbars, select the main menu command **Tools** → **Settings...**  
The **Electric Settings** dialog opens.

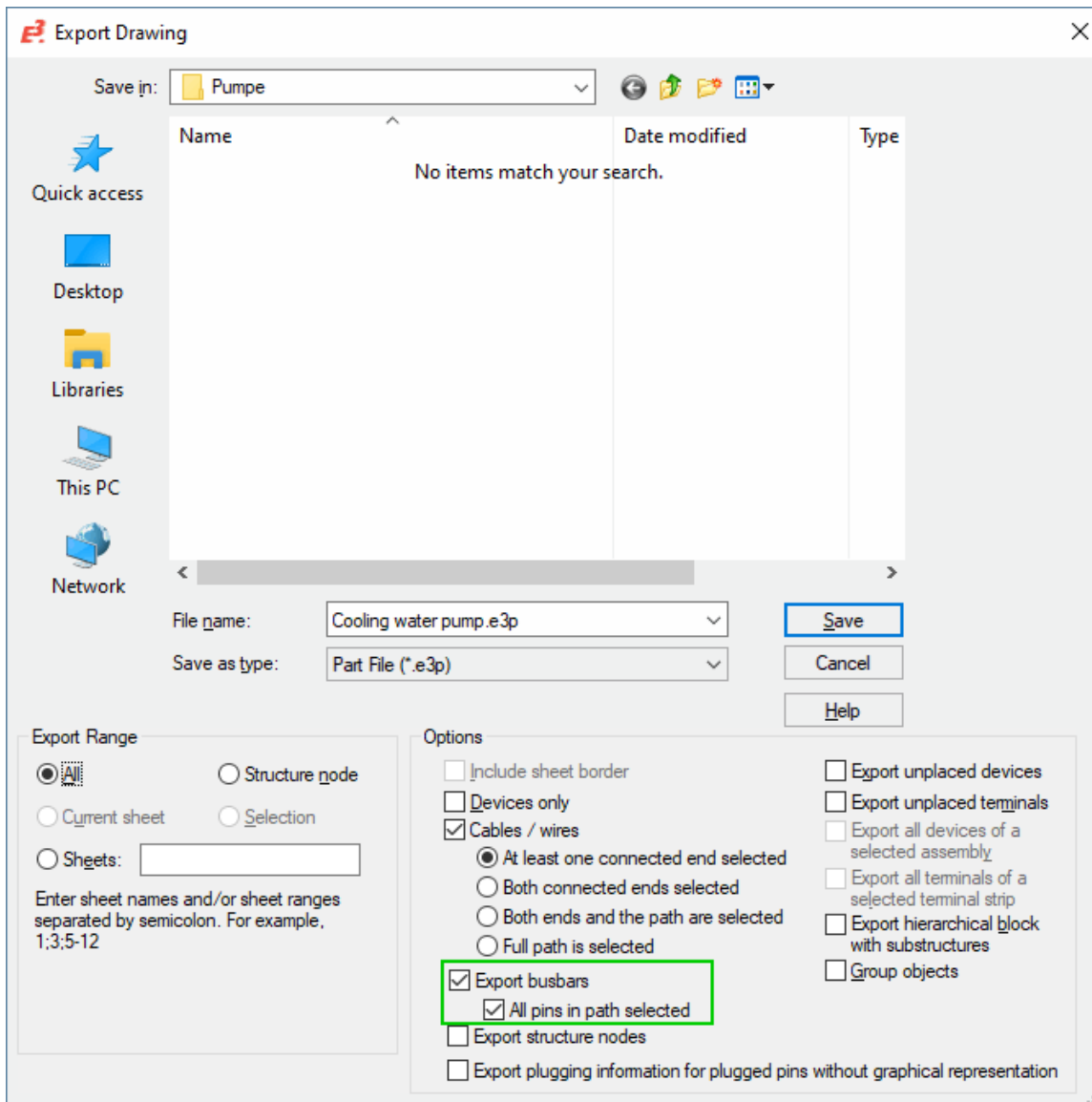
Activate the setting **Busbars** or optionally **All pins in path selected** under **Placement** → **Export/Copy**. The settings are user-specific:



As long as the setting **Busbars** is active, busbars are also copied when exporting. If the setting **All pins in path selected** is active, busbars are only copied during export if all pins and net segments of the busbar have been selected.

2. To include busbars when exporting a single drawing or subcircuit, select the main menu command **File** → **Export** → **Drawing...**  
The **Export drawing** dialog opens.

Activate the setting **Busbars** or optionally **All pins in path selected**:



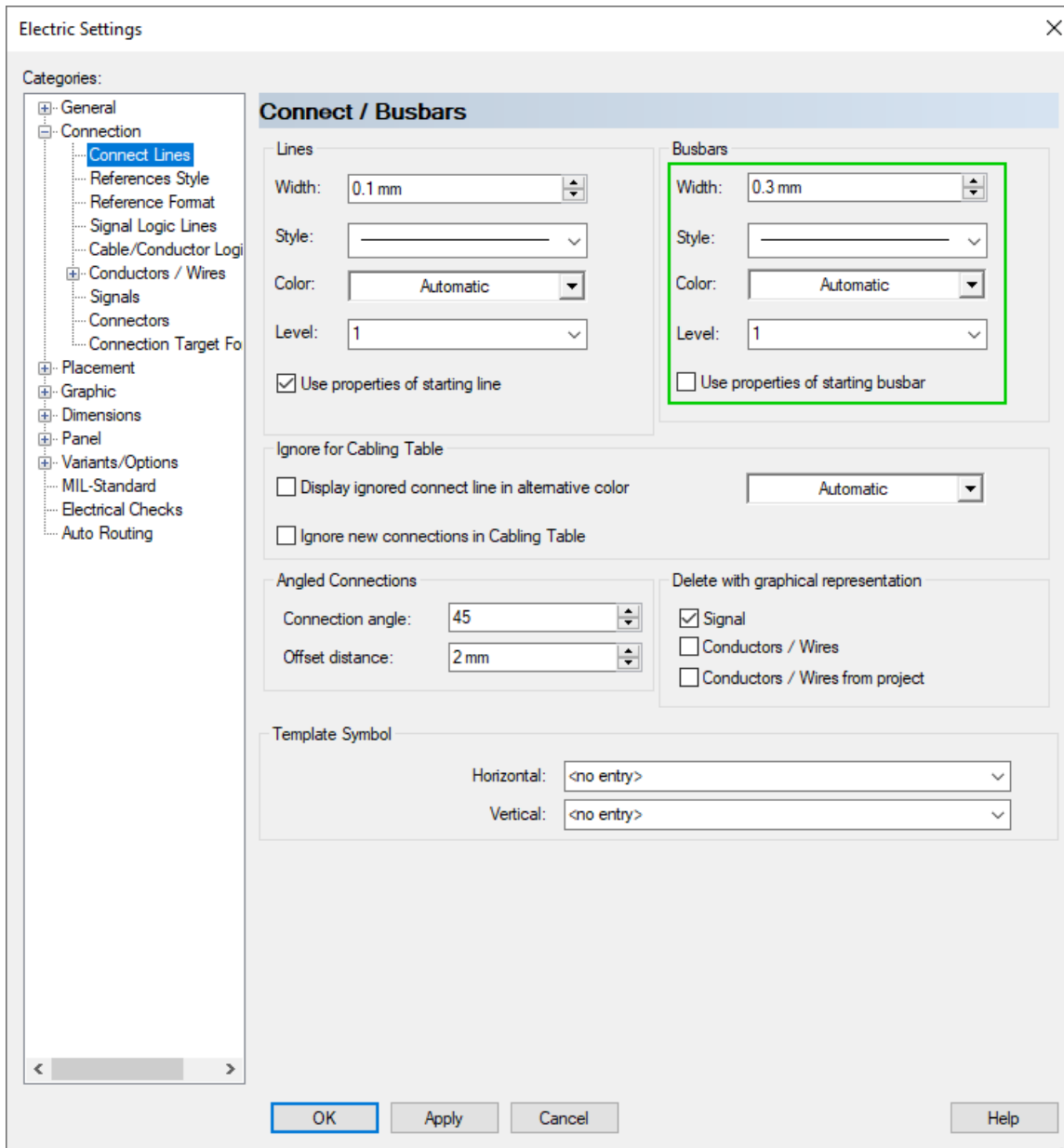
Busbars are exported according to the settings.

## Settings for Busbar Connection Lines

To define the settings for busbar connection lines, select the main menu command **Tools** → **Settings...**  
The **Electric Settings** dialog opens.

Adapt the connect line properties used for busbars under **Connection** → **Connect Lines** → **Busbars**.

The following settings are available:



- **Width:** Specifies the line width for busbar connect lines.
- **Style:** The desired line style can be selected from the drop-down list.
- **Color:** Specifies the color for the busbar connect lines.  
If **Automatic** is selected, the color defined in the **Color Definition** dialog is used.
- **Level:** Specifies the level, on which the connect lines are placed.
  - Levels between 1 and 256 are permitted.
  - Levels can be displayed globally or hidden.
- **Use properties of starting busbar:** When connecting a connection to an already existing graphical connection, its properties (width, style, color and level) are adopted depending on this option. If two connections are connected to one another, the properties of the first digitized connection line are used.

## New Attribute Owner

To be able to work with busbars, there are two new attribute owners:

- **Busbar** – the owner for attributes belonging to devices of the type *Busbar*
- **Busbar type** – the owner for attributes belonging to components of the type *Busbar*

To structure projects from **E<sup>3</sup>.series** Version 2022, Build 23.00 with attributes, the owner **Busbar** must be added to all attributes used in the structure template file.

**Example:** If it is defined in the structure template file that projects are structured with the attribute **sub-project**, it must be defined in the database editor that **sub-project** has the owners **structure node, sheet, cable, connector, device, block connector, hose/tube** and **busbar**.

### New Connection Type

The new connection type **Busbar** is used to define busbar pins.

Busbars can only be connected with pins that have the connection type **busbar**.

### Busbar Signals

Signal logic lines that are located between pins of the type **busbar** are displayed in a different color.

Each busbar forms its own signal network. Similarly, each signal network can be assigned only one busbar.

The following illustration shows the difference between signal logic lines of connections between **busbar** pins and signal logic lines of connections of other pin types with the default settings:



The line type of open signal lines can be set under the settings for signal logic lines.

The red-orange line represents an open signal logic line between two **busbar** pins. The blue line represents an open signal logic line between two pins of any other pin type.

**Note:** Signal logic lines of busbars are not displayed on panel sheets.

### Create or Change Busbars in the Database

New busbars can be created or changed using the component wizard in the database.

This allows a user to define new busbars as components that are not included in the **E<sup>3</sup>.series** supplied database, as well as modify existing busbars to meet your individual requirements.

To create or change a busbar in the database, select the main menu command **Tools** → **Start Database Editor**.

The database editor opens.



- **Change an existing busbar**

Use the following steps to modify an existing busbar:

Select a busbar whose properties you want to change in the database editor. Right-click on the busbar and select the context menu command **Edit**.

The **Component Properties** dialog opens.

Then change all properties in the tabs **Component**, **Value List**, **Busbar** and **Subcircuit** for the busbar to meet your requirements.

- **Create a new busbar**

Use the following steps to create a new busbar:

Right-click in the database editor component tree and select the context menu command **New Component**.

The component wizard opens.

Then change all properties in the tabs **Component**, **Value List**, **Busbar** and **Subcircuit** for the busbar to meet your requirements.

## Create and delete busbar connections

Connections between **busbar** objects are made with busbar connections. The pins of the objects must therefore have the connection type **busbar**.

### Start command

Busbar connections can be generated in the following ways:

- use the main menu command **Insert -> Busbar Connection**, or
- click on the **Insert busbar connection** icon in the **Connections** toolbar, or
- right-click on a busbar pin and select the context menu command **Busbar Connection** (in this case the connection is started on this pin), or
- press the key C on a busbar pin.

### Draw connection

A connection can be drawn as follows:

1. When the cursor is dragged over a valid busbar pin, it changes its appearance.
2. Left-click to start the connection on this pin.
3. Move the cursor away from the starting point and click to insert the necessary deviation points into the connection line.

### End connection

Connections can be closed as follows:

- click on a valid endpoint to establish the connection, or
- right-click and select the context menu command **Open line end**, or
- double-click anywhere in the drawing to create a connection with an open end.

This can be continued later.

### Insert connection with any angle

- press the SHIFT key while digitizing.

## Connect to same pin multiple times

- Connection lines of busbars cannot be placed on top of one another. For this reason, slanting connection lines are necessary if multiple connections are to be made to the same pin.  
Press the SHIFT key while digitizing.

## Delete connections

Busbar connections can be deleted as follows:

- mark the busbar connection and press the **Del** key, or
- mark a busbar connection and select the main menu command *Edit -> Delete*, or
- right-click and select the context menu command **Delete**.

Information regarding connections between **busbar** type objects is displayed in the **Busbar Information** dialog.

The dialog can be opened using the **Device Properties** or **Connection Properties**.

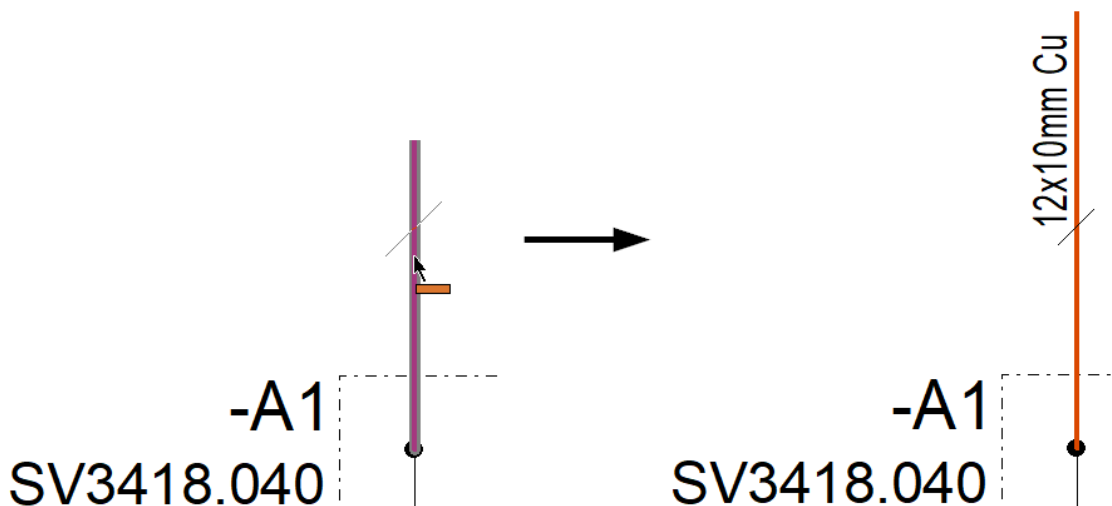
## Assign busbar to busbar connections

With busbars, which are assigned to busbar connections, the electrical properties of the connection are defined. There are two different ways to assign busbars to busbar connections:

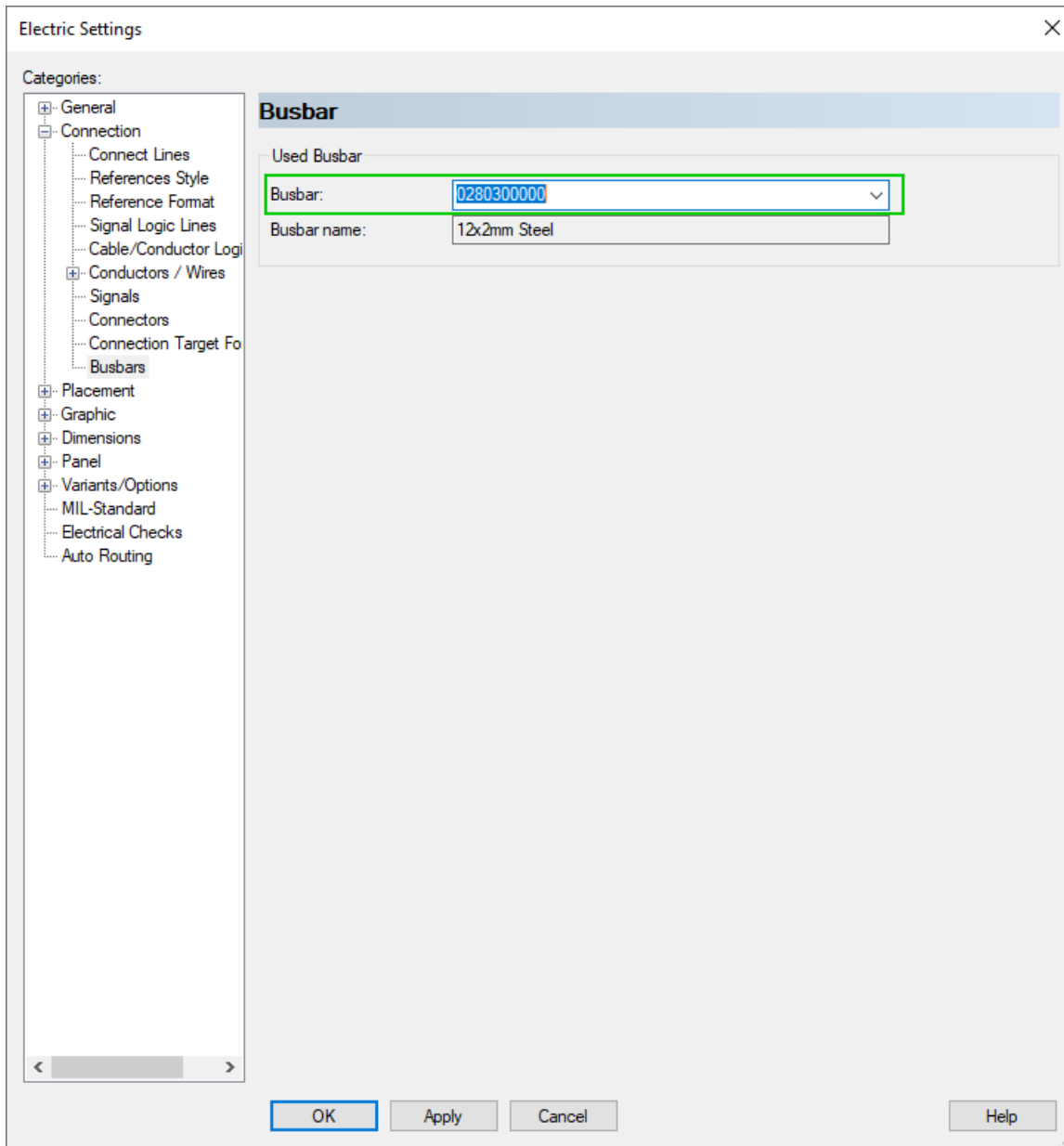
1. Search for the desired busbar to be assigned to a busbar connection in the database editor.

Then drag and drop the busbar onto a busbar connection and release the mouse button.


The following illustration shows a busbar connection during and after a busbar is assigned. The left side shows how the busbar is dragged onto the busbar connection with the mouse button held down and where the attribute template symbol is placed. The right side shows how the busbar is assigned to the busbar connection after the mouse button is released again. The default template symbol is also shown that is set for connections:



2. Open the **Busbar** toolbar and select the busbar to be used by default from the drop-down list. Alternatively, the default busbar to be used can be set under **Settings** → **Connection** → **Busbars**:



Select  in the toolbar or **Insert** → **Default Busbar** to select the desired busbar. Then click on a busbar connection to assign the busbar to it.

Alternatively,  can be activated to automatically assign the default busbar to all busbar connections created afterward.

## Place and connect busbars

Busbars can be placed and moved in **E<sup>3</sup>.series** like all other objects. All basic functions to place or move busbars can be executed by drag & drop or context menu commands.

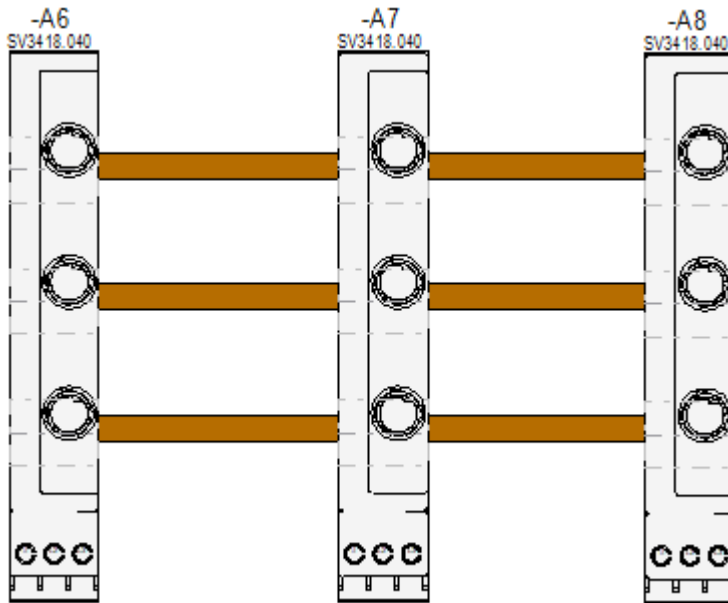
When moving placed busbars, there are the following restrictions. The restrictions depend on this:

- on what type of sheet the busbars are placed,
- whether the busbars are grouped with other objects and
- whether the busbars are part of an electrical connection.

Busbars, which are placed on **Schematic** sheets and grouped with other objects, cannot be moved.

For busbars, which are placed on **Panel** sheets, the following rules are valid.

The following illustration shows three busbars, which are placed together on a panel sheet with the devices **-A6**, **-A7** and **-A8**:



In the illustration, no groupings and adjustments have been made to the circuit diagram yet. All of the following rules are explained using the illustration. First, a hypothetical situation in the project is described and then the general rule is explained.

### Situation in the Project

The upper and middle busbars form a group.

Regardless of whether the busbars are placed in the circuit diagram, the top and middle busbars can only be moved together.

### General Rule

Busbars that are placed on panel sheets and form a group cannot be moved individually.

To move an object, all other objects of the same group must be selected and moved as well.

The devices **-A6**, **-A7**, **-A8** and the three busbars form a group.

The busbars are not placed on the devices.

The busbars are not **placed** in the schematic.

All 6 objects can only be moved together. The electrical connections between the objects remain intact.

Only the devices **-A6**, **-A7** and **-A8** form a group.

The group is placed on the busbars.

The busbars are not routed in the circuit diagram.

The busbars cannot be moved independently of the group. If the group is moved without the busbars, the electrical connection between the objects is disconnected.

The devices **-A6**, **-A7** and the upper as well as the middle busbar form a group.

The pins of the devices **-A6** and **-A7** are connected to all three busbars.

The busbars are

Busbars which are placed on panel sheets that form a group with devices, while not being placed on devices and not routed in the circuit diagram, can be moved.

The entire group is moved in the process.

The electrical connection between the busbars and devices remains intact.

Busbars, which are placed on panel sheets, are not routed in the circuit diagram and on which a group of devices is placed, cannot be moved independently of the group.

If the group is moved without the busbars, the electrical connection between the group and the busbars is disconnected.

Busbars, which are connected to device pins, cannot be moved separately from the connected devices.

not routed in the circuit diagram.

The busbars cannot be moved separately from each other.

### Rename Signal Names via the Signal Tree

Signal names can be changed directly via the signal tree.

This means that signals, such as sheets or devices, can be revised in the tree view. Furthermore, the renaming can be done more quickly without having to use the **Signal Properties** dialog.

To change a signal in the signal tree, first open the project signal tree under **View** → **Signal Tree**.

The **Signals** tab opens in the project window.

Then highlight a signal in the signal tree, press the F2 key and enter the new name of the signal.

The signal is renamed and sorted according to its new name in the signal tree.

**Note:** The signal **\*\*NC\*\***, signals with a signal definition and signals from the XML structure file, cannot be renamed.

**Reference:** Designer-19892



## Americas

### North America

Zuken USA Inc.  
Westford, MA 01886, USA  
Tel: +1 978 692 4900

### Asia

#### Japan

Zuken (Worldwide Head Office)  
Yokohama, Kanagawa 224-8585, Japan  
Tel: +81 45 942 1511

#### China

Zuken (Shanghai) Technical Center Co.,  
Ltd.  
Room 301, No.555 Nanjing West Road,  
Shanghai, 200041, People's Republic of  
China  
Tel: +86-21-3218-1784

#### Korea

Zuken Korea Inc.  
Seoul 135-283, Korea  
Tel: +82 2 5648031

#### Singapore

Zuken Singapore Pte Ltd.  
#22-05 Gateway East, Singapore 189721  
Tel: +65 6392 5855

#### Taiwan

Zuken Taiwan Inc.  
Taipei 110, Taiwan  
Tel: +886 2 7718 1116



## Europe

### Germany

Zuken GmbH (European HQ)  
D-85399 Hallbergmoos, Germany  
Tel: +49 89 7104059 00

Zuken E3 GmbH  
D-89079 Ulm, Germany  
Tel: +49 7305 9309 0

Zuken E3 GmbH  
D-30659 Hannover, Germany  
Tel: +49 511 8595 9489

### Switzerland

Zuken E3 GmbH  
CH-5504 Othmarsingen, Switzerland  
Tel: +41 62 561 08 00

### United Kingdom

Zuken UK Ltd.  
Bristol, BS32 4RF, UK  
Tel: +44 1454 207 801

### France

Zuken S.A.  
#91974 Les Ulis Cédex, France  
Tel: +33 1 69 29 48 00

### Italy

Zuken S.r.l.  
20090 Milanofiori Assago, Milan, Italy  
Tel: +39 02 575 921

### Netherlands

Zuken GmbH  
NL-6075 HA Herkenbosch, The Netherlands  
Tel: +31 475 520 998