commend

PRODUCT MANUAL

ENGLISH

ComPLC

MANUAL VERSION 4.1/0321

This manual refers to
software version 4.3

COMMEND INTERNATIONAL GMBH
Saalachstraße 51
A-5020 Salzburg – Austria
www.commend.com

Product manual ComPLC

Edition March 2021

Version 4.1/0321

Legal Notice:

Number of pages: 95

Content

1. General information

1.1 PDF format

The PDF manual can be used with the following viewers:

- Adobe Reader from version 6.0
- Foxit Reader from version 5.0

1.2 Info boxes

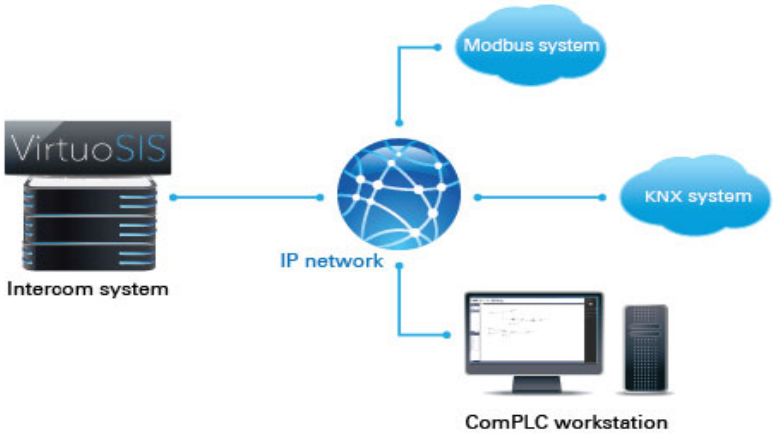
ATTENTION
This information box addresses all necessary configuration options and actions, which will allow error-free operation. In addition, this box warns you if a certain configuration, option or entered value could lead to a malfunction of the application or a loss of data.

NOTE
This information box serves multiple purposes: The information included ranges from basic configuration tips to additional advice and comments on occasional settings and events.

GOOD TO KNOW
This information box provides background knowledge, which helps to understand the basic principles behind different configurations and the application itself.

1.3 Introduction

ComPLC is an application to design and execute logical timed sequences and operations in a descriptive user interface. ComPLC supports the most important Intercom elements as well as KNX or Modbus interfaces. These options make it possible to interlink the world of Intercom with building automation of KNX and the most common PLC systems of Modbus in a very flexible manner. The solution you create with ComPLC runs on the virtual application card of VirtuoSIS.



ComPLC overview

NOTE: Further information
For more information about VirtuoSIS, see manual "**VirtuoSIS Setup Guide**".

One of the key benefits is the implemented "Online Mode". A simple click on a button in the user interface will list all traced operations of the custom-configured logic controls in real time and allows direct tracking of design errors. For recurring sequences (same logic but different inputs and outputs), ComPLC enables you to execute your designed logic for different circumstances. The highly flexible structure of the designer's user interface provides an excellent overview even of large projects.

Once the sequences are designed, the generated logic can automatically be uploaded onto the virtual application card of VirtuoSIS.

1.3.1 Key facts

- Windows-based graphical design application
- Stand-alone execution of the designed sequences on VirtuoSIS
- The virtual application card of VirtuoSIS is configured automatically by ComPLC designer
- Use of ICX messages
- Support of webhook commands via HTTP
- Configuration of individual functional components
- Ready for KNX interface to connect up to 50 KNX NET/IP gateway devices
- Ready for Modbus interface to connect up to 50 slave devices
- Integrated online mode for tracing operations in real time
- Send emails triggered by events
- Support of SNMP functions
- Ready for Avigilon interface to send and receive alarms
- Integration of Symphony MX call state information

1.4 System requirements

For information about hardware and software requirements, see data sheet "**ComPLC**".

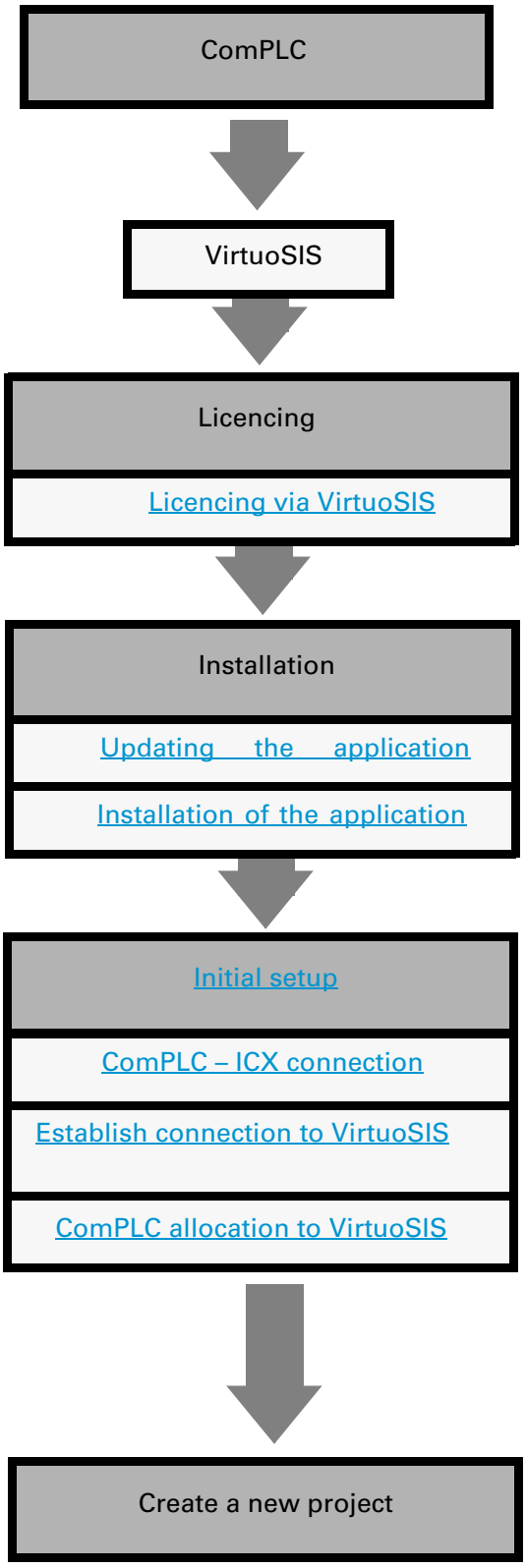
NOTE: G8-VOIPSERV is no longer supported
As of software version 4.0, ComPLC no longer supports G8-VOIPSERV. To keep the G8-VOIPSERV functionality and be able to use the new features of later versions, it is possible to keep ComPLC version 3.2 alongside newly installed ComPLC instances as of version 4.0. (Only one allocation for VirtuoSIS is possible, see "ComPLC allocation to VirtuoSIS" on page 10)

1.5 Licence requirements

For an overview of the licences for ComPLC, see data sheet "**ComPLC**".

2. Setup guide

ComPLC is operated via VirtuoSIS. There are several configuration steps, which have to be carried out beforehand. On the following pages, you will find a detailed step-by-step setup guide to initialise and to configure ComPLC. See the following illustration for a basic overview of the required configuration of ComPLC via VirtuoSIS:



2.1 Licencing

The operation of certain components requires specific licences (see data sheet “**ComPLC**”). Listed below, you will find the information on how and where to assign them in CCT 800. Further information about licence-based components can be found on page 35.

NOTE: Further information
For more information about licencing, see manual “**Intercom Server Configuration**”.

2.1.1 Licencing via VirtuoSIS

The ComPLC licences have to be allocated to any available client of the “SIS-CSA” virtual application card of VirtuoSIS.

	Address	License Product	Type	License Code	Generation	State
▸	2 / 01	L-PLC-P	Run	[redacted]	0	Live
	2 / 01	L-PLC-200	Run	[redacted]	0	Live

Licence process for VirtuoSIS

- Drag and drop the licence on the desired client of the virtual application card “SIS-CSA”.
- Click on the button **Synchronize** to accept the changes. A restart of VirtuoSIS is required.

3. Installation

3.1 Installation of the application ComPLC

ATTENTION: Pay attention for the system requirements
Observe the system requirements for your PC (see data sheet "**ComPLC**")!

- Start the setup file (.msi) and follow the on-screen instructions to install the application ComPLC.
- The application files are saved at "*Program (x86)\Commend\ComPLC 4*".
- The individual configuration settings are saved at "*%ProgramData%\Commend\Solutions\ComPLC 4*".

3.2 Updating the application ComPLC

If an older version of the application ComPLC is installed on your computer, remove the application ComPLC by uninstalling the application. After ComPLC has been uninstalled successfully, you can proceed with the installation of the newest version of ComPLC (see above).

ATTENTION: Don't uninstall version 3.2 when using G8-VOIPSERV!
As of software version 4.0, ComPLC no longer supports G8-VOIPSERV. To keep the G8-VOIPSERV functionality and be able to use the new features of later versions, it is possible to keep ComPLC version 3.2 alongside newly installed ComPLC instances as of version 4.0. Only one allocation to VirtuoSIS is possible, see "ComPLC allocation to VirtuoSIS" on page 10)

3.3 Initial setup

After installing ComPLC (see page 9), you can proceed with the following instructions, which will guide you through the initial configuration. In order to provide an easy setup, the described steps will not focus on all available functions, but only on the basic options, which are necessary for the initiation. Before proceeding with the configuration via CCT 800, the required licencing process has to be performed on the Intercom Server (see page 8).

3.3.1 Establish connection to VirtuoSIS

Configuration ComPLC

- Start the application "ComPLC".
- Click on the button ⚙ in the toolbar. The following settings have to be configured:
  - **Server**: Select the Intercom Server that shall establish a connection to the application ComPLC.
  - **IP**: In this field, enter the IP address of the logic host (VirtuoSIS).
  - **Use default credentials**: Activate this checkbox to enable the automatic login process to the Intercom Server with default credentials (user and password).
  - **Controldesk**: In this field, enter the call number of the control desk that synchronises the inbound and outbound components via ICX messages.

    NOTE: Further information
    For more information about the configuration of control desks, see manual "**Intercom Server Configuration**".

- Click on the button ➔ to close the dialogue. All changes are applied automatically.

3.3.2 ComPLC allocation to VirtuoSIS

NOTE: Initial Configuration of ComPLC via VirtuoSIS
This configuration only has to be carried out if the application ComPLC is operated via VirtuoSIS for the first time.

If it is operated via VirtuoSIS, the application ComPLC has to be allocated manually via CCT 800 to a licenced client of the virtual application card “SIS-CSA” of the respective VirtuoSIS.

Configuration of ComPLC

- Start the application ComPLC.
- Click on the button to create a new project, or click on the button to open an existing project.
- Click on the button to upload the configuration to VirtuoSIS.

NOTE: Connect ComPLC with VirtuoSIS
To upload configurations a connection between ComPLC and VirtuoSIS is required (see above).

NOTE: In case of an error
In case of an error (e.g. connection fault), a dialogue displays an error message. According to the error type, the upload dialogue will not be indicated or will be closed automatically after 20 seconds (see page 36).

Configuration CCT 800

- Menu **Intercom Server > Receive**
- Click on the Button **Receive** to receive the CCT 800 configuration from VirtuoSIS.
- **Interfaces > Data Interface >** tab **Apps**
- In the drop-down list **App** of a licenced client of the virtual application card “SIS-CSA”, select the entry “ComPLC”.

NOTE: Only for registered ComPLC version
The entry “ComPLC” can only be selected once for each ComPLC version that is registered at the respective VirtuoSIS.

- Menu **Intercom Server > Send**
- Click on the Button **Send** to send the CCT 800 configuration to VirtuoSIS.

3.3.3 ComPLC – ICX connection

In order to enable the output of ICX messages to ComPLC, it is necessary to configure an ICX connection for a control desk.

NOTE: Further information
For more information about the configuration of control desks, see manual “**Intercom Server Configuration**”.

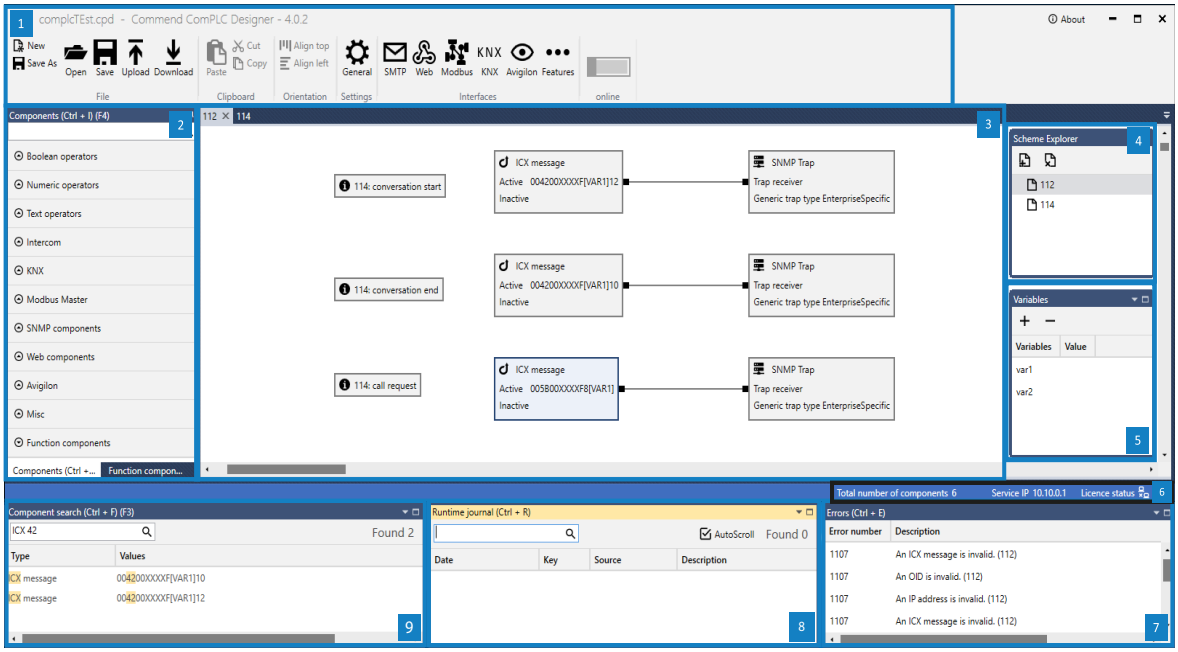
Configuration CCT 800

- **Subscriber > Control Desks >** tab **IP/RS232-ICX**
- In the drop-down lists **ICX 1 connection** and **ICX 2 connection**, select the interface for the connection between the respective control desk and ComPLC:
  - **VirtuoSIS**: Select the client that has been allocated to the application ComPLC (see page 10).
- Menu **Intercom Server > Send**
- Click on the button **Send** to send the CCT 800 configuration to the Intercom Server.

4. The application ComPLC

4.1 Application interface

After ComPLC has been started, the application interface of ComPLC is shown:



Overview: application interface of ComPLC

The application interface of ComPLC is separated into seven different parts with the following functions:

- 1 **Toolbar**: Executes basic functions (e.g. upload, save or cut), configures general settings (e.g. KNX or Modbus) and manages the online mode (see page 13).
- 2 **Component catalogue**: In this dialogue, all available components can be selected (see page 38).
- 3 **Scheme dialogue**: In this dialogue, all active schemes, instances or instance tables are indicated.
- 4 **Scheme explorer**: Manages and configures schemes and instances (see page 24).
- 5 **Variables**: Manages and configures variables (see page 31).
- 6 **Licence bar**: Shows the total number of placed licenced and non-licenced based components of all schemes (see page 35).
- 7 **Errors**: Shows occurring errors (e.g. error at upload or download or invalid object values; (see page 36).
- 8 **Runtime journal**: Shows runtime information errors and ICX messages (see page 37)
- 9 **Component search**: In this dialogue, all in the logics defined components can be listed (see page 36).

GOOD TO KNOW: Compatibility of ComPLC configuration files
Configuration files of former ComPLC versions can be imported into a succeeding ComPLC version, but not vice versa.

NOTE: SDS by default inactive
By default the function SDS is not activated. To enable SDS active the checkbox SDS in the menu **Features** (see page 20).

4.2 Upload configuration

Upload the ComPLC configuration to VirtuoSIS.

Configuration ComPLC

- Click on the button ⭱ to upload the ComPLC configuration to VirtuoSIS. The upload dialogue "File is uploading..." appears.
- After a successful upload, the dialogue "File is uploading..." will be closed automatically.

NOTE: In case of an error
In case of an error (e.g. connection fault) the upload dialogue will be closed automatically after 20 seconds or will not even be indicated (depending on the type of error), but an error message is displayed in the error message box (see page 36).

4.3 Download configuration

Download the ComPLC configuration from VirtuoSIS.

Configuration ComPLC

- Click on the button ⭳ to download the ComPLC configuration from VirtuoSIS.
- If there are unsaved changes in the selected ComPLC configuration, a save file dialogue is indicated. The dialogue "File is downloading..."appears.
- After a successful download, the dialogue "File is downloading..." will be closed automatically and the respective ComPLC configuration is opened.

NOTE: In case of an error
In case of an error (e.g. connection fault) the download dialogue will be closed automatically after 20 seconds or will not even be indicated (depending on the type of error), but an error message is displayed in the error message box (see page 36).

4.4 Toolbar

Toolbar of ComPLC

The following functions are available:

Button	Function
	New: Create a new configuration file.
	Save as: Save the configuration file with a new name.
	Open: Open an existing configuration file.
	Save: Save the configuration file.
	Upload: Upload the ComPLC configuration to VirtuoSIS (see page 12).
	Download: Download the ComPLC configuration from VirtuoSIS (see page 12).
	Paste: Insert the saved content of the clipboard.
	Cut: Cut the selection and saves it in the clipboard.
	Copy: Copy the selection to the clipboard.
	Align top: Align the selected components vertically.
	Align left: Align the selected components horizontally.
	General: Open the general setting dialogue (see page 14).
	SMTP: Open the SMTP setting dialogue (see page 15).
	Web: Open the Web setting dialogue (see page 16).
	Modbus: Open the Modbus setting dialogue (see page 16).
KNX	**KNX**: Open the KNX setting dialogue (see page 19).
	Avigilon: Open the Avigilon Setting dialogue (see page 20)
	SDS: Open the SDS setting dialogue (for more information about shooter detection systems, see manual "**ComPLC – SDS**").
	Features: Open the Features setting dialogue (see page 20)
	Online: Switch on/off the online mode (see page 21).

4.5 Section: Settings

Toolbar section: Settings

The following settings are available:

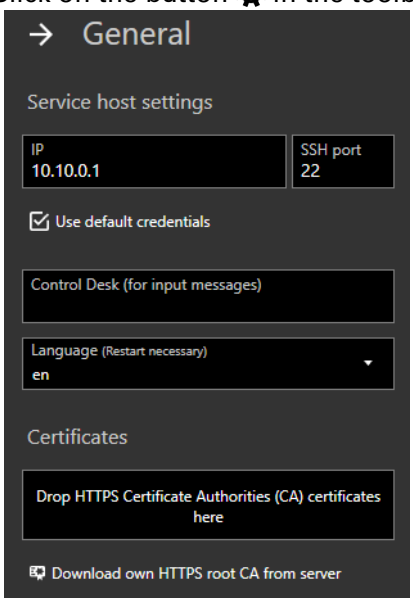
- **General** (see below)

4.5.1 General

In this section, the VirtuoSIS connection, user access and language of the application interface of ComPLC can be configured.

Service host settings

- Click on the button in the toolbar. The following dialogue appears:



Dialogue "General"

- **IP**: In this field, enter the IP address of the host system (VirtuoSIS).
- **SSH port**: In this field, the port of the SSH file transport protocol can be changed.
  → default: "22"
- **Use default credentials**: Activate this checkbox to enable the automatic login process to the Intercom Server with default credentials (user and password).
- **Controldesk**: In this field, enter the call number of the control desk that synchronises the inbound and outbound components.

> **NOTE: Further information**
> For more information about the configuration of control desks, see manual "**Intercom Server Configuration**".

- **Language**: In this drop-down list, the language of the application ComPLC can be selected.

> **NOTE: Manual restart required**
> In order to switch the interface language, ComPLC has to be restarted manually.

Certificates

Certificate Authorities (CA) are necessary to authenticate certificates for outbound HTTPS requests used in webhooks, the **Avigilon Control Center Web Endpoint** or the Symphony MX device API.

NOTE: Avigilon HTTPS certificate is mandatory
To connect ComPLC with Avigilon, a certificate is required! To use Avigilon, always upload an Avigilon HTTPS certificate in ComPLC. To create an Avigilon certificate, see the product information of the **Avigilon Control Center Web Endpoint**.

NOTE: HTTPS certificate for MX Device API access
An HTTPS certificate is mandatory for secure access to the MX Device API.

- **Download own HTTPS root CA certificate from server**
  For authentication, download the CA certificate here.
- Click on the button ➔ to close the dialogue. All changes are applied automatically.

4.6 Section: Interfaces

Toolbar section: Interfaces

The following interfaces are available:

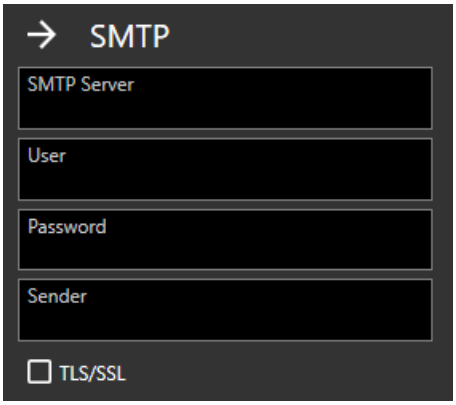
- **SMTP** (see page 15)
- **Web** (see page 16)
- **Modbus** (see below)
- **KNX** (see page 19)
- **Avigilon** (see page 20)
- **SDS** (for more information about SDS, see manual "**ComPLC – SDS**")
- **Features** (see page 20)

4.6.1 SMTP

In this section, the general e-mail settings can be configured, which are used for outbound e-mail components (see page 88).

Configuration ComPLC

- Click on the button ✉ in the toolbar. The following dialogue appears:



Dialogue "SMTP"

- **SMTP Server**: In this field, enter the IP address of the respective SMTP server.→ default port number: “25”

> **GOOD TO KNOW: Change default ports**
> To change the default port, enter the respective port number behind the server address (e. g. “mail.commend.com:568”).

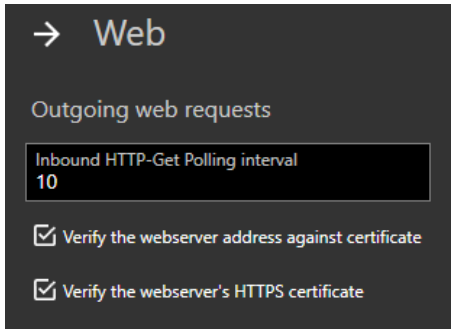
- **User** and **Password**: In these fields, enter the required credentials to connect with the desired SMTP server.
- **Sender**: In this field, enter the e-mail address of the sender.
- **TSL/SSL**: Activate this checkbox to protect the communication: TLS protocol (Transport Layer Security), SSL (Secure Sockets Layer). → default port number: “465”
- Click on the button to close the dialogue. All changes are applied automatically.

4.6.2 Web

In this section, the general web settings can be configured, which are used for web components (see “Web components” on page 79).

Configuration ComPLC

- Click on the button in the toolbar.The following dialogue appears:



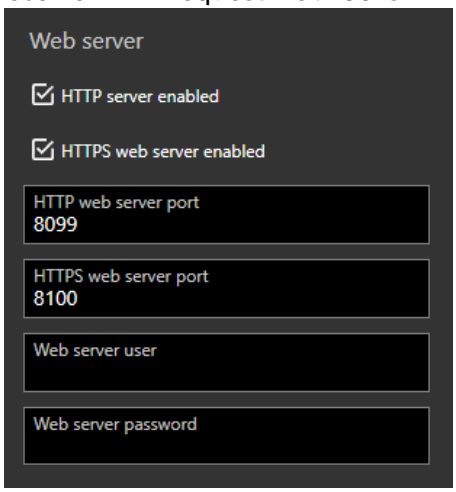
Dialogue “Web”

- **Inbound HTTP-Get Polling Interval**: In this field, enter the polling interval of the inbound HTTP-get request in seconds.
- **Verify the webserver address against certificate**: Activate this checkbox to verify the web server.
- **Verify the webserver‘s HTTPS certificate**: Activate this checkbox to verify the HTTPS certificate of the web server.

> **NOTE: Always use certificates for a secure system environment**
> For IT security reasons always use ComPLC with activated certificate verification. To import certificates into ComPLC, see “General” on page 14.

4.6.3 Webserver

- **HTTP server server enabled**: Activate this checkbox to enable the web server to receive HTTP request webhooks.
- **HTTPS web server server enabled**: Activate this checkbox to enable the web server to receive HTTP request webhooks.



Dialogue "Web server enabled"

- **HTTP web server port**: In this field, the port to receive HTTP request webhooks can be changed. → default: "8099
- **HTTPS web server port**: In this field, the port to receive HTTPS request webhooks can be changed. → default: "8100"
- **Web server username**: In this field, a user name can be entered which has to match the user name in the incoming HTTP request webhook.
- **Web server password**: In this field, a password can be entered which has to match the password in the incoming HTTP request webhook.

> **GOOD TO KNOW: Further information**
> Further information about HTTP request webhooks can be found on page 79.

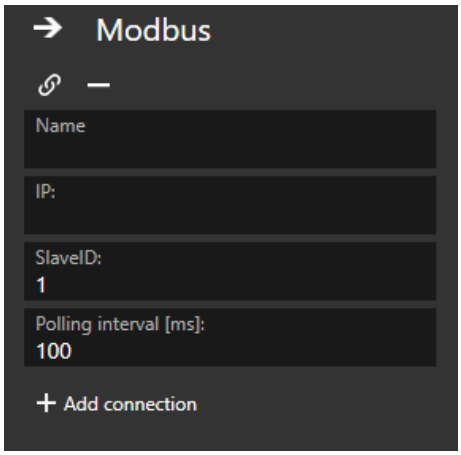
- Click on the button ➔ to close the dialogue. All changes are applied automatically.

4.6.4 Modbus

With this feature the connection settings of the Modbus slave devices can be configured.
With ComPLC version 3.0 or higher, up to 50 different Modbus connections can be configured. For each Modbus connection, a licence "L-PLC-MODBUS" is required (see data sheet "**ComPLC**").

Configuration ComPLC

› Click on the button in the toolbar and then on the button + Add connection in the right sidebar. The following dialogue appears:



Dialogue "Modbus"

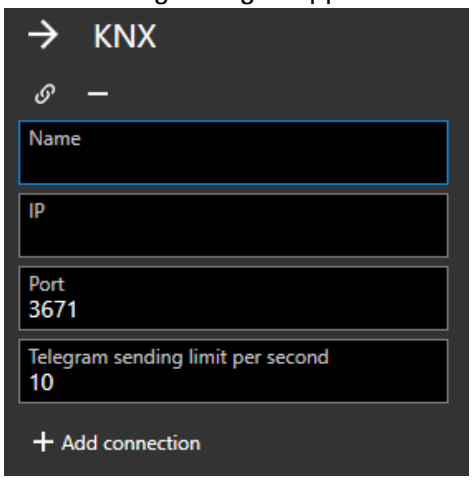
- **Name**: Enter the desired name of the Modbus connection.
- **IP**: Enter the IP address of the Modbus slave device.
- **SlaveID**: Enter the ID of the Modbus slave device (1 to 247) → default: "1".
- **Polling interval [ms]**: In this field, the polling time of the Modbus connection can be changed (in milliseconds). → default: "100"
- Click on the button **Add connection** to add an Modbus connection.
- Click on the button to assign the Modbus connection to all Modbus components without a Modbus connection.
- Click on the button to delete the Modbus connection.
- Click on the button to close the dialogue. All changes are applied automatically.

4.6.5 KNX

In this interface section, the general IP gateway of the KNX connection can be configured. With ComPLC version 3.0 or higher, up to 50 different KNX connections can be configured. For each KNX connection, a licence "L-PLC-KNX" is required (see data sheet "**ComPLC**").

Configuration ComPLC

› Click on the button KNX in the toolbar and then on the button + Add connection in the right sidebar. The following dialogue appears:



Dialogue "KNX"

› **Name**: Enter the description of the KNX connection.
› **IP**: Enter the KNX IP gateway.
› **Port**: Enter the port of the KNX gateway. → default: "3671"
› **Telegram sending limit per minute**: In this field, the amount of telegrams can be changed that will be sent via the respective KNX connection. → default: "10"
› Click on the button + Add connection to add an KNX connection.
› Click on the button to assign the KNX connection to all KNX components without a KNX connection.
› Click on the button − to delete the KNX connection.
› Click on the button to close the dialogue. All changes are applied automatically.

4.6.6 Avigilon

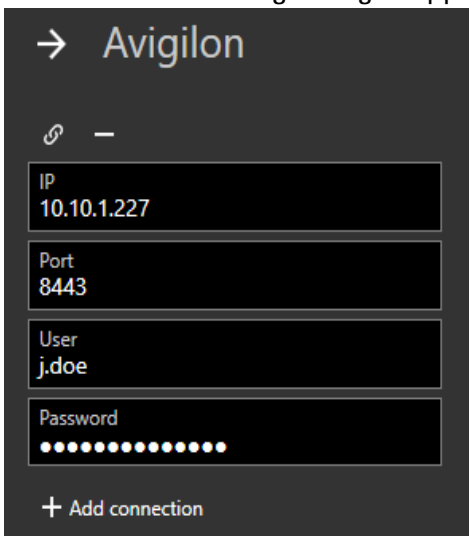
In this interface section, the Avigilon connection can be configured.

NOTE: Avigilon HTTPS certificate is mandatory
A certificate is required to connect ComPLC with Avigilon! To use Avigilon in ComPLC, upload the Avigilon HTTPS certificate to ComPLC, see "General" on page 14.

NOTE: Install Avigilon "ACC Web Endpoint Service"
To run the Avigilon interface, Avigilon´s "ACC Web Endpoint Service" needs to be installed on the Avigilon server.

Configuration ComPLC

› Click on the button **Avigilon** in the toolbar and then on the button + Add connection in the right sidebar. The following dialogue appears:



Dialogue "Avigilon".

› **IP**: Enter the ACC server IP address.
› **Port**: Enter the port of the Avigilon "Web Endpoint" → default: "8443"
› **User**: Enter the user name of the connection.
› **Password**: Enter the password of the connection.
› Click on the button + Add connection to add an Avigilon connection.
› Click on the button to assign the Avigilon connection to all Avigilon components without an Avigilon connection.
› Click on the button − to delete the Avigilon connection.
› Click on the button to close the dialogue. All changes are applied automatically.

4.6.7 SDS

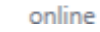
For more information about the SDS Guardian Indoor Gunshot Detection system, see manual "**ComPLC – SDS**".

4.6.8 Features

With this function, the interface section can be customised.

NOTE: SDS
By default, the checkbox SDS is not activated.

4.7 Section: Online

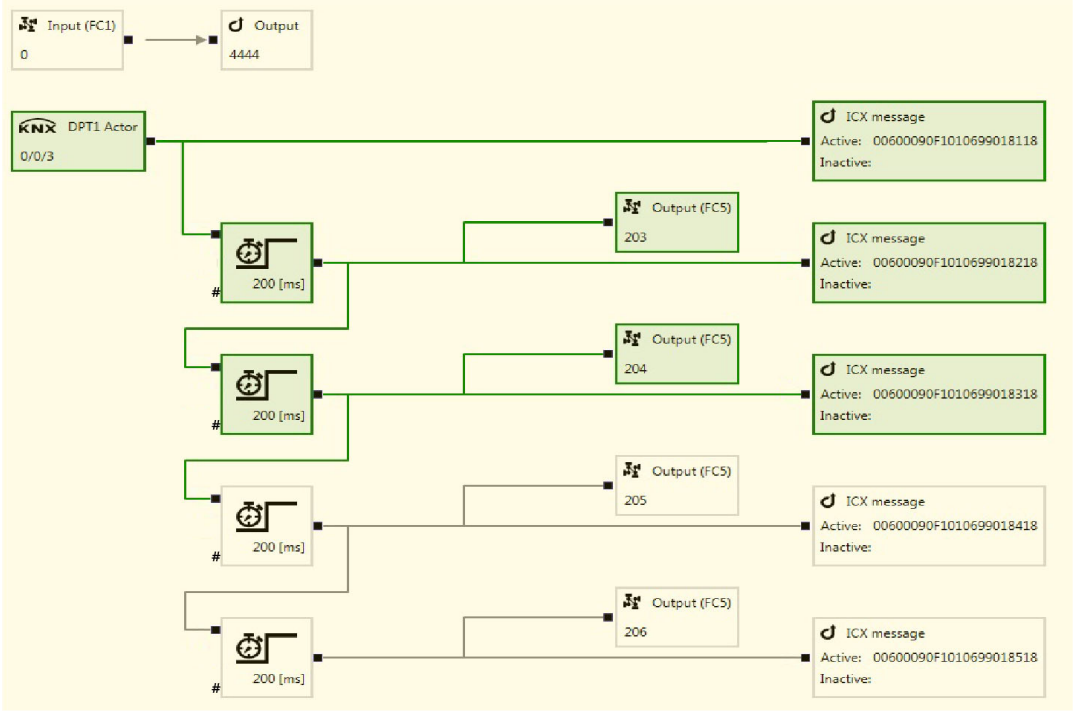


Toolbar section: online switch

4.7.1 Online mode

ATTENTION: Permanent connection required
A permanent active connection between ComPLC and the Intercom Server is required to activate the online mode!

The online mode traces all current operations of the configured logic, which is operating via VirtuoSIS in real-time. The operation can be traced over several schemes. See the following illustration:



Activated online mode

The current input/output states (also the outgoing connections) are indicated in an appropriate colour. Components with a numeric output are always indicated grey, whereas the current values of these components are indicated in the upper right corner. The component itself and its outgoing connection switches the colour as soon as the input/output state has been changed (e.g. by an incoming ICX message or change of the component value). The following states can be indicated:

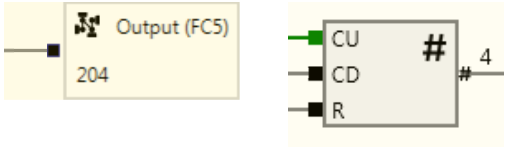
State “green”

› The input/output state of the component is TRUE. See the following illustration:

Input state TRUE in online mode

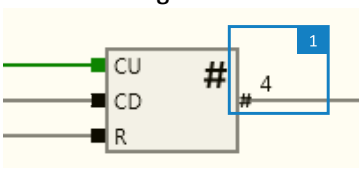
State “grey”

- The input/output state of the component is FALSE or
- It is a component with a numeric output or
- It is a deactivated outgoing connection. See the following illustration:



Component states: Input state FALSE/Numeric output in online mode

The value of a numeric output is indicated in the upper right corner of the component 1. See the following illustration:

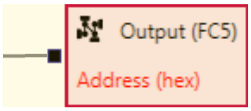


Component “counter”: component value indication in online mode

NOTE: Further information
Further information about numeric components can be found on page 51.

State “red”

- An error has been indicated for this component (e.g. invalid value or IP address or loop detection). See the following illustration:



Input error in online mode

State “orange”

- The input/output state of the component is UNKNOWN (e.g. entered call number does not exist). See the following illustration:

Input state UNKNOWN in online mode

Configuration ComPLC

- In order to use the online mode, the identical ComPLC configuration must be used in the application ComPLC and in the Intercom Server. Therefore, one of the following methods can be used:
  - Download the ComPLC configuration from the Intercom Server (see page 12),
  - or upload the ComPLC configuration to the Intercom Server (see page 12).
- Click on the slider  in the toolbar to switch on/off the online mode. If ComPLC has an active connection to the Intercom Server, this slider is indicated in blue (online mode is activated).

NOTE: Restrictions

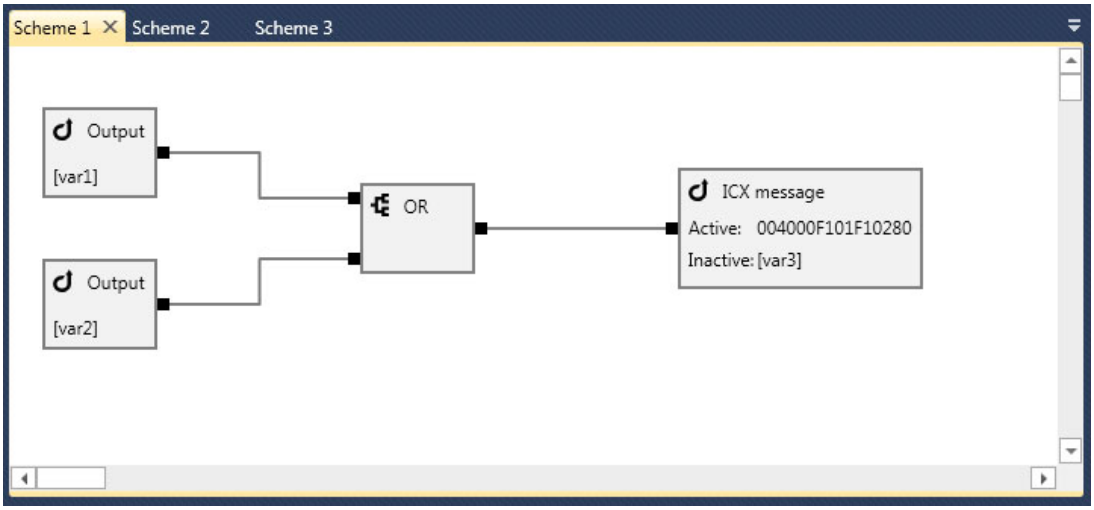
- If no active connection between ComPLC and the Intercom Server is available, the online mode will be deactivated automatically after several seconds.
- The online mode stops working, if the ComPLC configuration gets changed during the online mode (e.g. component value gets changed by an event).
- As long as the online mode is activated, the ComPLC configuration cannot be changed manually.

4.8 Scheme explorer

On the following pages, you will find a detailed step-by-step setup guide to configure schemes, instances and instance tables via ComPLC.

4.8.1 Scheme

A scheme is the platform to configure the logic of a ComPLC project. All components are placed, connected and configured in an active scheme. It is also possible to work with several schemes at the same time. Thus, the configured components are able to interact over several schemes.

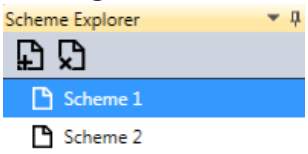


In the scheme dialogue, the ComPLC logic can be configured

The logic is performed automatically after the upload to VirtuoSIS. All current operations of the uploaded logic can be traced via the online mode in real-time (see page 21).

A scheme can be mirrored by several one-to-one duplicated instances to simultaneously perform the configured logic with different configured variables (multi-instance scheme), but with different results (see page 28). By default, a scheme works as single-instance scheme.

Configuration of a single-instance scheme



Scheme explorer (single-instance scheme)

- **Add scheme**: Click on the button .
- **Delete scheme**: Select the desired scheme and click on the button .
- Additional options: Right-click on the desired scheme in the scheme explorer. The following options are available:
  - **Multiple instances**: Switch between the use of a single-instance or multi-instance scheme (see page 28).

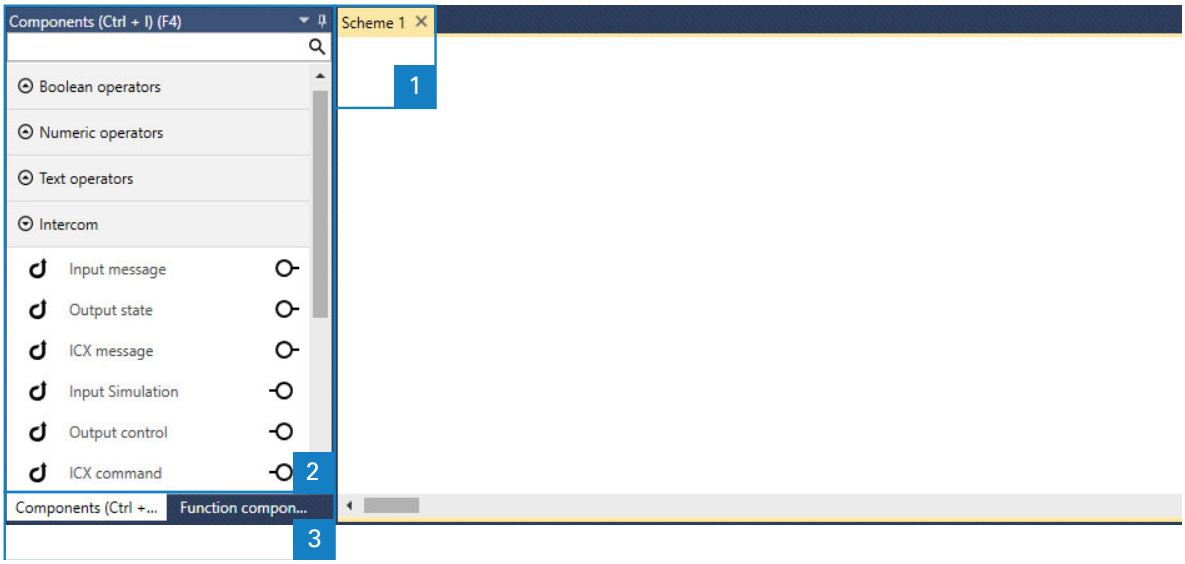
    ATTENTION: Instances will be deleted by switching to a single-instance scheme
    With switching to a single-instance scheme, all instances allocated to this scheme will be deleted! Save your project before switching to single-instance schemes.

  - **Remove selected scheme**: Delete the scheme and all allocated instances.
  - **Rename**: Change the scheme name.

4.8.2 Place components

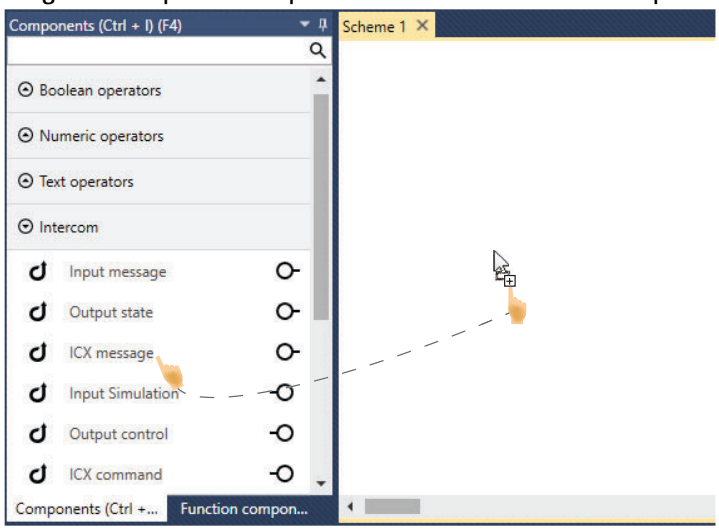
In order to configure and use components in ComPLC, they have to be placed in the scheme explorer.

Configuration ComPLC



Components can be placed in the scheme

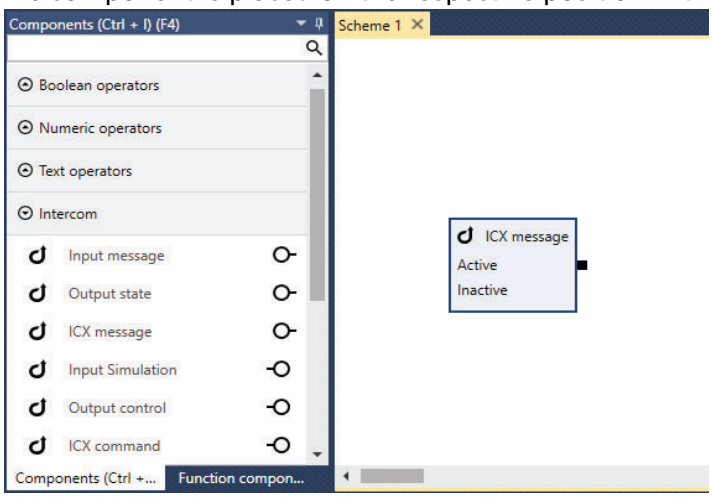
- Click on the scheme tab 1 in the scheme explorer.
- Select the desired component in the dialogue “component” 2. Switch between the component types by clicking on the respective component tab 3.
- Drag and drop the component and onto the desired position in the scheme explorer:



Drag and drop the component in the active scheme

NOTE: Placement of components
It is not possible to place a component within an instance or during the online mode.

› The component is placed on the respective position within the active scheme:



Components are placed within the scheme

Place the identical component in an other scheme.

› Right-click on the component and select the button **Copy** or **Cut** or select the component and click on the button **Copy** or **Cut** in the toolbar.
› Click on the desired scheme tab 1 in the scheme explorer.
› Right-click in the scheme explorer and select the button **Paste** or click on the button **Paste** in the toolbar.

GOOD TO KNOW: Place components via “quick insert”
A quick way to place components in a scheme is the “quick insert” function:
Place the cursor inside the scheme dialogue.
Press “F4” or “Ctrl + I” to start the components search and enter the desired component.
Navigate via arrow keys to the desired component and press “Enter”. The component appears next to the cursor in the scheme dialogue.

Delete component

› Right-click on the component and select the button **Delete**.

4.8.3 Create a connection

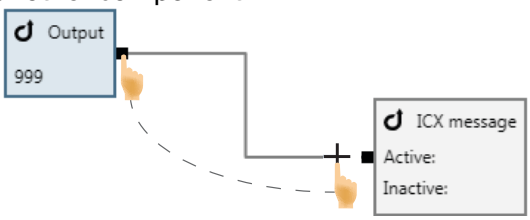
A connection enables the interaction of two or more connected components. Components can only influence directly other components via a connection.

Configuration ComPLC

› Move the cursor over the output/input connection of the component. A plus symbol is indicated instead of the mouse cursor:

Move the cursor over the input/output connection

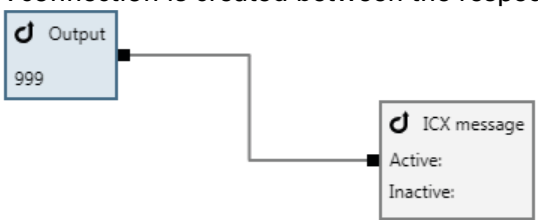
› Click on the output/input and drag and drop the cursor to the input/output connection of another component:



Drag and drop the cursor to another input/output connection

NOTE: Movement of connections
It does not matter whether the connection is moved from the output to the input or vice-versa.

› A connection is created between the respective components:



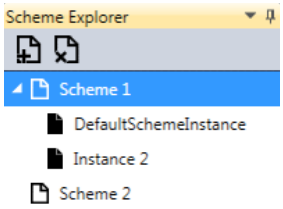
Connection between two components

NOTE: Restrictions
- Connections between two inputs or outputs are not possible.
- Two or more incoming connections to an input are not possible.
- A connection between components over several schemes is only possible using flag components (see "Inbound/Outbound Flags (Boolean)" on page 49).
- Numeric and Boolean connectors cannot be connected directly. A numeric connection is marked with "#", whereas a Boolean connection is marked by a black square. For further information on numeric and Boolean components: see "Boolean operators (n)" on page 39 and see "Numeric operators (#)" on page 51.

Delete connection

› Right-click on the connection and select the button **Delete**.

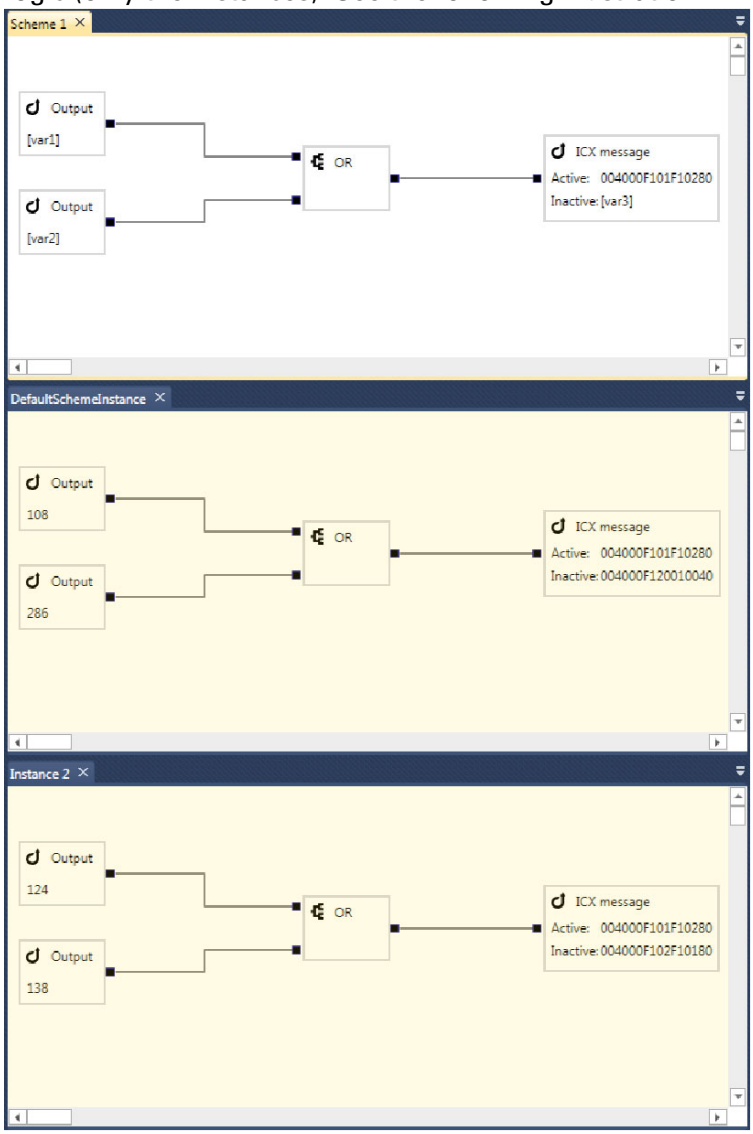
4.8.4 Instances



Scheme explorer (multi-instance scheme)

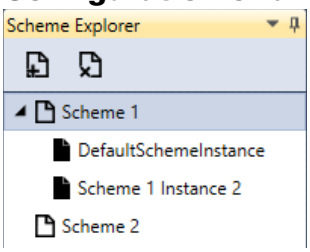
An instance is a one-to-one duplicate of the allocated scheme (multi-instance scheme). Any changes that are performed in the scheme (also in online mode) will be applied for all allocated instances (e.g. replace a component, change a fixed value, place a new component or change a connection). All allocated instances of a scheme simultaneously perform the configured logic with different configured variables.

The online mode (see page 21) traces all current operations of each instance that performs the configured logic separately, depending on the configured variable values. Thus, the instances present an extended overview of the real-time operation with different variable values at the same time. With one or more instances, the scheme itself will not perform any logic (only the instances). See the following illustration:



The instance is a one-to-one duplicate of the template scheme

Configuration of a multi-instance scheme



Scheme explorer (multi-instance scheme)

ATTENTION: At least one instance required
The following configuration steps require at least one instance! Further information about the configuration of additional instances can be found on page 24!

- Right-click on the desired scheme in the scheme explorer. The following options are available:
  - **Multiple instances**: Switch between the use of a single-instance or multi-instance scheme.

    ATTENTION: Instances will be deleted by switching to a single-instance scheme
    With switching to a single-instance scheme, all instances allocated to this scheme will be deleted!

  - **Open instance table**: Open the instance table for this scheme. The instance table provides an overview of all instances and variables of this schemes (see page 30).
  - **Remove selected scheme**: Delete the selected scheme and all allocated instances.
  - **Add instance**: Add a new instance that is assigned to this scheme.
  - **Rename**: Change the scheme name. Press the <enter> key to confirm the entry.
- Right-click on the desired instance (if available). The following options are available:
  - **Delete**: Delete the instance.
  - **Rename**: Change the instance name. Press the <enter> key to confirm the entry.

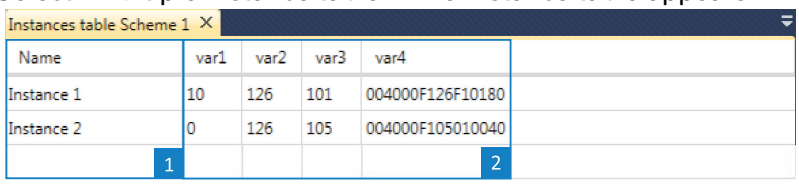
4.8.5 Instance table

ATTENTION: At least one instance required
The following function is only available if at least one instance is available for the scheme (see page 28)!

The instance table provides an overview of all instances and variables of a scheme. With this table, it is possible to change variable values and to create or delete instances. It is not possible to create variables or change the variable names via the instance table.

Configuration ComPLC

- Right-click on the scheme in the scheme explorer and activate the checkbox „Multiple instances“
- Select “Multiple instance table”. The instance table appears:



The instance table presents an overview of all instances and variables

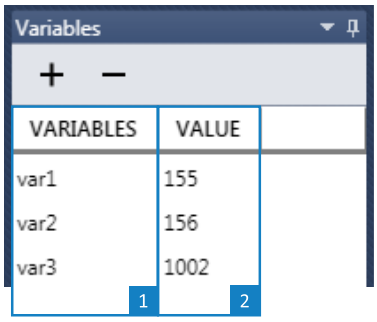
1 Shows all instances of the scheme.

2 Shows all variables of the scheme and the current values of each instance.

NOTE: Automatic change of settings in the instance table
Changes of instance names, variable names and variable values will be applied automatically in the instance table.

- **Edit variable values**:
  - Click on the variable value and change the entry. Press the <enter> key to confirm the entry. The variable value is displayed in the dialogue “Variables” for the respective instance.
- **Create an instance**:
  - Right-click on a row and click on the button **Insert Row** to add an row.
  - Click on the field **Name** of the new row and enter the instance name. Press the <enter> key to confirm the entry. The instance is displayed in the scheme explorer.
- **Delete an instance**:
  - Right-click on a row and click on the button **Remove Row** to delete the respective instance.
- **Copy entry into spreadsheet software**:
  - Select the desired fields and press the keys <ctrl> + <C> to copy the selected field.
  - Press the keys <ctrl> + <V> to insert the copied fields one-to-one in any spreadsheet software (e.g. Microsoft Excel).

4.9 Variables



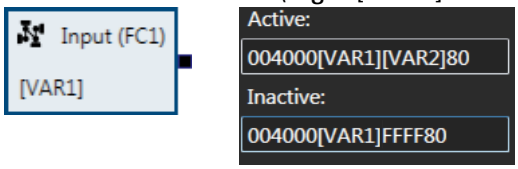
Dialogue "variables"

Variables can storage any configurable values (numbers and letters) and can be used for all component values (for input and output actions; e.g. call number, group address, ICX message, comment or e-mail subject). With changing the variable value, the new value will automatically be applied for all components with this assigned variable.

In multi-instance schemes (see page 28), a variable will be defined for each instance of a scheme, but its value can only be configured for the currently selected instance. Therefore, all instances of a single scheme have the same variables, but with different variable values. This provides an execution of the same ComPLC configuration with different parameters.

Configuration ComPLC

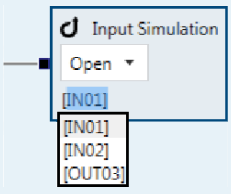
- **Add variable**: Click on the button + to add a variable with pre-defined name or click on an empty name field 1 and enter the variable name.
- **Delete variable**: Select the variable and click on the button −.
- **Rename**: Select the variable, click on the name field 1 and change the variable name.
- **Configure variable value**: Select the variable, click on the value field 2 and enter the variable value.
- **Assign a variable**: Variables can be assigned to all components with a value field. In order to assign a variable to a component, enter the variable name with square brackets on both ends in the value field (e.g. "[VAR1]" for variable "VAR1"). See the following screenshot:



Variable "VAR1" is assigned to both components

GOOD TO KNOW: Auto-complete function

With entering "[" in the value field of any component, all variables will be displayed that can be assigned to this component. See the following screenshot:



Variables can be selected via the auto-complete function

4.10 Runtime variables

Runtime variables are individually usable variables that can be used in the entire ComPLC project by ICX, webhook or e-mail components. Runtime variables will be used as wildcards in ICX strings, webhook commands and emails to automatically save or replace a certain part of the string (numbers and characters), which can be re-used in other components.

4.10.1 Create and store runtime variables

In order to create runtime variables and/or set the variable value, the desired runtime variable expression (see syntax below) has to be inserted into an ICX message of an inbound ICX message component (see page 61) or message of an inbound webhook component (see page 79). It is possible to use several runtime variables within the same string.

The runtime variable expression (e.g. "{CALLER:8}") serves as wildcards within the string and the attached length defines the number of reserved digits in the string and the maximum number of storable digits (see example below).

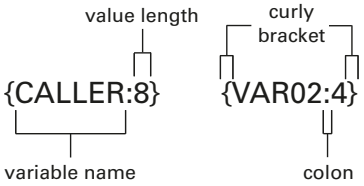
As soon as the respective component gets triggered, non-existing runtime variables will be created and store the respective string as variable value, or already available runtime variables update their variable value (older variable values will automatically be replaced).

NOTE: No auto complete of variable value
Free digits in the variable value will not be automatically completed if the saved value is shorter than the maximum size of the runtime variable.

GOOD TO KNOW: Change runtime variable value from outside the ComPLC system
Runtime variable values can be changed from outside the ComPLC system with HTTP request webhooks. Further information about the configuration of HTTP request webhooks can be found on page 79.

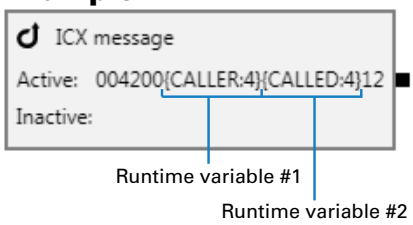
Syntax

The expression consists of the variable name and the length (number of storable digits), which are divided by a colon, and is put into curly brackets. The variable can contain numbers and characters. See the following illustration:



A runtime variable consists of several parts

Example



Example: Inbound ICX message component with two runtime variables

The inbound ICX message component will be triggered by each incoming ICX message starting with “004200” and ending with ”12”. The string in-between is replaced by wildcards of two runtime variables (“CALLER” and “CALLED”) with a length of “4”. Therefore, digits 7 to 10 will be saved in variable “CALLER” and digits 11 to 14 will be saved in variable “CALLED”. At the incoming ICX message “004200 F101 F102 12”, the values of the runtime variables “CALLER” is “F101” and of “CALLED” it is “F102”.

4.10.2 Use runtime variables

In order to use a runtime variable, the desired runtime variable expression (see syntax below) has to be inserted into the respective component. E. g. ICX command of an outbound ICX command component (see page 63), the command of an outbound webhook component (see page 79), the variable of a outbound SNMP variable (see page 76) or the subject or text of an email component (see page 88). It is possible to use several runtime variables within the same string.

The runtime variable expression (e.g. “{CALLED}”) serves as wildcards within the string. The length, and therefore the number of wildcards within the string, is already pre-defined by the saved variable value (see example below).

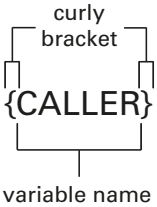
NOTE: Only the entire variable value can be used
It is not possible to use only a part of the saved variable value.

When the respective component gets triggered, the wildcards will automatically be replaced by the currently saved variable value.

ATTENTION: Consider varying length of variable value
Consider that the length of the currently saved value must not be equal with the maximum length of the runtime variable!

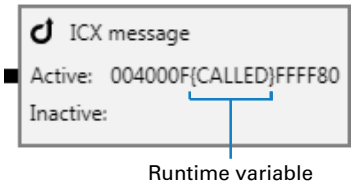
Syntax

The expression consists of the variable name and is put into curly brackets. See the following illustration:



A runtime variable consist of several parts

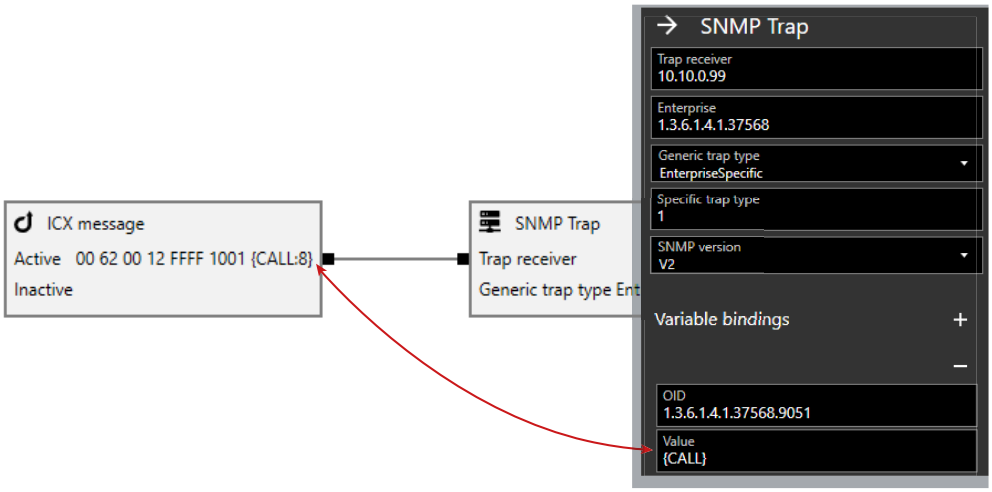
Example 1– ICX command



Example: Outbound ICX command component with one runtime variable

If the outbound ICX command component is triggered, the ICX command “004000 F{CALLER} FFFF 88” will be put out and the wildcards will be replaced by the referenced variable value. At a variable value of “102” (length is “3”), digits 8 to 10 of the ICX command will be filled by the runtime variable and the ICX command “004200 F102 FFFF 80” will be put out.

Example 2 – SNMP Trap



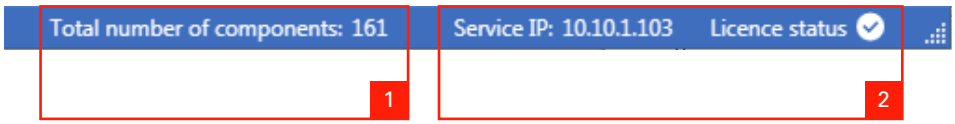
Example: Inbound ICX component + outbound SNMP Trap component with one runtime variable

If the ICX message starting with “00 62 00 12 FFFF 1001” and ending with the referenced variable value (length is “8”) is received, the SNMP trap will be triggered.

Runtime variables – supported components

- Store variables:
  - Inbound ICX (active and inactive message),
- Use variables:
  - Outbound ICX (active and inactive message)
  - Outbound E-Mail (subject and email message)
  - SNMP trap (value in variable bindings)

4.11 Licence bar



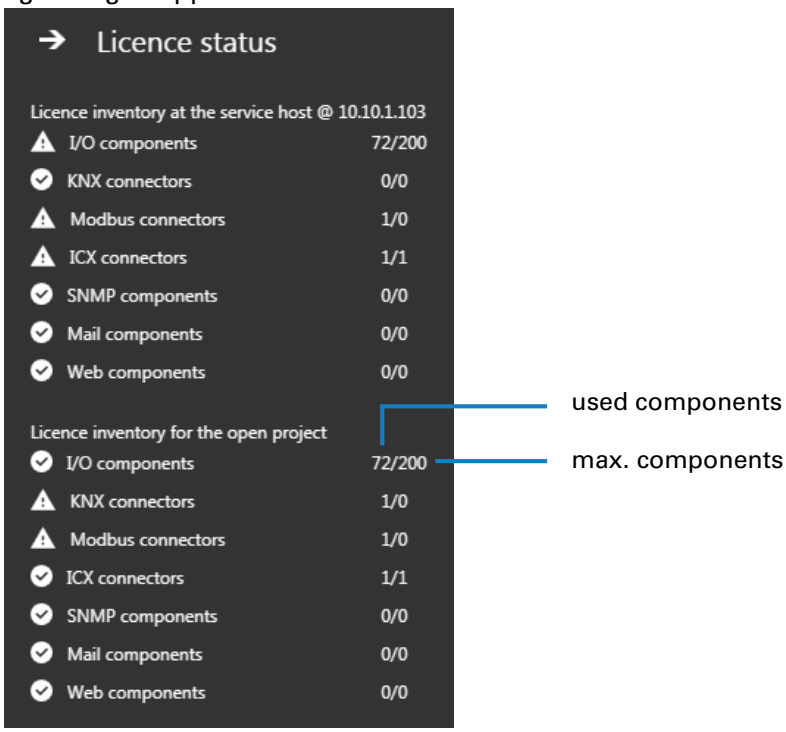
The licence bar shows the number of all placed components

The licence bar indicates the total number of placed components in all active schemes, the IP address of the host system and the current licence status:

1 **Total number of components**: Shows the number of all placed licence-based and non-licence-based components in all active schemes.

2 **Connection and licence status**: Shows the IP address of the connected host system (see page 14) and the current licence status. If all required licences are available on the respective host system, a check mark will be shown. Otherwise, a warning triangle will be shown.

› Click on the field 2 in order to open an advanced licence information dialogue. The following dialogue appears:



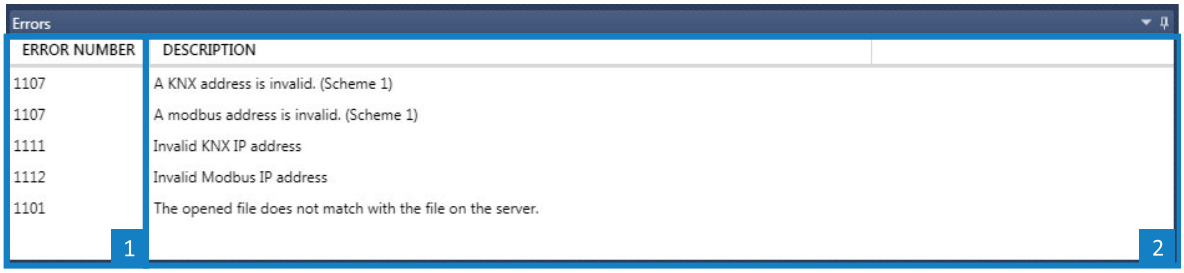
Shows a list of required and available licences for all component types

This dialogue is divided into two lists, showing both the maximum usable and currently used number of components, separated into different categories according to the required licences. The list below shows the licence status in the current ComPLC project and the list above shows the licence status in the ComPLC project that is running on the connected host system.

If the required licences are available for a category, a check mark will be shown. Otherwise, a warning triangle will be indicated.

For further information about licence-based components and licences, see data sheet "**ComPLC**".

4.12 Error message box



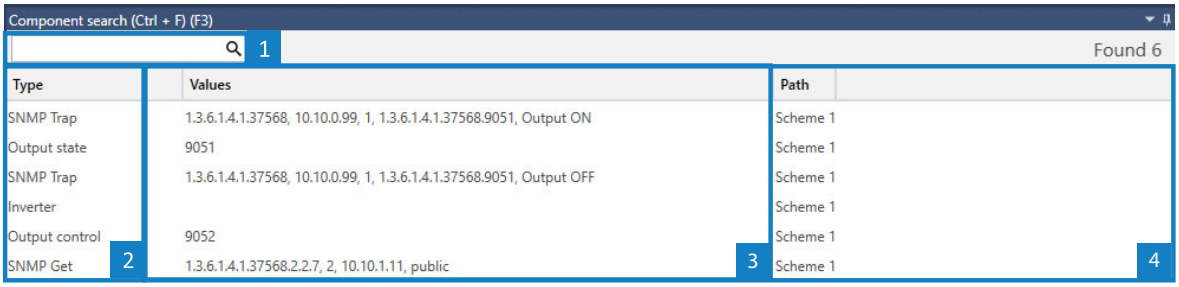
Dialogue "error message box"

The error message box indicates all occurring errors of the application ComPLC during the online mode, upload or download of the configuration. The following error information is available:

- 1 **Error number**: Shows the error number (for the technical support only).
- 2 **Description**: Shows a description about the occurred error. The affected scheme is displayed in brackets.

To navigate to the respective component in the scheme dialogue, double-click on the desired error message or select it and press enter.

4.13 Component search dialogue



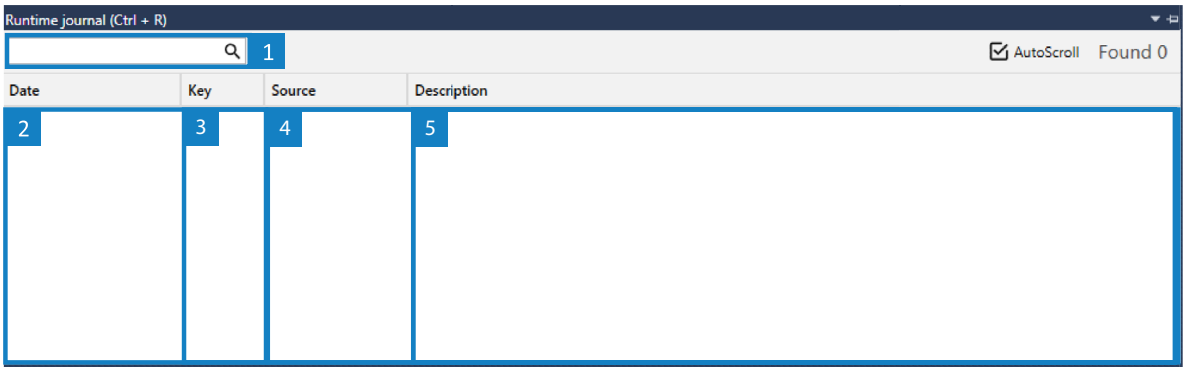
Dialogue "component search"

Previously defined components in a scheme can be found in the dialogue "Component search". To open the dialogue and to search for previously applied components press "F3" or "Ctrl + F". The following function/information is available:

- 1 **Component Search**: Enter keywords to search for already applied components.
- 2 **Type**: Indicates the type of the component.
- 3 **Values**: Indicates the set values of the component.
- 4 **Path**: Indicates the connected scheme of the component.

To navigate to the regarding component in the scheme dialogue, double click on the desired component or select it and press enter.

4.14 Runtime journal dialogue



Dialogue "Runtime journal"

Runtime information errors and ICX messages can be found in the dialogue "Runtime journal". To open the dialogue and to search for messages press, "Ctrl + R". The following function/information is available:

NOTE:
"Runtime journal" is only available in online mode.

- 1 **Journal search**: Enter keywords to filter for messages
- 2 **Date**: Indicates when the message has been created
- 3 **Key**: Indicates the key-value of the message
- 4 **Source**: Indicates the source of the message
- 5 **Description**: Description of the message

4.15 Components

A component is a logic element with either an output and/or one or several inputs. Depending on the trigger conditions, a component performs a certain action. Combined components influence each other to achieve a specific behaviour according to the configuration. A component is triggered either by incoming messages (e.g. ICX message, KNX telegram or Modbus value) or via the input. The component with a Boolean output can only have one state at the same time, whereas a component with a numeric output can only have one numeric value at the same time. The following input/output states are available for Boolean components:

- **0**: Input/output state is FALSE (deactivated).
- **1**: Input/output state is TRUE (activated).
- **Unknown**: The component output state was never set.

ATTENTION: Disable online mode
Object values can only be configured if the online mode is disabled!

If the input port of a component is not connected to another component output, the state “n.c.” is indicated. This applies for Boolean as well as numeric components.

4.15.1 Component toolbox

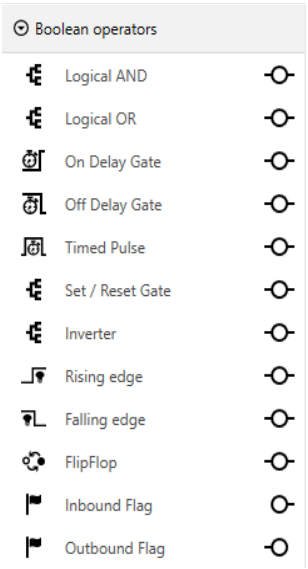
Components can easily be found using the search bar in the “Components” dialogue. To search for components click in the search bar or press F4 or Ctrl + I.

The components are separated into several types and dialogues. The following component types are available:

- **Boolean operators** see page 93
- **Numeric operators** see page 53
- **Text operators** see page 56
- **Intercom** see page 56
- **KNX** see page 66
- **Modbus Master** see page 72
- **SNMP components** see page 76
- **Web components** see page 79
- **SDS** see page 20
- **Avigilon** see page 20
- **Misc** see page 86
- **Function component** see page 89

4.15.2 Boolean operators (■)

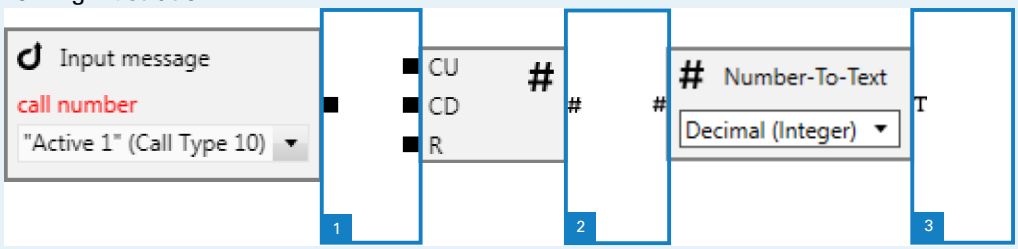
The operator performs a logical operation on one or more inputs states (depending on the component), and produces a single output state or numeric value.
Boolean operators can only progress the truth of variables according to the performed action (TRUE, FALSE). The connection points of Boolean operators are characterised by „■“.



Component category “Boolean operators”

GOOD TO KNOW: Difference: “Boolean”, “numeric” and “textual”

Up to ComPLC 1.0, only Boolean components are available. With ComPLC 2.0 or higher, it is also possible to progress numeric values with a defined data type instead of the truth of a variable. With ComPLC 3.0 or higher, it is possible to define textual values. The input and output are specific-marked to indicate them as Boolean, numeric or textual connectors. See the following illustration:



Connection types of components

1 **Boolean connection**: This connection type is marked by “■”
2 **Numerical connection**: This connection type is marked by “#”.
3 **Textual connection**: This connection type is marked by “T”
Only the same connector types can be connected (independent whether output or input).That means, the direct connection of a Boolean and numeric connector is not possible.

› Click on the tab “Operators” to display all operator components. The following operator components are available:
  - Logical AND see page 40
  - Logical OR see page 41
  - On Delay Gate see page 42
  - Off Delay Gate see page 43
  - Timed Pulse see page 44
  - Set/Reset Gate see page 45
  - Inverter (NOT) see page 46
  - Rising edge see page 47
  - Falling edge see page 47
  - FlipFlop see page 48
  - Inbound Flag/Outbound Flag see page 49

4.15.3 Logical AND

Component “AND”

This component is triggered if both input state are TRUE.

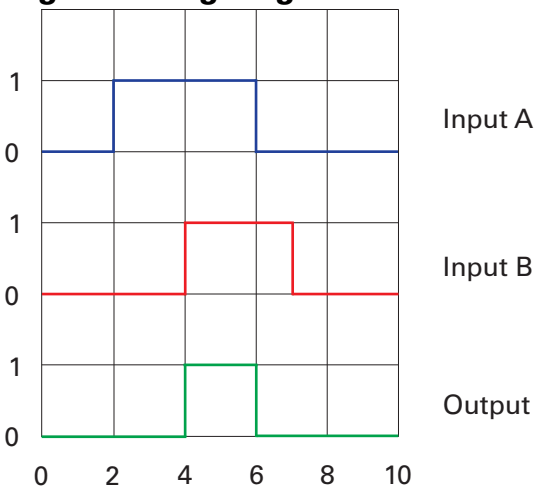
Status change (FALSE → TRUE)

A TRUE output state results if both inputs (A and B) are TRUE.

Status change (TRUE → FALSE)

A FALSE output state results if one or both inputs states (A or B) are FALSE.

Digital timing diagram



Component “and”: digital timing diagram

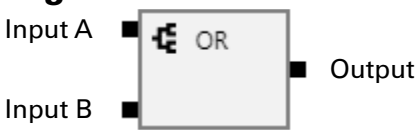
Truth table

Input A	Input B	Output
0	0	0
0	1	0
0	unknown	0
0	n.c.	0
1	0	0
1	1	1
1	unknown	unknown
1	n.c.	1
unknown	0	0
unknown	1	unknown
unknown	unknown	unknown
unknown	n.c.	unknown
n.c.	0	0
n.c.	1	0
n.c.	unknown	0
n.c.	n.c.	unknown

No configuration required

GOOD TO KNOW: Add/delete ports
Right-clicking on the desired component opens a context menu where ports can be added or deleted.

4.15.4 Logical OR



Component "or"

This component is triggered if one or both input state are TRUE.

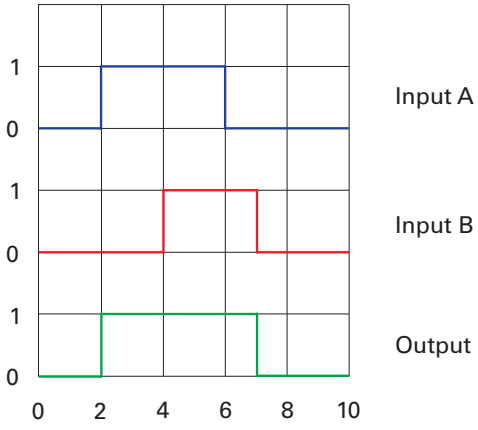
Status change (FALSE → TRUE)

A TRUE output state results if one or both inputs states (A or B) are TRUE.

Status change (TRUE → FALSE)

A FALSE output state results if both input states are FALSE.

Digital timing diagram



Component "or": digital timing diagram

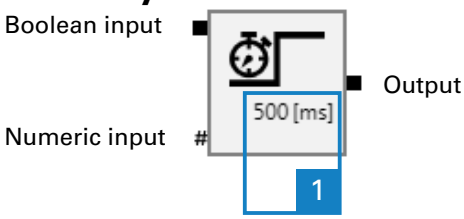
Truth table

Input A	Input B	Output
0	0	0
0	1	1
0	unknown	unknown
0	n.c.	0
1	0	1
1	1	1
1	unknown	1
1	n.c.	1
unknown	0	unknown
unknown	1	1
unknown	unknown	unknown
unknown	n.c.	unknown
n.c.	0	0
n.c.	1	1
n.c.	unknown	unknown
n.c.	n.c.	unknown

No configuration required

GOOD TO KNOW: Add/delete ports
A right click on the desired component opens a context menu where ports can be added or deleted.

4.15.5 On Delay Gate



Component "on delay gate"

This component delays the change of the output state from FALSE to TRUE.

Status change (FALSE → TRUE)

A TRUE output state results if the configured delay has expired 1.

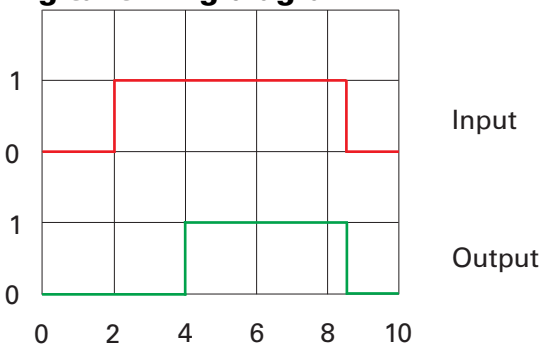
Status change (TRUE → FALSE)

A FALSE output state results if the input state is FALSE.

Configuration ComPLC

› 1 In this field, the delay of the output state can be changed (in milliseconds). With ComPLC version 3.0 or higher, this value can individually be configured by a numerical value via the numerical input (e.g. with a counter; see page 52).
→ default: "500 ms"

Digital timing diagram

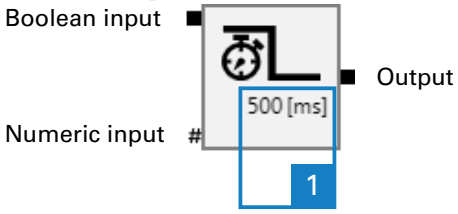


Component "on delay gate": digital timing diagram

Truth table

Input	Output
0	0
0	1 (delayed)
unknown	unknown
n.c.	unknown

4.15.6 Off Delay Gate



Component "off delay gate"

This component delays the change of the output state from TRUE to FALSE.

Status change (FALSE → TRUE)

A TRUE output state results if the input state is TRUE.

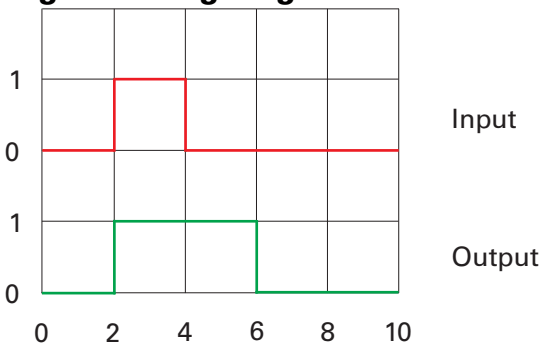
Status change (TRUE → FALSE)

A FALSE output state results if the configured delay has expired 1.

Configuration ComPLC

› 1 In this field, the delay of the output state can be changed (in milliseconds). With ComPLC version 3.0 or higher, this value can individually be configured by a numerical value via the numerical input (e.g. with a counter; see page 52).
→ default: "500 ms"

Digital timing diagram



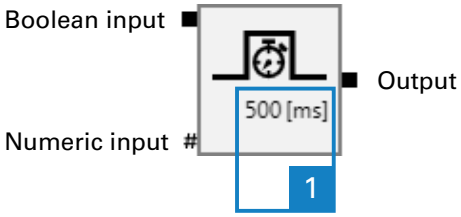
Component "off delay gate": digital timing diagram

Truth table

Input	Output
0	0 (delayed)
0	0 (no delay) [1]
1	1
unknown	unknown
n.c.	unknown

[1] Input state changes from “unknown” to “0”.

4.15.7 Timed Pulse



Component “timed pulse”

This component issues a timed pulse at the output.

Status change (FALSE → TRUE)

A TRUE output state results if the input state is TRUE.

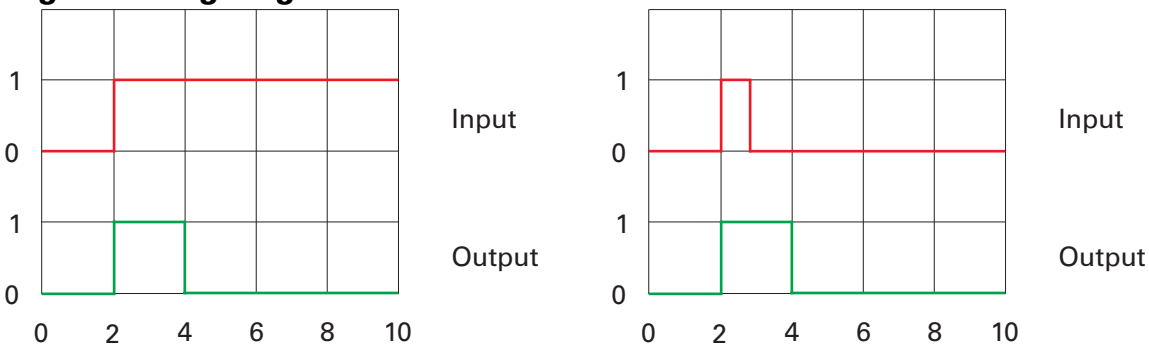
Status change (TRUE → FALSE)

A FALSE output state results if the configured pulse time has expired 1.

Configuration ComPLC

› 1 In this field, the duration of the timed pulse can be changed (in milliseconds). With ComPLC version 3.0 or higher, this value can individually be configured by a numerical value via the numerical input (e.g. with a counter; see page 52).
→ default: “500 ms”

Digital timing diagram



Component “timed pulse”: digital timing diagrams

Truth table

Input	Output
0	0
1	1 (pulse)
unknown	unknown
n.c.	unknown

4.15.8 Set/Reset Gate

Component "set/reset gate"

This component has two stable output states. One state is referred as "set" (TRUE) via input A and the other as "reset" (FALSE) via input B.

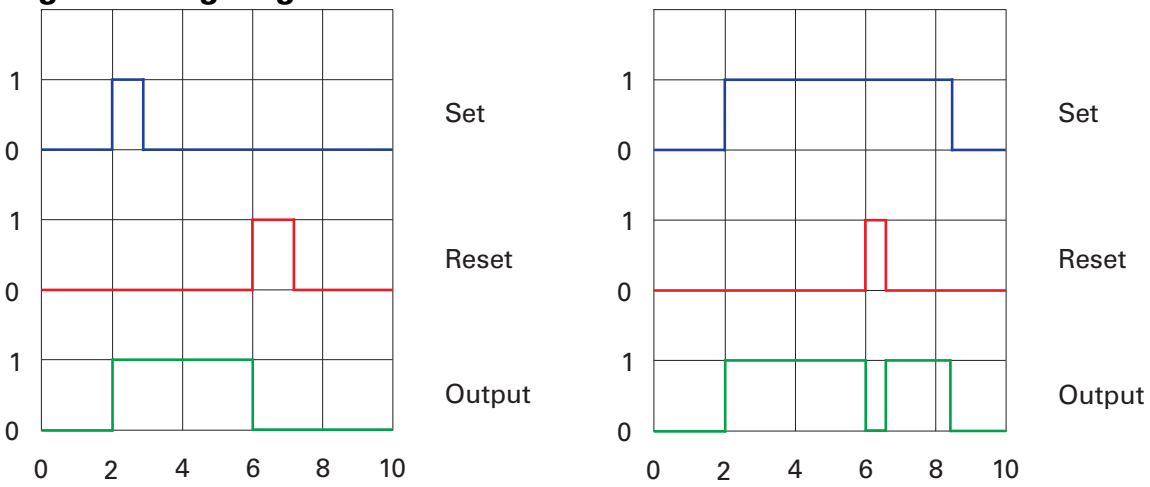
Status change (FALSE → TRUE)

A TRUE output state results if the input state A "set" is TRUE and the state of the input B "reset" is FALSE.

Status change (TRUE → FALSE)

A FALSE output state results if the input state B "reset" is TRUE.

Digital timing diagrams



Component "set/reset gate": digital timing diagrams

Truth table

Set	Reset	Output
0	0	out
0	1	0
1	0	1
1	1	0
0	n.n	out
1	n.c.	1
n.c.	0	out
n.c.	1	0

No configuration required

4.15.9 Inverter (NOT)

Component "inverter"

This component negates the input state.

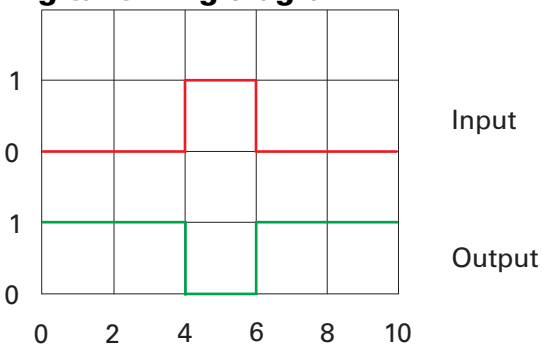
Status change (FALSE → TRUE)

A TRUE output state results if the input state is FALSE.

Status change (TRUE → FALSE)

A FALSE output state results if the input state is TRUE.

Digital timing diagram



Component "inverter": digital timing diagram

Truth table

Input	Output
0	1
1	0
unknown	unknown
n.c.	unknown

No configuration required

4.15.10 Rising edge

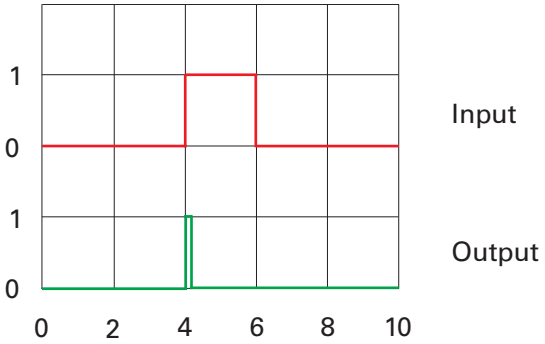
Component "rising edge"

This component issues a TRUE/FALSE pulse at a rising edge of the input state.

Status change (FALSE → TRUE)

If the input state changes from FALSE to TRUE ("rising edge"), the output state issues a TRUE or FALSE pulse.

Digital timing diagram



Component "rising edge": digital timing diagram

Truth table

Input	Output
0	0
1	1 (pulse)
unknown	unknown
n.c.	unknown

No configuration required

4.15.11 Falling edge

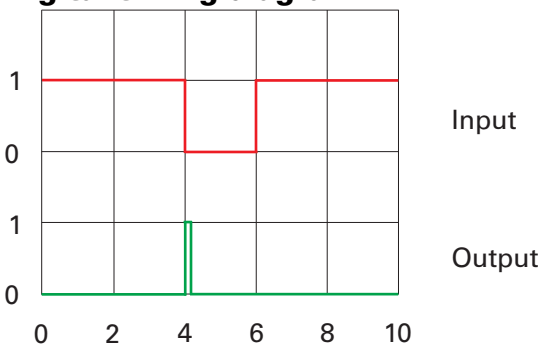
Component "falling edge"

This component issues a TRUE/FALSE pulse at a falling edge of the input state.

Status change (FALSE → TRUE)

If the input state changes from TRUE to FALSE ("falling edge"), the output state issues a TRUE or FALSE pulse.

Digital timing diagram



Component “falling edge”: digital timing diagram

Truth table

Input	Output
0	1 (pulse)
1	0
unknown	unknown
n.c.	unknown

No configuration required

4.15.12 FlipFlop

Component “FlipFlop”

This component toggles between two stable output states (TRUE and FALSE) at a rising edge of the input state. Both states are controlled by the input.

Start-up behaviour

Initially, the output state is FALSE.

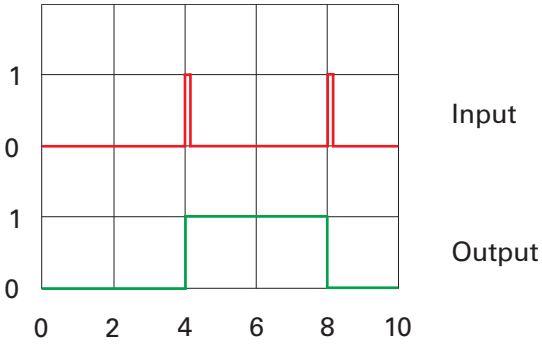
Status change (FALSE → TRUE)

A TRUE output state results if the input state changes from FALSE to TRUE (“rising edge”).

Status change (TRUE → FALSE)

A FALSE output state results if the input state changes from FALSE to TRUE (“rising edge”).

Digital timing diagram



Component “Flip-flop”: digital timing diagram

Truth table

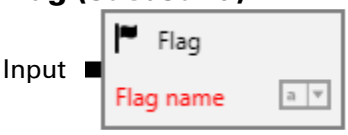
Input	Output
0	0
1	toggle
unknown	unknown
n.c.	unknown

No configuration required

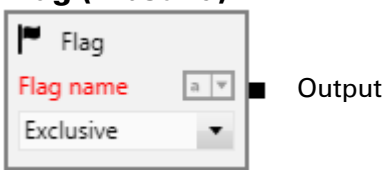
4.15.13 Inbound/Outbound Flags (Boolean)

The flag components can be used for cross-scheme connection. Several “outbound” and “inbound” flag components can be connected within one or over several schemes to a circuit via the configurable “flag name”. The connected flag components (circuit) operate like one component with several inputs or outputs, thus the connected flag components have not to be connected manually via lines. Each “outbound” flag component influences all connected “inbound” flag components in a configurable way. It is possible to place flag components of a circuit in different schemes.

Flag (outbound)



Flag (inbound)



Component “flag (outbound)”Component “flag (inbound)”

Configuration ComPLC

- **Flag name**: In this field, enter the name of the flag circuit. All flags with the same flag name within all active schemes are connected.

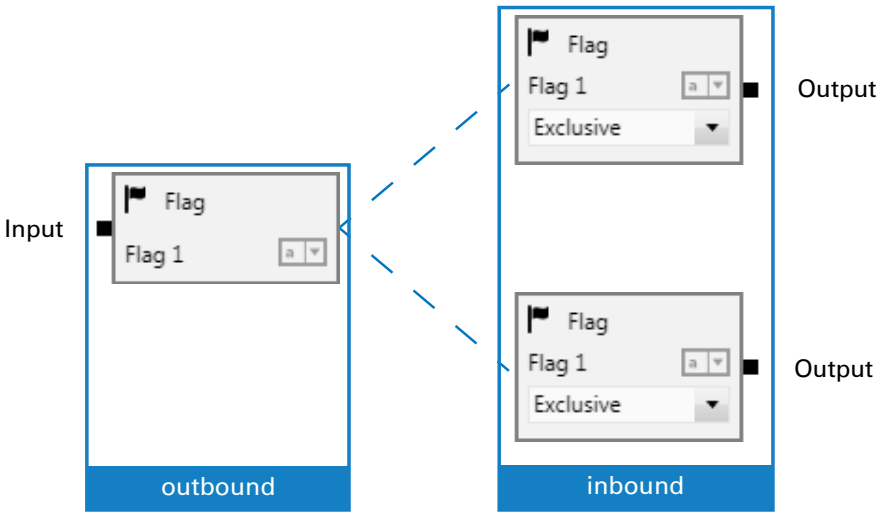
  NOTE: Same name for all connected flags
  It is required to configure the same flag name for all connected components.

- Click on the button [a] to enable the drop-down list for a quick search of all available flag circuits. In this drop-down list, an existing flag circuit can be selected instead of entering the flag name in the field “flag name”. In this case, only Boolean flag components are indicated.
- In the drop-down list, the operation mode can be selected for the inbound flag component. The following operation modes are available:
  - Exclusive see page 50
  - And see page 40
  - Or see page 41

    → default: “Exclusive”

NOTE:
- It is required to configure the same operation mode for all connected “inbound” components.
- The operation mode can only be configured for inbound flag components.

Exclusive



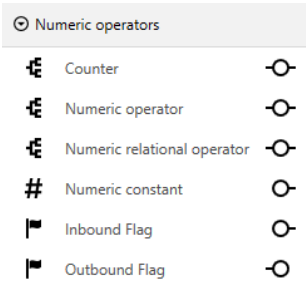
"flag (exclusive circuit)"

An exclusive circuit always consists of one outbound flag component and one or more inbound components. A TRUE output results only for all inbound flag components if the input of the outbound flag component is TRUE.

NOTE: Error message
If two or more outbound flag components are configured with the same name, an error message will be indicated with activating the online mode or at upload of the configuration.

4.15.14 Numeric operators (#)

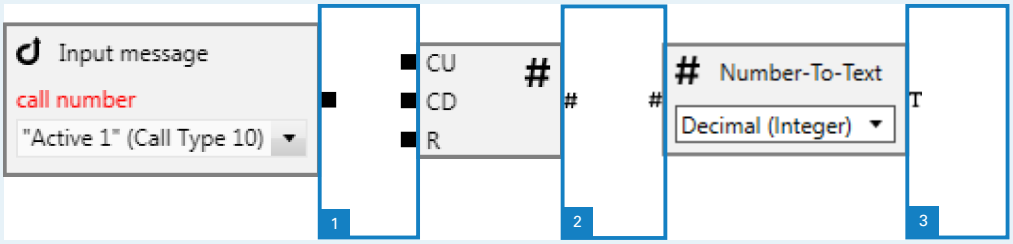
The operator performs a logical operation on one or more inputs states (depending on the component), and produces a single output state or numeric value. Numeric operators can progress numeric values with a defined data type. The connection points of Boolean operators are characterised by„#“.



Component category “Numeric operators”

GOOD TO KNOW: Difference between “Boolean“ and “numeric“

Up to ComPLC 1.0, only Boolean components are available. With ComPLC 2.0 or higher, it is possible to progress numeric values with a defined data type instead of the truth of a variable. With ComPLC 3.0 or higher, it is possible to define textual values. The input and output are specific-marked to indicate them as Boolean, numeric or textual connectors. See the following illustration:



Connection types of components

1 **Boolean connection**: This connection type is marked by “■”

2 **Numerical connection**: This connection type is marked by “#”.

3 **Textual connection**: This connection type is marked by “T”

Only the same connector types can be connected (independent whether output or input). That means, the direct connection of a Boolean, numeric or textual connector is not possible.

Click on the tab “Numeric operators” to display all Intercom components. The following Intercom components are available:

- Counter see page 52
- Numeric operator see page 51
- Numeric relational operator see page 54
- Numeric constant see page 54
- Inbound Flag/Outbound Flag see page 55

4.15.15 Counter

Component "counter"

This component counts at a rising edge of the Boolean input state. The resulting numeric value is given out at the numeric output. Data type of the numeric output is "double precision". Further information about numeric components can be found on page 51.

NOTE: Consider the connection
The component output can only be connected with a numeric input, whereas the input of this component can only be connected with a Boolean output. Numeric connectors are always marked with "#".

Start-up behaviour

Initially, the numeric output value is UNKNOWN.

Input CU (count up)

- **Status change (FALSE → TRUE)**
  If the input state changes from FALSE to TRUE ("rising edge"), the component value is increased by "1". At the output, the current component value is given out.
- **Status change (TRUE → FALSE)**
  No action

Input CD (count down)

- **Status change (FALSE → TRUE)**
  If the input state changes from FALSE to TRUE ("rising edge"), the component value is decreased by "1". At the output, the current component value is given out.
- **Status change (TRUE → FALSE)**
  No action

Input R (reset)

- **Status change (FALSE → TRUE)**
  If the input state changes from FALSE to TRUE ("rising edge"), the component value is set to "0". At the output, the component value "0" is given out.

  NOTE: Active input "R"
  As long as the input state of the connector "R" is TRUE, the inputs "CU" and "CD" have no effect.

- **Status change (TRUE → FALSE)**
  No action

Truth table

Input R	Input CD	Input CU	Output value
0	0	0	no action
0	0	1	increased by “1”
0	1	0	decreased by “1”
0	1	1	unknown
1	0	0	reset
1	0	1	reset
1	1	0	reset
1	1	1	reset

No configuration required

4.15.16 Numeric operator

Component “numeric operator”

This component carries out a numeric operation with two incoming numeric values at input A and B and gives out the resulting numeric value at the numeric output. As soon as one of the input values changes, this operation is carried out. Data type of the numeric input and output is “double precision”. Further information about numeric components can be found on page 53.

NOTE: Consider the connection
The output and input of this component can only be connected with a numeric connection. Numeric connectors are always marked with “#”.

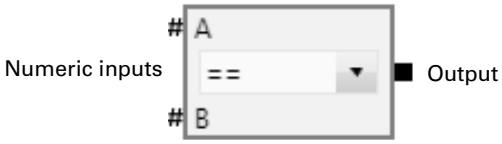
Start-up behaviour

Initially, the numeric output value is UNKNOWN.

Configuration ComPLC

› In the drop-down list, the numeric operator can be selected. The following numeric operations are available:
  - Addition “+”
  - Division “/”
  - Multiplication “*”
  - Subtraction “-”
    → default: “+”

4.15.17 Numeric relational operator



Component “numeric relational operator”

This component compares two incoming numeric values at input A and B with a numeric relational operator and gives out the resulting truth value at the output (TRUE or FALSE). As soon as one of the input values changes, this operation is carried out. Data type of the numeric input is “double precision”. Further information about numeric components can be found on page 51.

NOTE: Consider the connection
The input of this component can only be connected with a numeric output, whereas the output of this component can only be connected with a Boolean input. Numeric connectors are always marked with “#”.

Start-up behaviour

Initially, the output state is UNKNOWN.

Configuration ComPLC

› In the drop-down list, the numeric relational operator can be selected. The following numeric relational operations are available:
  - Equal “==”
  - Less than “<”
  - Greater than “>”
  - Unequal “!=”
  - Less than or equal “<=”
  - Greater than or equal “>=”

  → default: “==”

4.15.18 Numeric constant

Component “numeric constant”

This component represents a numeric constant, which is given out at the numeric output. Data type of the numeric output is “double precision”. Further information about numeric components can be found on page 51.

NOTE: Consider the connection
The output of this component can only be connected with a numeric input. Numeric connections are always marked with “#”.

Start-up behaviour

Initially, the output numeric value corresponds to the numeric constant value.
→ default: “0”

Configuration ComPLC

› In the field, the numeric constant can be entered.

NOTE: Decimal required
The numeric constant has to be entered in decimal.

4.15.19 Flags (numeric)

The flag components can be used for cross-scheme connection. Several “outbound” and “inbound” flag components can be connected within one or over several schemes to a circuit via the configurable “flag name” (see page 49). The connected flag components (circuit) operate like one component with several inputs or outputs, thus the connected flag components have not to be connected manually via lines. Each “outbound” flag component influences all connected “inbound” flag components in a configurable way. It is possible to place flag components of a circuit in different schemes.

Flag (outbound) **Flag (inbound)**

Component “flag (outbound)”Component “flag (inbound)”

Numeric flag components can only transmit numeric values. Therefore, it is only possible to connect a numeric output to an outbound numeric flag component, respectively a numeric input to an inbound flag component. Numeric connections are always marked with “#”. Further information about numeric components can be found on page 51.

Configuration ComPLC

- **Flag name**: In this field, enter the name of the flag circuit. All flags with the same flag name within all active schemes are connected.

> **NOTE: Same name for all connected flags**
> It is necessary to configure the same flag name for all connected components.

- Click on the button a to enable the drop-down list for a quick search of all available flag circuits. In this drop-down list, an existing flag circuit can be selected instead of entering the flag name in the field “flag name”. In this case, only numerical flag components are indicated.

In online mode, the value of inbound and outbound numeric flag components are indicated in the upper right corner. See the following illustration:

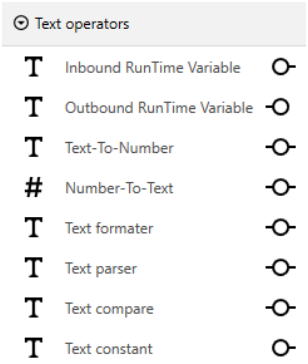
Component “flag (numeric)”: component value indication in online mode

- 1 Current flag value, which is the same for all connected numeric flag components.

> **NOTE: Error message**
> If two or more outbound flag components are configured with the same name, an error message will be indicated with activating the online mode or at upload of the configuration to VirtuoSIS.

4.15.20 Text operators

Text operators can be used to publish text by using runtime variables, to convert string and numeric inputs, to parse XML strings, to perform string formatting, to compare input strings or to specify a textual constant.

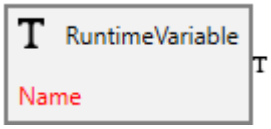


Component category "Text operators"

Click on the tab "Text operators" to display all Intercom components. The following Intercom components are available:

- Inbound RunTime Variable (see page 56)
- Outbound RunTime Variable (see page 57)
- Text-To-Number (see page 57)
- Number-To-Text (see page 57)
- Text formater (see page 58)
- Text parser (see page 58)
- Text compare (see page 58)
- Text constant (see page 59)

4.15.21 Inbound RunTime Variable



Component "Inbound RunTimeVariable"

This component publishes a textual value of a runtime variable. If the runtime variable is not set, "unknown" is published. For more information see also "Use runtime variables" on page 33.

NOTE: Reusable runtime variables
It is possible to copy parts of incoming ICX messages into reusable runtime variables (see "Runtime variables" on page 32).

Start-up behaviour

"Unknown"

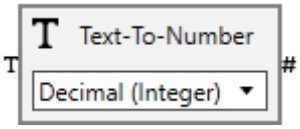
4.15.22 Outbound RunTime Variable

Component "Outbound RunTimeVariable"

This component allows the setting of the value of a runtime variable. As soon as the input port value changes, the runtime variable with the configured name is set to this value. If the same runtime variable is set from various places, the last change will be prioritised. For more information, see also "Use runtime variables" on page 33.

NOTE: Reusable runtime variables
It is possible to copy parts of incoming ICX messages into reusable runtime variables (see "Runtime variables" on page 32).

4.15.23 Text-To-Number



Component "Text-To-Number"

This component converts the string input port value to a numeric value.

Start-up behaviour

Initially the component value is *"unknown"*.

Input value change

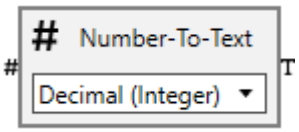
As soon as one of the input value changes, the operation is performed depending on the selected format.

Available formats

- Decimal (Integer): converts the given text into a numeric integer value (max. 2.147.483.648, min. -2.147.483.648).
- Decimal (Float): converts the given text into a numeric floating point value (no real limits, the higher (positive or negative) the number, the more inaccurate is the result, max. 15 decimal digits of precision), *"."* and *","* are accepted as decimal separators.
- Hexadecimal: converts a given hexadecimal noted string into an integer value.

If the input string is not parsable into the configured format, *"unknown"* is published.

4.15.24 Number-To-Text



Component "Number-To-Text"

This component converts the numeric input value into a string representing the numeric value in the given format.

Start-up behaviour

Initially the component value is *"unknown"*.

Input value change

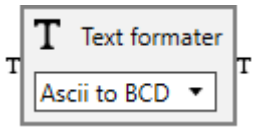
When the input value changes, the operation is performed depending on the selected format.

Available formats

- Decimal (Integer): converts the given text into a numeric integer value (max. 2.147.483.648, min. -2.147.483.648).
- Decimal (Float): converts the given text into a numeric floating point value (no real limits, the higher (positive or negative) the number, the more inaccurate is the result, max. 15 decimal digits of precision), "." and "," are accepted as decimal separators.
- Hexadecimal: converts a given hexadecimal noted string into a integer value.

If the input string is not parsable into the configured format, "unknown" is published.

4.15.25 Text formater



Component "Text formater"

This component performs a string formatting with the given format option.

Start-up behaviour

Initially the component value is "unknown".

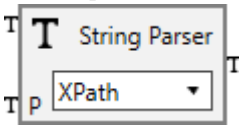
Input value change

As soon as one of the input value changes, the operation is performed depending on the selected format.

Available formats

- Ascii to BCD: converts the input string in a "binary coded decimal" string (used e.g. in ICX messages).
- BCD to Ascii: decodes a string in BCD format to an ASCII string.
- ToLower: converts all letters of the input string to lowercase letters.
- ToUpper: converts all letters of the input string to uppercase letters.

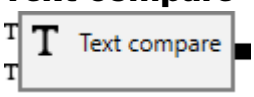
4.15.26 Text parser



Component "Text parser"

This component parses XML strings (input T) into text via XPath. XPath uses path expressions (input P) to select nodes or node-sets in an XML document.

4.15.27 Text compare



Component "Text compare"

This component is able to compare input strings.

Start-up behaviour

Initially the component value is "unknown".

Input value change

As soon as one of the input values changes, the component compares the two input strings. If they are equal, "true" is transmitted. Otherwise "false" is transmitted.

4.15.28 Text constant

Component "Text constant"

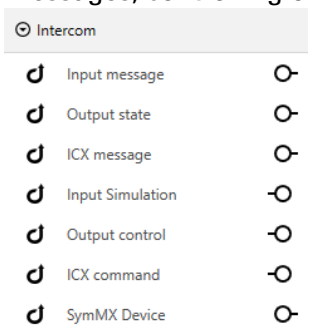
This component represents a textual constant that can be used for various text components.

Start-up behaviour

Always represent the configured value.

4.15.29 Intercom

The Intercom components carry out Intercom actions like sending and receiving ICX messages, controlling outputs or simulate input states.



Component category "Intercom"

› Click on the tab "Intercom" to display all Intercom components. The following Intercom components are available:
  - Input message see page 60
  - Output (inbound) see page 61
  - ICX message see page 61
  - Input Simulation see page 62
  - Output control (outbound) see page 63
  - ICX command see page 63
  - SymMX Device see page 64

4.15.30 Input message

Component "input message"

This component is triggered by certain incoming ICX messages with configured call numbers, control desks and call types.

Start-up behaviour

- Initially, the output state is FALSE.
- In order to receive the current output state (TRUE or FALSE) from VirtuoSIS, a control desk ICX synchronization message is sent (task number "80" and type number "6E").

Status change (FALSE → TRUE)

A TRUE output state results when an incoming ICX message with the respective call type is received (task number "5B", "7B" and type number = configured call type).

Status change (TRUE → FALSE)

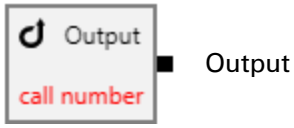
A FALSE output state results when an incoming ICX message deletes the previously received ICX message (task number "5B", "7B" and type number **"**30").

Configuration ComPLC

- **Call number**: In this field, enter the call number of the desired input that triggers this component.
- In the drop-down list, select the desired call type that triggers this component.

NOTE: Further information
For more information about call types, see manual "**ICX Protocol**".

4.15.31 Output state



Component "output (inbound)"

This component is triggered by certain incoming ICX messages with configured call numbers.

Start-up behaviour

- Initially, the output has an UNKNOWN state.
- In order to receive the current output state (TRUE or FALSE) from VirtuoSIS, an output ICX synchronization message is sent (task number "80" and type number "51").

Status change (FALSE → TRUE)

A TRUE output state results when the respective incoming ICX message is received (task number "5B", "7B" and type number "41").

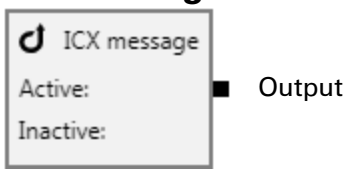
Status change (TRUE → FALSE)

A FALSE output state results when the respective incoming ICX message deletes the command of the previously received ICX message (task number "5B", "7B" and type number "40", "43", "45").

Configuration ComPLC

- **Call number**: In this field, enter the call number of the desired output that triggers this component.

4.15.32 ICX message



Component "ICX message" (inbound)

This component is triggered by incoming generic ICX messages.

Start-up behaviour

As soon as an "inactive" message is configured, the initial state has to be defined. The following states are available:

- KeepLastValue - the initial state will be UNKNOWN until the active or inactive message is received.
- False - the initial value is set to FALSE
- True - the initial value is set to TRUE

Status change (FALSE → TRUE)

A TRUE output state results when the respective incoming ICX message is received that was configured in the field "Active".

Status change (TRUE → FALSE)

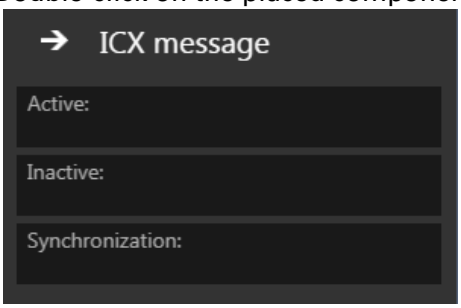
A FALSE output state results when the respective incoming ICX message is received that was configured in the field "Inactive".

Special behaviour

Without a configured "inactive" message, the components startup value is "false". If the active message is received, the components emits a true/false pulse only.

Configuration ComPLC

› Double-click on the placed component to open the following configuration dialogue:



Component "ICX message" configuration dialogue

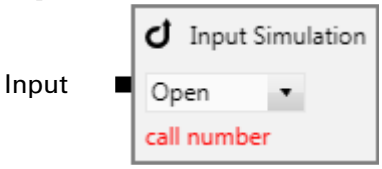
NOTE:
› "X" characters may be used as wildcards within the entered ICX message to ignore specific parts of the incoming ICX message.
› ICX messages have to be entered in long format (only in combination with VirtuoSIS).

› **Active**: In this field, enter an ICX message to change the output state to TRUE if it is received from VirtuoSIS.
› **Inactive**: In this field, enter an ICX message to change the output state to FALSE, if it is received from VirtuoSIS.

NOTE: If an inactive ICX message is not configured
If no "inactive" ICX message is configured, the output issues a TRUE/FALSE pulse when the "active" ICX message is received.

› **Trigger message**: Enter an ICX synchronization message to obtain the current status from the intercom system. This message is sent to the Intercom Server in case of a restart.
› **Value after restart and synchronization:** After entering an ICX message in the field **Inactive**, this field appears. Select the value of the output state after restart and synchronization.
› Click on the button ➔ to close the dialogue.

4.15.33 Input Simulation



Component "input simulation"

This component simulates configured input levels by sending ICX messages to VirtuoSIS.

Status change (FALSE → TRUE)

If the input state changes to TRUE, an ICX message with a configurable simulated input level is sent to VirtuoSIS (task number "80" and type number "A3"; input = configured input call number; input level = configured input level).

Status change (TRUE → FALSE)

If the input state changes to FALSE, an ICX message with a non-configurable simulated input level is sent to VirtuoSIS (task number “80” and type number “A3”; input = configured input call number; input level = open).

Configuration ComPLC

- Select the desired input level (open, 15k, 5k6, 1k5 or short) that is simulated by an outgoing ICX message.
- **Call number**: In this field, enter the call number of the desired input that receives the outgoing ICX message.

4.15.34 Output control

Component “output (outbound)”

This component switches output levels via outgoing ICX messages to VirtuoSIS.

Status change (FALSE → TRUE)

If the input state changes to TRUE, an ICX message to activate an output is sent to VirtuoSIS (task number “80” and type number “40”; output = configured output call number; parameter 2 = 0100 “on”).

Status change (TRUE → FALSE)

If the input state changes to FALSE, an ICX message to deactivate an output is sent to VirtuoSIS (task number “80” and type number “A3”; output = configured output call number; parameter 2 = 0000 “off”).

Configuration ComPLC

- **Call number**: In this field, enter the call number of the desired output that receives the outgoing ICX message.

4.15.35 ICX command

Component “ICX command”

This component sends generic ICX messages to VirtuoSIS.

Status change (FALSE → TRUE)

If the input state changes to TRUE, the ICX message entered in the field “Active” is sent to VirtuoSIS.

Status change (TRUE → FALSE)

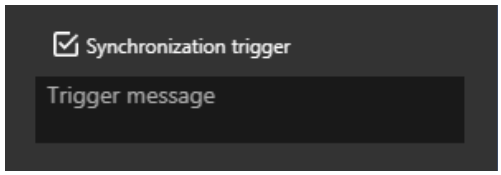
If the input state changes to FALSE, the ICX message entered in the field “Inactive” is sent to VirtuoSIS.

Configuration ComPLC

› Double-click on the placed component to open the following configuration dialogue:

Component "ICX command" configuration dialogue

› **Active**: In this field, enter an ICX message that is sent to VirtuoSIS if the input state changes to TRUE.
› **Inactive**: In this field, enter an ICX message that is sent to VirtuoSIS if the input state changes to FALSE.
› **Synchronization trigger**: Activate this checkbox to activate the triggering of the component via a configurable ICX synchronization message. The following setting will be activated:



Configure an ICX synchronization messages

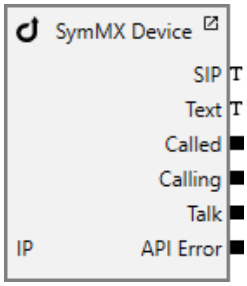
› **Trigger message**: In this field, enter an ICX synchronization message. If this ICX message is received, the current activated ICX message in the field **Active** or **Inactive** (see above) is sent.

ATTENTION: Long ICX format required for VirtuoSIS
This ICX message has to be configured in long format when using VirtuoSIS!

› Click on the button ➔ to close the dialogue.

4.15.36 SymMX Device

NOTE: HTTPS certificate for MX Device API access
An HTTPS certificate is mandatory for secure access to the MX Device API (see "General" on page 14).



Component "SymMX Device"

› **SIP**: A string port that shows the SIP URI of the called or calling party.
› **Text**: If the called or calling party is configured as a contact on the device, the contact's name is shown at this port.

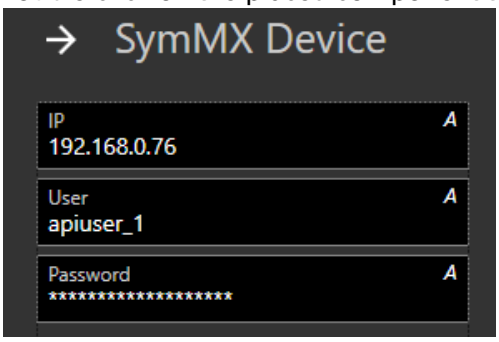
- **Called**: This Boolean port is active if the device state is “Incoming Call“.
- **Calling**: This Boolean port is active if the device state is “Outgoing Call“.
- **Talk**: This Boolean port is active if the device state is “In Call“.

NOTE: Multiple call states
If there are multiple incoming or outgoing calls, the call shown by the component is not defined. The device state “In Call“ is always given priority and shown by the component.

- **API Error**: This Boolean port is active if there is a connection error between ComPLC and the device via the MX Device API. If this is the case, consult the runtime journal (see page 37) or the log files for a detailed error description.

Configuration ComPLC

- Double-click on the placed component to open the following configuration dialogue:



Component “SymMX Device” configuration dialogue

- **IP**: In this field, enter the IP address of the Symphony MX device.
- **User**: In this field, enter the token key of the user.

NOTE: Permission “API Access for Symphony MX Device“ required
To be able to access the MX Device API, the user role allocated to the user must have the permission “API Access for Symphony MX Device“. For more information, see the Symphony MX device‘s manual.

- **Password**: In this field, enter the token value of the user.
- A : Indicates that auto complete of variables is available.

4.15.37 KNX

KNX is a two wired based building automation bus. The communication is realised by a KNX net/IP gateway (KNX net/IP coupler), which is using a KNX net/IP tunnelling protocol. In case of a lost connection, ComPLC is trying to reconnect automatically. Further information about the configuration of KNX connections can be found on page 19. The following connections are used:

- KNX net/IP coupler: Siemens N 148/22 IP interface (SWG 148-1AB22)
- KNX net/IP ports: UDP 3671/3672

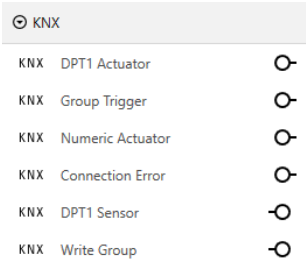
The communication between several installed devices is carried out via a group address. The following level address structures are available:

- **2-level structure**: Main group/subgroup (historically from ETS1, practically not used anymore).
- **3-level structure**: Main group/subgroup (used in ETS2 and ETS3, also suitable for larger projects, fixed folder sizes and quantities on 3 hierarchical levels).

NOTE:

- The level address structure can be changed via the project properties of each individual project.
- The group address “0/0/0” is an invalid address. Components with configured “0/0/0” address will not be processed.

An actuator can listen to several group addresses. However, a sensor can only send one group address per telegram. The group addresses are assigned to the group objects of the respective sensors and actuators.



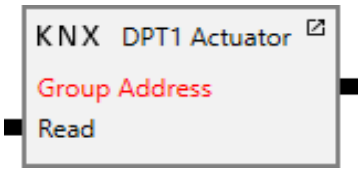
Component category “KNX”

- Click on the tab “KNX” to display all KNX components. The following KNX components are available:
  - DPT1 Actuator see page 67
  - Group Trigger see page 68
  - Numeric Actuator see page 68
  - Connection Error see page 69
  - DPT1 Sensor see page 70
  - Write Group see page 71

GOOD TO KNOW: How to assign a KNX connection to a KNX component?

In order to assign an existing KNX connection to a KNX component, carry out a double-click on the placed KNX component in the scheme and select the desired KNX connection in the drop-down list **Active connection** in the appearing dialogue (only one KNX connection can be assigned to a KNX component at the same time). Further information about the configuration of KNX connections can be found on page 19.

4.15.38 DPT1 Actuator



Component "DPT1 actuator"

This component is triggered by incoming "group write" telegrams of the configured KNX address and keeps its value until the reversed value (see table below) is received. This component supports the KNX data type "DPT1".

Start-up behaviour

- Initially, the output state is TRUE or FALSE depending on the received read answer. If no read answer is received, the component state stays UNKNOWN.
- In order to get the current output state (TRUE or FALSE) a read request of the configured group address is sent to the KNX bus.

Status change (FALSE → TRUE)

A TRUE output state results when a KNX write telegram (or read answer) is received with an "active" value (see table below).

Status change (TRUE → FALSE)

A FALSE output state results when a KNX write telegram (or read answer) is received with an "inactive" value (see table below).

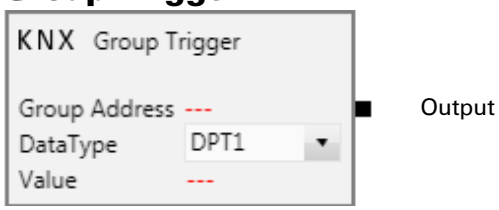
DPT table

DPT	Name	Active	Inactive
1.001	DPT_Switch	On	Off
1.002	DPT_Bool	True	False
1.003	DPT_Enable	Enable	Disable
1.004	DPT_Ramp	Ramp	No ramp
1.005	DPT_Alarm	Alarm	No alarm
1.006	DPT_BinaryValue	High	Low
1.007	DPT_Step	Increase	Decrease
1.008	DPT_UpDown	Down	Up
1.009	DPT_OpenClose	Close	Open
1.010	DPT_Start	Start	Stop
1.011	DPT_State	Active	Inactive
1.012	DPT_Invert	Inverted	Not inverted
1.013	DPT_DimSendStyle	Cyclically	Start/Stop
1.014	DPT_InputSource	Calculated	Fixed
1.015	DPT_Reset	Reset command	No action
1.016	DPT_Ack	Acknowledge	No action
1.017	DPT_Trigger	Trigger	No action
1.018	DPT_Occupancy	Occupied	Not occupied
1.019	DPT_Window_Door	Open	Closed
1.021	DPT_LogicalFunction	Logical function AND	Logical function OR
1.022	DPT_Scene_AB	Scene B	Scene A
1.023	DPT_ShutterBlinds_Mode	Move up/down + StepStop mode (blind)	Only move up/down mode (shutter)

Configuration ComPLC

- **Group Address**: Enter the group address of the device that triggers this component.
- **Read**: If a rising edge is detected at this port, ComPLC manually tries to read the configured group address from the KNX bus.

4.15.39 Group Trigger



Component "group trigger"

This component is triggered by incoming "group write" telegrams of the configured group address and value.

Start-up behaviour

Initially, the output state is FALSE.

Status change (FALSE → TRUE → FALSE)

The component publishes a TRUE value, immediately followed by a FALSE value (trigger) as soon as KNX write telegram with the configured value is received.

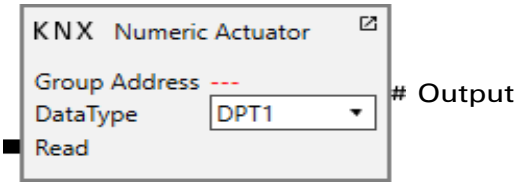
Configuration ComPLC

- **Group Address**: In this field, enter the group address that triggers this component.
- **DataType**: In this drop-down list, the data type for standardised communication can be selected. The following data types are available:

DPT	KNX function
1	Switch
3	Dimming (position, control, value)
5	8-bit unsigned value
6	8-bit signed value
7	16-bit unsigned value
8	16-bit signed value
9	Floating point
14	IEEE floating point

- **Value**: In this field, enter the value that triggers this component.

4.15.40 Numeric Actuator



Component "numeric actuator"

This component is available with ComPLC 3.0 or higher. The component is triggered by incoming "group write" telegrams of the configured group address. The received value of this telegram will be put out at the output.

Start-up behaviour

Initially, the output state is UNKNOWN.

Configuration ComPLC

- **Group Address**: In this field, enter the group address that triggers this component.
- **DataType**: In this drop-down list, the data type for standardised communication can be selected. The following data types are available:

DPT	KNX function
1	Switch
3	Dimming (position, control, value)
5	8-bit unsigned value
6	8-bit signed value
7	16-bit unsigned value
8	16-bit signed value
9	Floating point
14	IEEE floating point

- **Read**: If a rising edge is detected at this port, ComPLC manually tries to read the configured group address from the KNX bus.

4.15.41 Connection Error

Component "connection error"

This component indicates a connection error between the application ComPLC and the configured KNX net/IP coupler.

Start-up behaviour

Initially, the output has an UNKNOWN state.

Status change (FALSE → TRUE)

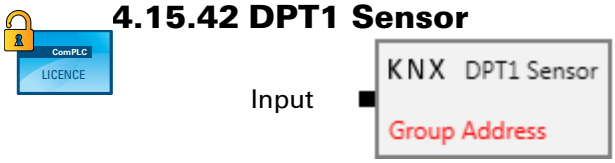
A TRUE output state results as soon as the connection between the application ComPLC and the configured KNX net/IP coupler is broken.

Status change (TRUE → FALSE)

A FALSE output state results as soon as the connection between the application ComPLC and the configured KNX net/IP coupler is (re-)connected.

No configuration required

4.15.42 DPT1 Sensor



Component "DPT1 sensor"

This component sends "write" telegrams and answers to "read" telegrams. This component supports the KNX data type "DPT1".

NOTE: In case of a broken connection
In case of a broken connection between the application ComPLC and the configured KNX net/IP coupler the respective value is sent again after reconnection.

Status change (FALSE → TRUE)

If the input state changes to TRUE, a write telegram is sent with value "1" (TRUE) to the configured group address.

NOTE: Sending a "read" answer
If a read telegram with the respective group address is received, a "read" answer is sent with the current value (0 or 1).

Status change (TRUE → FALSE)

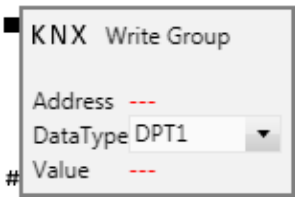
If the input state changes to FALSE, a write telegram is sent with value "0" (FALSE) to the configured group address.

NOTE: Sending a "read" answer
If a read telegram with the respective group address is received, a "read" answer is sent with the current value (0 or 1).

Configuration ComPLC

› **Group Address**: In this field, enter the group address that receives the write telegram if these components gets triggered.

4.15.43 Write Group



Component "write group"

This component can send write telegrams with a configured value.

Status change (FALSE → TRUE)

If the input state changes to TRUE, a write telegram is sent with the configured value to the configured group address.

Status change (TRUE → FALSE)

No action

Configuration ComPLC

- **Address**: In this field, enter the group address that receives the write telegram if these components gets triggered.
- **DataType**: In this drop-down list, the data type for standardised communication can be selected. The following data types are available:

DPT	KNX function
1	Switch
3	Dimming (position, control, value)
5	8-bit unsigned value
6	8-bit signed value
7	16-bit unsigned value
8	16-bit signed value
9	Floating point
14	IEEE floating point

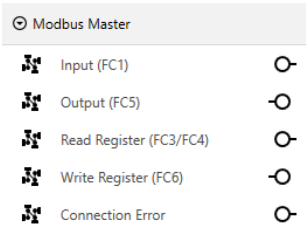
- **Value**: In this field, enter the value that is sent with the outgoing write telegram (0 to 63). With ComPLC version 3.0 or higher, this value can individually be configured by a numerical value via the numerical input (e.g. with a counter; see page 52).

4.15.44 Modbus Master

Modbus (TCP) is a widely used protocol to communicate to PLCs (Programmable Logic Controllers) and remote input/output units. Modbus (TCP) is a master/slave protocol (the master polls the slave to receive current values), whereby ComPLC is always the Modbus (TCP) master. In case of a lost connection, ComPLC is trying to reconnect automatically. Further information about the configuration of Modbus connections can be found on page 18. The following technical requirements has to be observed:

- Each Modbus data point is addressed by a 16-bit decimal value (“0” to “65535”)
- The Modbus address has to be entered in decimal format.
- The Modbus input components are queried within an interval of 10 ms.
- **Modbus port**: TCP 502

NOTE: Pay attention for invalid addresses
Depending on the interfaced slave device, it is possible that the entire interface will not work if there is any invalid address assigned to an input or output.

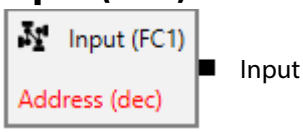


Component category “Modbus master”

- Click on the tab “Modbus master” to display all Modbus components. The following Modbus components are available:
  - Input (FC1) see page 73
  - Output (FC5) see page 75
  - Read Register (FC3/FC4) see page 73
  - Write Register (FC6) see page 75
  - Connection Error see page 75

GOOD TO KNOW: Assigning a Modbus connection to a Modbus component
In order to assign an existing Modbus connection to a Modbus component, carry out a double-click on the placed Modbus component in the scheme and select the desired Modbus connection in the drop-down list **Active connection** in the appearing dialogue (only one Modbus connection can be assigned to a Modbus component at the same time). Further information about the configuration of Modbus connections can be found on page 18.

4.15.45 Input (FC1)



Component "input (FC1)"

This component is triggered by a received value of the configured address by using Modbus function code "FC1".

Start-up behaviour

- Initially, the output has an UNKNOWN state.
- The current output state is initialised during the first "read" interval. If the configured Modbus slave is not reachable, the output state stays UNKNOWN.

Status change (FALSE → TRUE)

A TRUE output state results when the value "1" (TRUE) of the configured address is received.

Status change (TRUE → FALSE)

A FALSE output state results when the value "0" (FALSE) of the configured address is received.

Configuration ComPLC

- **Address (dec)**: In this field, enter the address of the desired Modbus data point. This address is polled with a configurable polling interval (see "Modbus" on page 18).

4.15.46 Output (FC5)

Component "output (FC5)"

This component sends "write" data points using Modbus function code "FC5".

NOTE: In case of a broken connection
In case of a broken connection between the application ComPLC and the configured Modbus TCP slave device, the respective value is sent again after reconnection.

Status change (FALSE → TRUE)

If the input state changes to TRUE, the value "1" (TRUE) is sent to the configured Modbus address.

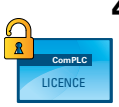
Status change (TRUE → FALSE)

If the input state changes to FALSE, the value "0" (FALSE) is sent to the configured Modbus address.

Configuration ComPLC

- **Address (dec)**: In this field, enter the Modbus address that receives the write telegram if these components are triggered. This address is polled at a configurable polling interval (see "Modbus" on page 18).

4.15.47 Read Register (FC3/FC4)



Component "read register (FC3)"

This component periodically reads the numeric values of polling intervals (unsigned word) of an output register by using Modbus function code "FC3" or "FC4". Further information about numeric components can be found on page 51.

Start-up behaviour

Initially, the output has an UNKNOWN state.

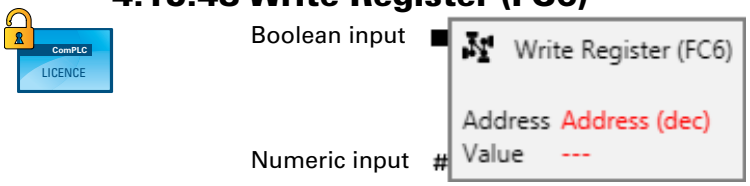
Read value change

The numeric output publishes the value which is received from the configured address to all connected components.

Configuration ComPLC

› **Address**: In this field, enter the address of the desired Modbus register. This address is polled with a configurable polling interval (see page 16).

4.15.48 Write Register (FC6)



Component "write register (FC6)"

This component changes numeric values (unsigned word) of an output register by using Modbus function code "FC6". Further information about numeric components can be found on page 51.

Status change (FALSE → TRUE)

If the input state changes to TRUE, the configured value is sent to the configured address of an output register.

> **NOTE: Connection error**
> In case of a connection error between ComPLC and the Modbus slave device, the configured value is sent again after reconnection as long as the input state of the respective component is TRUE.

Status change (TRUE → FALSE)

No action

Configuration ComPLC

- **Address**: In this field, enter the Modbus address that receives the telegram if these components gets triggered.
- **Value**: In this field, enter the value that is sent to the output register. With ComPLC version 3.0 or higher, this value can individually be configured by a numerical value via the numerical input (e.g. with a counter; see page 52).

4.15.49 Connection Error

Component "connection error"

This component indicates a connection error between the application ComPLC and the configured Modbus slave.

Start-up behaviour

Initially, the output has an UNKNOWN state.

Status change (FALSE → TRUE)

A TRUE output state results as soon as the connection between the application ComPLC and the configured Modbus slave is broken.

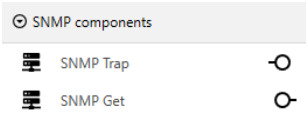
Status change (TRUE → FALSE)

A FALSE output state results as soon as the connection between the application ComPLC and the configured Modbus slave is (re-)established.

No configuration required

4.15.50 SNMP components

With SNMP components generic SNMP Traps can be send and “SNMP get” queries of configured OIDs can be performed.



Component category “SNMP components”

- Click on the tab “SNMP components”. The following SNMP components are available:
  - SNMP Trap (see page 76)
  - SNMP Get (see page 78)

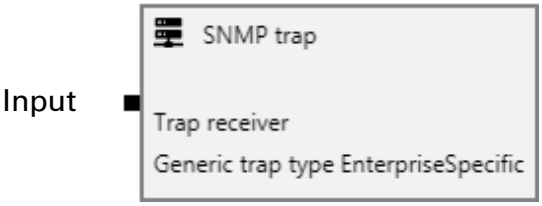
GOOD TO KNOW: What is SNMP?

The “Simple Network Management Protocol” (SNMP) is an Internet Standard protocol for collecting and organizing information about managed devices on IP networks. Note that there are three versions of the SNMP protocol:

- SNMPv1 is the original version of the protocol
- SNMPv2 contains improvements in performance, flexibility and security
- SNMPv3 includes cryptographic security (not supported)

The different versions are not compatible; therefore you need to make sure all devices working together support the same version.

4.15.51 SNMP trap



Component “SNMP trap”

This component sends SNMP traps.

Status change (FALSE → TRUE)

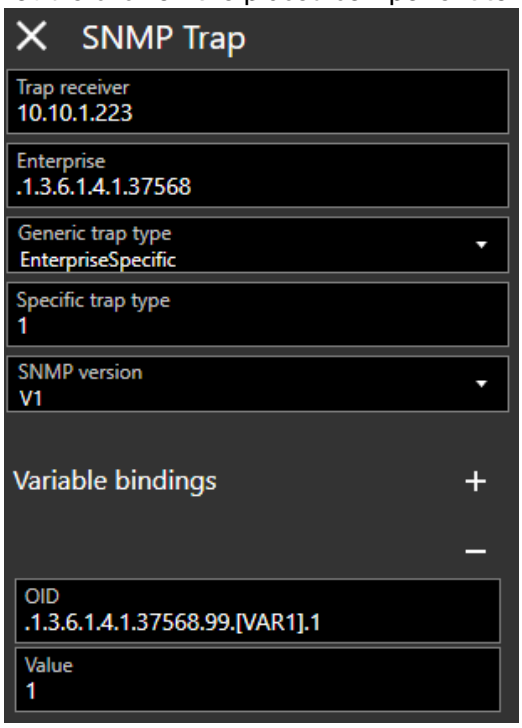
If the input state changes to TRUE, the configured SNMP trap is sent to the configured receiver.

Status change (TRUE → FALSE)

No action

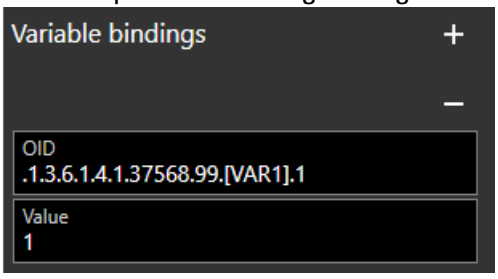
Configuration ComPLC

› Double-click on the placed component to open the following configuration dialogue:



Component "SNMP trap" configuration dialogue

› **Trap receiver**: In this field, enter the IP address or host of the SNMP trap receiver. This IP address is displayed in the respective placed component.
› **Enterprise**: In this field, enter the numerical OID ("object identifier") of the SNMP sender.
› **Generic trap type**: In this drop-down list, the SNMP trap type can be selected. The selected type is displayed in the respective placed component.
› **Specific trap type**: In this field, a specific SNMP trap type can be entered. Therefore, the SNMP trap type "EnterpriseSpecific" has to be selected in the drop-down list "Generic trap type".
› **SNMP version**: In this drop-down list, the SNMP version can be selected.
→ default: "v1"
› **Variable bindings**: Click on the button ⊞ to add a variable, which will be sent with the SNMP trap. The following settings will be activated:



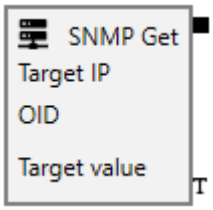
Configure different variables for specific needs

› **OID**: In this field, enter the OID of the respective value.
› **Value**: In this field, enter the value. If this value is signed as integer, the value is sent with data type "integer". All other values are sent with data type "octet string".
» Click on the button ⊟ to remove the respective variable.
› Click on the button ➔ to close the dialogue.

3.2

GOOD TO KNOW: Runtime variables can be used as values
In the field **Value** also runtime variables can be entered.

4.15.52 SNMP Get



Component "SNMP get"

This component performs "SNMP get" queries of configured OIDs. A "SNMP get" query is carried out every five seconds. The component can compare received value with a given target value. A string output port represents the last received value.

Start-up behaviour

- Initially, the output has an UNKNOWN state.
- The current output state is initialised during the first query interval.

Status change (FALSE → TRUE)

A TRUE output state results as soon as the configured "target value" is equal to the received "SNMP get" answer.

Status change (TRUE → FALSE)

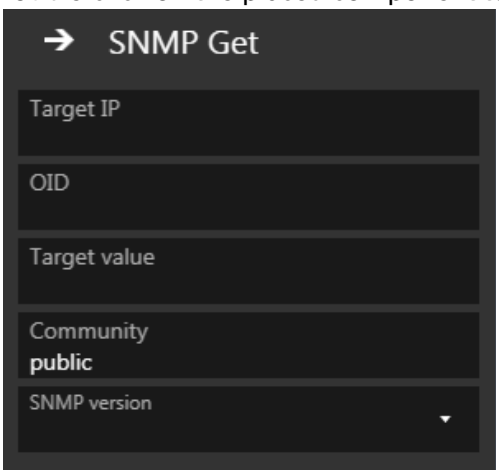
A FALSE output state results as soon as the configured "target value" is not equal to the received "SNMP get" answer or the configured target IP is not reachable.

String port

Publishes the last received value. In case of timeout or SNMP errors this port will respond with an UNKNOWN state.

Configuration ComPLC

- Double-click on the placed component to open the following configuration dialogue:



Component "SNMP get" configuration dialogue

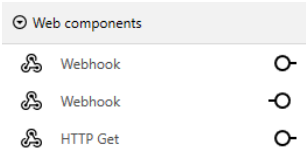
- **Target IP**: In this field, enter an available IP address for the SNMP query. This IP address is displayed in the respective placed component.
- **OID**: In this field, enter an available numerical OID ("object identifier") for the SNMP query. This OID is displayed in the respective placed component.
- **Target value**: In this field, enter a target value of the queried "target IP". This value is displayed in the respective placed component.
- **Community**: In this field, the password to get access to the "SNMP MIBs" can be changed. → default: "public"

- **SNMP version**: In this drop-down list, the SNMP version can be selected.
  → default: “v1”
- Click on the button ➔ to close the dialogue.

4.15.53 Web components

Webhooks – for general use

A webhook is a simple event notification via HTTP to carry out different actions in a ComPLC or third-party system. Each webhook consists of the destination URL with the desired parameters.



Component category “SNMP components”

The following webhook components are available:

- **Webhook (inbound)**: Receive HTTP request webhooks to carry out actions in the ComPLC system (see page 79).
- **Webhook (outbound)**: Send HTTP request webhooks to carry out actions in another ComPLC system or any third-party system (see page 81).
- **HTTP Get**: Sends configurable HTTP request webhooks. (see page 83)

Construction of the URL to change runtime variable values

In order to change the value of runtime variables (see page 32) in ComPLC systems with HTTP request webhooks via a third-party system or any ComPLC system, use the URL “http://user:password@<host-IP>:<port>/variables?variable=<name>&value=<value>” and replace all wildcards with the following parameters:

- **host-IP**: IP address of the respective host system (VirtuoSIS), which contains the desired component to be controlled (see page 14).
- **port**: Port to communicate with the respective host system (VirtuoSIS), which contains the desired component to be controlled (see page 14).
- **name**: Variable name.
- **value**: New variable value.

GOOD TO KNOW: Change runtime variable with a ComPLC webhook component

If the runtime variable shall be changed by a webhook outbound component from any ComPLC system, enter the URL above in the field **URL** of the respective component (see page 81).

4.15.54 Webhook (inbound)

Component “Webhook (inbound)”

This component gets triggered by incoming HTTP request webhooks with a certain ID in order to set its output state with a specific command. When the components gets triggered, this component will response with a HTTP response status code. HTTPS is supported.

Start-up behaviour

Initially, the output has an FALSE state.

Status change

The output state results as soon as a HTTP request webhook with the given value in the URL is received (see below).

HTTP response status code

If the component gets triggered by a HTTP request webhook, this component will response with a status code to the sender. The following status codes are available:

Code	Message
200	Standard response for successful HTTP requests.
400	Malformed request syntax, invalid request message framing (see page 81).
401	Authentication has not been provided or failed (see page 15).

Configuration ComPLC

- **ID**: In this field, enter the ID of the HTTP request webhook that triggers the component.

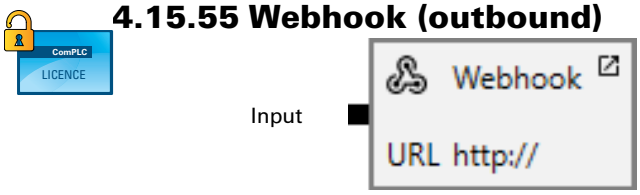
> **NOTE: Web server has to be configured**
> In order to receive HTTP request webhooks, the web server has to be configured (see page 15).

Construction of the URL to change component values

In order to change the value of components in ComPLC systems with HTTP request webhooks via a third-party system or any ComPLC system, use the URL "http://user:password@<host-IP>:<port>/components/bool?id=<ID>&value=<value>" and replace all wildcards with the following parameters:

- **host-IP**: IP address of the respective host system (VirtuoSIS), which contains the desired component to be controlled (see page 14).
- **port**: Port to communicate with the respective host system (VirtuoSIS), which contains the desired component to be controlled (see page 14).
- **ID**: Identifier of the HTTP request webhook. Enter this ID in the field **ID** of the respective webhook inbound component (see above).
- **value**: Command to set the output state. Select one of the following commands:
  - **active**: Set the output state of the respective component to TRUE.
  - **true**: Set the output state of the respective component to TRUE.
  - **inactive**: Set the output state of the respective component to FALSE.
  - **false**: Set the output state of the respective component to FALSE.
  - **toggle**: Invert the output state of the respective component.
  - **invert**: Invert the output state of the respective component.
  - **trigger**: Set the output state of the respective component to TRUE and back to FALSE.
  - **pulse**: Set the output state of the respective component to TRUE and back to FALSE.

4.15.55 Webhook (outbound)



Component "Webhook (outbound)"

This component sends configurable HTTP request webhooks (e. g. to control external devices). HTTPS is supported.

Start-up behaviour

Initially, the output has an UNKNOWN state.

Status change (FALSE → TRUE)

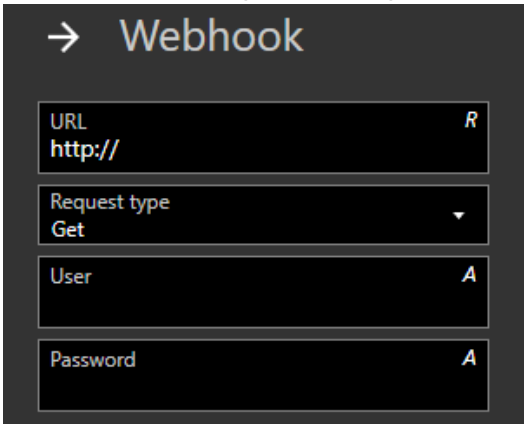
If the input state changes to TRUE, the configured HTTP request is sent.

Status change (TRUE → FALSE)

No action.

Configuration ComPLC

› Double-click on the placed component to open the following configuration dialogue:



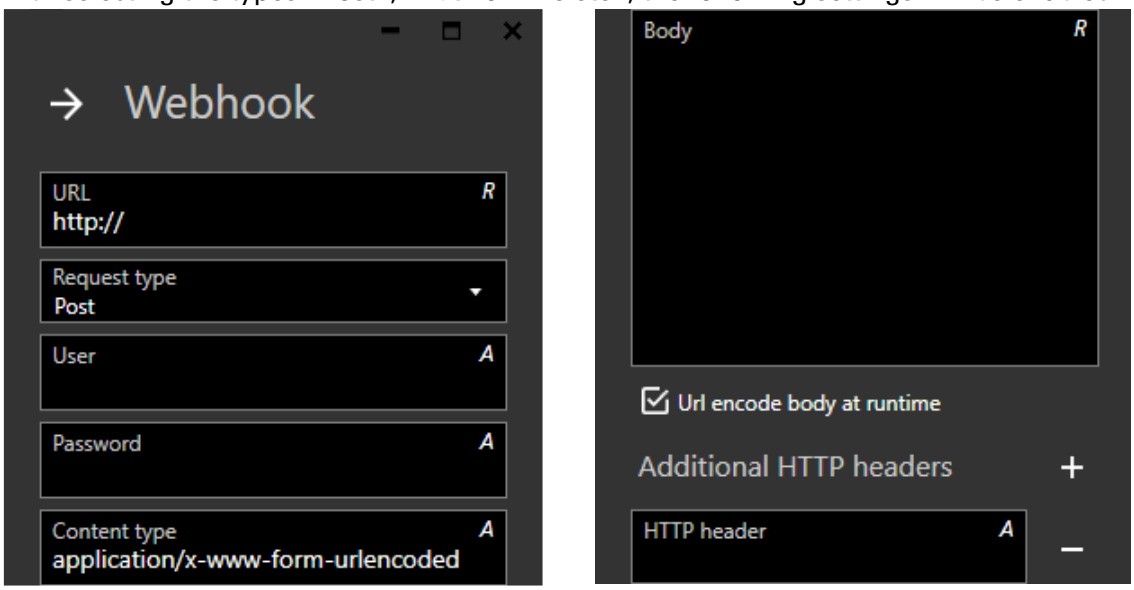
Component "Webhook (outbound)" configuration dialogue

› **URL**: In this field, enter the destination URL of the HTTP request webhook (see page 80).
  » R : Indicates that runtime variables can be used.
› **Request type**: In this drop-down list, the HTTP request type can be selected. The following types are available:
  » **Get**: Retrieve the information that is identified by the URL.
  » **Post**: Add the enclosed data as a new subordinate of the resource identified by the URL.
  » **Put**: Save the enclosed data under the given URL.
  » **Delete**: Delete the files under the given URL.

GOOD TO KNOW: More information about request types
In order to get more information about the available HTTP request types, visit the web page www.w3.org.

› **User**: In this field, enter the user name.
  » A : Indicates that auto complete of variables is available.
› **Password**: In this field, enter the password of the user.
  » A : Indicates that auto complete of variables is available.

- With selecting the types “Post”, “Put” or “Delete”, the following settings will be enabled:



Component “Webhook (outbound)” configuration dialogue

- **User**: In this field, enter the user name.
  - A : Indicates that auto complete of variables is available.
- **Password**: In this field, enter the password of the user.
  - A : Indicates that auto complete of variables is available.
- **Content type**: In this field, the standardised content type of the HTTP request webhook can be changed (see page 83). → default: “application/x-www-form-urlencoded”
  - A : Indicates that auto complete of variables is available.
- **Body**: In this field, enter the desired parameters. Only ASCII characters can be used. Enter the attribute description, followed by the character “=” and the desired value. (The values are encoded in key-value tuples separated by “&”, with a “=” between the key and the value, e.g. key=val&key1=value1&... .)

  NOTE: Case-sensitive
  Parameter values in the field **Body** are case-sensitive.

  - R : Indicates that runtime variables can be used.
  - ☑ **Url encode body at runtime**: Can only be enabled if default content type “application/x-www-form-urlencoded” is used. Otherwise an error message will appear.
- **Additional HTTP headers**: To define more than one HTTP header click on ➔ in the “Webhook” dialogue to enable the “Additional HTTP headers” dialogue and enter the desired HTTP header.
  - A : Indicates that auto complete of variables is available.

Construction of the content type

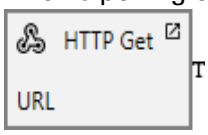
Enter the structure “<Type>/x-<Subtype>” in the field **Content type** and replace both wildcards with the following parameters:

- **Type** (transferred data type):
  - **application**: Other kind of data, typically either uninterpreted binary data or information
  - **audio**
  - **image**
  - **message**: Encapsulated message
  - **multipart**: Data consisting of multiple parts of independent data types
  - **text**
  - **video**
- **Subtype**: Any character except spaces or special characters (“/”, “)”, “<”, “>”, “@”, “,”, “;”, “:”, “\”, <“>, “/”, “[” ,“]” ,“?” ,“.” , “=”).

NOTE: Use quoted-strings for parameter values
Implement special characters in quoted-strings to use them within parameter values.

4.15.56 HTTP Get

Allows polling of HTTP/HTTPS content from a given URL.



Component “HTTP Get”

This component sends configurable HTTP request webhooks.

Start-up behaviour

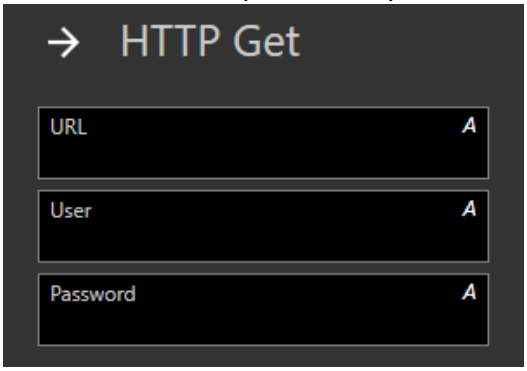
Initially, the output has an UNKNOWN state.

Status change

The outbound String-Port represents the received HTTP content.
In case of a timeout, UNKNOWN is published (connect timeout = 2 sec. receive timeout = 3 sec.)

Configuration ComPLC

- Double-click on the placed component to open the following configuration dialogue:



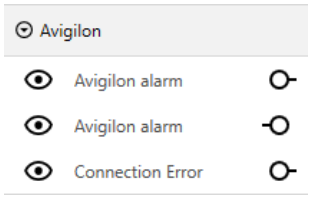
Component “HTTP Get” configuration dialogue

- **URL**: In this field, enter the requested URL of the HTTP content.
- **User**: In this field, enter the user name.
- **Password**: In this field, enter the password of the user.
- A : Indicates that auto complete of variables is available.

4.15.57 Avigilon

> **NOTE: Avigilon HTTPS certificate is mandatory**
> To connect ComPLC with Avigilon a certificate is required! To use Avigilon, always upload an Avigilon HTTPS certificate to ComPLC.

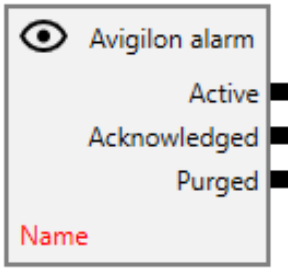
The Avigilon components carry out specific actions.



Component category "Avigilon"

- Click on the tab "Avigilon" to display all components. The following components are available:
  - Avigilon alarm (inbound)
  - Avigilon alarm (outbound)
  - Connection error

4.15.58 Avigilon alarm (inbound)

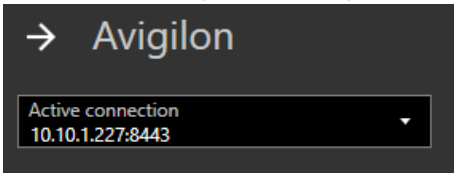


Component "Avigilon alarm

- **Active**: Signals that the Avigilon alarm is active.
- **Acknowledged**: Signals an acknowledged alarm.
- **Purged**: Signals that the Avigilon alarm is purged.
- **Name**: Enter the name of the alarm in the Avigilon system.

Configuration ComPLC

Double-click on the placed component to open the following configuration dialogue:

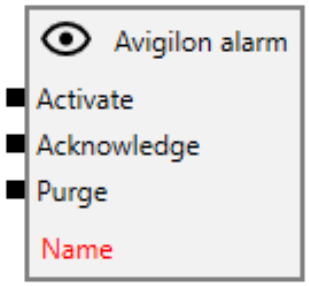


Component "Avigilon (inbound)" configuration dialogue

Active connection: In this field, select the Avigilon connection.

> **NOTE:**
> Alarm names are case sensitive.

4.15.59 Avigilon alarm (outbound)

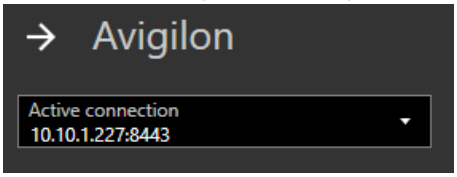


Component "Avigilon alarm (outbound)"

- **Activate**: Activate the Avigilon alarm.
- **Acknowledge**: Acknowledge the Avigilon alarm.
- **Purge**: Purge the Avigilon alarm.
- **Name**: Enter the name of the alarm in the Avigilon system.

Configuration ComPLC

Double-click on the placed component to open the following configuration dialogue:



Component "Avigilon (outbound)" configuration dialogue

Active connection: In this field, select the desired Avigilon connection.

NOTE:
Alarm names are case sensitive.

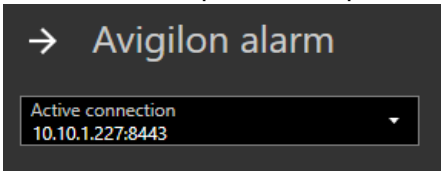
4.15.60 Avigilon connection error

Component "Connection Error"

This component indicates a connection error between the application ComPLC and the configured Avigilon alarm system.

Configuration ComPLC

Double-click on the placed component to open the following configuration dialogue:

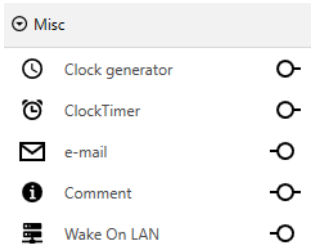


Component "Avigilon alarm" configuration dialogue

Active connection: In this field, select an active Avigilon connection. → default: "8443"

4.15.61 Misc

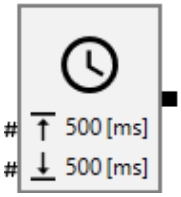
The miscellaneous components carry out specific actions.



Component category "Misc"

› Click on the tab "Misc" to display all miscellaneous components. The following components are available:
  » Clock generator see page 86
  » ClockTimer see page 87
  » Email see page 88
  » Comment see page 89
  » Wake on LAN (see page 89)

4.15.62 Clock generator



Component "clock generator"

› : Signal high timespan
› : Signal low timespan

This component triggers at the configured time interval.

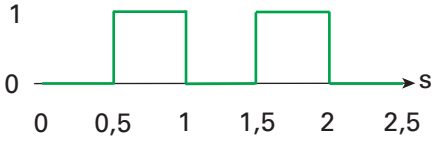
Status change (FALSE → TRUE)

A TRUE output state if the signal is high.

Status change (TRUE → FALSE)

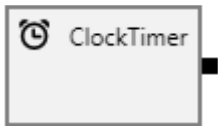
A FALSE output state if the signal is low.

Digital timing diagram



›

4.15.63 ClockTimer



Component "clock timer"

This component is triggered at the configured day and time. Depending on the Intercom Server that runs ComPLC, this component receives the current time in the following way:

› **VirtuoSIS**: Use the current system time from the host.

Status change (FALSE → TRUE)

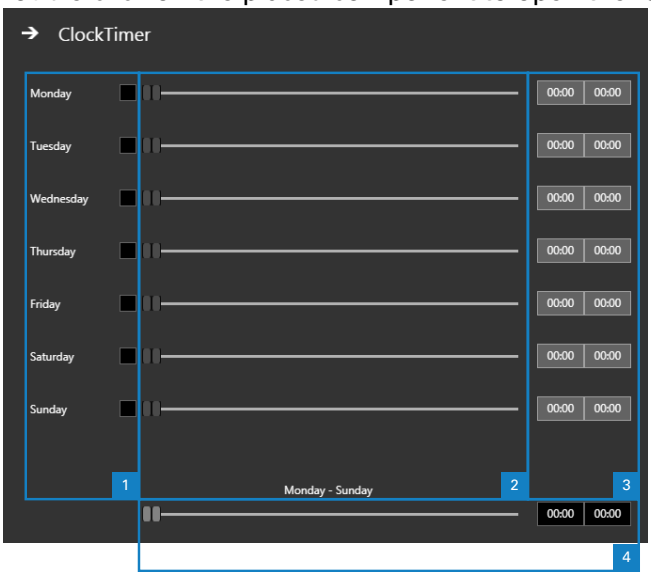
A TRUE output state results if the current day and time matches with the configuration.

Status change (TRUE → FALSE)

A FALSE output state results if the current day and time does not match with the configuration.

Configuration ComPLC

› Double-click on the placed component to open the following configuration dialogue:



Component "clock timer" configuration dialogue

› 1 Activate the desired checkbox to trigger the component at the respective day.

› 2 Put the sliders on the desired position to set a time range for triggering the component at the respective day (left slider for start and right slider for end). The adjusted time can also be configured in the field 3 to the right.

› 3 In the first field, enter the start time and in the second field the end time (format "hours:minutes"), to set a time range for triggering the component at the respective day. The adjusted time can also be configured in the bar 2 on the left.

› 4 Move the sliders to the desired position (left slider for start and right slider for end) and enter the start time in the first field and the end time in the second field (format "hours:minutes") to set a time range for several days.

NOTE: Separate configuration
The time range and operation state of each day can still be configured separately.

› Click on the button ➔ to close the dialogue.

4.15.64 e-mail

Component "e-mail"

This component sends e-mails via SMTP. In order to configure the SMTP server settings (see page 15).

Status change (FALSE → TRUE)

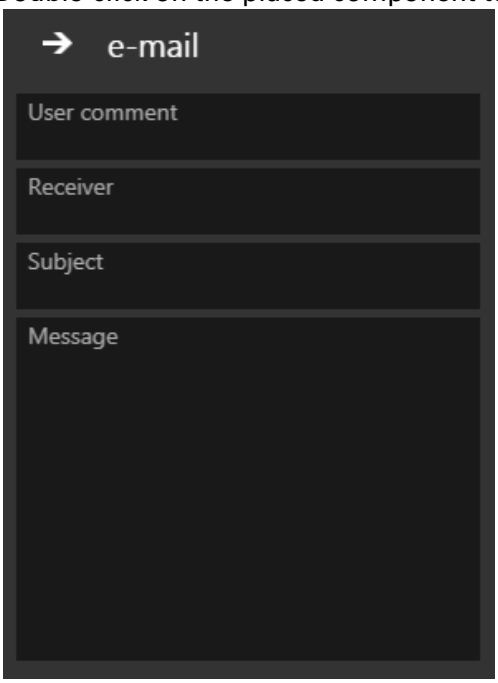
If the input state changes to TRUE, the configured e-mail is sent to the configured receiver.

Status change (TRUE → FALSE)

No action

Configuration ComPLC

› Double-click on the placed component to open the following configuration dialogue:



Component "e-mail" configuration dialogue

› **User comment**: In this field, enter the description of the component (this comment does not affect the transmission). This description is displayed in the respective placed component.
› **Receiver**: In this field, enter the email address of the e-mail receiver.

> **NOTE: Several receivers**
> Several e-mail receivers are separated with a semicolon.

› **Subject**: In this field, enter the subject of the email (this comment does not affect the transmission).
› **Message**: In this field, enter the email text.
› Click on the button to close the dialogue.

4.15.65 Comment

Component "comment"

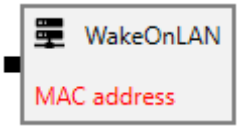
This component serves for comments in the scheme. This component does not affect the configured logic, nor is affected by other components.

Configuration ComPLC

› Click on the field next to the button .

In this field, enter the desired comment.

4.15.66 Wake on LAN (WoL)

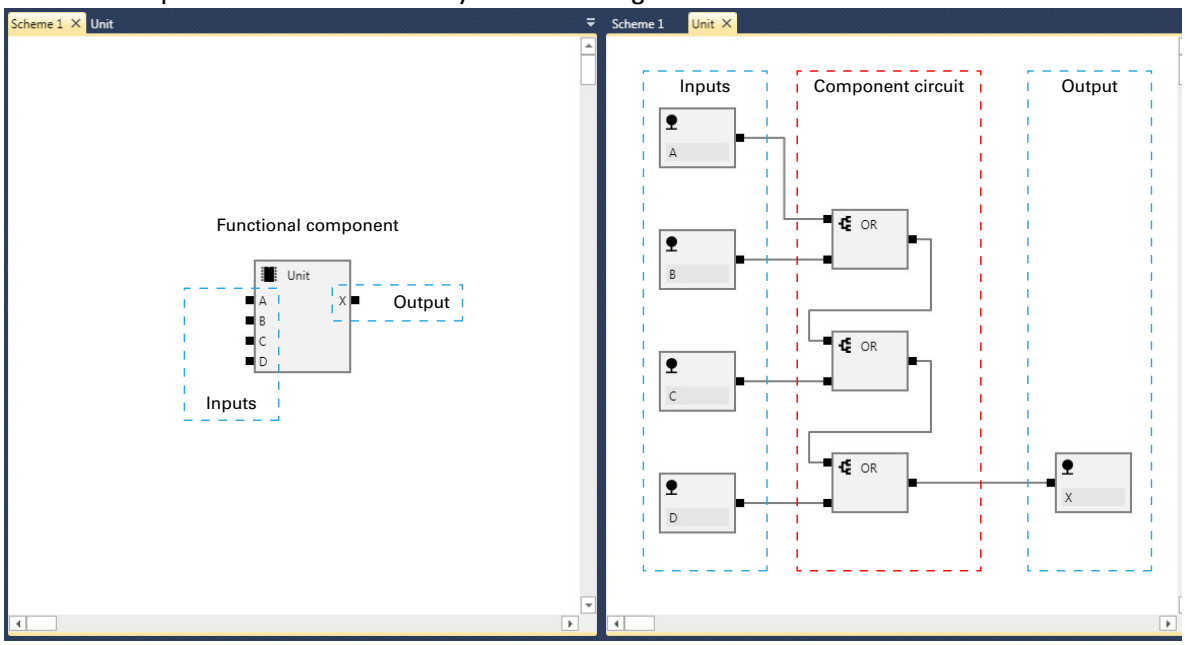


Component "wake on LAN"

If this component becomes active, a "Magic packet" is sent as broadcast containing the configured MAC address.

4.15.67 Function component

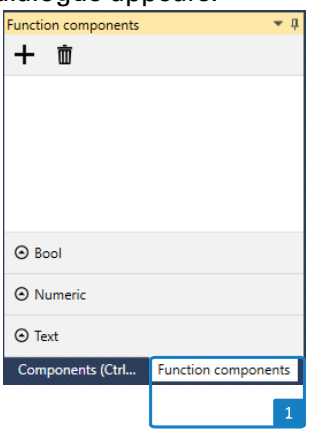
With function components, every circuit can be combined to a single component, which can be used unlimited in the entire ComPLC project. This facilitates time consuming configuration of frequently used component circuits. Each placed component operates as independent unit with an own instance, whereby the unit carries out the configured circuit regardless other placed units in the ComPLC project. Every function component consists of any number of Boolean and/or numerical inputs and outputs, via which the function component can read-in or read-out values. In the instance, you can connect all inputs and outputs of the unit with your desired component circuit to create your own integrated circuit.



Configure your own customised integrated circuit

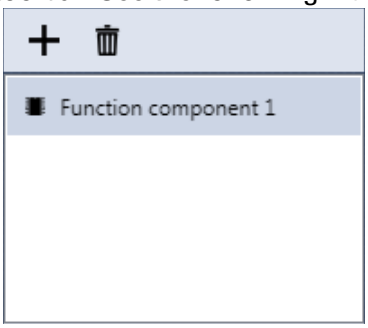
Configuration ComPLC

› Click on the tab “Function component” 1 to display all function components. The following dialogue appears:



Component category “Function component”

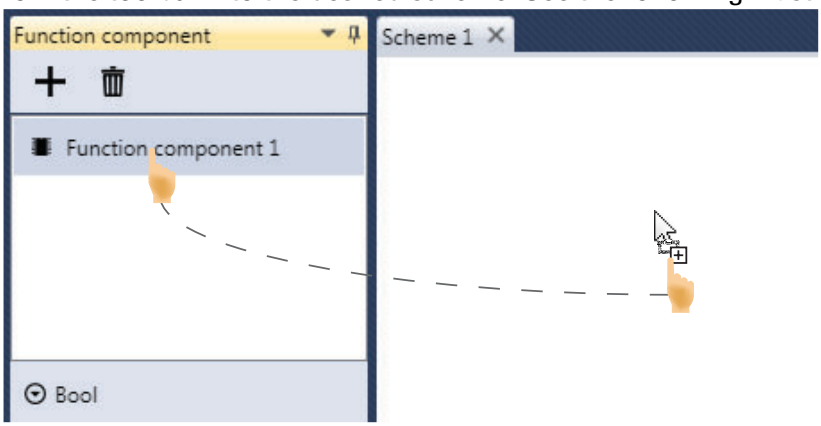
› Click on the button + to create a function component. This component is indicated in the toolbar. See the following illustration:



Several function components can be added

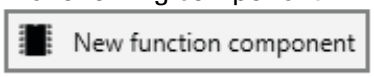
› **Rename**: Right-click on the desired function component in the toolbar, select “Rename”, then change the component name and press the *<enter>* key to confirm the entry.

› In order to configure the function component, drag & drop the added function component from the toolbar into the desired scheme. See the following illustration:



Drag & drop the component into the desired scheme

› The following component will be placed:



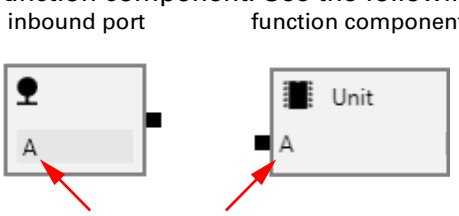
This component has no function yet

GOOD TO KNOW: Component has no function yet
Due to no inputs or outputs are pre-configured after it has been added, this component has no function yet.

› Double-click on the respective function component in the toolbar or scheme to open a new instance tab.

GOOD TO KNOW: Instance represents the internal configuration
This instance represents the internal configuration of the respective function component. In this tab, all ComPLC components (e.g. Intercom, KNX or operator) can be used to create any circuit. Each function component has its own instance.

› With the following components, the external input and output ports of the function component can be created:
  - **Boolean inbound port**: Represents the Boolean input of the function component.
  - **Boolean outbound port**: Represents the Boolean output of the function component.
  - **Numeric inbound port**: Represents the numeric input of the function component.
  - **Numeric outbound port**: Represents the numeric output of the function component.
  - **Text inbound port**: Represents the textual input of the function component.
  - **Text outbound port**: Represents the textual output of the function component.
› Drag & drop the desired input or output port component into the instance of the respective function component. See the following illustration.
› In the name field of the component, enter the desired description of the input or output port of the function component. This description will be indicated at the respective port of the function component. See the following illustration:



Name the ports of the function component

› Configure your desired component circuit in the respective instance tab of the function component (see example below).

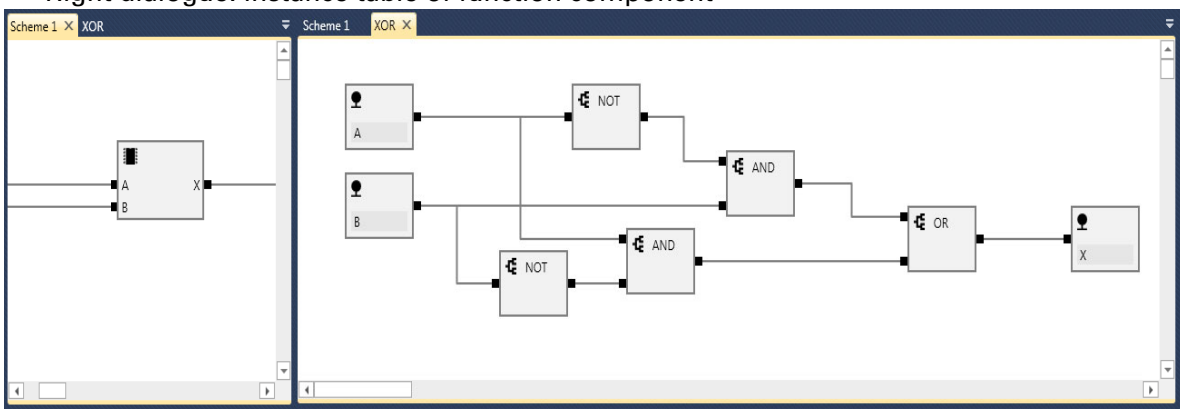
NOTE: Flags only valid within the same component
Flag components are only ever valid within the same function component (see page 89).

› **Delete**: Select the desired function component in the toolbar, click on the button and confirm the appearing note.

NOTE: Delete units in scheme first
In order to delete a function component, all units placed in a scheme has to be removed first manually.

Example: XOR gate

- Left dialogue: use in scheme
- Right dialogue: instance table of function component



Gate logic of a XOR gate

5. Appendix

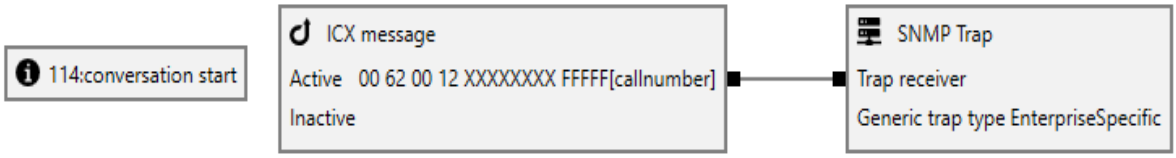
5.1 Example: SNMP Traps

This example demonstrates how to send SNMP traps related to the call state of an Intercom station. As soon as a conversation with a call number within the range of the variable [callnumber] is established, an SNMP trap with the value 1 will be sent.

GOOD TO KNOW: What is SNMP?
The “Simple Network Management Protocol” (SNMP) is an Internet Standard protocol for collecting and organising information about managed devices on IP networks.
Note that there are three versions of the SNMP protocol:
- SNMPv1 is the original version of the protocol
- SNMPv2 contains improvements in performance, flexibility and security
- SNMPv3 includes cryptographic security (not supported)

The different versions are not compatible, therefore you need to make sure all devices working together support the same version

GOOD TO KNOW: What are traps?
Independently from the values of the MIB, so-called traps will be sent. Traps are unrequested messages of an agent to the management station, reporting that a specific event has occurred (e.g. conversation between two SIP stations has been established).

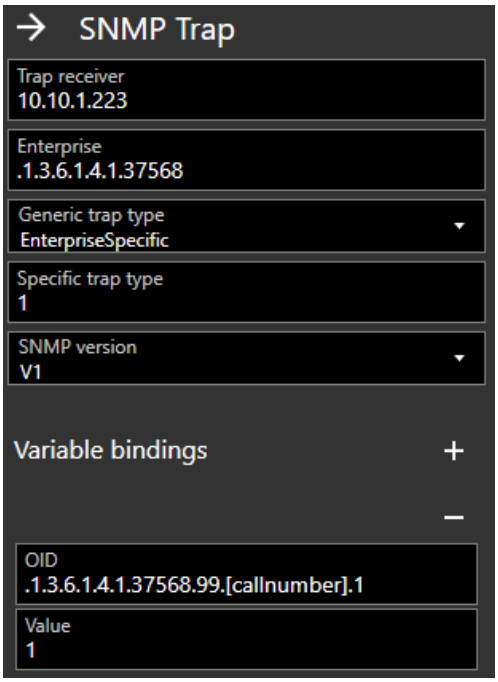


Example: SNMP traps

SNMP trap — conversation start

- Add a new logic: When a connection with a specified variable is established an SNMP trap is sent to the respective management software.

› Double click on the SNMP component. The following dialogue appears:



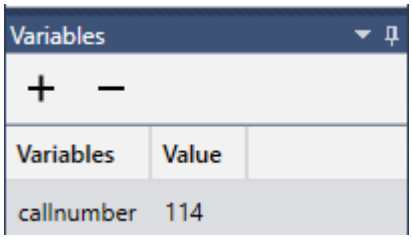
Dialogue "General"

› In the field "Trap receiver", enter the IP address of the receiving device.
› In the field "Enterprise", enter your enterprise ID.
› In the field "Generic trap type", choose a generic trap type.
› In the field "Specific trap type", you can enter a specific trap type.
› In the field "SNMP version", choose an SNMP version (we recommend V2).
› To open die dialogue "Variable bindings" click on the button ⊞.
› In the field "OID", enter the enterprise ID and an output number or a variable (for more information on variables see page 31).
› In the field "Value", enter a value or a text for the event (e. g. "1" or "conversation").

GOOD TO KNOW: Master templates
You can define templates and create various versions of a scheme (see "Instances" on page 28).

Define variable

› Define the variable [callnumber] in the dialogue "Variables" (see "Variables" on page 31):



Example: variable "callnumber" = 114

» When a connection is established with "callnumber" 114 an SNMP trap is sent to a management software.

6. Technical Support

More information about our products and services can be found at:

www.commend.com