C-Bus® Network Automation Controller

Wiser™ for C-Bus® Automation Controller

User Manual

This document describes the programming interface for the Automation Controller. The software programming interface is embedded in the Automation Controller and requires a web browser. Pre-programming and configuration cannot be performed without an Automation Controller product.

5500NAC

5500SHAC

clipsal.com

CLIPSAL®

by Schneider Electric

Legal information

The Schneider Electric brand and all registered trademarks of Schneider Electric Industries SAS referred to in this manual are the sole property of Schneider Electric SA and its subsidiaries. They may not be used without the written approval of the owner. This manual and all its content are protected in accordance with the laws on copyright for texts, drawings and models, as well as in accordance with the law on trademarks. You agree not to duplicate the manual, in whole or in part, on media of any kind, without the written approval of Schneider Electric, other than for personal, non-commercial use as defined by the code of law. Furthermore, you agree not to create any hyperlinks to this manual or its contents. Schneider Electric does not grant the right or permission to use the manual or its content for personal and non-commercial use, with the sole exception of the non-exclusive right to view the current version of the manual and its content, at the viewer's own risk. All other rights reserved. Electrical equipment must be installed, operated, maintained and repaired by qualified personnel only. Schneider Electric accepts no liability for the consequences of using this material.

This Schneider Electric product is provided with certain open source software components (collectively, "OSS") developed by third parties. The OSS is licensed in accordance with the OSS license terms set forth on page 210.

Trademarks

- Linux® is the registered trademark of the Linus Torvalds in the U.S. and other countries.
- iTunes™ is a registered or unregistered trademark of the Apple Inc. in the USA and/or other countries.
- Google Chrome™ and Google Play™ are trademarks or registered or unregistered trademarks of the Google Inc. in the USA and/or other countries.
- Firefox® is a trademark or registered trademark of the Mozilla Corporation in the USA and/or other countries.
- Microsoft Windows® is a trademark or registered trademark of the Microsoft Corporation in the USA and/or other countries.
- Mac OS X® are trademarks or registered trademarks of Apple Corporation in the USA and/or other countries.

Warnings

Read through the following instructions carefully and familiarise yourself with the device prior to installation, operation and maintenance. The warnings listed below can be found throughout the documentation and indicate potential risks and dangers, or specific information that clarifies or simplifies a procedure.

The addition of a symbol to “Danger” or “Warning” safety instructions indicates an electrical danger that could result in serious injuries if the instructions are not followed.

This symbol represents a safety warning. It indicates the potential risk of personal injury. Follow all safety instructions with this symbol to avoid serious injuries or death.

DANGER

DANGER indicates an imminently hazardous situation that will inevitably result in serious or fatal injury if the instructions are not observed**.**

WARNING

WARNING indicates a possible danger that could result in death or serious injuries if it is not avoided.

CAUTION

CAUTION indicates a possible danger that could result in minor injuries if it is not avoided.

NOTICE

NOTICE provides information about procedures that do not present any risk of physical injury.

Further information

The information provided must be complied with, otherwise program or data errors may occur.

Additional information is provided here to make your work easier.

Depictions in this document

Style and text features used

Text feature	Meaning
[F6]	**Keys** on the keyboard
[Ctrl] + [N]	Press both keyboard keys at the same time
Configurator	**Software module** (Start page) Click the button to open the software module.
Objects tab	**Tab** Click the tab to open its window. The window contains elements like lists, buttons, editors. Some windows are divided into several parts.
Objects list	**List** Click the tab (***Objects***) to display the list.
Event column	**Column**
Add new object button	**Button** Click the button to open a window, activate/ deactivate a function or to display parameters and values.
Configurator → ***Objects*** tab → ***Add new objects*** button	**Path**
Edit object *Application (Decimal Format)* • *56 - Lighting* • *228 - Measurement* • *250 - User parameters*	**Window** **Parameters** **Values**
Are you sure you want to delete all entries?	System notifications
Choice: **0 s**(instantaneous)/4 s ...	Pre-set values in the software are highlighted in bold in the tables.
Ground Floor/lighting/room 1	Composed address in the ***Name*** column in the ***Objects*** list
0/56/1	Composed address in the ***Group address*** column of the ***Objects*** list
Operation chapter	Cross-references
www.lua.org	URL address
`SetRelayState(true)`	LUA script
`Project-Hostname_yyyy_mm_dd.hh.mm.tar.gz`	File name

C-Bus operation

Requirements for safe operation

Knowledge of the basic rules for operating a computer and a browser such as Google Chrome or Firefox is a prerequisite for operation.

Knowledge of C-Bus operation is required. This includes using the C-Bus Toolkit software to select and commission C-Bus applications for export to a CGL file.

To use client-server communication with Modbus or BACnet, knowledge of these systems is required.

Special features of the C-Bus Toolkit

The C-Bus Toolkit Software enables configuration, exporting and importing of a C-Bus project.

Appropriate C-Bus Toolkit version

Use the current version of C-Bus Toolkit (1.15.x or higher). The included drivers are needed to access the Automation Controller via USB-B.

Firmware

It is recommended to update the firmware to install the latest features, security updates and bug fixes.

https://www.clipsal.com/Trade/Support/Software

Table of contents

1 For your safety

DANGER

HAZARD OF ELECTRIC SHOCK, EXPLOSION, OR ARC FLASH

- It is illegal for persons other than an appropriately licensed electrical contractors or other persons authorised by legislation to work on the fixed wiring of any electrical installation.
- To comply with all safety standards, the product must be used only for the purpose described in this instruction and must be installed in accordance with the wiring rules and regulation in the location where it is installed.
- There are no user serviceable parts inside the product.

Failure to follow these instructions will result in death or serious injury.

CAUTION

EQUIPMENT DAMAGE HAZARD

- Install the device according to instructions in the printed Quick Start Guide and in this document.
- Pay attention to the specifications and wiring diagrams related to the installation.
- Do not use this product for any other purpose than specified in the printed Quick Start Guide and in this instruction.

Failure to follow these instructions can result in minor injuries or equipment damage.

1.1 Qualified personnel

This document is aimed at personnel responsible for setting up, installing, commissioning and operating the device and the system in which it is installed. Personnel must possess basic knowledge in the following areas:

- Connection to installation networks
- Connecting several electrical devices
- Laying electric cables
- Connecting and establishing C-Bus networks and other used Building Management System (BMS) like BACnet or Modbus
- Commissioning C-Bus installations and other used BMS.

Detailed expertise gained by means of training in the C-Bus system and other used BMS is a prerequisite.

2 General overview

This document describes the programming interface for the Automation Controller. The software programming interface is embedded in the Automation Controller and requires a web browser. Pre-programming and configuration cannot be performed without an Automation Controller product.

Access the embedded web server via the Ethernet or USB type B connections on the Automation Controller (see *Getting started* on page 27).

2.1 Getting started

Access the embedded web server via the Ethernet or USB type B connections on the Automation Controller. The chapter *Getting started* on page 27 provides information about administrator access, saving data, setting date and time, firmware upgrade and monitoring the Automation Controller.

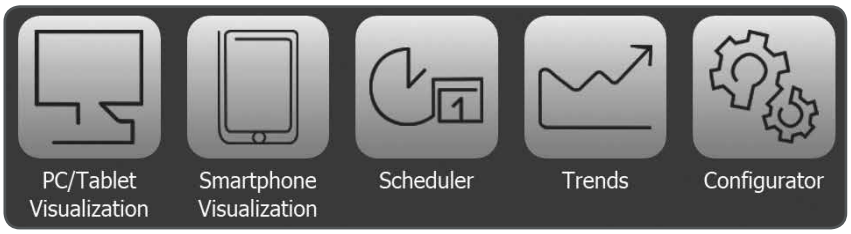
The default IP address for the ethernet connection is 192.168.0.10. The default IP address for the USB type B connection is 192.168.254.10. The administrator username is admin and the default password is admin.

2.2 Modules of the web server

The following modules of the web server can be accessed from the ***Start*** page:

- PC/Tablet Visualization: user module
- Smartphone Visualization: user module
- Scheduler: user module
- Trends: user module
- Configurator.

The following screen shot shows the buttons of the ***Start*** page:



Administrators can configure the four user modules via the Configurator. Once visualization pages, schedulers or trend logs have been configured, the result can easily be viewed. These four modules can be accessed by end users.

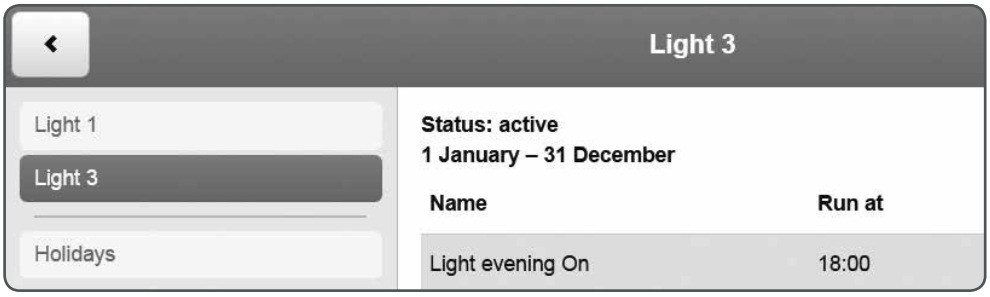
2.2.1 Modules for the end user

The following images show examples of pages for each user module.

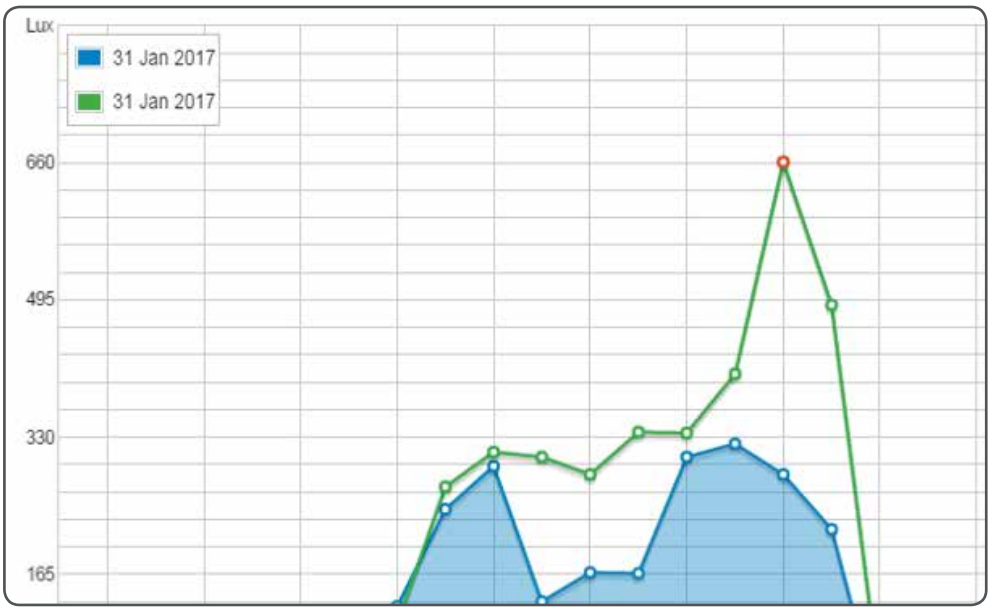
Page in the ***PC/Tablet Visualization*** user module:

Page in the ***Smartphone Visualization*** user module:

Page in the ***Scheduler*** user module:



Page in the ***Trends*** user module:



2.2.2 Configurator

Access the configurator from the ***Start*** page. Tabs on the configurator page provide access to different aspects of configuration. The configurator main page can:

- Access general and visualization configuration
- Open the System page to access network, update and reset functions, and status information.

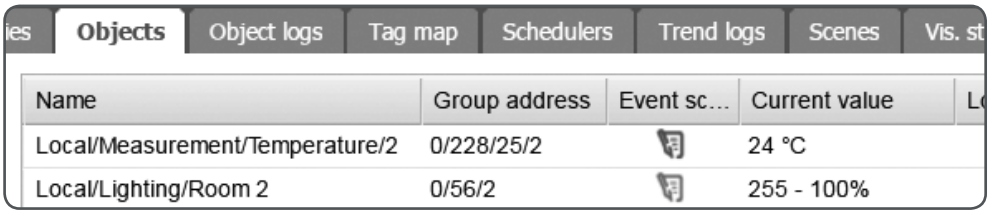
Utilities tab on the ***Configurator*** page:

C-Bus objects

Objects are the basis of all communication via the Automation Controller and represent different C-Bus applications for sending and receiving values. Objects can also be used for visualization, scripting, exchange with BACnet IP or Modbus and the control of outputs and inputs. The chapter *C-Bus objects* on page 38 describes different applications, how to add and edit objects and how to exchange applications configured with the C-Bus Toolkit.

Objects are used in the following steps of configuration.

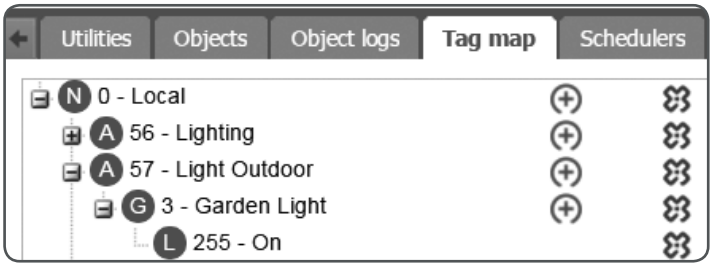
Objects tab with ***Object*** list on with 2 C-Bus objects:



Tag map

In addition to the objects list, a tag map is available (see *Tag map* on page 110). In this map, object names (tags) can be edited and used in other steps of configuration.

Tag map with network, application, group and level:

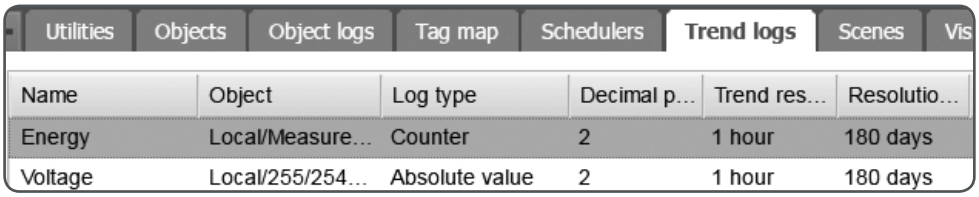


Scheduler and trends

Schedulers provide control of different building functions based on date and time (see *Schedulers* on page 112). Trend logs (data logging) store the selected data and compare that data over different time periods (see *Trend logs* on page 117).

Objects can be added in the configurator. Users can access schedulers or trends form the ***Start page*** or from a visualization. (Images of visualization pages are shown on page 17.)

Trend logs tab with two objects for logging:



Scenes

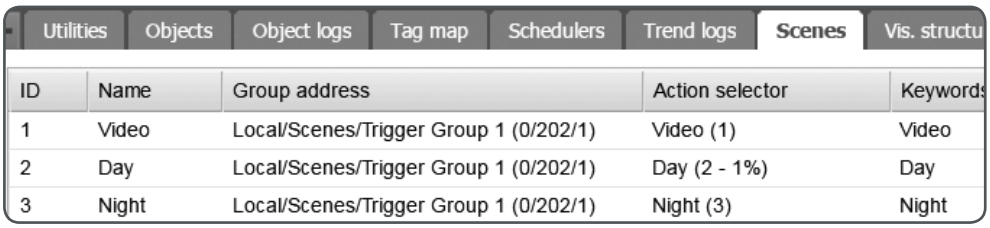
Scenes allows changes to multiple room functions at the touch of a button. For example, use a scene to dim the room lighting to a specific value, move the blinds into the desired position and switch on the power outlets.

Scenes configured in the Automation Controller can be controlled from a visualization page, the scheduler, or from other C-Bus devices such as push buttons.

The basis of scene communication is a *trigger group* (see *Add new trigger group* on page 43). The chapter *Scenes* on page 121 describes the configuration of scenes and all components (objects) linked with those scenes.

When configuring scenes on a visualization page or schedulers, add a trigger group object (*Objects* on page 144 and *Schedulers and events* on page 113).

Scenes tab with one trigger group and three scenes:



ID	Name	Group address	Action selector	Keywords
1	Video	Local/Scenes/Trigger Group 1 (0/202/1)	Video (1)	Video
2	Day	Local/Scenes/Trigger Group 1 (0/202/1)	Day (2 - 1%)	Day
3	Night	Local/Scenes/Trigger Group 1 (0/202/1)	Night (3)	Night

Visualization

The Automation Controller provides an embedded *Smartphone* and *PC/Tablet* visualization (see *Modules of the web server* on page 17).

The elements of a visualization, such as objects, can be configured on visualization plans (pages) that can be accessed on both visualization modules. The ***Smartphone Visualization*** provides a list with control objects and links. The ***PC/Tablet Visualization*** provides additional elements such as images and graphs.

The chapter *Visualization* on page 124 describes the main steps of configuration, how to create a structure, how to use the plan editor and how to configure different elements such as control types for C-Bus applications.

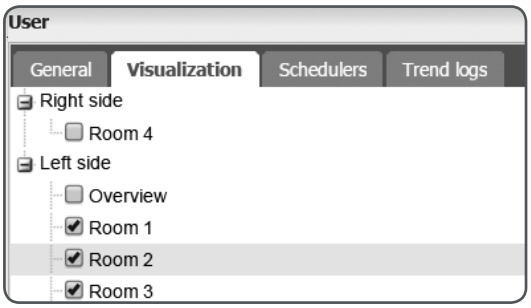
Visualization tab with plan editor and structure:

User access

The Automation Controller is designed for a maximum of 50 users for visualization. The Wiser™ for C-Bus® Automation Controller is limited to a maximum of 8 users.

The chapter *User access* on page 156 describes how to configure individual access rights. Access can be given without restriction for all users, with a common PIN code for all users, or with individual user logins. Different user access levels can be configured for visualization pages, schedulers and trends.

User window with selection of pages for a user (opened on the ***User Access*** tab):

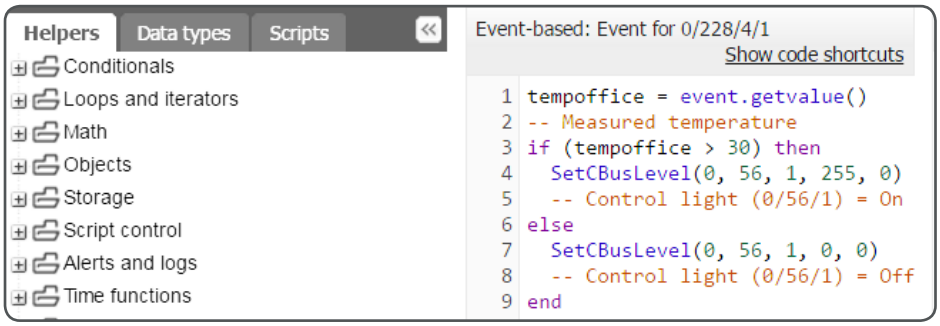


Scripting

A *script* is a small, non-compiled program written in the scripting language *LUA*. Event-based scripts are commonly used. In this case, define conditions and actions to perform when the object values meet certain criteria.

The chapter *Scripting* on page 160 provides an introduction to different types of scripts, script editor usage, the script commands available in the helpers tab, common functions like sending emails, tools and alerts. Find script examples in *Inputs and Outputs* on page 173 and *Modbus settings using scripts* on page 193.

Script editor with a script example (opened from a list on the ***Scripting*** tab):



CNI Functionality

CNI is enabled by default on port 10001. When enabled, the Automation Controller can act as a C-Bus network interface for commissioning and maintenance.

The CNI functionality in the Automation Controller can be enabled/disabled and the port can be changed, if desired, through the Automation Controller's configuration page.

In toolkit configure connection details for each C-Bus network are as follows:

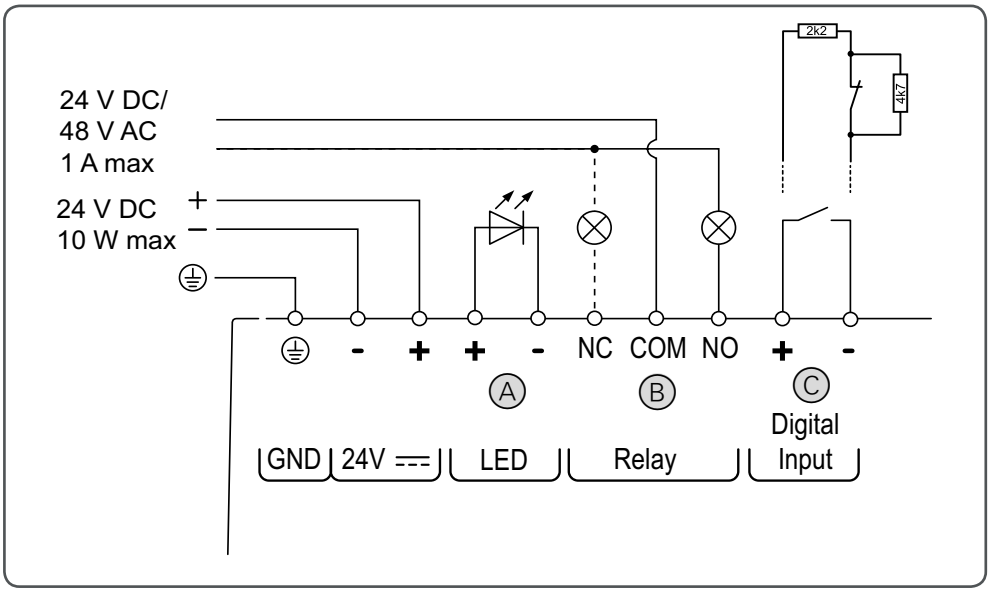
- Type: CNI
- Address: IP address
- The default IP address of the Automation Controller is 192.168.0.10 (or 192.168.254.10 if connected using the USB-B interface)
- Port: 10001.

Inputs and Outputs

The Automation Controller is equipped with a LED driver output, a relay output and a digital input. The chapter *Inputs and Outputs* on page 173 describes how to control these interfaces via scripts.

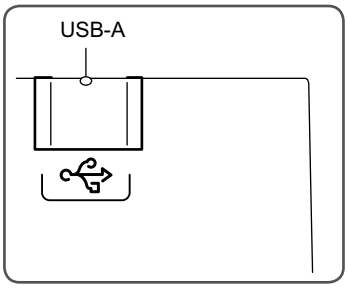
The LED and relay output can be controlled via C-Bus applications. The digital input of the Automation Controller is compatible with either a potential-free contact or a monitored cable using End of Line Resistance.

LED driver output Ⓐ, relay output Ⓑ and digital input Ⓒ:



USB-A

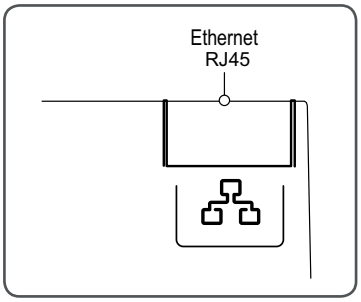
A USB type A connector in the Automation Controller provides connection with USB expansion devices (see *USB-A* on page 178). FAT and FAT32 file system formatted flash drives can be attached. Data transmission is performed via script configuration commands.



USB-A

Ethernet

The Automation Controller includes an RJ45 interface for 10/100 Base-T UTP Ethernet. Use the Ethernet connection to integrate IP devices (see *Camera* on page 155), web services (see *NTP client/server* on page 32) and additional building management functions (see *Modbus overview* on page 185 and *BACnet* on page 201). C-bus networks configured with the C-Bus Toolkit software can be commissioned and maintained in online mode via the configured Ethernet connection of the Automation Controller (on page 40).



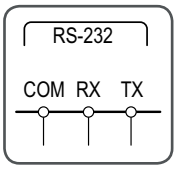
Ethernet RJ45 connector

RS-232

The RS-232 serial interface is one of the most widely used communication standards for data transmission between two devices over short distances.

Interaction with other equipment is possible via the isolated RS-232 interface in the Automation Controller (see *RS-232* on page 181). The script configuration commands are listed in the ***Helpers*** tab of the script editor.

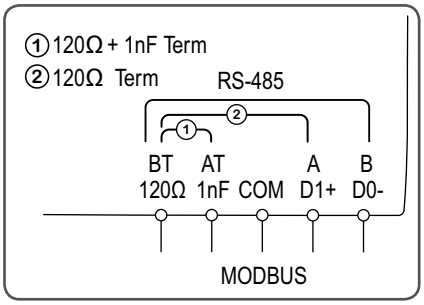
RS-232 interface connections: Common, Receive, Transmit.



RS-232

RS-485

The Automation Controller has an isolated RS-485 interface, which can be used for serial data transmission between two or more devices (see *RS-485* on page 183).



RS-485 connectors

Modbus

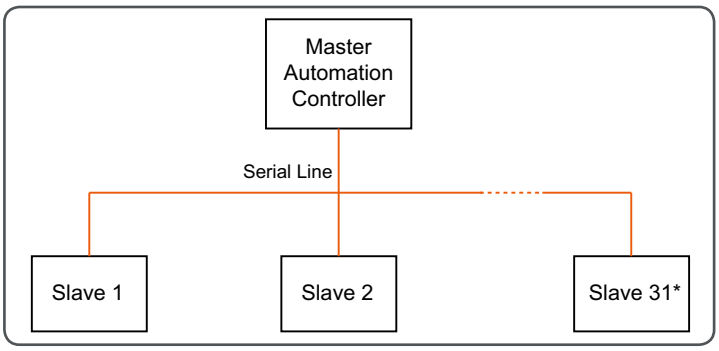
Modbus is an open standard for client/server communication. The client sends a request message and the addressed device (the server) sends a response message. The values of a server are saved in registers which can be accessed by the client. In Modbus *RTU mode*, the **client** is usually referred to as the **master** and the **server** as the **slave.**

The Automation Controller supports Modbus RTU serial communication via the RS-485 interface and Modbus TCP communication via the RJ45 Ethernet interface. It has been tested to conform to Modbus over serial line standards at the Modbus-SL Interoperability test lab in Marktheidenfeld.

By Modbus specification a serial line is limited to 31 slaves. The Network Automation Controller is not limited but designed for 31 slaves. The Wiser for C-Bus Automation Controller is limited to a maximum of 6 Modbus slaves.

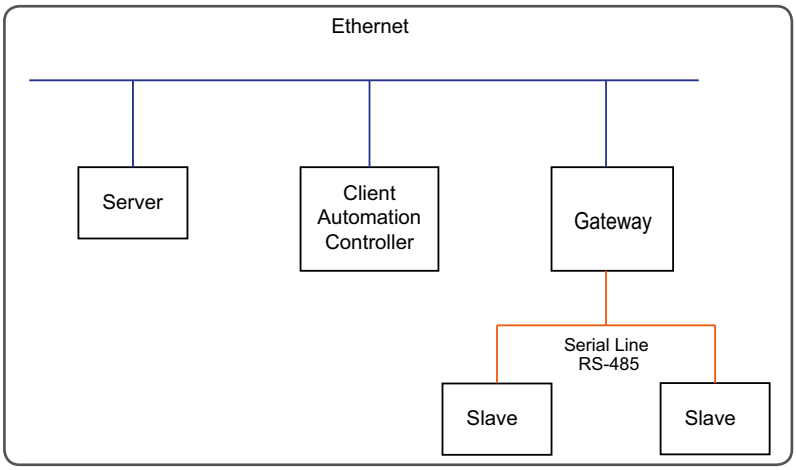
Using the preinstalled profiles, Modbus devices can be easily configured (see *Configure preinstalled profiles* on page 187). The devices are displayed in a list, with links to Modbus registers with objects of the C-Bus application *250 - User Parameter.*

Principle of Modbus RTU communication with the Automation Controller:



** By Modbus specification a serial line is limited to 31 slaves.*

Principle of Modbus TCP communication with the Automation Controller:



BACnet

BACnet is designed to allow communication with building automation and control systems for applications such as heating, ventilation, air conditioning control, lighting control, access control, fire detection systems and their associated equipment. The BACnet protocol provides exchange information for building automation devices, regardless of the particular building service they perform.

The Automation Controller supports the BACnet IP format using the RJ45 Ethernet connector.

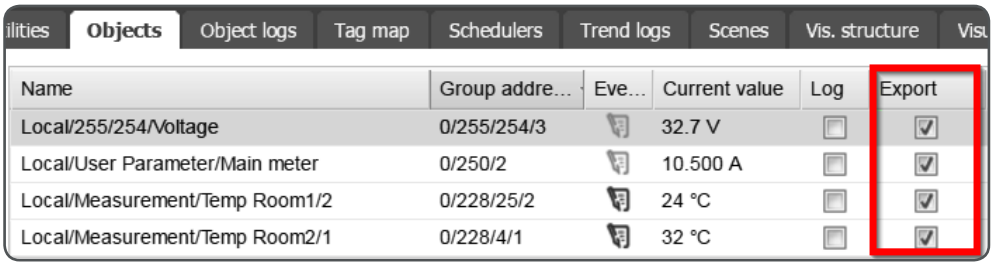
The Automation Controller can act only as a BACnet server (not as a client). The Automation Controller serves data which can be read by BACnet client devices (such as Buildings Management Systems) and BACnet client devices can write data to the server.

The Automation Controller has been certified by BACnet Testing Laboratories (BTL) as a BACnet Application Specific Controller (B – ASC).

The chapter *BACnet* on page 201 describes how to configure the Automation Controller as a BACnet server. This is done by activating the object export function and configuring BACnet communication.

The Network Automation Controller is designed for a maximum of 500 BACnet data points. The Wiser™ for C-Bus® Automation Controller is limited to a maximum of 50 BACnet data points.

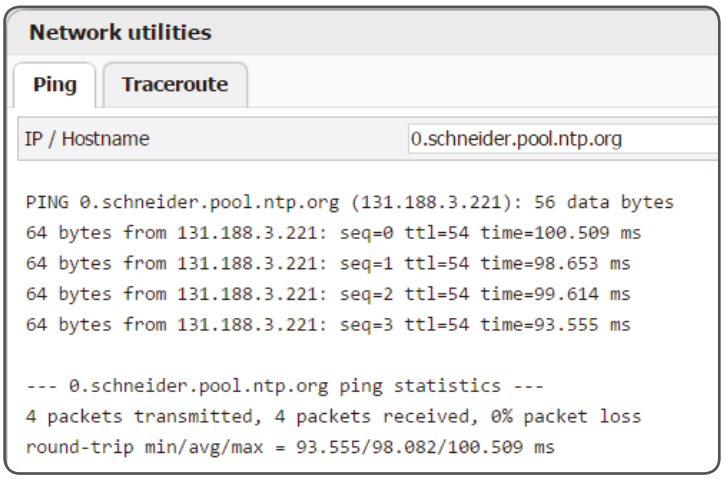
Objects tab with objects available for BACnet selected in the ***Export*** column:



Advanced network functions

The chapter *Advanced network functions* on page 205 describes network settings, utilities and remote functions.

Network utilities (opened in the ***Services*** tab of the ***System*** page):



2.3 Limitations of the Automation Controller

The Network Automation Controller is designed for a maximum of:

- Objects (C-Bus and internal): 2000*
- Users for visualization: 50*
- Modbus devices: 31*
- BACnet data points: 500*.

* Limits not physical but dependent on maximum CPU load.

The Wiser for C-Bus Automation Controller is designed for a maximum of:

- Objects (C-Bus and internal): 2000*
- Users for visualization: 8
- Modbus devices: 6
- BACnet data points: 50.

* Limits not physical but dependent on maximum CPU load.

The bottom bar of the ***Configurator*** page displays information about the processor load, the used memory and the status of C-Bus. For more detailed information about processor load, used memory and partitions, see *Status of the Automation Controller* on page 35.

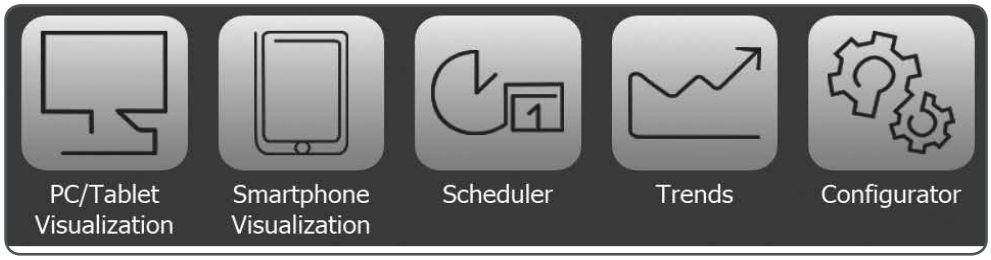
3 Getting started

The Automation Controller is programmed via its embedded web server. Access the web server using a Google Chrome or Firefox web browser. When accessing the web server, the start page is displayed. From the start page, access the configurator and the following user modules:

- PC/Tablet Visualization
- Smartphone Visualization
- Scheduler
- Trends.

As administrator, configure the modules via the configurator and control user access to these four modules (see *User access* on page 156).

The following screen shot shows the buttons of the ***Start page***:



Tabs on the configurator page give access to the configuration options. From the main page of the configurator, allows access to general and visualization configuration as well as the system page. From the system page, access network, update and reset functions, and status information.

3.1 Access to the Controller

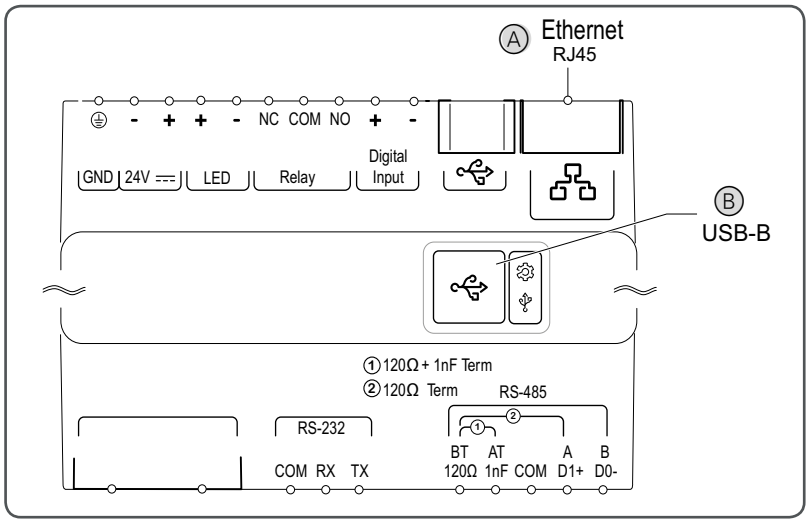
Access the embedded web server of the Automation Controller using a web browser.

Use either Google Chrome or Firefox. No other browser is supported.

Physical connection between the Automation Controller and the PC hosting the browser can be established via either:

Ⓐ Ethernet connection, or

Ⓑ USB Type B connection.



3.1.1 Access via Ethernet and IP address

Preconditions

- The Automation Controller will be supplied with an external 24 V DC regulated power supply (10 W min.) or USB (5 V, 500 mA).
- The default IP address is: 192.168.0.10.

Steps

① Connect an Ethernet cable to the PC.
② Change the IP address of the computer to the same range, e.g. 192.168.0.9; subnet mask 255.255.255.0.
③ Start Google Chrome or Firefox and go to 192.168.0.10.
④ Click the ***Configurator*** button.
⑤ Enter the username. The default username is: `admin`.
⑥ Enter the password. The default password is: `admin`.

3.1.2 Access via USB Type B front connection and IP address

Preconditions

- The Automation Controller be supplied with an external 24 V DC regulated power supply (10 W min.) or USB (5 V, 500 mA).
- The IP address is 192.168.254.10.
- USB drivers are included with the latest C-Bus Toolkit installation. Use the most recent version (1.15 or higher) and install the full package including USB drivers.

Prior to first install, accept installation of USB drivers for the Automation Controller.

Steps

① Connect the USB-B port on the Automation Controller to a USB port on the PC. The PC is given a DHCP IP address in the range of 192.168.254.1 – 192.168.254.9.
② Run Google Chrome or Firefox and go to 192.168.254.10.
③ Click the ***Configurator*** button.
④ Enter the user name. The default user name is: `admin`.
⑤ Enter the password. The default password is: `admin`.

The USB connection to the network adaptor is independent of the Ethernet connection and so both can be used concurrently.

3.1.3 Access via hostname

Access via *hostname* is possible when a network router is present and the Automation Controller is properly configured. See *Change IP settings of the Ethernet interface* on page 29.

The default hostname is the product part number:

- LSS5500NAC for the Network Automation Controller
- LSS5500SHAC for the Wiser™ for C-Bus® Automation Controller.

Access example: http://LSS5500NAC.local

The hostname is used for identification of Automation Controller in installation as well as in files names (e.g. backup).

Change the hostname on the *System* page.

Path: ***Configurator*** → ***Utility*** tab → ***System*** button → ***System*** tab → ***Hostname***.

3.1.4 Change password

Change the default password on the *System* page.

Path: ***Configurator*** → ***Utility*** tab → ***System*** button → ***System*** tab → ***Admin Access***

3.1.5 Direct access to a module

Access a module directly from the browser by using the module's address.

Table 1: *Module addresses*

Page/Module	Address (with default IP address)
PC/Tablet Visualization	http://192.168.0.10/scada-vis
Smartphone/Visualization	http://192.168.0.10/scada-vis/touch
Scheduler	http://192.168.0.10/scada-vis/schedulers
Trends	http://192.168.0.10/scada-vis/trends
Start page	http://192.168.0.10/home
Configurator	http://192.168.0.10/scada-main

When accessing a module in this way, provide the username and password.

Settings for users are described in the chapter *User access* on page 156.

3.1.6 Change IP settings of the Ethernet interface

Change the IP settings on the *System* page.

Path: ***Configurator*** → ***Utility*** tab → ***System*** button → ***Network*** tab → ***Interfaces*** → ***IP address.***

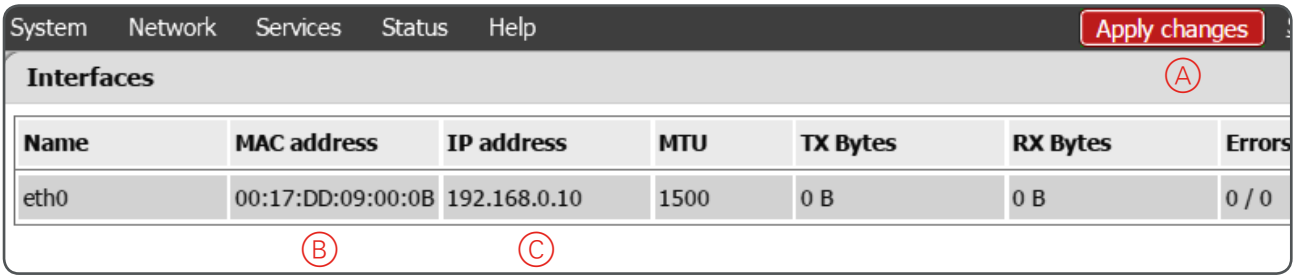
Click the IP address to open the window *Interface eth0.*

Table 2: *IP settings*

Protocol	Static IP Static IP address (default is 192.168.0.10). DHCP DHCP protocol used to get IP configuration, e.g. from a router or gateway with DHCP server.
IP address	Enter a static IP address.
Network mask	Network mask (default is 255.255.255.0).
Gateway IP	IP address of the router or gateway.
DNS server 1	Primary DNS server IP address (resolution of address names). In general set the IP address of the network router.
DNS server 2	Secondary DNS server IP address. Visit public-dns.info for a list of public DNS servers.
MTU	**Maximum transmission unit:** The largest size of the packet which can be passed in the communication protocol (default is 1500).

- Add new IP settings to personal documentation before applying the changes. When selecting the DHCP, note the MAC address of the network interface. This helps to identify the IP address set by the DHCP server. It is recommended to test the access via hostname (on page 28) before changing the IP settings.
- With USB-B, it provides independent local access to the Automation Controller. The linked IP address can not be changed from its default (192.168.254.10).

- When changes are made, the ***Apply changes*** button appears in the top-right corner of the window (see Ⓐ in the next figure). Click this button to apply the changes—the Automation Controller will automatically reboot. Once the reboot process is complete, use the new IP address to access the Automation Controller.



Ⓐ ***Apply changes*** button
Ⓑ MAC address
Ⓒ IP address.

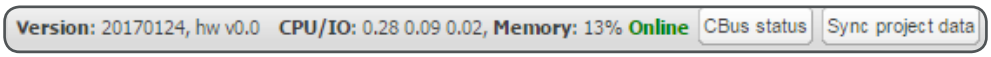
3.2 Reset and save data

3.2.1 Save data during configuration

To save data during configuration:

Path: ***Configurator*** → ***Sync project data*** button.

Click the ***Sync project data*** button on the bottom bar of the configurator:



The project will be immediately synchronised to the microSD card. If power is removed from the Automation Controller without pressing this button, any configuration change may be lost.

3.2.2 Backup and Restore

All objects, trends, logs, scripts, icons, images, backgrounds and visualizations are backed up.

Path: ***Configurator*** → ***Utility*** tab.

Backup regularly and before the following functions are executed:

- Factory reset
- Clean-up selected functions
- Installation of updates
- Firmware update.

Default file name

`Project-Hostname_ yyyy_mm_dd.-hh.mm.tar.gz`

The file name includes the device time and date when the backup is made.

The file name can be changed as desired (*.tar.gz).

Backup

Click the ***Backup*** button.

The following settings are not backed up:

- System configuration
- Network settings
- Passwords
- C-Bus settings.

Ensure that the **backup size is no greater than 32 MB** as this is the maximum restore size.

Restore

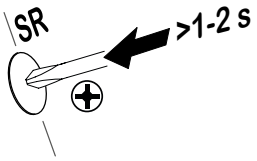
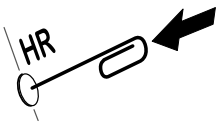
① Click the ***Restore*** button.
② Select the file and save.

- Do not switch off the Automation Controller during the restore procedure. The Automation Controller will automatically reboot.
- Clean the browser cache after a restore. Use the settings of the browser or the short cuts [Crtl] + [N] or [Crtl] + [F5].
- **Backup files >32 MB cannot be restored**.

3.2.3 Reboot or hardware reset

Perform a reboot or hardware reset via either the configurator or buttons on the Automation Controller.

Table 3: *Reboot options*

Function	Description	Access
Reboot	Forces running processes to stop and then reboots the Automation Controller.	***Configurator*** → ***Utility*** tab → ***System*** button → ***System*** tab → ***Reboot***
	Forces running processes to stop and then reboots the Automation Controller.	
Shutdown	System shuts down and data is saved. To run the system, power must be switched off and back on again or switching on via Hardware Reset is possible.	***Configurator*** → ***Utility*** tab → ***System*** button → ***System*** tab → ***Shutdown***
Hardware Reset	Power switches off and back again. Data is **not** saved. Use to restart an Automation Controller that has shut down.	

- Use the *Shutdown* function to power off the Automation Controller. The database is saved.
- Use the *Sync project data* function **before** performing a Hardware Reset to avoid the loss of data (the database is not saved during a Hardware Reset). See *Reset/Clean-up* on page 32.
- If the Automation Controller has locked up and does not respond to the Software Reset Button, the Hardware Reset can be used. Unsaved data will be lost.

3.2.4 Reset/Clean-up

It is possible to delete the following items:

- Objects
- Object logs
- Object and high priority logs
- Alerts
- Logs
- Error logs
- Script storage.

If *Objects* is selected, they will be deleted from the visualization part as well.

Perform a back up **before** using this function. See *Backup* on page 31.

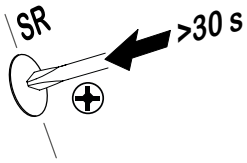
Path: ***Configurator*** → ***Utility*** tab → ***Reset/clean-up*** button.

3.2.5 Factory reset

Perform a factory reset via either the configurator or the software reset button on the Automation Controller.

- Perform a back up **before** using the *Factory reset* function.
- IP settings and security credentials are set to defaults:
  - IP: 192.168.0.10
  - User name: `admin`
  - Password: `admin`

Table 4: *Factory reset options*

Function	Description	Access
Factory Reset	Deletes all configurations and resets to the factory default settings.	***Configurator*** → ***Utility*** tab → ***Factory reset*** button
Factory Reset (Software Reset button)	Deletes all configurations and resets to the factory default settings.	

3.3 Set date and time

3.3.1 NTP client/server

By default, the Automation Controller gets its date and time from remote network time protocol (NTP) servers. On the system page, configure the NTP client for the Automation Controller.

Path: ***Configurator*** → ***Utility*** tab → ***System*** button → ***Services*** tab → ***NTP client/server***.

The following remote servers are selected by default:

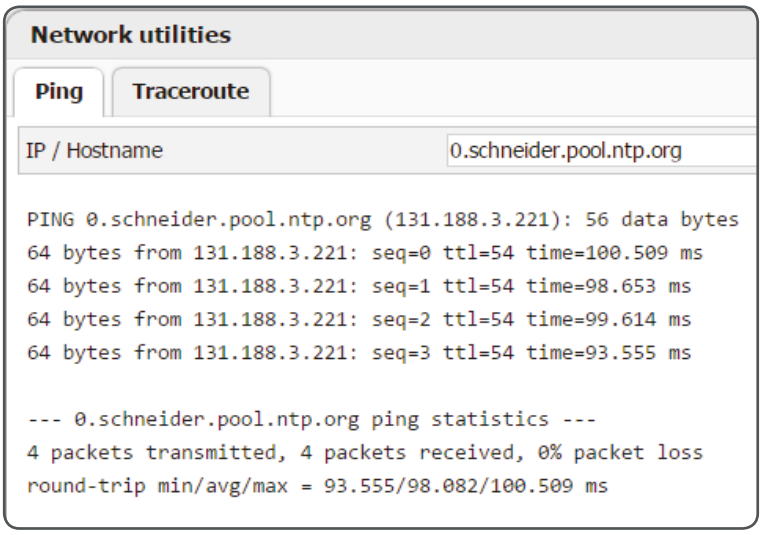
- Server 1: 0.schneider.pool.ntp.org
- Server 2: 1.schneider.pool.ntp.org
- Server 3: 2.schneider.pool.ntp.org
- Server 4: 3.schneider.pool.ntp.org

Enable/disable the default servers and edit other servers.

Enable the sending of the local server status so that the Automation Controller can act as a local time server for other Automation Controllers.

To use this function, enter the IP address of the Automation Controller in the *NTP server/client* settings of other Automation Controllers.

- After enabling the local time server function on the Automation Controller, perform a reboot. See *Reboot or hardware reset* on page 31.
- Check if the time server can be accessed. Ping the NTP server via the ***Network utilities***. (***Configurator*** → ***Utility*** tab → ***System*** button → ***Status*** tab).



3.3.2 Date and time

Path: ***Configurator*** → ***Utility*** tab → ***Date and time*** button.

Edit the following parameters. (See the next screen shot).

Get from system

Synchronise the Automation Controller to the date and time zone from the connected PC.

Time zone

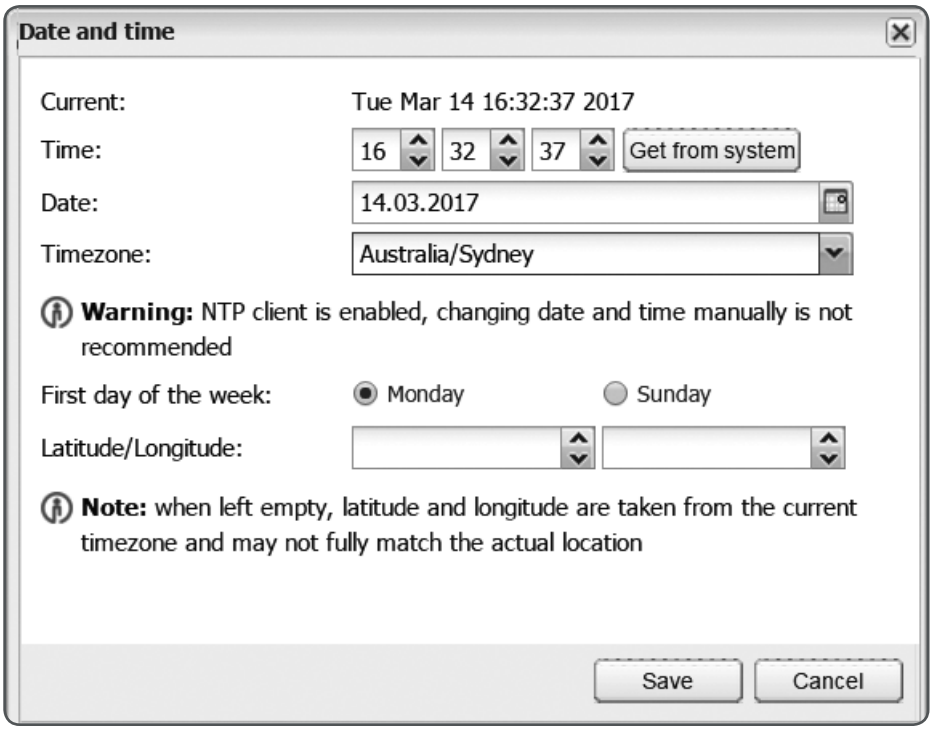
Select the correct time zone. Even if the NTP client is enabled, check that the time zone is correct.

First day of week

This option changes the calender views, e.g. calendars displayed in schedulers.

Latitude and Longitude

Sunrise and sunset are calculated from the selected time zone. Specify exact sunrise and sunset times for your location by entering the location latitude and longitude. Sunrise and sunset can be used for time functions.



3.4 Updates and Firmware upgrade

3.4.1 Updates

Updates provide a way to install improvements and new features to the Automation Controller. Updates may require a particular firmware version to be installed. See *Upgrade firmware* on page 34.

Path: ***Configurator*** → ***Utility*** tab → ***Install updates*** button.

Update files have the file extension *.lmup (LMUP).

Click the ***Install updates*** button and select the file. The Automation Controller reboots after a successful installation.

- **Do not switch off** the Automation Controller during the installation.
- Clean the browser cache after the installation. Use the settings in your browser or the short cuts [Crtl] + [R] or [Crtl] + [F5].

3.4.2 Upgrade firmware

It is recommended to update the firmware to install the latest features, security updates and bug fixes. Scan the QR code on your Automation Controller using the Facility hero application to get information specific to your device. Download the Facility hero application from Google Play or from iTunes.

Firmware version

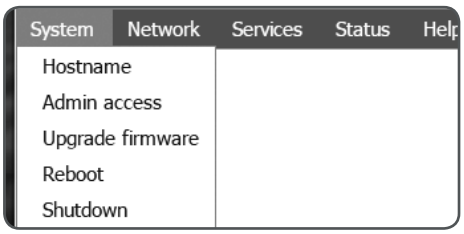
Check the firmware version installed in the Automation Controller before upgrading.

The firmware version is displayed in the left corner of the bottom bar of the ***Configurator*** page (e.g. v1.0).

Installation

Install a firmware upgrade via the ***System*** page.

Path: ***Configurator*** → ***Utility*** tab → ***System*** button → ***System*** tab → ***Upgrade firmware***.



The file name is: `*.img`

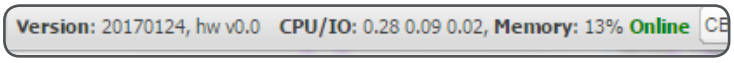
Click ***Upgrade Firmware*** and select the file.

A firmware upgrade can take up to several minutes. During the upgrade, the device will not respond but will reboot several times.

- **Do not switch off** the Automation Controller during the installation.
- Clean the browser cache after the installation. Use the settings in your browser or the short cuts [Crtl] + [N] or [Crtl] + [F5].

3.5 Status of the Automation Controller

Information about the processor load, the used memory and the status of C-Bus is displayed on the bottom bar of the ***Configurator*** page:



Information about the processor load, memory, partitions and System log is shown on the ***System*** page.

Path: ***Configurator*** → ***Utility*** tab → ***System*** button → ***Status*** tab → ***System status.***

System information is shown in ***Error logs*** and ***Alerts*** tab of the ***Configurator***.

3.5.1 Processor load

The power LED on the Automation Controller blinks at a rate proportional to the processor load (Green, blinking red).

The processor load is also displayed on the bottom bar of the ***Configurator*** page, under the heading CPU/IO. The processor load numbers (examples 0.28, 0.09, 0.02 shown above) represent averages over progressively longer periods of time: 1 min, 5 min and 15 min. Lower numbers indicate better performance.

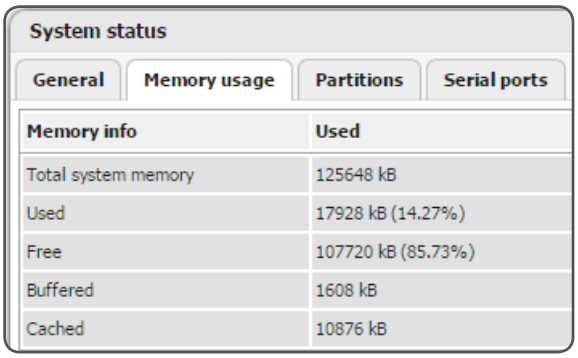
If the load exceeds 0.70 for the 5 min or 15 min average, check the possible reasons, which may include:

- Number of active users accessing the visualization
- Number and resolution of images used for visualization
- Number of objects in the Objects list
- Frequency of updating objects:
  - Delta for sending a new value
  - Delta for change on the client side (BACnet COV setting)
  - Cycle time for sending
  - Polling cycle of the master/client – slave/server communication (Modbus).
- Number of active scripts:
  - Sleep time interval (resident scripts with sleep interval 0 have high impact on CPU load)
  - Using resident scripts instead of scheduled scripts.
- Logging policy (e.g. log all new objects). Excessive object logging degrades performance.

3.5.2 Memory

The used system memory is displayed on the bottom bar of the ***Configurator*** page (see the previous screen shot).

View detailed memory usage on the ***System*** page (***Memory usage*** tab).



3.5.3 Partitions

View the values of total, used and free space for storage partitions on the ***System*** page ***Partitions*** tab. For example, check the free space for data on the MicroSD card (see: /data in the partition table).

3.5.4 Bus Status

The status of C-Bus is displayed in the bottom bar of the ***Configurator*** page:

C-Bus powered	**Online**
No C-Bus power	**Offline**

When C-Bus is connected and a clock is running, the status is *Online.* Otherwise, *Offline* is displayed. See *Sniffer function* on page 107.

C-Bus must be physically connected via one of the C-Bus RJ45 connectors.

A *C-Bus status* button is located on the bottom bar of the ***Configurator*** page. Click this button to get detailed information about C-Bus status.

C-Bus status is also displayed by the C-Bus LED on the Automation Controller. The C-Bus LED indicates the following states:

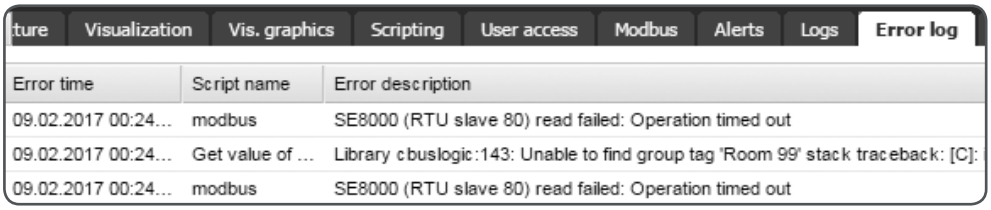
C-Bus powered and clock running	**Green**
C-Bus voltage warning	**Green flashing**
No C-Bus power	Off

Test C-Bus online and offline status on the ***Objects*** tab. See *Edit and test objects* on page 106.

3.5.5 Error logs

Error logs are displayed in the ***Error log*** tab of the ***Configurator*** page. Errors messages come from the system. Errors in the configuration or in a running script can be detected.

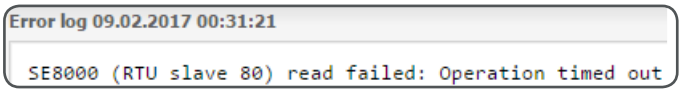
Error log



Example Modbus slave configuration

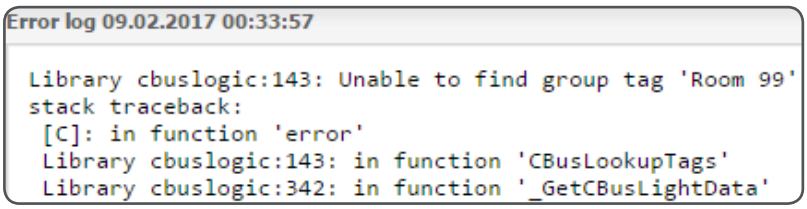
In this example, a Modbus slave/server is configured but does not respond, e.g. the device is not connected to the Modbus line.

Click the line to get the message:



Example script

In this example, a lighting channel (local network/Lights 1/room 99) does not respond, e.g. the address *Room 99* does not exist in ***Objects*** list or ***Tag map***.

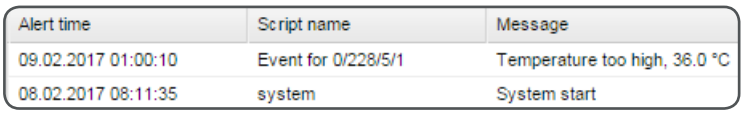


3.5.6 Alerts

System messages and alert information for user-edited scripts are displayed in the ***Alerts*** tab of the ***Configurator*** page. See *Alerts* on page 171.

Example

In this example, system start alerts and a user-edited temperature alert are shown.



4 C-Bus objects

4.1 Overview

The objects in the ***Objects*** list are the basis of all communication via the Automation Controller. These objects represent different C-Bus applications for sending and receiving values. In the case of the *Lighting application*, the *Target level* and the *Ramp rate* are sent. The objects can also be used for visualization and scripting.

With the special application *250 - User Parameter,* values can be used for visualization or scripting. Note that user parameters are not exported to a C-Bus Toolkit project.

The application *User Parameter* must be used for the connection to Modbus register values (see *Modbus Mapping* on page 190). By default, the Automation Controller works as a Modbus client/master.

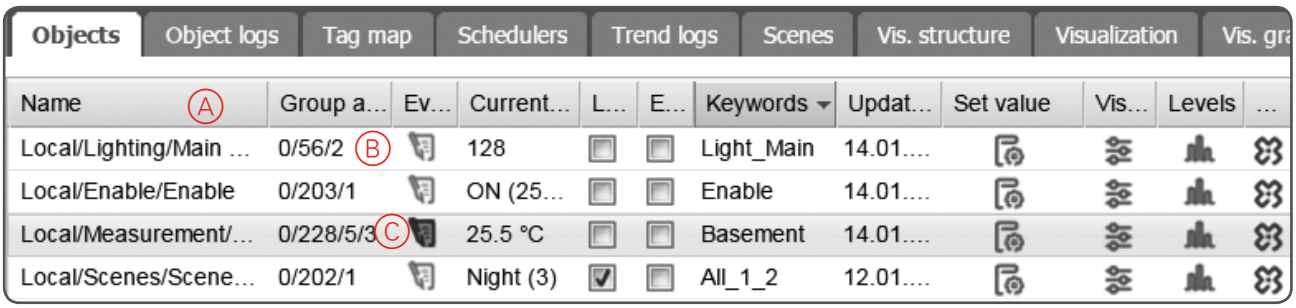
Objects can be exported for communication with BACnet (see *Object export* on page 202). The Automation Controller can share its BACnet information with a BACnet client (such as a BACnet IP-BMS client).

C-Bus objects can be imported, manually added or automatically detected and displayed.

In addition to the ***Objects*** list, a ***Tag map*** list is available. When importing or adding new objects, the corresponding entry is available in the tag map. In the tag map, change all names (tags) and add new applications, groups and levels. See *Tag map* on page 110.

4.1.1 Properties of C-Bus objects

All added or imported objects are visible in the ***Objects*** list. If activated, new objects are automatically displayed.



All objects in the list have a unique composed address with 3 or 4 entries. These addresses are visible in the *Group address* column. All objects in this column are addressed as (network address)/(application identifier)/ + object information. The network address for the local network of the Automation Controller is 0.

For the following examples, refer to the previous screen shot.

Example: Lighting (0/56/2) Ⓑ

Local network	Lighting application (default)	Group address
0	56*	2

* 56 = default lighting application, 48-95 reserved for lighting applications (must be decimal format)

Example: Measurement (0/228/5/3) Ⓒ

Local Network	Measurement application	Device ID	Channel number
0	228	5	3

Example for Tags Ⓐ

The names added to the composed address are called *tags*. The tagged names for example Ⓑ are: Local/Lighting/Main office = (0/56/2).

Edit objects

Objects in the list can be edited by clicking the row. In this mode, undertake editing of the following functions and information:

- Keywords
- Log
- High priority log
- Export
- Object comment.

These functions are described in *Additional parameters for all applications* on page 105.

The network, application and group names are visible in the ***Name*** column of the object list.

The names can not be changed in the ***Objects*** list. To edit names, use the tag map. See *Tag map* on page 110.

4.1.2 C-Bus settings

Select the default network, application and device description in the C-Bus settings.

Path: ***Configurator*** → ***Utilities*** tab → ***C-Bus settings*** tab.

The following table shows the default C-Bus settings.

Table 5: *Default C-Bus settings*

Local network	0 - Local The C-Bus Toolkit auto-assigns networks in descending order, starting at 254 (254, 253, ...). Applications imported from the Toolkit are added to the local network (0) of the Automation Controller.
Default lighting application	56 - Lighting
Device description	**Defaults** • LSS5500NAC (Network Automation Controller) • LSS5500SHAC (Wiser for C-Bus Automation Controller) The device description is used by the Control Systems IP Utility software that comes with Toolkit to help identify different devices on the C-Bus network. It may be customised to your requirements in this dialog, or from the IP Utility.
Enable CNI functionality	If enabled, the Automation Controller can act as a C-Bus Network interface for commissioning and maintenance. In C-Bus Toolkit, the Automation Controller can be used as an interface to C-Bus networks. In Toolkit configure connection details for each C-Bus network: • Type: CNI • Address: IP address The default IP address of the Automation Controller is 192.168.0.10 (or 192.168.254.10 if connected using the USB-B interface) • Port: 10001 The default port of the Automation Controller CNI is 10001
CNI port	10001 (default)

4.1.3 Automation Controller as network interface

In C-Bus Toolkit, the Automation Controller can be used as an interface to C-Bus networks. See *C-Bus settings* on page 39. C-bus networks configured with the C-Bus Toolkit software can be commissioned and maintained in online mode via the configured Ethernet connection of the Automation Controller.

4.1.4 Three ways to fill the objects list

Import

A project created with the ***Toolkit*** software can be imported as CGL file (***Utilities*** tab → ***Import Toolkit CGL file*** button).

Sniffer function

The bus sniffer detects objects from the C-Bus network and automatically adds them to the list.

Objects are visible when C-Bus is connected and the *discover* function is activated (***Utilities*** → ***General Configuration*** tabs).

Add new objects

New objects can be added manually (***Objects*** tab → ***Add new object*** button).

Use any of the three ways to find the objects for visualization and exchange with other systems, however the most typical way is to prepare a Toolkit project and import it:

① Export the project in the Toolkit software.
② Import the CGL file into the Automation Controller.
③ Edit the objects to be used.
④ Add new objects manually or via the bus sniffer function.

4.2 Export from Toolkit and import into the Automation Controller

4.2.1 CGL export in Toolkit

Export any network in one of the projects configured in the C-Bus Toolkit. When importing to the Automation Controller, all applications are added to the local network (0) of the Automation Controller.

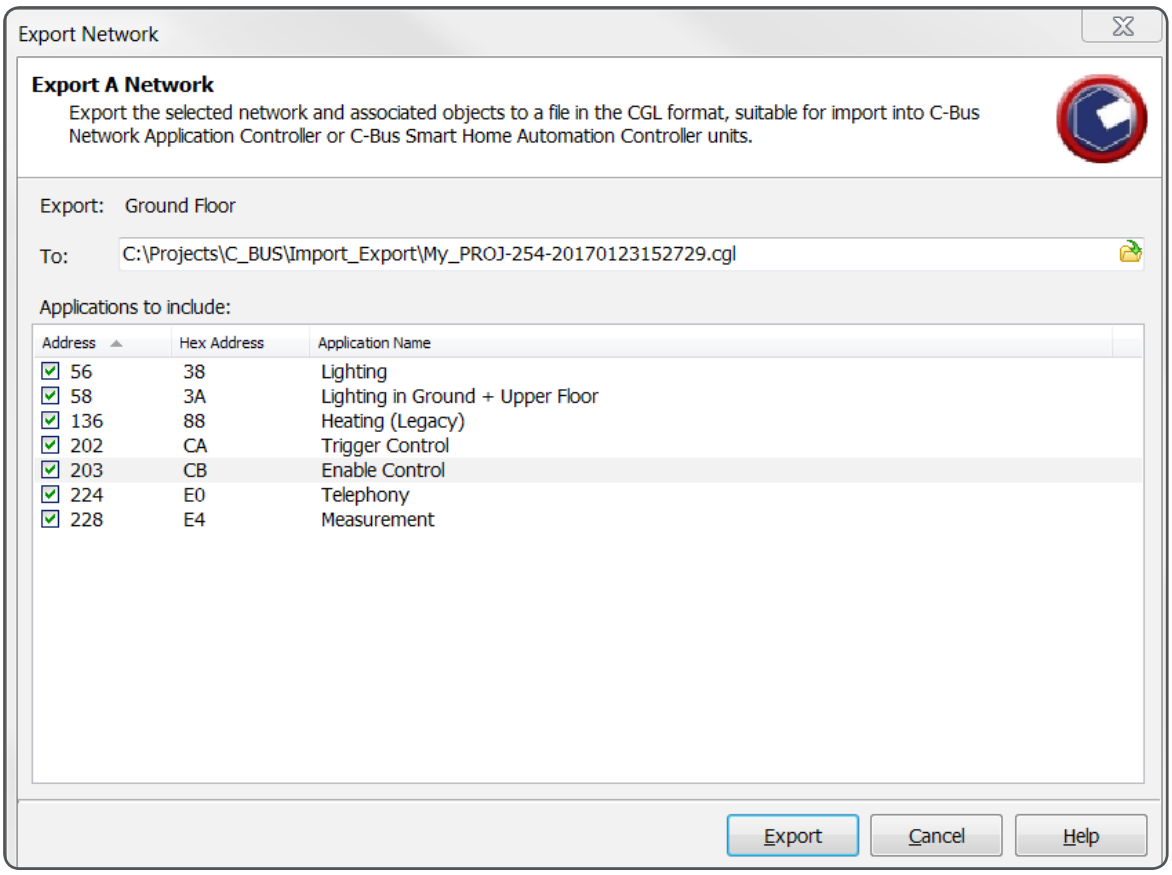
Install the complete C-Bus Toolkit, including C-Gate software and any USB drivers that Toolkit asks to install. The C-Gate software is required to export from the Toolkit and import to the Automation Controller. C-Gate can also be used as part of a C-Bus control system.

Steps

① Open Toolkit.
② Select the Toolkit project and a network. 254 is the default number of the local network in the Toolkit. 0 is the number of the local network in the Automation Controller (*C-Bus settings* on page 39).
③ Click the ***Export CGL*** button of the main menu of Toolkit.
④ Select the applications to export. As a default, all application addresses are selected (checked).
⑤ Select the folder, where the file is saved (see icon on the left side of the path).

⑥ Click the ***Export*** button. The message *The export was successful* appears.
⑦ The CGL file is now in the selected folder:
(e.g. `Import_Export\My_PROJ-254-20170123150827.cgl`).

When exporting, the *Export Network* window is displayed:



4.2.2 Import Toolkit CGL file into the Automation Controller

Import a *network* with *applications* using the ***Utilities*** tab.

Path: ***Configurator*** → ***Utilities*** tab → ***Import Toolkit CGL file*** button.

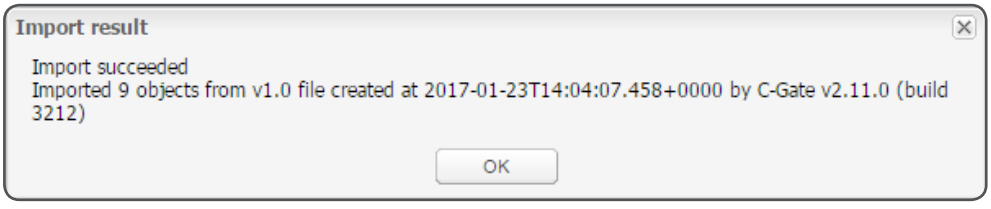
Once the Toolkit CGL file has been imported, the applications are listed in the ***Objects*** tab and are available in the ***Tag map*** tab.

Steps

In the ***Configurator*** page of the Automation Controller:

① Click the ***Import Toolkit CGL file*** button.
② Select the file (*.cgl).
③ Click the ***Save*** button.

An *Import result* message is displayed after import.



Existing object names of the same application and group address are overwritten with the imported values. To change the names (tags), open the *Tag map* list. See *Tag map* on page 110.

Update object list

To use additional objects, import an updated C-Bus Toolkit file.

Alternatively, manually add new objects in the ***objects*** list. See *Add new objects* on page 42.

When the sniffer function is activated and C-Bus is online, new objects are automatically added to the ***objects*** list. See *Sniffer function* on page 107.

4.3 Add new objects

Manually add new objects to the ***Objects*** list. Once saved, the new composed addresses are visible in the ***Name*** and the ***Group Address*** columns of the ***Objects*** list.

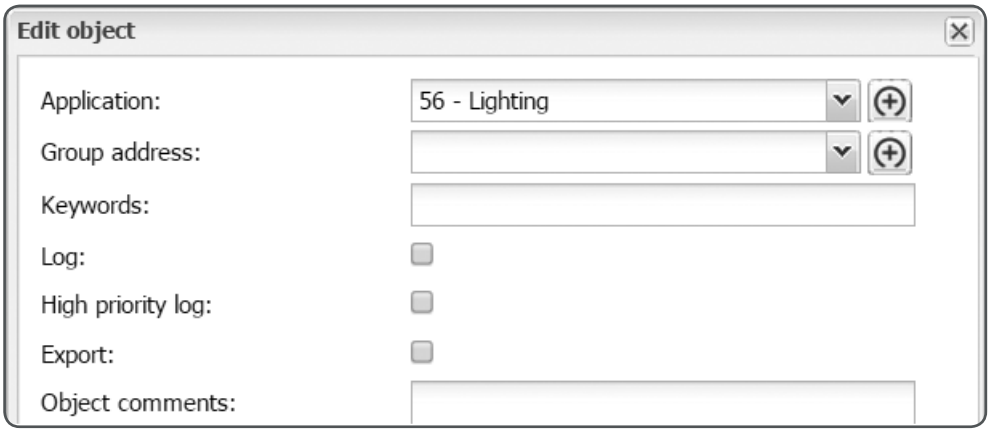
The following applications are predefined.

Table 6: *Predefined addresses*

Number	Application
56 (48-127)	Lighting
202	Trigger control
203	Enable
228	Measurement
250	User Parameter
255	Unit Parameter

Each application has specific parameters that must be set when editing the address elements of a new object. Some options are common for all applications.

Click on the ***Add new object*** button at the bottom of the ***Objects*** tab to open an *Edit object* window.



The window is displayed with empty fields and the default application *56 - Lighting* selected. Click the ***Application*** drop-down list to select from the currently defined applications.

Click on the Application ***Add*** button to create a new lighting application.

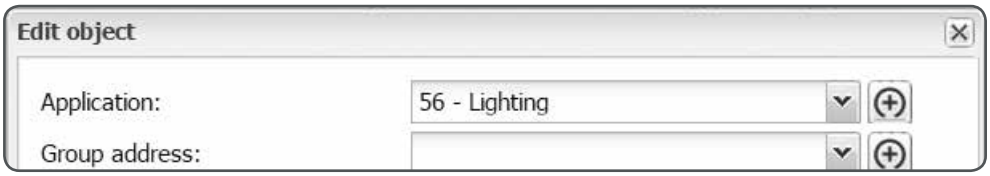
- Change names (tags) linked to the composed addresses in the ***Tag map*** tab.
- Activate/deactivate the *Log*, *High priority log* and the *Export* functions.

- Once a new application is saved, its application address cannot be edited. To change the address, first delete the existing address and then create a new one with the correct number.

4.3.1 Address new lighting applications

The *Lighting* application is used for lighting and lighting-related applications. This includes switching different loads such as fans, curtains and shutters. The numbers 48–127 are reserved for lighting and lighting-related applications.

Click the ***Add new object*** button at the bottom of the ***objects*** tab to open an *Edit object* window:



Add a new application

By default, the lighting application (56) is opened. Click the Application ***Add*** button to create a new lighting application.

ID	For lighting applications, select one of following numbers: 48–127.
Name	Max. 32 characters, except "/". The name is linked with the ID. Change the name in the ***Tag map*** tab.

Add new group address

Click the Group address ***Add*** button to create a new group address.

This group address is linked with the new application.

ID	Select: 0–254.
Name	Max. 32 characters, except "/". The name is linked with the ID. Change the name in the ***Tag map*** tab.

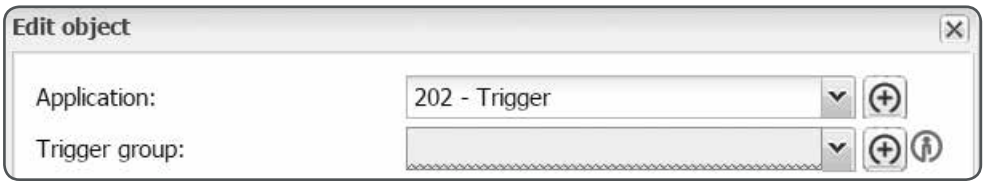
4.3.2 Add new trigger group

The *Trigger* application is similar to the lighting application and is used to trigger scenes. Different scenes are triggered with different values of the action selector. Configure scenes in the ***Scenes*** tab.

Click the ***Add new object*** button at the bottom of the ***objects*** tab to open an *Edit object* window.

Select the trigger application

In the Application list, select: 202 - Trigger.



Add trigger group

Click the Trigger group ***Add*** button to create a new *Trigger group* (0–254).

This Trigger group is linked with the *Trigger* application.

ID	Select: 0–254.
Name	Max. 32 characters, except "/". The name is linked with the ID. Change the name in the ***Tag map*** tab.

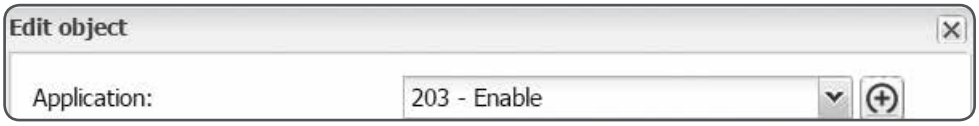
4.3.3 Add new enable group

The *Enable* application is similar to the lighting application and is used to enable/ disable additional functions of a C-Bus device.

Click the ***Add new object*** button at the bottom of the ***objects*** tab to open an *Edit object* window.

Select the Enable application

In the Application list, select: **203 - Enable**.



Add network variable

Click the Network variable ***Add*** button to create a new *Network variable*

(0–254). This network variable is linked with the *Enable* application.

ID	Select: 0–254.
Name	Max 32 characters, except "/". The name is linked with the ID. Change the name in the ***Tag map*** tab.

4.3.4 Add new measurement device and channel

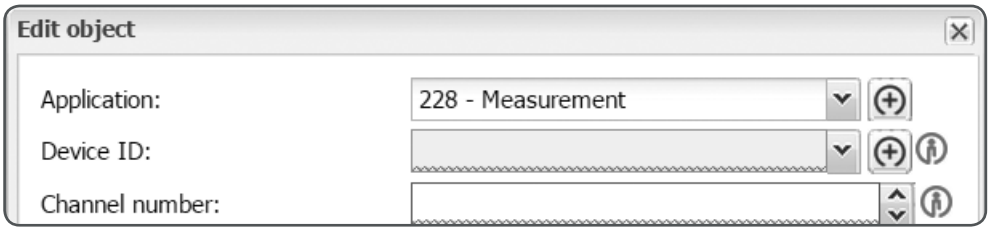
The *Measurement* application is used to send and receive measured values and units (e.g. temperature with °C). The measured value is transferred with a 24 bit floating point number and the unit is coded with 8 bit.

The measurement application is linked with a *Device ID* and a *Channel number*.

Click the ***Add new object*** button at the bottom of the ***objects*** tab to open an *Edit object* window.

Select the measurement application

In the Application list, select: 228 - Measurement.



Device ID	Select: 0–254. The device ID is the *group* in the ***Tag map*** tab. Add a name (tag) in the ***Tag map*** tab.
Channel number	Select: 0–254. The channel number has no tag and so is not visible in the ***Tag map*** tab.

In the ***Group address*** column of the ***Objects*** list, the composed address for a measurement object looks like this example:

`0/228/1/3`

Local Network	Measurement application	Device ID	Channel number
0/	228/	1/	3
Groundfloor	Measurement	Temperature	3

Add a new device ID

Click the Device ID ***Add*** button to add a new *Device ID* (0–254).

ID	Select: 0–254.
Name	Max 32 characters, except "/". The name is linked with the ID. Change the name in the ***Tag map*** tab.

Add a channel number

Select a *channel number* (0–254) from the ***Channel number*** list.

4.3.5 Security application

A C-Bus enabled security system can be added to your Automation Controller through the security application. The security application is used for monitoring and controlling C-Bus enabled security panels.

Typical uses include:

- Remote security keypad emulation
- Arming security systems
- Setting security system mode
- Monitoring alarms
- Monitoring security zones.

Table 7 is an example of a security system installation in a house with 8 monitored zones.

Table 7: *Example of security system zones*

Zone number	Description (Location)
1	Lounge
2	Dining Room
3	Bedroom 1
4	Bedroom 2
5	Bedroom 3
6	Games Room
7	Kitchen
8	Bathroom

Table 8 on page 46 is a list of the read-only objects which report status of the security application. Unless listed, they are all boolean types. Objects with a number of 20 (Zone isolation), and higher are per-zone—note that zones are indexed from 1.

Table 8: *Read-only objects that report status of the security application*

Address	Variable Name	Usage	Value	Settable?
0/208/0	Key press	CBusSecurityKeypress(net, key) Send key on the security application to network net. Key can either be a number from 0-255, a single character or one of the following strings: • Enter • Shift • Panic • Fire • Arm • Away • Night • Day • Vacation.		Yes
0/208/0	Raise Alarm	CBusSecurityRaiseAlarm(net) Returns the value of a user parameter or nil if not found.		Yes
0/208/0	Arm	CBusSecurityArm(net, level) Send an arm system to level message on the security application to net where level is one of the following strings (or nil for highest): • Away • Night • Day • Vacation • Highest.		
0/208/0	Raise Tamper	CBusSecurityRaiseTamper(net) Send a raise tamper message on the security application to network net.		Yes
0/208/0	Drop Tamper	CBusSecurityDropTamper(net) Send a drop alarm message on the security application to network net.		Yes
0/208/1	Alarm sounding	Is the alarm sounding (i.e. emitting noise or raising some other sort of warning)?	True if the alarm is sounding	No
0/208/2	All zones OK	Are all security zones secure (sealed)?	True if all zones are sealed or isolated	No
0/208/3	Arm failed	Did the arm sequence fail?	True if the alarm system failed to arm	No
0/208/4	Arm ready	Is the system ready to arm?	False if any zone failed to arm	No
0/208/5	Armed state	Armed state: 0 = Disarmed 1 = Fully Armed 2 = Partially Armed (Other) (Other values are panel dependent)	Current arm state	No
0/208/6	Battery charging	Is the battery charging?	True if the backup battery is being charged.	No
0/208/7	Entry delay	Is the entry delay in progress?	True if entry delay is running	No

Address	Variable Name	Usage	Value	Settable?
0/208/8	Exit delay	Is the exit delay in progress?	True if exit delay is running	No
0/208/9	Fire alarm	Is the fire alarm sounding?	True if fire is detected	No
0/208/10	Gas alarm	Is the gas alarm sounding?	True if gas is detected	No
0/208/11	Line cut alarm	Is the line cut alarm sounding?	True if phone line cut detected	No
0/208/12	Low battery	Is the battery charge low?	True if backup battery is sub 1 hour	No
0/208/13	Mains failure	Is there a mains power failure?	True if mains power not detected.	No
0/208/14	Normal operation	Is the operation normal (no alarms or warnings)	True if no alarms detected.	No
0/208/15	Other alarm	Is there another alarm sounding?	True if a special alarm condition is detected.	No
0/208/16	Panic	Is the system in the panic state?	True of the system has emitted a panic message.	No
0/208/17	Password status	Status of the password entry: 1 = password succeeded 2 = password failed 3 = password disabled 4 = password enabled		No
0/208/18	Password OK	Was the last password correct?	True if last password entry successful.	No
0/208/19	Tamper	Is the system in the tamper state?	True if system detects a tamper condition	No
0/208/20	Zone isolated	Is a security zone isolated (disabled)?	True if system has isolated this zone	No
0/208/21	Security Zone Status	Security zone status: 0 = Sealed 1 = Unsealed 2 = Open 3 = Short		No
0/208/22	Security Zone Name	Name of a security zone.		No

When adding a security object for the first time, all the associated tags are created for the security application. If a security object is deleted, so are the associated tags. Adding the object back will also re-create the tag/s.

The Automation Controller maintains state internally of the security application based on messages it receives from a security panel operating on the network. All except the exposed security objects are read-only and reflect direct or derived state reported by the panel. The only control the Automation Controller has of the security application is via the parameter ***Command*** *(0/208/0)* which provides a way for visualisation and scripting to send the following messages and arguments:

- Simulate keypress (argument: keypress)
- Raise the alarm (no argument)
- Arm the system (argument: level to arm to)
- Raise tamper (no argument)
- Drop tamper (no argument)
- Display message on panel (argument: message to be displayed).

Due to the nature of operation, messages will not necessarily have an immediate effect on the state of the system. For example, if a script or UI event causes an Arm system message to be sent the value of Armed state will not change until the panel processes the message and send a status message back.

Setting up Security

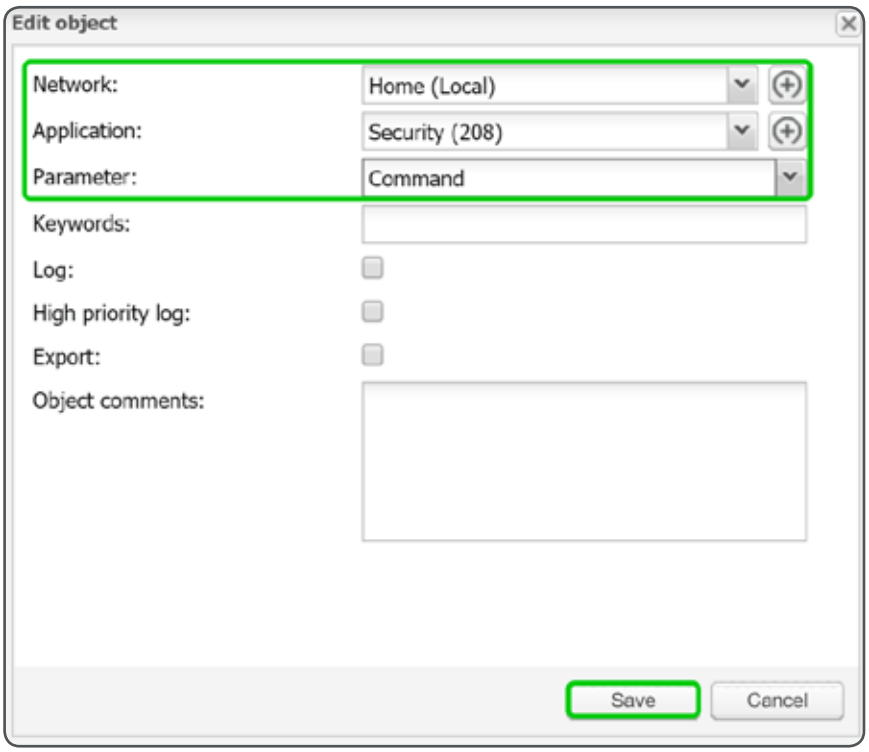
This is an example project which describes how to add the security objects to the Automation Controller.

To create a Security related object

Path: ***Configurator*** → ***Objects*** tab → ***Add new object*** button → ***System*** tab → ***Admin Access***

In the ***Edit object*** window, select:

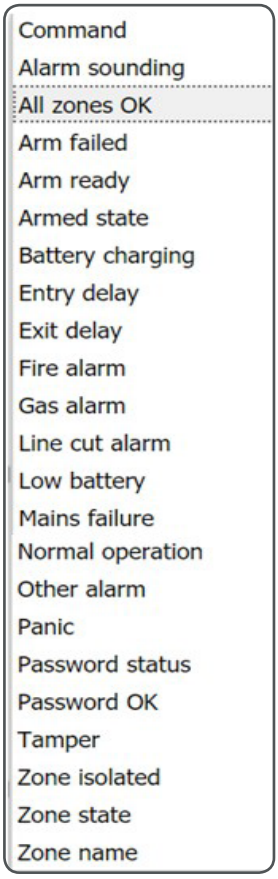
- Network = ***Home (Local)***
- Application = ***208 - Security***
- Parameter = ***Command***.



Click ***Save***. The parameter ***Command*** is created for the security application.

Creating the Security related objects

Refer to *How to create a Security related object* on page 48 to add each of the following parameters:

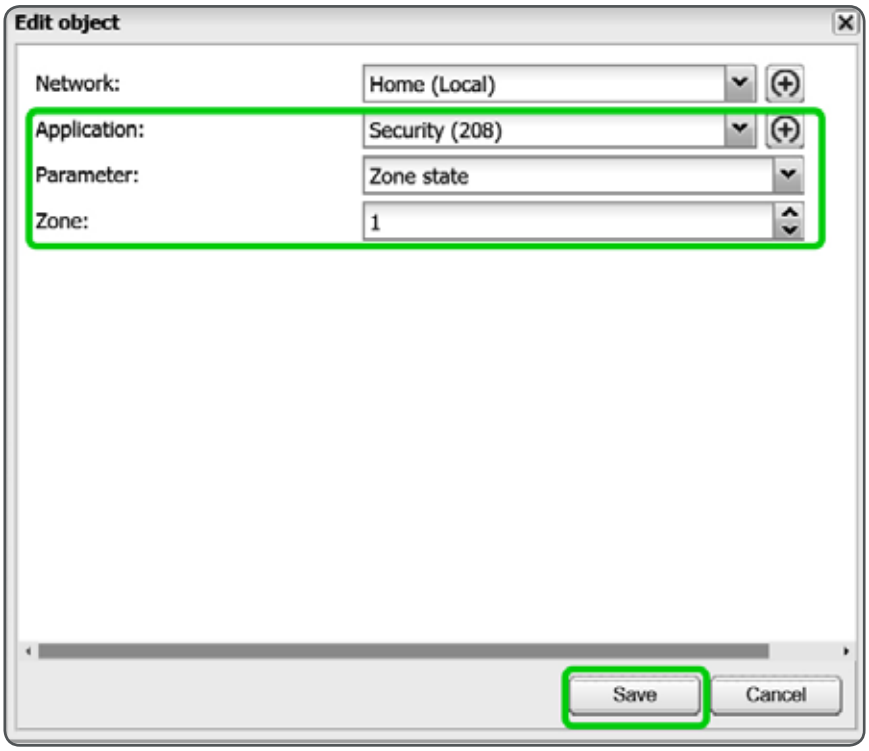


Adding the security zones

Refer to *How to create a Security related object* on page 48 to add Zones 1 to 8, one at a time.

In the ***Edit object*** window, select:

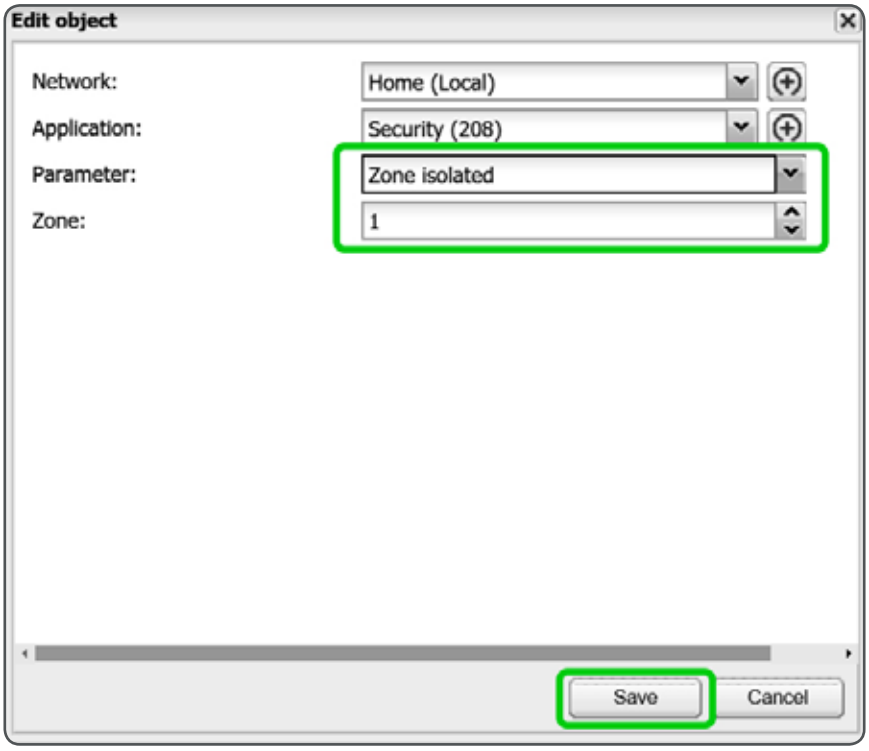
- Network = ***Home (Local)***
- Application = ***208 - Security***
- Parameter = ***Zone state***
- Zone = ***1***, then ***Save***.



Repeat to add Zones 2 to 8.

Adding Zone isolated objects

Select *Zone Isolated* in the ***Parameter*** field, then select ***Zone*** = *1* and click ***Save***.



Repeat for Zones 2 to 8.

The added parameters will show in the ***Objects*** tab, identifying the name and group address.

Name	Group address ▲
Home/Security/Command	0/208/0/0
Home/Security/Alarm sounding	0/208/1/0
Home/Security/All zones OK	0/208/2/0
Home/Security/Armed state	0/208/5/0
Home/Security/Fire alarm	0/208/9/0
Home/Security/Normal operation	0/208/14/0
Home/Security/Zone isolated/1	0/208/20/1
Home/Security/Zone isolated/2	0/208/20/2
Home/Security/Zone isolated/3	0/208/20/3
Home/Security/Zone isolated/4	0/208/20/4
Home/Security/Zone isolated/5	0/208/20/5
Home/Security/Zone isolated/6	0/208/20/6
Home/Security/Zone isolated/7	0/208/20/7
Home/Security/Zone isolated/8	0/208/20/8
Home/Security/Zone state/1	0/208/21/1
Home/Security/Zone state/2	0/208/21/2
Home/Security/Zone state/3	0/208/21/3
Home/Security/Zone state/4	0/208/21/4
Home/Security/Zone state/5	0/208/21/5
Home/Security/Zone state/6	0/208/21/6
Home/Security/Zone state/7	0/208/21/7
Home/Security/Zone state/8	0/208/21/8

Preparing security icons

Path: ***Vis.Graphics*** tab → ***Add icons*** button.

Select ***Choose File*** to add the icons file, then click ***Save***.

Prepare icons for each possible value for each Status and Command:

① Click ***Add new Graphic***.

② Assign the ***Status Name*** and ***Possible Value*** to each icon.

Table 9: *Status names and possible values*

Status Name	Possible Values
Normal operation	True/False
Alarm sounding	True/False
All zones OK	True/False
Fire alarm	True/False
Armed state	Disarmed/Fully armed/Partially armed
Zone 1 isolated	True/False
Zone 2 isolated	True/False
Zone 3 isolated	True/False
Zone 4 isolated	True/False
Zone 5 isolated	True/False
Zone 6 isolated	True/False

Status Name	Possible Values
Zone 7 isolated	True/False
Zone 8 isolated	True/False
Zone 1 state	Sealed/Unsealed/Open/Short
Zone 2 state	Sealed/Unsealed/Open/Short
Zone 3 state	Sealed/Unsealed/Open/Short
Zone 4 state	Sealed/Unsealed/Open/Short
Zone 5 state	Sealed/Unsealed/Open/Short
Zone 6 state	Sealed/Unsealed/Open/Short
Zone 7 state	Sealed/Unsealed/Open/Short
Zone 8 state	Sealed/Unsealed/Open/Short

Command description, Control Type and Values are shown in *Table 10*.

Home/Security/Command (0/208/0/0).

Table 10: *Command descriptions, control types and values*

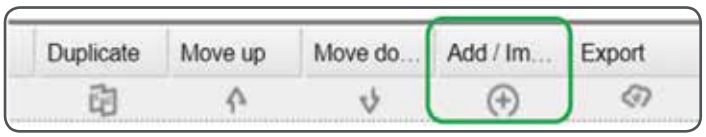
Command description	Control type	Value
Arm in day mode	Arm	Day
Arm in away mode	Arm	Away
Arm in vacation mode	Arm	Vacation
Keypad 0	Keypress	0
Keypad 1	Keypress	1
Keypad 2	Keypress	2
Keypad 3	Keypress	3
Keypad 4	Keypress	4
Keypad 5	Keypress	5
Keypad 6	Keypress	6
Keypad 7	Keypress	7
Keypad 8	Keypress	8
Keypad 9	Keypress	9
Keypad *	Keypress	*
Keypad Enter	Keypress	Enter
Keypad Arm	Keypress	Arm

Repeat to add all new icons required.

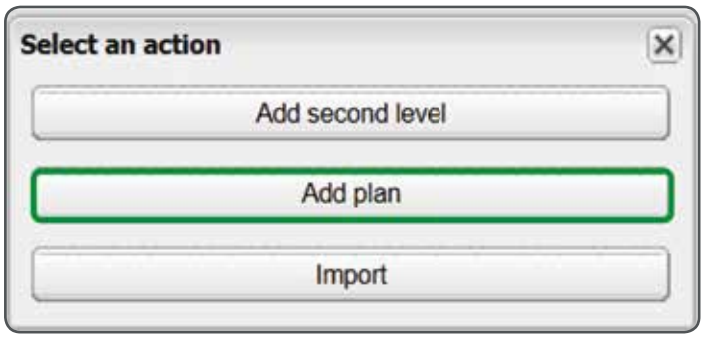
Add a security user interface page/plan

Creating the plan

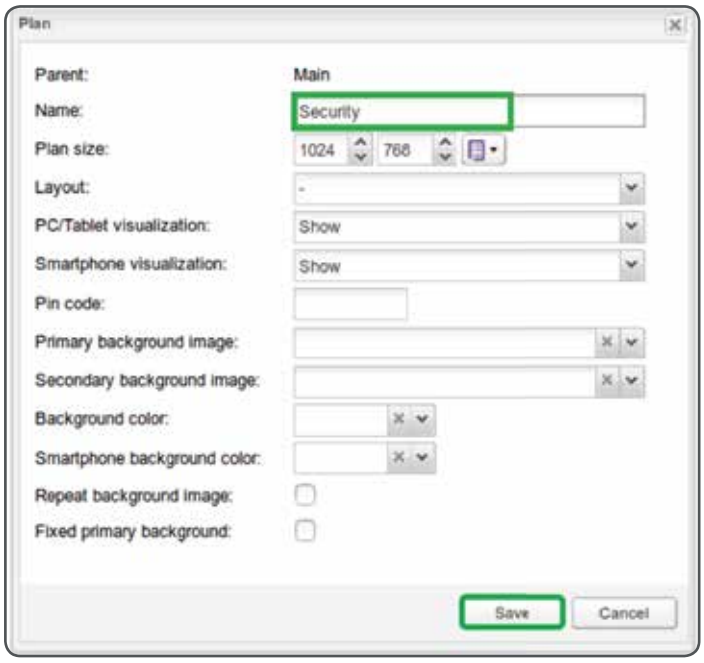
Path: ***Vis.Structure*** → ***Add/Import*** button → ***System*** icon



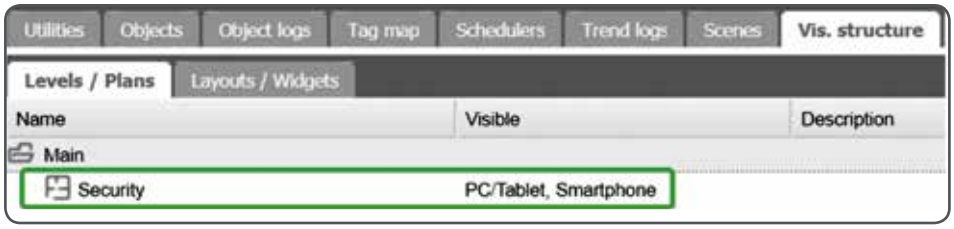
When the ***Select an action*** box appears, select ***Add plan***.



When the ***Plan*** window appears, at ***Name*** enter *Security,* then click ***Save***.



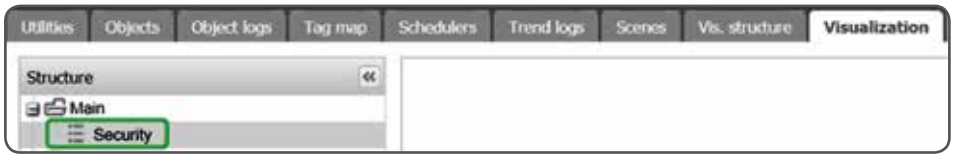
The ***Vis. Structure*** tab, ***Levels/Plan*** page then displays ***Security*** in the Main list.



Create visualization elements (icons) for commands on security page/plan

Ensure no visual element is selected on the ***Levels/plan*** page.

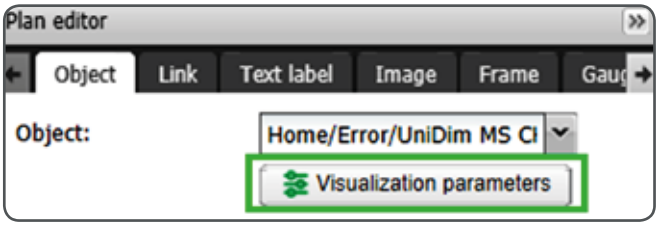
Path: ***Configurator*** → ***Visualization*** tab → ***Structure*** view → ***Security*** object.



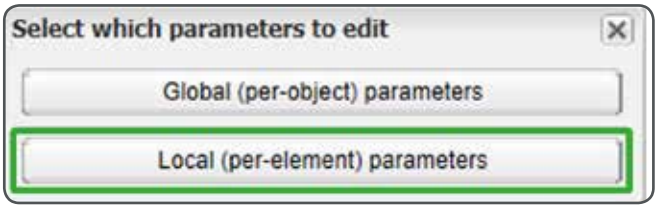
Click ***Unlock current plan for editing*** at the bottom right to enter edit mode.

To add object visualization elements, click the ***Object*** drop down arrow, then select *Home/Security/Command (0/208/0/0)*.

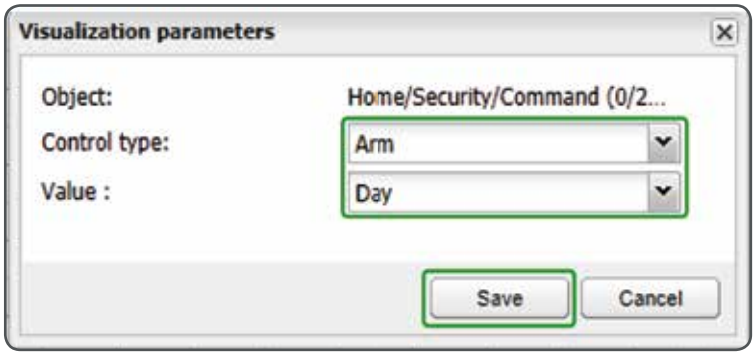
Click ***Visualization parameters***.



Click the ***Local (per-element) parameters*** button.

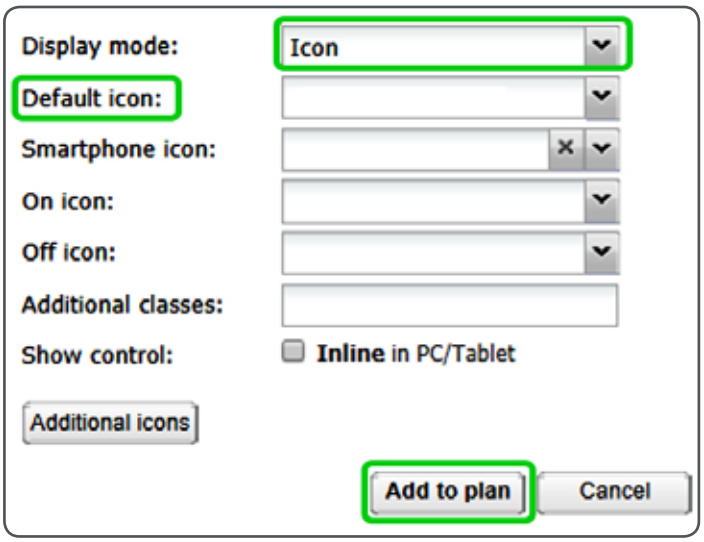


In the ***Visualization parameters*** window, enter ***Control Type*** = *Arm* and ***Value*** = *Day*, then click ***Save***.



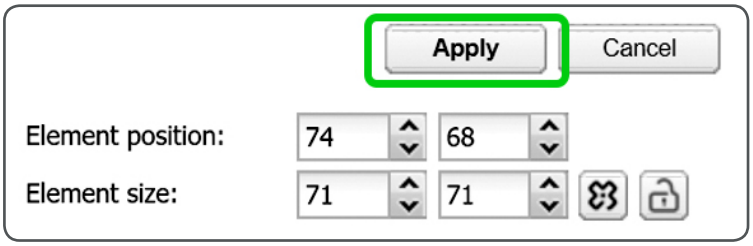
In the ***Visualization\Object*** tab, select ***Display mode*** = *Icon* and select the Default icon for the command *Armdaytime* mode.

Click ***Add to plan*** to add the icon on to the security page grid.



Click the newly added icon on the grid to adjust the position and size, then drag the icon to the desired location.

With the icon selected, use ***Element position*** and ***Element size*** to align the icon. Click ***Apply***.



Repeat the above steps to add icons for all visualization elements.

Click ***Save and reload plan***. When selected, the icon will now send related C-Bus commands.

Clicking ***Save and reload plan*** after adding each element avoids having to exit edit mode for each element.

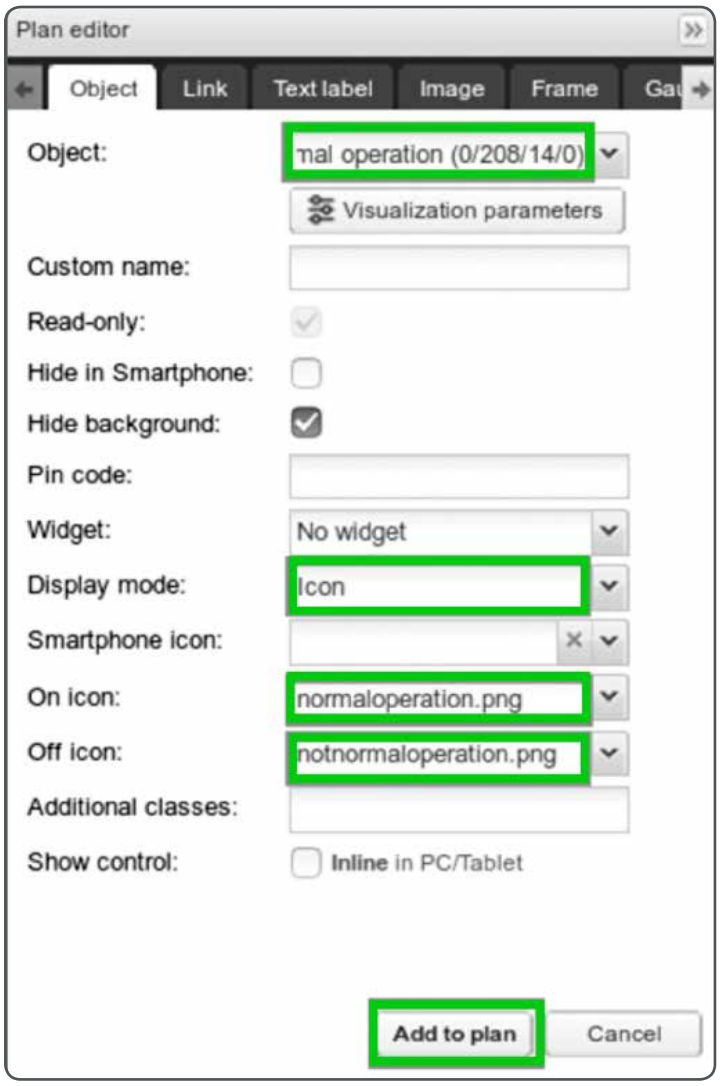
Create visualization elements for status on security page/plan

Adding icons to the Security page

① Ensure no visual element is selected on the page/plan.
② Path: ***Configurator*** → ***Visualization*** tab → ***Structure*** view → ***Security*** object.
③ Click ***Unlock current plan for editing***.
④ In the ***Plan Editor***, select the ***Object*** tab.
⑤ At ***Object***, select *Local/Security/Normal operation (0/208/14/0)*.
⑥ At the ***Display Mode*** dropdown, select *Icon*.
⑦ At ***On icon***, select an icon for visualization representing true (on).
⑧ At ***Off icon***, select an icon for visualization representing false (off).

Normal operation has 2 possible values: *True* and *False*.

Click ***Add to plan***.



Select the newly created icon on the grid to adjust position and size.

Click ***Apply***.

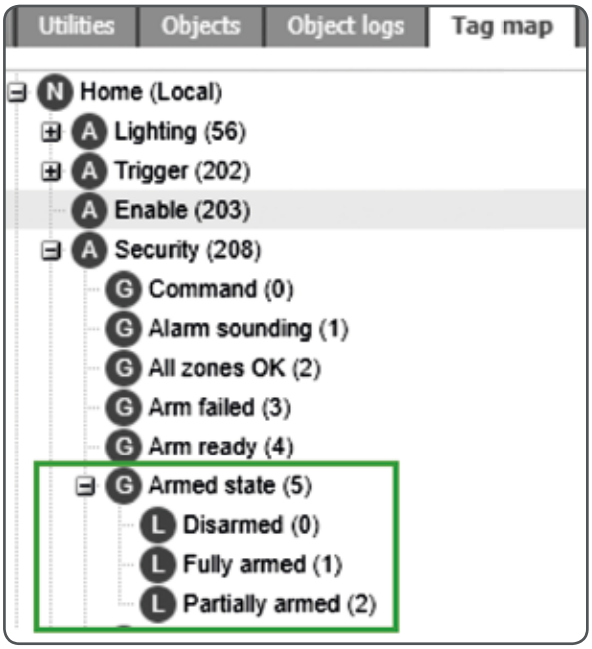
Click ***Save and reload plan***.

Repeat to add other icons to the security page.

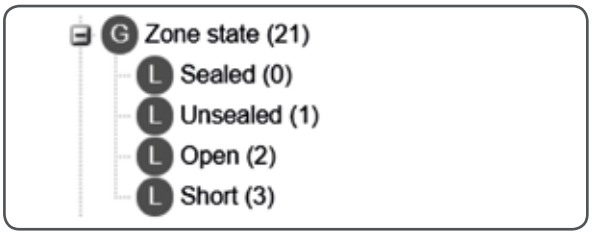
Viewing Armed state and Zone state values

Path: ***Tag map*** tab → ***Local*** tab → ***Structure*** view → ***Security*** object → ***Armed state*** object.

Expand ***Armed state*** to view the values: *Disarmed (0)*; *Fully armed (1)*; and *Partially armed (2)*.



Scroll to ***Zone state*** to view the values: *Sealed (0)*; *Unsealed (1)*; *Open (2)*; and *Short (3)*.



Setting up icons for Armed state

Ensure no visual element is selected on the levels/plan page.

Path: ***Configurator*** → ***Visualization*** tab → ***Structure*** view → ***Security*** object.

Click ***Unlock current plan for editing*** at the bottom right to enter edit mode.

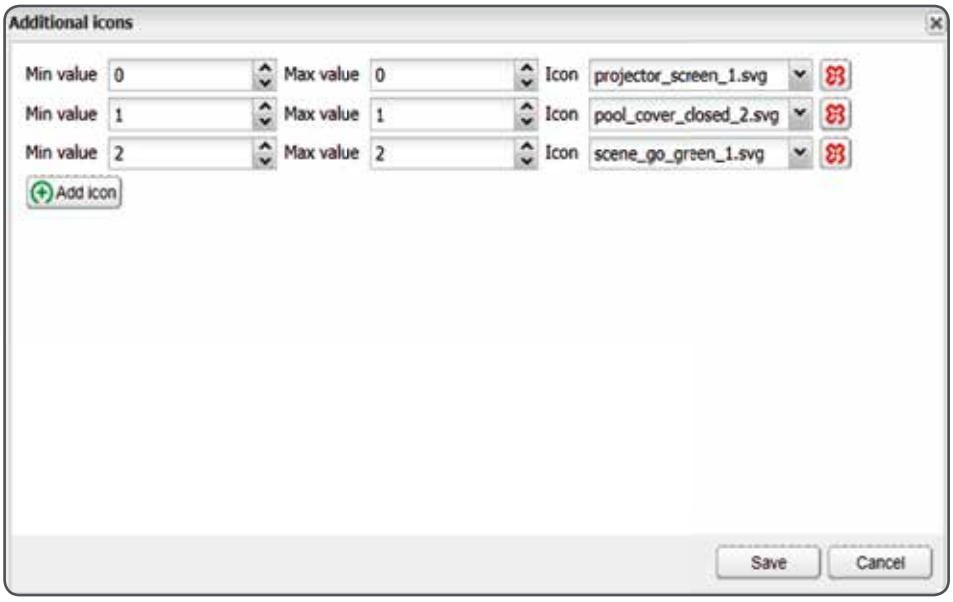
To add object visualization elements, click the ***Object*** drop down arrow, then select ***Local/Security/Armed state***.

Select a ***Default icon*** from the available selection, then click ***Additional icons***.

In the ***Additional icon*** window, click ***Add icon***.

Enter the ***Min value*** and ***Max value***, and select the icon for the values *Disarmed (0)*; *Fully armed (1)*; and *Partially armed (2)*.

Click ***Save***.



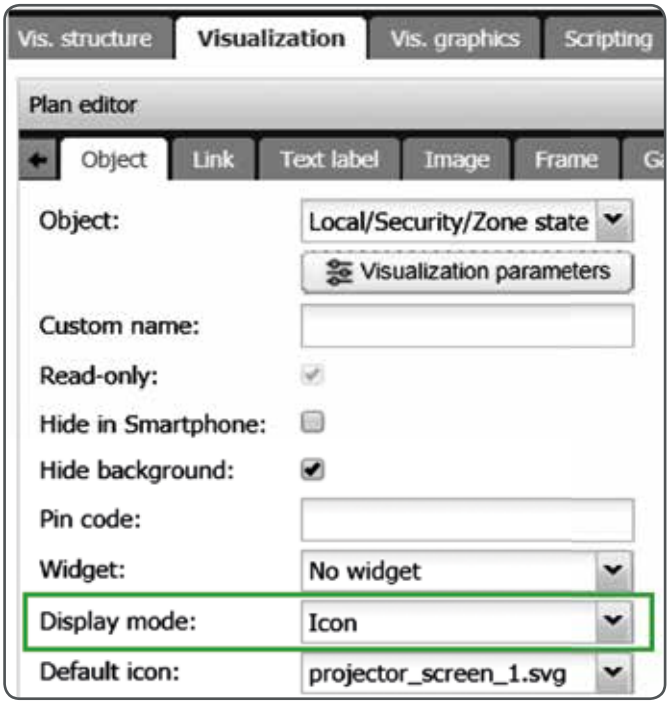
Setting up icons for Zone state

① Unlock the plan for editing. See *Setting up icons for Armed state* on page 57.

② Click the ***Object*** drop down arrow, then select *Local/Security/Zone state/1 (0/208/21/1)*.

③ Select ***Display mode*** = *Icon*.

④ Select the ***Default icon*** from the available selection.



Repeat for all Zone states required.

Click ***Save and reload plan***.

Creating text labels for the zones

Ensure no visual element is selected on the levels/plan page.

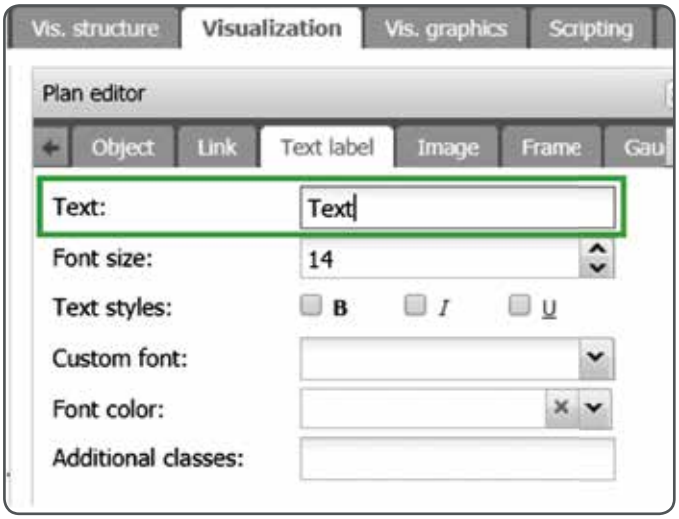
Path: ***Configurator*** → ***Visualization*** tab → ***Structure*** view → ***Security*** object.

Click ***Unlock current plan for editing*** at the bottom right to enter edit mode.

Select an ***Object*** drop down arrow to add a label to.

Click the ***Visualization*** tab, then select ***Text label*** from the ***Plan editor***.

In the ***Text*** field, enter the required text label.



Click ***Add to plan***.

Click the text label on the grid to adjust size and position.

Click ***Apply***.

Click ***Save and reload plan***.

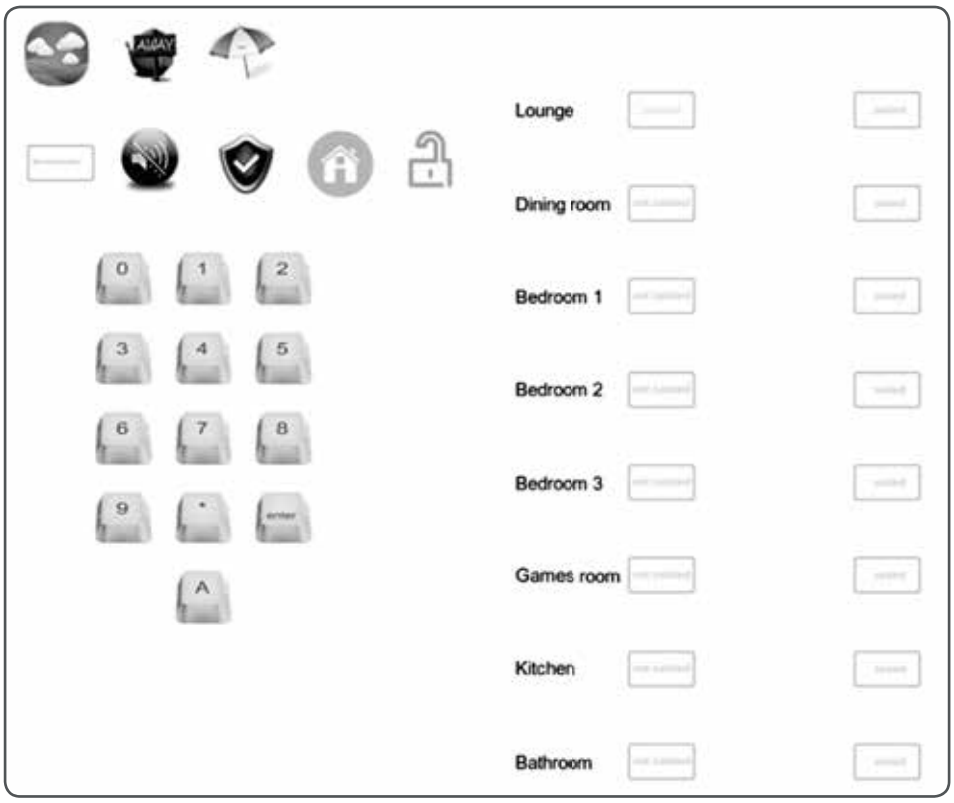
Repeat until all text labels required are in position. In this example, we have used the following labels.

Table 11: *Zone numbers and example labels*

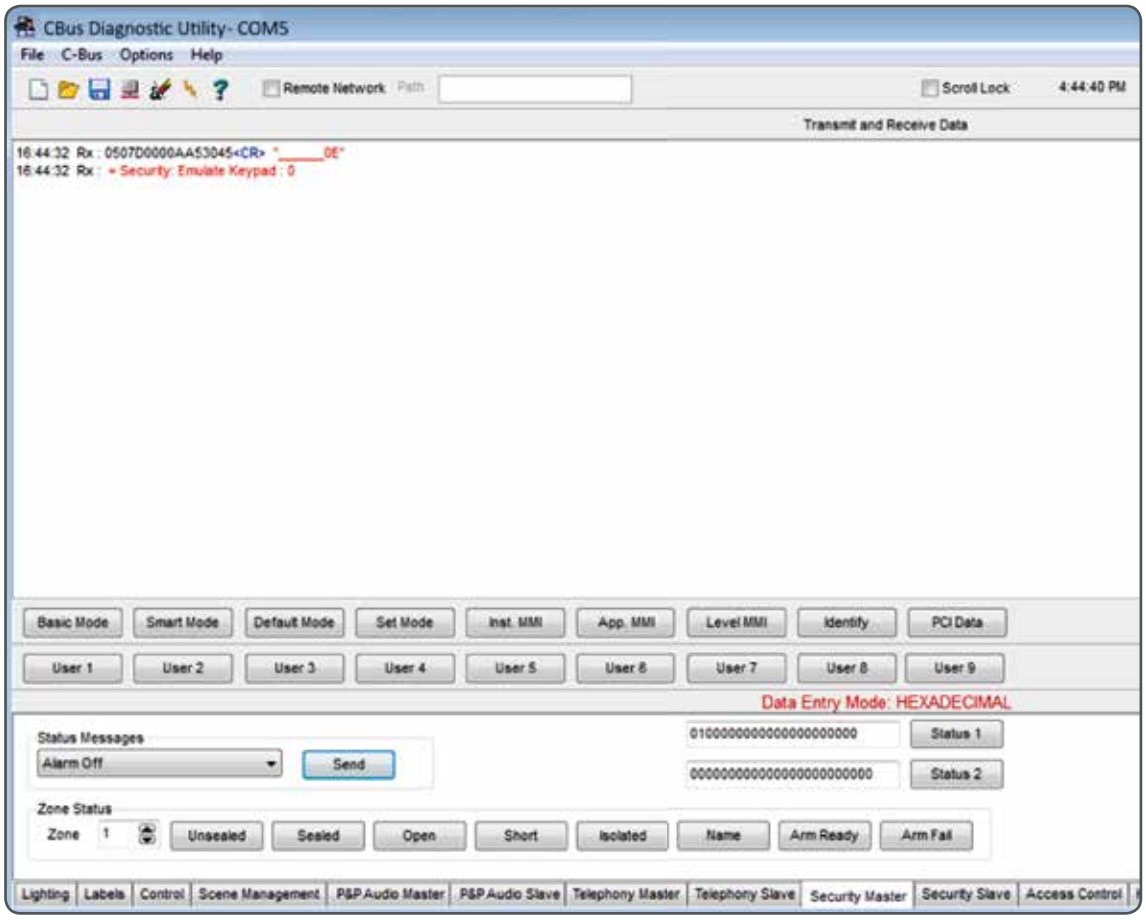
Zone number	Description (Location)
1	Lounge
2	Dining Room
3	Bedroom 1
4	Bedroom 2
5	Bedroom 3
6	Games Room
7	Kitchen
8	Bathroom

To run a test view, select the ***view icon*** in the bottom left window.

All the visual elements in this example security page could be arranged like this:



Clicking any command icon, e.g. a keypad button, sends a C-Bus message from the C-Bus Diagnostic Utility.



When using the Security panel or C-Bus Diagnostic Utility to change a related status, the status icon changes on the security page. The icon changes for ***Alarm sounding*** becoming *True* when issued an ***Alarm on*** message on Zone 1 from the C-Bus Diagnostic Utility.

Lua scripting example

The following Lua scripting is based on the previous security application example and demonstrates triggering a script on a security event.

Example 1

A disarm event triggers a script which activates the ***Welcome Home*** scene.

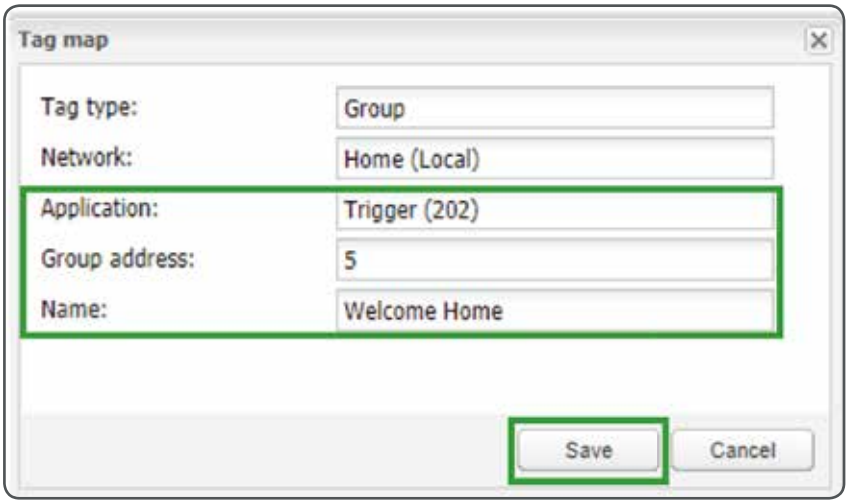
Click ***Tag map***.

Scroll to *Trigger Control (202)* and click ⊕.

Enter ***Group address*** = *5*.

Enter ***Name*** = *Welcome Home*.

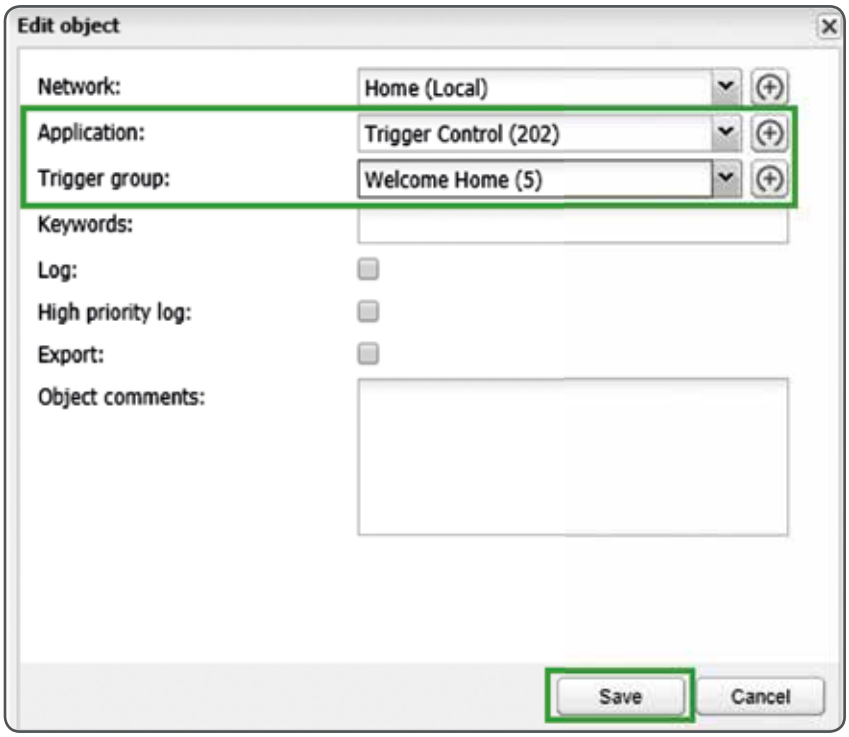
Click ***Save***.



Open the ***Objects*** tab.

Click ***Add new object***.

In the ***Edit object*** window select *Application = Trigger Control (202)*, then select ***Trigger group*** = *Welcome Home (5)*.



Click ***Save***.

Open the ***Scenes*** tab.

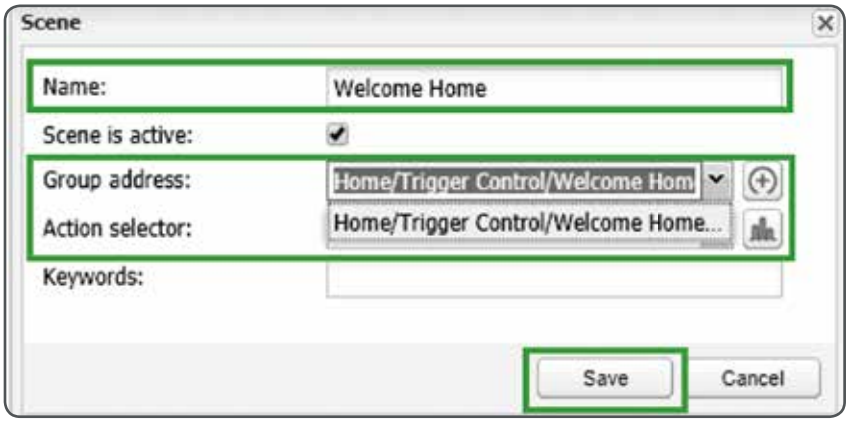
Select ***Add Scene.***

At the ***Scene*** window, set ***Name*** = *Welcome Home*.

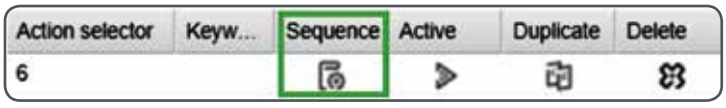
Select the ***Group address*** = *Home/Trigger Control/Welcome Home (0/202/5)*.

Enter ***Action selector*** = *6*.

Click ***Save***.



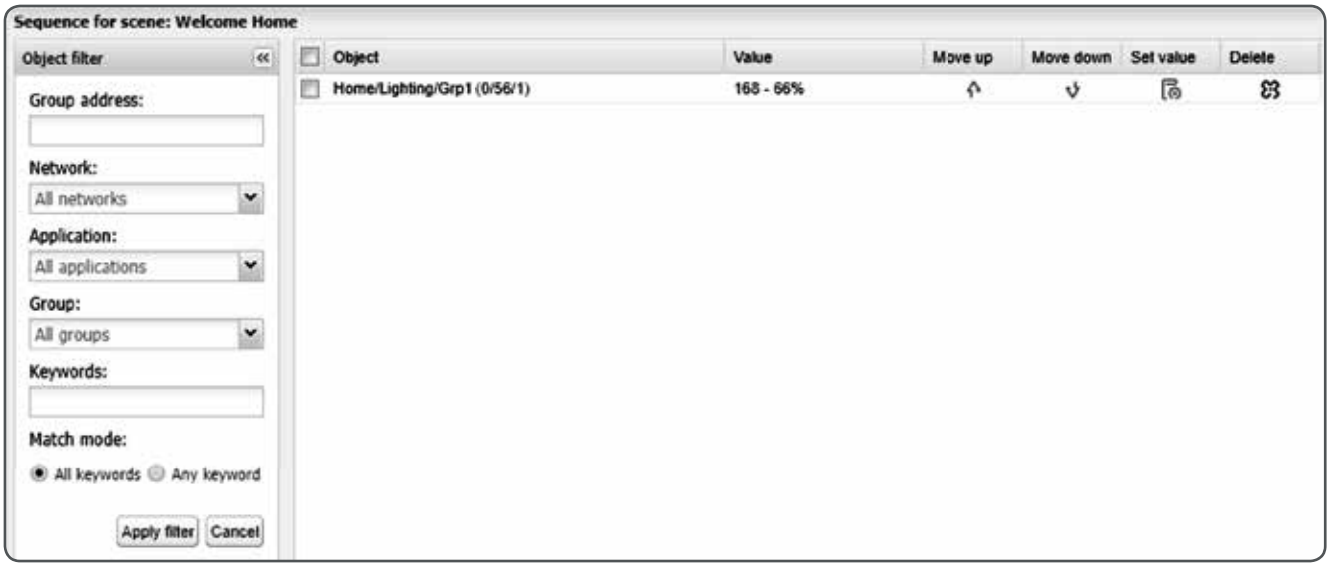
Click ***Sequence*** to add reactions for this scene.



Click ***Add objects***.

When the ***Sequence for scene: Welcome Home*** page displays, add ***Group Address*** = *Home/Lighting/grp1 (0/56/1)*.

Set ***Value*** at *168*.

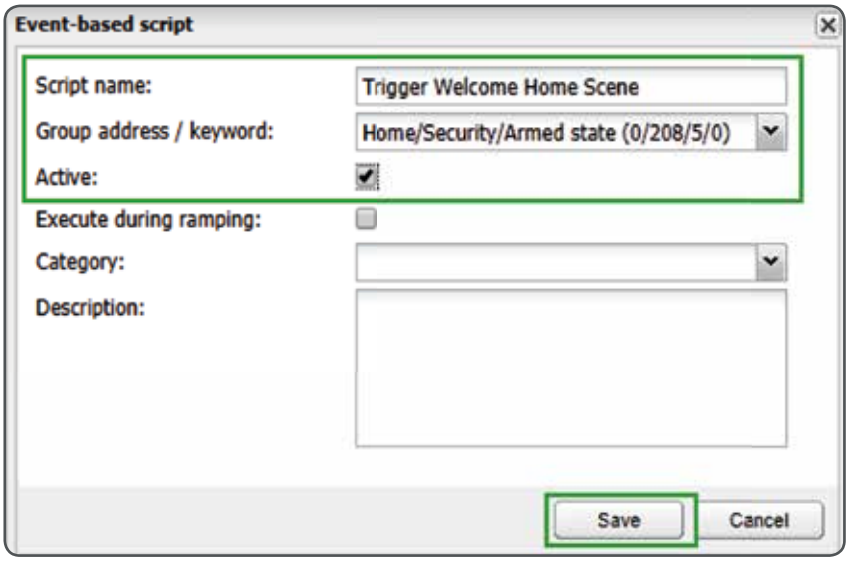


Lua script for disarm event

Path: ***Configurator*** → ***Scripting*** tab → ***Event-based*** button → ***Add new script*** button.

Enter the ***Script name*** *Trigger Welcome Home Scene* when disarmed and select the ***Group address*** *Home/Security/Armed state (0/208/5/0)*.

Check ***Active*** and click the ***Save*** button.



Scroll to the Script name *Trigger Welcome Home Scene*, then click ***Editor***.

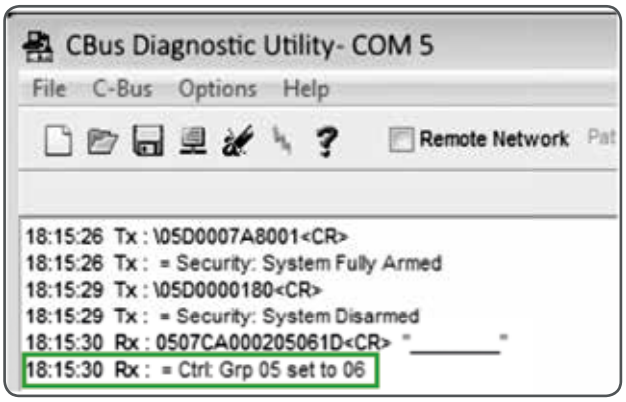
The following script sets the Trigger Welcome Home Scene.

```
-- if armed state has a value changed to 0 (disarmed)
if event.getvalue() == 0 then
  -- Set trigger level of group 5 to 6
  SetTriggerLevel=(5, 6)
end
```

Click ***Run script***.

At the ***Set object value*** window click ***Save***. The script list status changes to *Active*.

If ***Armed state*** is set to *Fully armed* and then to *Disarmed* on Zone 1 from the C-Bus Diagnostic Utility (the value need to be changed to *Fully armed* hence setting it to another value first). The scene ***Welcome Home*** sets the trigger group *5* to value *6*.



Example 2

An Alarm event can trigger a script to send email notifications. This example uses the Fire alarm event trigger.

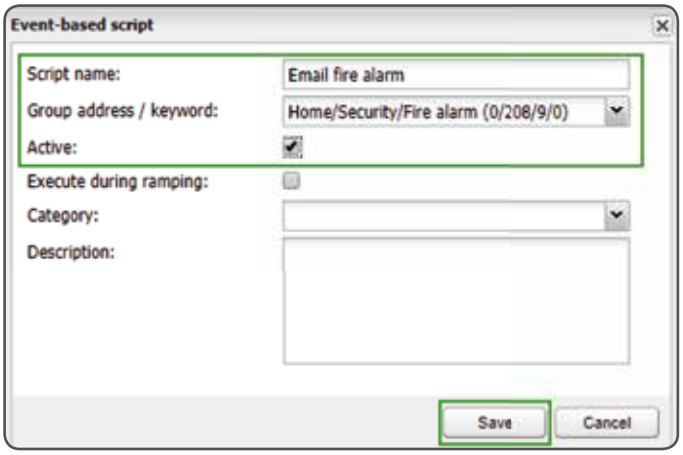
Path: ***Configurator*** → ***Scripting*** tab → ***Event-based*** button → ***Add new script*** button.

In the ***Event-based script*** window, enter the ***Script name*** = *Email fire alarm*.

Select ***Group Address*** = *Home/Security/Fire alarm (0/208/9/0)*.

Tick ***Active***.

Click ***Save***.



In the ***Script name*** window, scroll to *Email fire alarm*, then click ***Editor***.

The following script sets the Email fire alarm:

```
if event.getvalue() == true then
  return
end
-- make sure mail settings are set in user function library before
using this function
  local subject = 'Fire Alarm'
  local message = 'You have got a fire alarm\n'
  message = message .. 'Current status:\n'
  local detailedInfo={}
  detailedInfo.alarmSounding = CBusSecurityGet(0, 'Alarm sounding')
  detailedInfo.allZonesOK = CBusSecurityGet(0, 'All zones OK')
  detailedInfo.armedState = CBusSecurityGet(0, 'Armed state')
  detailedInfo.fireAlarm = CBusSecurityGet(0, 'Fire alarm')
  detailedInfo.normalOperation = CBusSecurityGet(0, 'Normal
operation')
  message = message .. tab2str(detailedInfo)
  mail('user@example.com', subject, message)
```

Enter a valid email address instead of *user@example.com* at Line 15 in the above example.

To enable email settings, refer to the section *E-mail* on page 168.

Click ***Run script***, then ***Save***.

In the ***Scripts*** tab, select ***Event-based***, and the event *Email fire alarm* will appear with a green arrow indicating *Active* status.

Test by setting the C-Bus Diagnostic Utility to send a *Fire Alarm Cleared* message followed by a *Fire Alarm Raised* message on Zone 1 by changing the value to *Fire Alarm Raised* and checking for a received email.

4.3.6 Multi-Room Audio and Media Transport Control – Control C-Bus

The C-Bus Automation Controller supports Multi-Room Audio (MRA) and Media Transport Control (MTC) applications, which can be controlled by C-Bus Matrix Switcher and Amplifier, and status information can be obtained from the Automation Controller. This section explains how to set up the Automation Controller to support MRA and MTC applications.

Prerequisites

The following Application Notes, must be read and configured before completing any MRA installation:

- Configuring MRA in Toolkit Software
- Configuring MRA with MRAPA Software
- Configuring MRA with PICED Software, and
- Configuring C-Bus Ripple Software.

Importantly, the settings and configurations must be completed for the following devices:

- one Matrix Switcher
- one Amp
- one EDLT, and
- a Ripple media server.

MRA Objects

An MRA object address must be specified by a network number, zone number and a function address. The Automation Controller does not use Matrix Switcher number in MRA objects, therefore the zone number is used to configure both Matrix Switcher number and zone number used in Toolkit, PICED and MRAPA. See *Table 12* for Zone number mapping.

Table 12: *Zone numbers for MRA objects*

Zone Number in Automation Controller	Matrix Switch Number	Zone number in Toolkit, PICED and MRAPA
0	1	1
1	1	2
2	1	3
3	1	4
4	1	5
5	1	6
6	1	7
7	1	8
8	2	1
9	2	2
10	2	3
11	2	4
12	2	5
13	2	6
14	2	7
15	2	8
16	3	1
17	3	2
18	3	3

Zone Number in Automation Controller	Matrix Switch Number	Zone number in Toolkit, PICED and MRAPA
19	3	4
20	3	5
21	3	6
22	3	7
23	3	8
24	4	1
25	4	2
26	4	3
27	4	4
28	4	5
29	4	6
30	4	7
31	4	8

The MRA object's function address could be from 0 to 10, as shown in *Table 13*.

Table 13: *MRA object function addresses*

Function Address	Meaning	Value Type	Value Range	Access Type
0	Volume	int	0-255	R/W
1	Balance	int	0-255	R/W
2	Bass	int	0-255	R/W
3	Treble	int	0-255	R/W
4	Mute	int	0, 2, 5, 7, 255 0 – Turn amplifier off. 2 (1%) – Turn amplifier on, Volume normal and speakers off. 5 (2%) – Turn amplifier on, Volume at pre-set and speakers off. 7 (3%) – Turn amplifier on, Volume at pre-set and speakers on. 255 (100%) – Turn amplifier on, Volume normal and speakers on.	R/W
5	Source Number	int	0-6 (As defined 1-7 in MARPA)	R/W
6	Dynamic 1 Label	string	Max 255 bytes	R/W
7	Dynamic 2 Label	string	Max 255 bytes	R/W
8	Source Descriptor	string	Max 255 bytes	R/W
9	Zone Descriptor	string	Max 255 bytes	R/W
10	MRA Command	string	See Table 3	See Table 3

MRA command is a predefined string command with one or additional parameters. See *Table 14* for MRA command definitions.

Table 14: *MRA command definitions*

Predefined Comman	P1	P2	Description	Example	Access Type
Next Feed	N/A	N/A	Go to next feed/source	“Next Feed”	R/W
Previous Feed	N/A	N/A	Go to previous feed/ source	“Previous Feed”	R/W
All Off	N/A	N/A	Send an All Off command to all zones	“All Off”	R/W
Zone Descriptor Request	N/A	N/A	Request zone description	“Zone Descriptor Request”	R/W
Feed Descriptor Request	N/A	N/A	Request feed/source description	“Feed Descriptor Request”	R/W
Current Feed Request	N/A	N/A	Request current feed/ source description	“Current Feed Request”	R/W
Dynamic1	N/A	N/A	Dynamic1 is pressed	“Dynamic1”	R/W
Dynamic2	N/A	N/A	Dynamic2 is pressed	“Dynamic2”	R/W
Priority On	level	feed	Set high priority on	“Priority On:80:3”	R/W
Priority Off	N/A	N/A	Set high priority off	“Priority Off”	R/W
Set Off Timer	time to off (in minutes)	N/A	Set power off timer	“Set Off Timer:120”	R/W
Cancel Off Timer	N/A	N/A	Cancel power off timer	“Cancel Off Timer”	R/W
Error Code	Error code number	N/A	Report error code	“Error Code:20”	R/W
Status Request	N/A	N/A	Request status	“Status Request”	R/W
Off Timer Expired	N/A	N/A	This is a notification when power off timer is expired. Read only.	“Off Timer Expired”	R

MTC Objects

A MTC object address must be specified by a network number, media link group and a function address. A MTC object’s function address is from 0 to 32, *Table 15* has the details.

Table 15: *MTC object function addresses*

Function Address	Meaning	Value Type	Value Range	Access Type
0	Play Stop	int	0:stop, 1:play	R/W
1	Pause Resume	int	0:pause, 255:resume	R/W
2	Category	int	0-127	R/W
3	Selection	int	0-32767	R/W
4	Track	int	0-2147483647	R/W
5	Shuffle	int	0: shuffle off, 255: shuffle on	R/W
6	Repeat	int	0: repeat off, 1: repeat current, 255: repeat all	R/W
7	Forward	int	0: normal speed, 2: 2x speed, 4: 4x speed, 6: 8x speed, 8: 16x speed, 10: 32x speed, 12: 64x speed	R/W
8	Rewind	int	0: normal speed, 2: 2x speed, 4: 4x speed, 6: 8x speed, 8: 16x speed, 10: 32x speed, 12: 64x speed	R/W

Function Address	Meaning	Value Type	Value Range	Access Type
9	Source Power Control	int	0: source power off, 255: source power on	R/W
10	Total Tracks	int	0-2147483647	R/W
11	Current Track Name	string	Max 255 bytes	R/W
12	Current Selection Name	string	Max 255 bytes	R/W
13	Current Category Name	string	Max 255 bytes	R/W
14	Next Track Name	string	Max 255 bytes	R/W
15	Next Selection Name	string	Max 255 bytes	R/W
16	Next Category Name	string	Max 255 bytes	R/W
17	Previous Track Name	string	Max 255 bytes	R/W
18	Previous Selection Name	string	Max 255 bytes	R/W
19	Previous Category Name	string	Max 255 bytes	R/W
20	Next2 Track Name	string	Max 255 bytes	R/W
21	Next2 Selection Name	string	Max 255 bytes	R/W
22	Next2 Category Name	string	Max 255 bytes	R/W
23	Previous2 Track Name	string	Max 255 bytes	R/W
24	Previous2 Selection Name	string	Max 255 bytes	R/W
25	Previous2 Category Name	string	Max 255 bytes	R/W
26	MTC Command	string	See Table 5	See Table 5
27	Enumerate Category Size	int	0-255	R/W
28	Enumerate Selection Size	int	0-255	R/W
29	Enumerate Track Size	int	0-255	R/W
30	Enumerate Category Names	string	Max 255 bytes	R/W
31	Enumerate Selection Names	string	Max 255 bytes	R/W
32	Enumerate Track Names	string	Max 255 bytes	R/W

MTC command is a predefined string command with some commands having one or two additional parameters. Parameter is followed by the command and separated by ":".

Table 16: *MTC command objects*

Predefined Command	P1	P2	Description	Example	Access Type
Next Category	N/A	N/A	Go to next category	"Next Category"	R/W
Previous Category	N/A	N/A	Go to previous category	"Previous Category"	R/W
Next Selection	N/A	N/A	Go to next selection	"Next Selection"	R/W
Previous Selection	N/A	N/A	Go to previous selection	"Previous Selection"	R/W
Next Track	N/A	N/A	Go to next track	"Next Track"	R/W
Previous Track	N/A	N/A	Go to previous track	"Previous Track"	R/W
Media Status	N/A	N/A	Request media status	"Media Status"	R/W
Enumerate Categories	start	N/A	Enumerate Categories	"Enumerate Categories:0"	R/W
Enumerate Selections	start	N/A	Enumerate Selections	"Enumerate Selections:0"	R/W
Enumerate Tracks	start	N/A	Enumerate Tracks	"Enumerate Tracks:0"	R/W

Example project

Prior to commencing an example project, compile a list of each source (an input into the MRA system) and zone (an area where an amplifier is installed). *Table 17* on page 70 and *Table 18* on page 70 show the sources and zones for an example project. These examples are used throughout the series of MRA programming application notes.

In *Table 14* Media Sources, the left column shows the source name, which is fixed in MRA. The middle column shows the type of input, while the right column shows a 'real world' name or description applied for easy identification. *Table 15* Output zones shows eight zones (or locations).

Table 17: *Media sources*

Source	Type	Description
Local*	Line-level input on amplifier	iPod
Analogue 1	Line-level input #1 into Matrix Switcher	Cable TV
Analogue 2	Line-level input #2 into Matrix Switcher	DVD
Analogue 4/AUX**	RCA sockets on the rear panel of the Matrix Switcher 3.5mm AUX input socket on front panel of Matrix Switcher	AUX
Tuner 1	Internal AM/FM tuner	Radio 1
Tuner 2	Internal AM/FM tuner	Radio 2
Streaming	Internal Streaming Module (deluxe model of Matrix Switcher only)	MP3

* The local input on the amplifier does not need to be configured for operation, but its position is reserved. This allows for a total of six additional source inputs to be configured on the Matrix Switcher

** The 3.5mm AUX socket on the front of the matrix switcher is internally (electrically) connected to Analogue Input 4. Either the AUX socket OR Analogue input 4 may be used, not both.

Table 18: *Output zones*

Zone No*	Description (location)
Zone 0	Living
Zone 1	Bathroom
Zone 2	Bedroom
Zone 3	Ensuite
Zone 4	Patio
Zone 5	Pool area
Zone 6	Dining
Zone 7	Study

* The zone number is from 0 to 7, which is corresponding to Matrix Switcher 1 and zone number 1 to 8 in Toolkit, MARPA and PICED settings. See Table 1 for details.

Media Link Group

The following media link groups are used for Streaming Audio, Radio 1 and Radio 2.

- Media Link Group 000 – Streaming Audio
- Media Link Group 001 – Radio Tuner 1, and
- Media Link Group 002 – Radio Tuner 2.

Table 19 shows all MRA objects used in the example project.

Table 19: *MRA objects used in the example project*

Network	Application	Zone Number	Function
Home(Local)	MRA	0	Volume
Home(Local)	MRA	0	Dynamic 1 Label
Home(Local)	MRA	0	Dynamic 2 Label
Home(Local)	MRA	0	MRA Command

Table 20 on page 71 shows all MTC objects used in the example project.

Table 20: *Objects used in the example project*

Network	Application	Media Link Group	Function
Home(Local)	Media Transport	Streaming Audio (0)	Play/Stop
Home(Local)	Media Transport	Streaming Audio (0)	Current Track Name
Home(Local)	Media Transport	Streaming Audio (0)	Current Selection Name
Home(Local)	Media Transport	Streaming Audio (0)	Current Category Name
Home(Local)	Media Transport	Streaming Audio (0)	MTC Command
Home(Local)	Media Transport	Radio Tuner 1	Current Selection Name
Home(Local)	Media Transport	Radio Tuner 1	MTC Command
Home(Local)	Media Transport	Radio Tuner 2	Current Selection Name
Home(Local)	Media Transport	Radio Tuner 2	MTC Command

This example project explains how to control C-Bus media system using an Automation Controller. These steps detail how to create control objects for zone 0. The steps for other zones are similar.

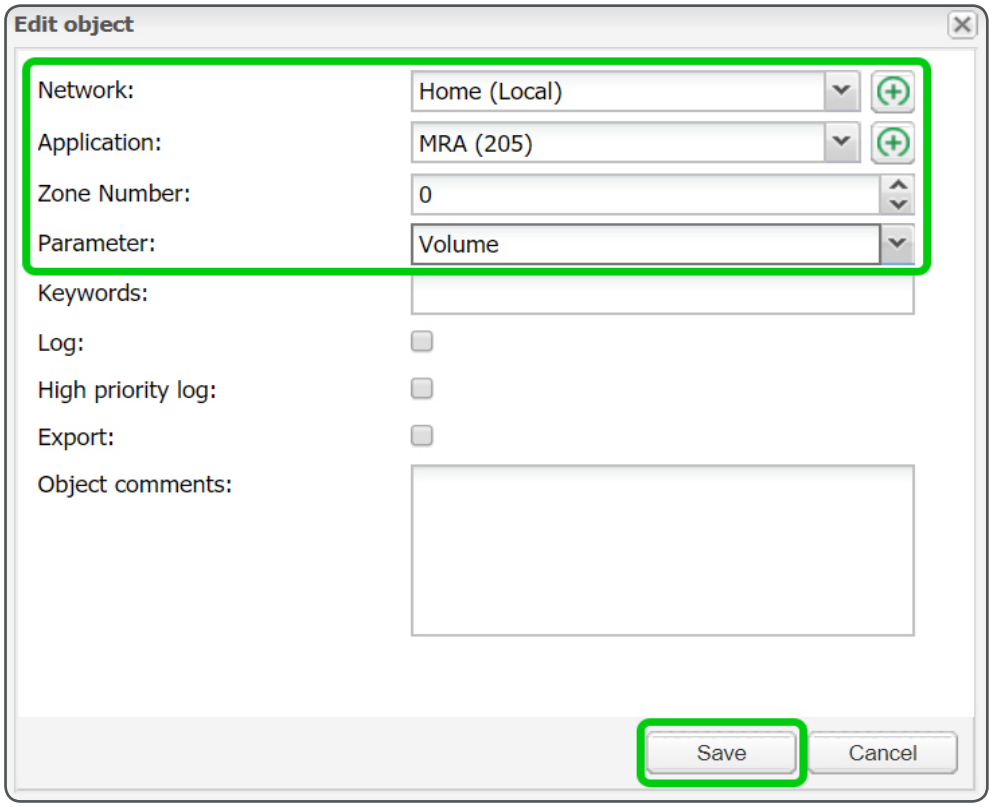
Create MRA Related Objects

Path: ***Configurator*** → ***Objects*** tab → ***Add new object*** .

In ***Edit object*** window enter:

- ***Network*** select *Home (Local)*
- ***Application*** select *MRA - 205*
- ***Zone Number*** select *0*
- ***Parameter*** select *Volume*.

Click ***Save***.



Add the remaining objects in *Table 19* on page 71. The object window will display all four entries in the object screen.

Name	Group address
Home/MRA/Zone 0/Volume	0/205/0/0
Home/MRA/Zone 0/Dynamic1 LABEL	0/205/0/6
Home/MRA/Zone 0/Dynamic2 LABEL	0/205/0/7
Home/MRA/Zone 0/MRA Command	0/205/0/10

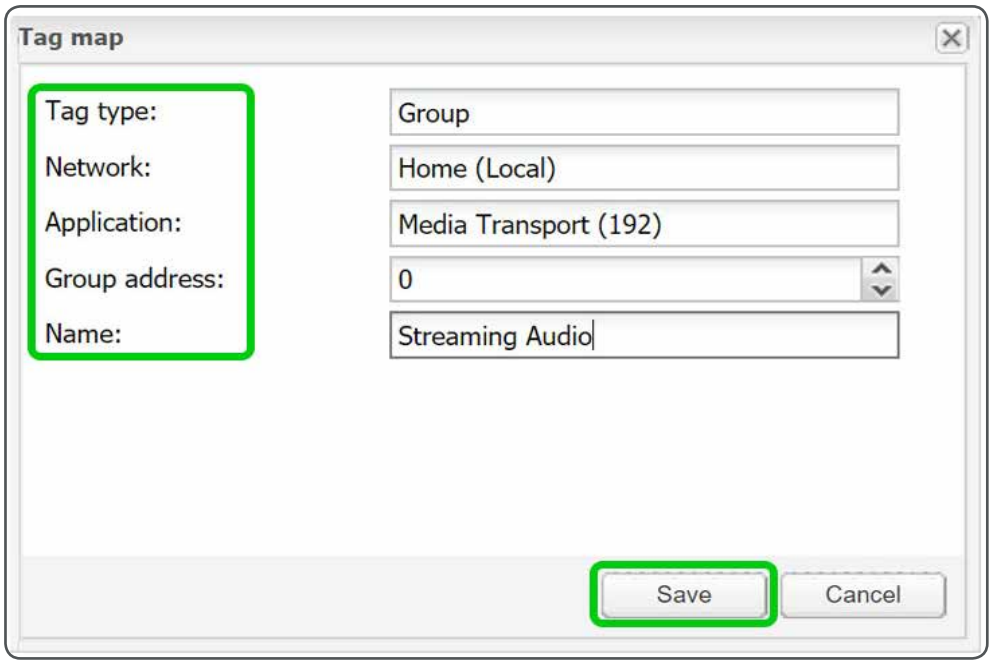
Create related MTC Objects

Path: ***Configurator*** → ***Objects*** tab → ***Add new object***.

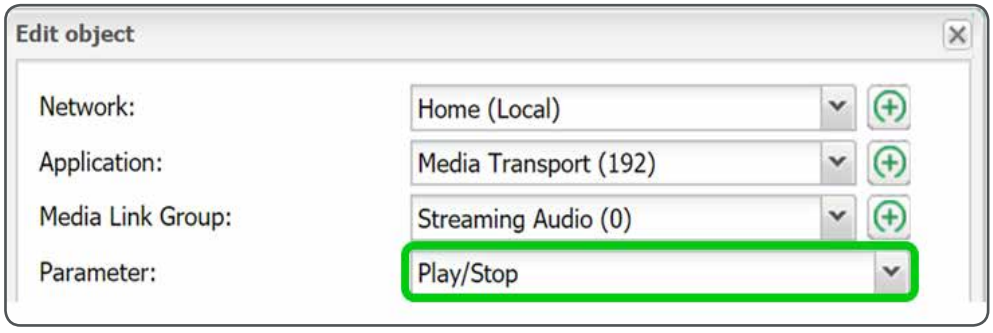
In ***Edit object*** window box enter:

- ***Network*** select *Home (Local)*
- ***Application*** select *Media Transport*
- ***Media Link Group*** select *0*
- ***Parameter*** enter *Streaming Audio*.

Click ***Save***.



At ***Parameter*** select ***Play/Stop.***



Add the remaining objects listed in *Table 20* on page 71.

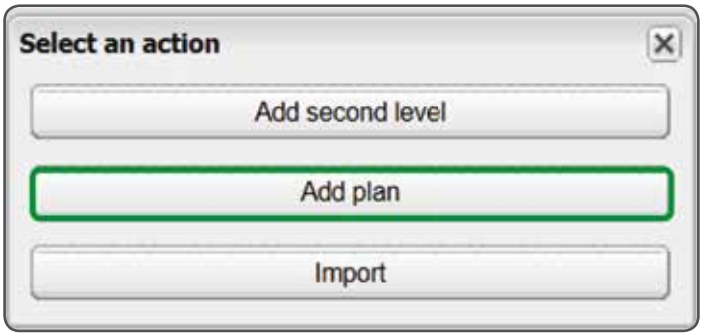
Click ***Save***.

Add the other objects listed in *Table 15* on page 68.

Add an MRA/MTC user interface page/plan

Path: ***Configurator*** → ***Vis. Structure*** tab → ***Add new object*** button → Add/Import ⊕.

At ***Select an action*** select ***Add plan***.



At the ***Plan*** window appears, enter *Media - Living* in the ***Name*** field and then click ***Save***.

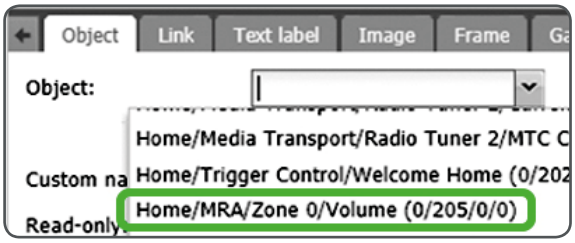
Create visualization elements for media living page/plan

Path: ***Configurator*** → ***Visualization*** → ***Media Living*** page → ***Unlock current plan for editing***.

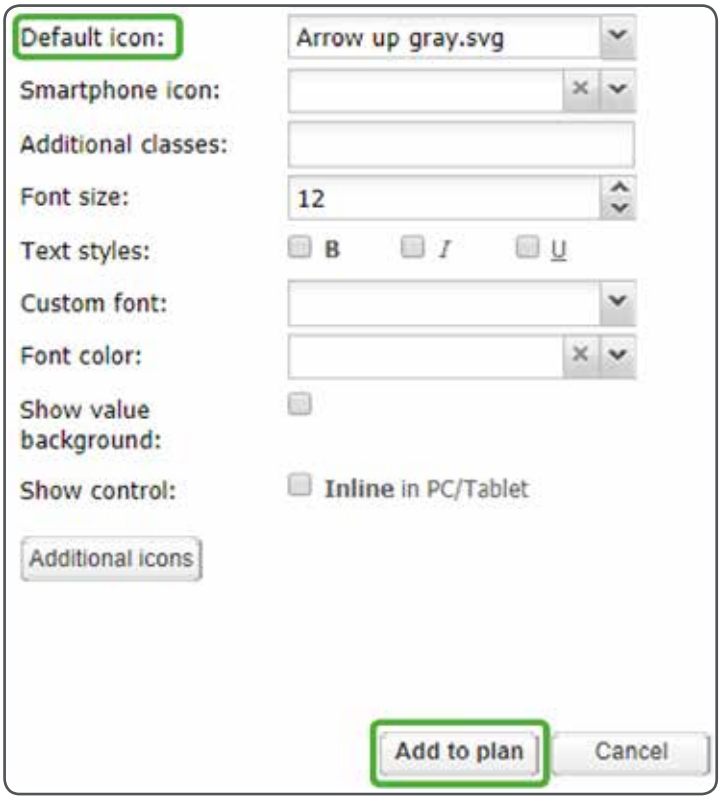
Ensure no visual elements are selected in the page/plan.

In the Plan editor, select the ***Object*** tab.

At the ***Object*** dropdown, select ***Home/MRA/Zone 0/Volume (0/205/0/0)***.



Change the ***Default icon*** to a volume icon.



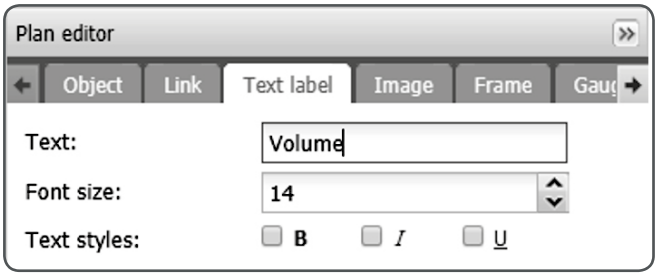
Click ***Add to plan***.

Drag to a desired location and click ***Apply***.

To add a text label, click the ***Text label*** tab.

Enter text in the ***Text*** field.

Adjust any preferred font characteristics, then click ***Add to Plan***.



Drag the text label to the preferred location on page.

Repeat the steps for ***Dynamic 1 Label*** and ***Dynamic 2 Label***.

Click ***Save and reload plan***.

Add MRA Command Object

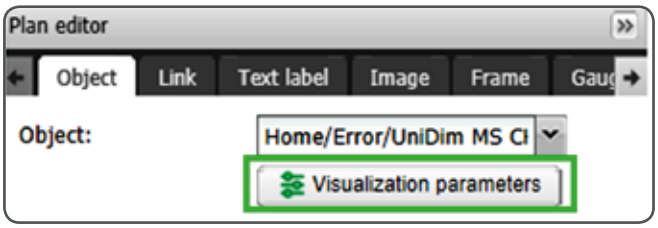
Path: ***Configurator*** → ***Visualization*** → ***Media Living*** page → ***Unlock current plan for editing***.

Ensure no visual elements are selected in the page/plan.

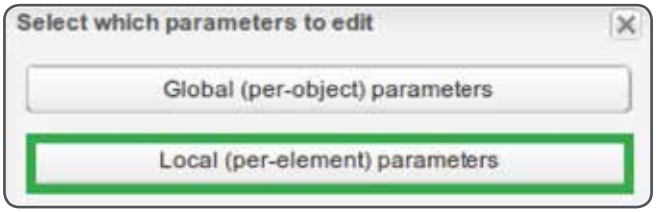
In Plan editor, select the ***Object*** tab.

At the ***Object*** dropdown, select ***Home/MRA/Zone 0/MRA Command (0/205/0/10)***.

Click ***Visualization parameters***.

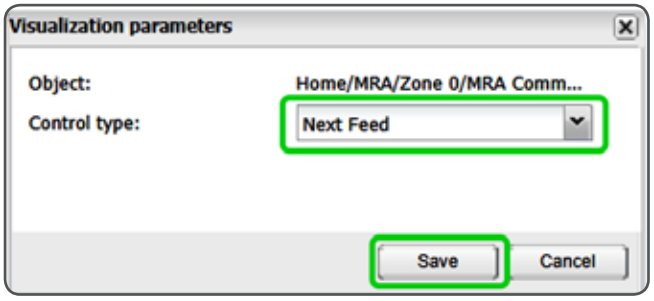


At ***Select which parameters to edit*** window click ***Local (per-element) parameters***.



At ***Visualization parameters*** window click ***Control Type*** box ***Next Feed***.

Click ***Save.***



In Plan editor change the ***Display mode*** to ***Icon.***

At ***Default icon*** select a preferred icon.

Click ***Add to plan***.

Drag the icon to a preferred location on the plan.

In Plan editor, select the ***Text label*** tab, and enter Text = ***Next Source/Feed***. Change any font preferences.

Click ***Add to plan***.

Drag the text label to the preferred location next to the icon.

Click ***Save and reload plan***.

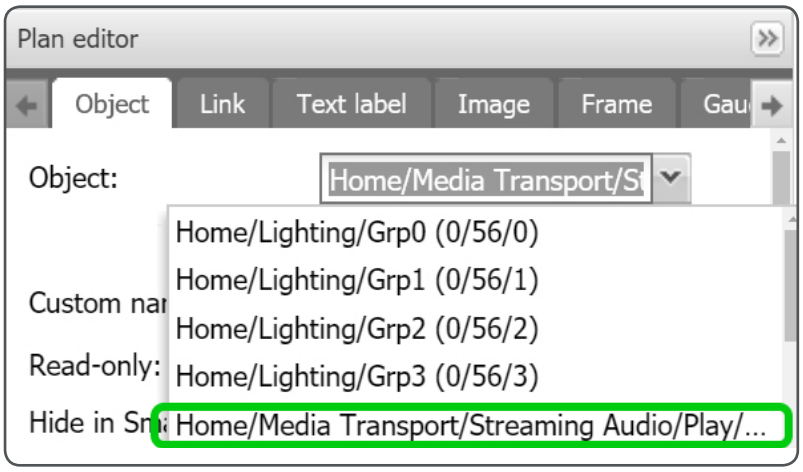
Adding visual elements for MTC objects

Path: ***Configurator*** → ***Visualization*** → ***Media Living*** page → ***Unlock current plan for editing***.

Ensure no visual elements are selected in the page/plan.

In Plan editor, select the ***Object*** tab.

At the ***Object*** dropdown, select ***Home/Media Transport/Streaming Audio/Play/Stop (0/192/0/0)***.

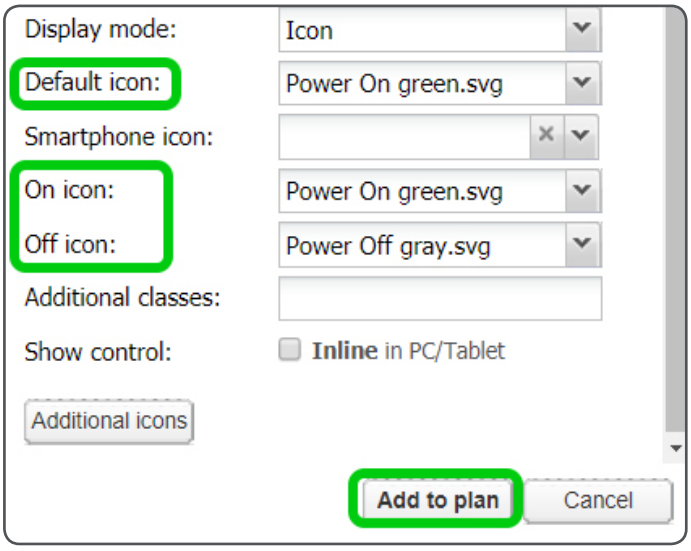


At Display mode select ***Icon***.

Select the preferred ***Default Icon*** from the dropdown.

Set the ***On icon*** to a preferred On Icon from the ***Select Image*** window.

Set the ***Off icon*** to a preferred Off Icon from the ***Select Image*** window.



Click ***Add to plan***.

Drag the icon to a preferred location on the plan, then select the icon, and in Plan Editor, select the ***Text label*** tab.

Enter text = ***Play/Stop***.

Adjust any preferred font characteristics, then click ***Add to plan***.

Move the label to the desired position above the icon in the plan.

Click ***Save and reload plan***.

Adding other visual elements

Path: ***Configurator*** → ***Visualization*** → ***Media Living*** page → ***Unlock current plan for editing***.

Ensure no visual elements are selected in the page/plan.

In Plan editor, select the ***Object*** tab.

At the ***Object*** dropdown, select ***Home/Media Transport/Streaming Audio/ Current Track Name (0/192/0/11)***.

At ***Default icon*** select a preferred icon.

Click ***Add to plan***.

Drag the icon to preferred location on the plan, and from ***Plan Editor/Text label***, enter ***Text*** = ***Streaming: Current Track Name***.

Adjust any font characteristics to Text label, then click ***Add to Plan***.

Adjust the Text label to preferred location next to icon. Click ***Save and reload plan***.

Repeat the steps to add visualization elements for the objects:

- Streaming Audio (0)\Current Selection Name
- Streaming Audio (0)\Current Category Name
- Current Selection Name\Radio Tuner 1
- Current Selection Name\Radio Tuner 2.

Adding MTC command visual element

Path: ***Configurator*** → ***Visualization*** → ***Media Living*** page → ***Unlock current plan for editing***.

Ensure no visual elements are selected in the page/plan.

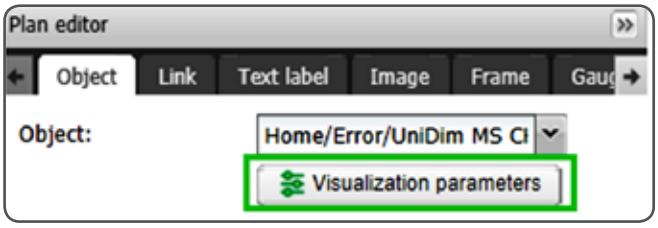
In Plan editor, select the ***Object*** tab.

At the ***Object*** dropdown, select ***Home/Media Transport/Streaming Audio/MTC Command (0/192/0/26) - Living***.

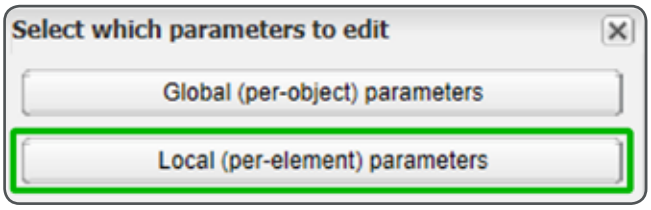
At Display mode select ***Icon***.

Select the preferred ***Default Icon*** from the dropdown.

Click ***Visualization parameters***.

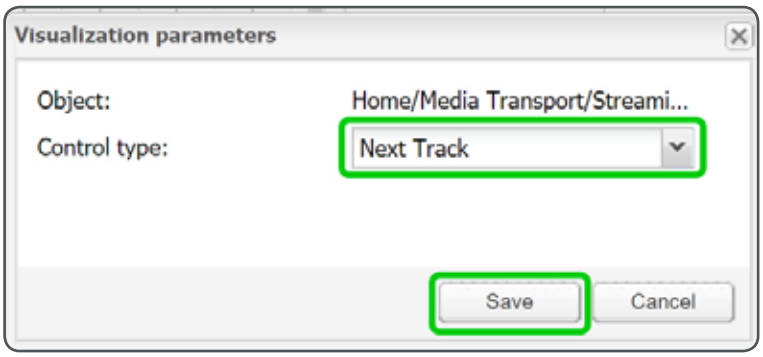


At ***Select which parameters to edit*** window click ***Local (per-element) parameters***.



At ***Visualization parameters*** window click ***Control Type*** box ***Next Track***.

Click ***Save.***

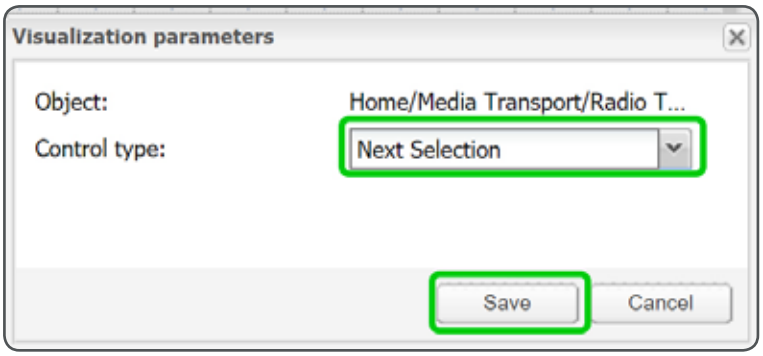


In Plan Editor, click ***Add to plan***.

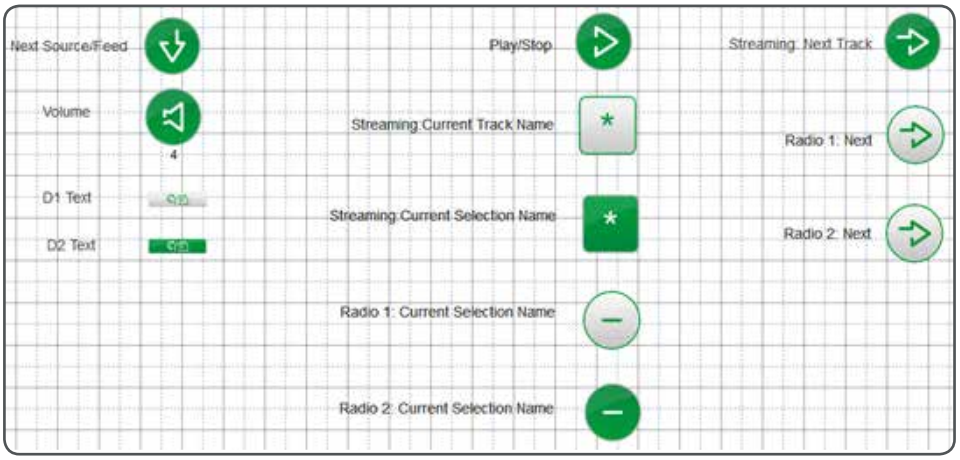
Drag the icon to a desired and add a text label ***Next Track***.

Repeat the steps to add visualization elements for the objects:

- Home/Media Transport/Radio Tuner 1/MTC Command (0/192/1/26)
- Home/Media Transport/Radio Tuner 2/MTC Command (0/192/2/26). In Visualization parameters for this object change the ***Control Type*** to ***Next Selection***.



When the example project plan displays similar to this plan, click ***Save and reload plan***.



4.3.7 Events and notifications

C-Bus error reporting can be added to your Automation Controller through use of the C-Bus Error application. The Error Application is used for monitoring events and notifications in the C-Bus system.

Table 21 shows an office with 2 C-Bus devices that support error reporting (two 4-channel universal dimmer with device ID 7 and device ID 8).

Table 21: *C-Bus Error report enabled device in an office*

Installed location	C-Bus device
Reception	universal dimmer channel 1
File Room	universal dimmer channel 2
Shipping Room	universal dimmer channel 3
Lunch Room	universal dimmer channel 4
VP Office	universal dimmer channel 1
Sales	universal dimmer channel 2

In the Automation Controller there are two types of error devices: physical and virtual.

- A physical device refers to a tangible device. The universal dimmer channel 1 is a physical device.
- A virtual device refers to an object that is made by logic, and will never send error reports.

Virtual devices are used to aggregate errors from multiple channels and devices. There are three types of virtual devices:

- **Project** – Aggregate any error seen by the Automation Controller
- **Network** – Aggregate any error seen on the associated network by the Automation Controller
- **Device** – Aggregate any error seen for the associated system ID and device ID by the Automation Controller.

Each of the physical/virtual object support ***most severe*** and ***most recent*** error types. Each physical/virtual object has a unique address assigned to it which is used to refer to it in the Automation Controller for scripting. Note that the address has no meaning in and of itself and is in no way related to a group address or other C-Bus address. ***Most recent*** error messages reflect the present status of the error condition. ***Most severe*** error messages reflect the most severe error condition which has existed since the condition was last cleared. A ***most severe*** error can be cleared using *CBusClearMostSevereError()* function in Lua.

Table 22 shows the physical objects created for this example project in the Automation Controller, assuming the local network (***net ID*** = *0*) is tagged as *Office*.

Table 22: *Physical error objects*

Name	Object Address	Most recent/ Most severe	Group Address	Device Type	Device ID	System ID	Channel
Office/Error/ UniDim MS CH 1	2	Most Severe	0/206/2	Physical	7	Universal Dimmer	1
Office/Error/ UniDim MR CH 1	3	Most Recent	0/206/3	Physical	7	Universal Dimmer	1
Office/Error/ UniDim MS CH 2	4	Most Severe	0/206/4	Physical	7	Universal Dimmer	2

Name	Object Address	Most recent/ Most severe	Group Address	Device Type	Device ID	System ID	Channel
Office/Error/ UniDim MR CH 2	5	Most Recent	0/206/5	Physical	7	Universal Dimmer	2
Office/Error/ UniDim MS CH 3	6	Most Severe	0/206/6	Physical	7	Universal Dimmer	3
Office/Error/ UniDim MR CH 3	7	Most Recent	0/206/7	Physical	7	Universal Dimmer	3
Office/Error/ UniDim MS CH 4	8	Most Severe	0/206/8	Physical	7	Universal Dimmer	4
Office/Error/ UniDim MR CH 4	9	Most Recent	0/206/9	Physical	7	Universal Dimmer	4
Office/Error/ UniDim 2 MS CH 1	18	Most Severe	0/206/2	Physical	8	Universal Dimmer	1
Office/Error/ UniDim 2 MR CH 1	19	Most Recent	0/206/3	Physical	8	Universal Dimmer	1
Office/Error/ UniDim 2 MS CH 2	20	Most Severe	0/206/4	Physical	8	Universal Dimmer	2
Office/Error/ UniDim 2 MR CH 2	21	Most Recent	0/206/5	Physical	8	Universal Dimmer	2

Table 23 shows the virtual error objects to be used.

Table 23: *Virtual error objects*

Name	Variable Address	Most recent/ Most severe	Group Address	Device Type	Device ID	System ID	Channel
Office/Error/ UniDim MS	0	Most Severe	0/206/0	Virtual Unit	7	NA	NA
Office/Error/ UniDim MR	1	Most Recent	0/206/1	Virtual Unit	7	NA	NA
Office/Error/ UniDim 2 MS	10	Most Severe	0/206/10	Virtual Network	NA	NA	NA
Office/Error/ UniDim 2 MR	11	Most Recent	0/206/11	Virtual Network	NA	NA	NA
Office/Error/ WholeNet MS	14	Most Severe	0/206/13	Virtual Network	NA	NA	NA
Office/Error/ WholeNet MR	15	Most Recent	0/206/14	Virtual Network	NA	NA	NA
Office/Error/ WholeProject MS	16	Most Severe	0/206/15	Virtual Project	NA	NA	NA
Office/Error/ WholeProject MR	17	Most Recent	0/206/16	Virtual Project	NA	NA	NA

Table 23 on page 81 shows that *Office/Error/UniDim*, *Office/Error/WholeNet*, and *Office/Error/WholeProject* are all virtual devices as their values are only updated by logic and based on their children's value. For instance, *Office/Error/UniDim MS* has 4 physical children:

- Office/Error/UniDim MS CH 1
- Office/Error/UniDim MS CH 2
- Office/Error/UniDim MS CH 3
- Office/Error/UniDim MS CH 4.

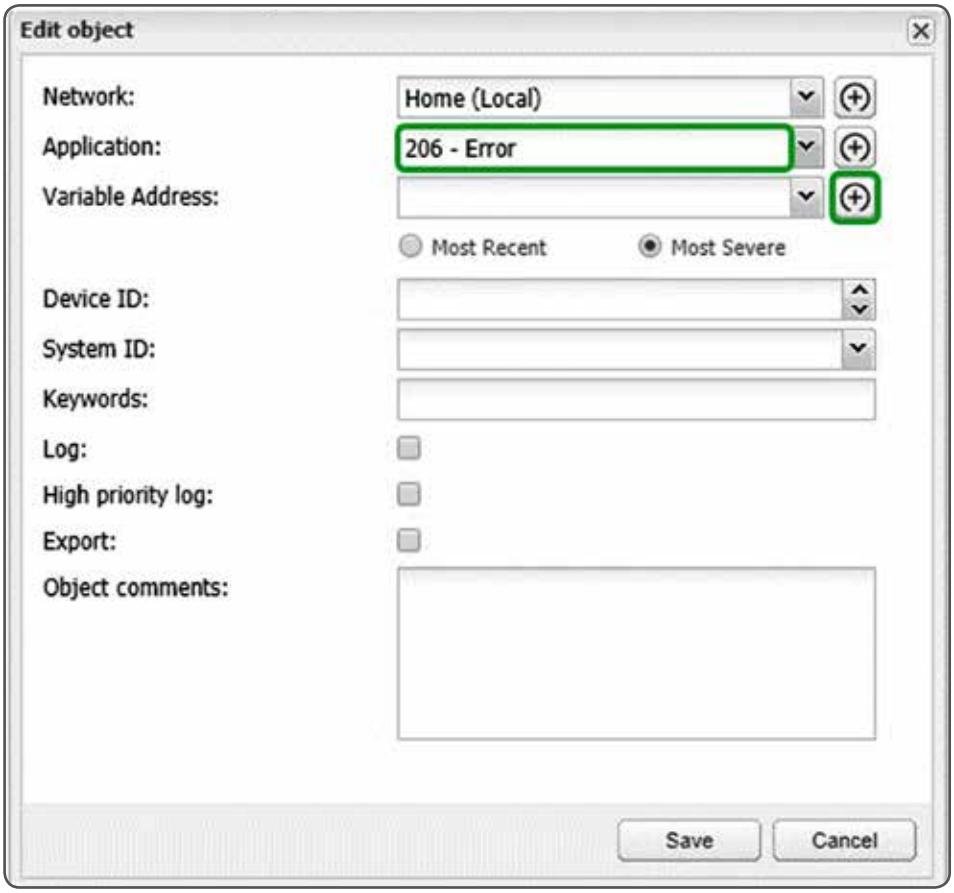
Similarly, *Office/Error/WholeNet MS* has 2 children: *Office/Error/UniDim MS* and *Office/Error/Key Unit MS*. The following example project will detail how to add them into Automation Controller.

Example project: Creating related error objects

Path: ***Configurator*** → ***Objects*** tab → ***Add new object*** button.

In the ***Edit object*** window, at ***Application*** select *Error (206)* from the drop down list.

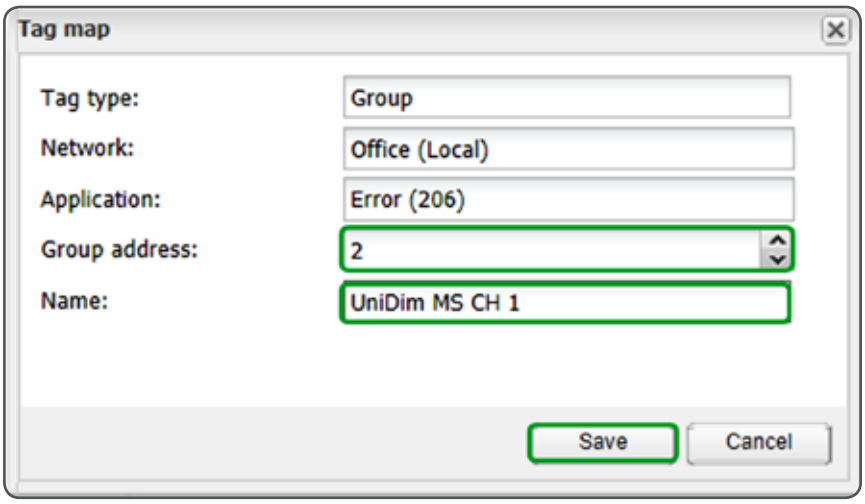
At ***Variable Address*** click the + sign to add a variable address.



In the ***Tag Map*** window enter ***Group address*** = *2*.

In the ***Name*** box enter *UniDim MS CH 1*.

Click ***Save***.



Check the radio box next to ***Most Severe***.

At ***Device type*** select *Physical*.

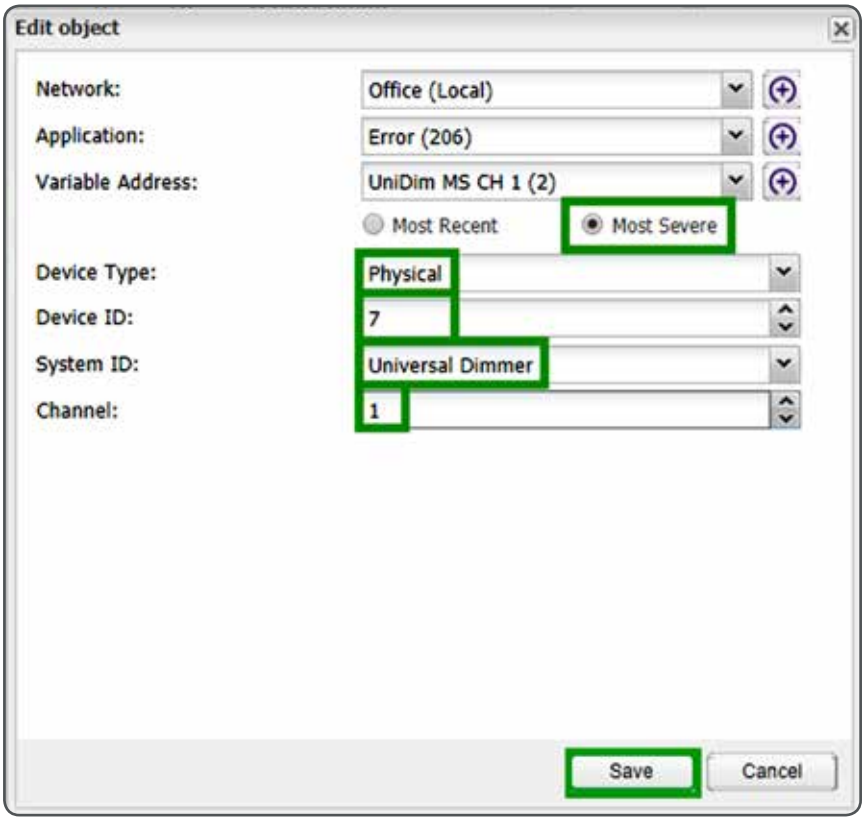
At ***Device ID*** select *7*. Refer to the reporting unit configuration to match with the correct device ID.

Device ID is the ID number of the devices sending the message (this is often, but not necessarily the same as the unit address)

At ***System ID*** select *Universal Dimmer*.

At ***Channel*** select *1*.

Click ***Save***. The first error object is now created.



Repeat the steps until all physical error objects in *Table 22* on page 80 are created, and then repeat the steps until all virtual error objects in *Table 23* on page 81 are created.

The finished objects are shown below.

Name	Group address
Home/Error/UniDim MS	0/206/0
Home/Error/UniDim MR	0/206/1
Home/Error/UniDim MS CH 1	0/206/2
Home/Error/UniDim MR CH 1	0/206/3
Home/Error/UniDim MS CH 2	0/206/4
Home/Error/UniDim MR CH 2	0/206/5
Home/Error/UniDim MS CH 3	0/206/6
Home/Error/UniDim MR CH 3	0/206/7
Home/Error/UniDim MS CH 4	0/206/8
Home/Error/UniDim MR CH 4	0/206/9
Home/Error/UniDim 2 MS CH 1	0/206/10
Home/Error/UniDim 2 MR CH 1	0/206/11
Home/Error/UniDim 2 MS CH 2	0/206/12
Home/Error/UniDim 2 MR CH 2	0/206/13
Home/Error/UniDim 2 MS	0/206/14
Home/Error/UniDim 2 MR	0/206/15
Home/Error/WholeNet MS	0/206/16
Home/Error/WholeNet MR	0/206/17
Home/Error/WholeProject MS	0/206/18
Home/Error/WholeProject MR	0/206/19

Preparing icons

Each object shown in *Table 22* on page 80 and *Table 23* on page 81 may have both error level and severity level.

All C-Bus devices that support error reporting have five severity levels. The example in universal dimmer has seven error levels and key unit has no error levels.

View existing icons and add related icons if necessary. Prepare three icons for five severity levels:

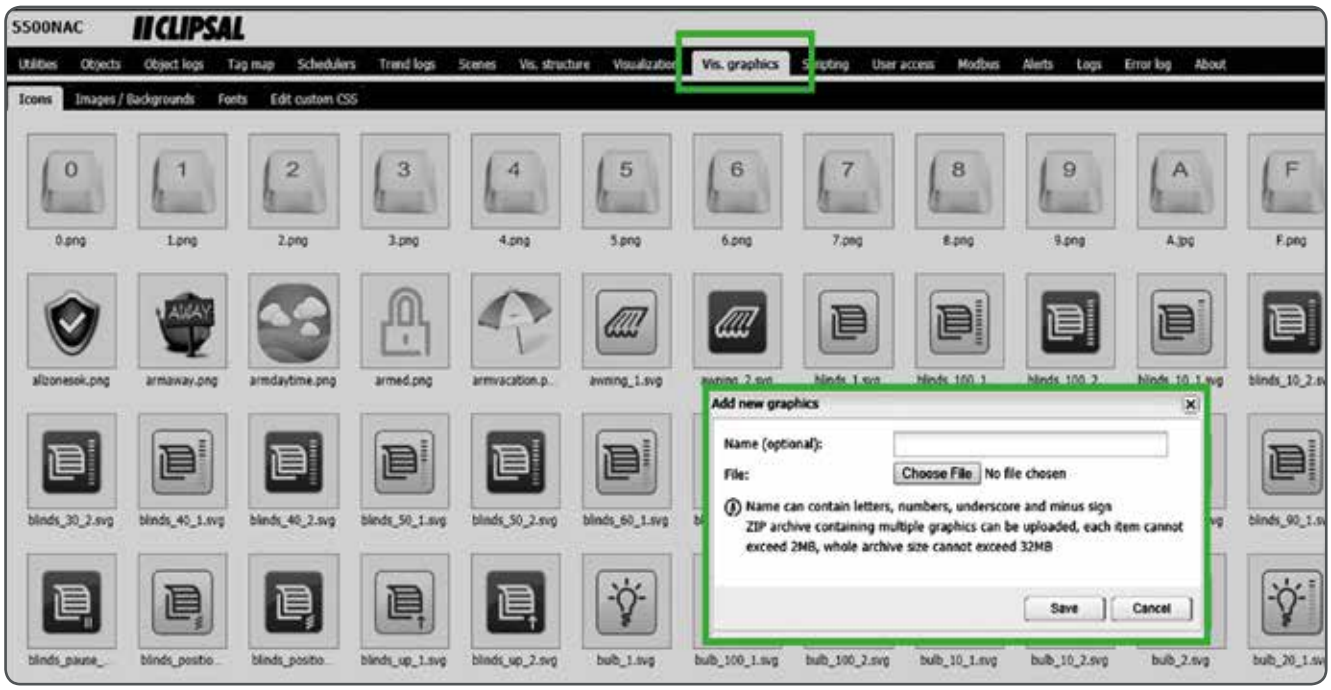
- Green light for ***All OK*** and ***OK***
- Yellow light for ***Minor Failure***
- Red light for General Failure and ***Extreme Failure***.

Prepare seven icons ready for the seven error levels of the universal dimmer example

Path: ***Configurator*** → ***Vis. Graphics*** tab.

Repeat until all required icons are added.

If more icons are needed, click the ***Add icons*** button at bottom left.



Prepare image

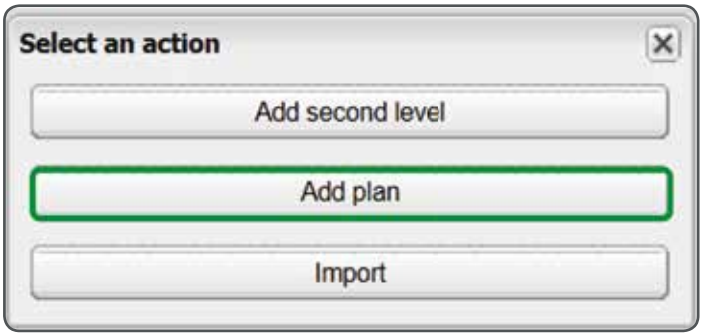
Path: ***Configurator*** → ***Vis. Graphics*** tab → ***Images/Backgrounds*** tab.

Click the ***Add images*** button to add a floor plan.

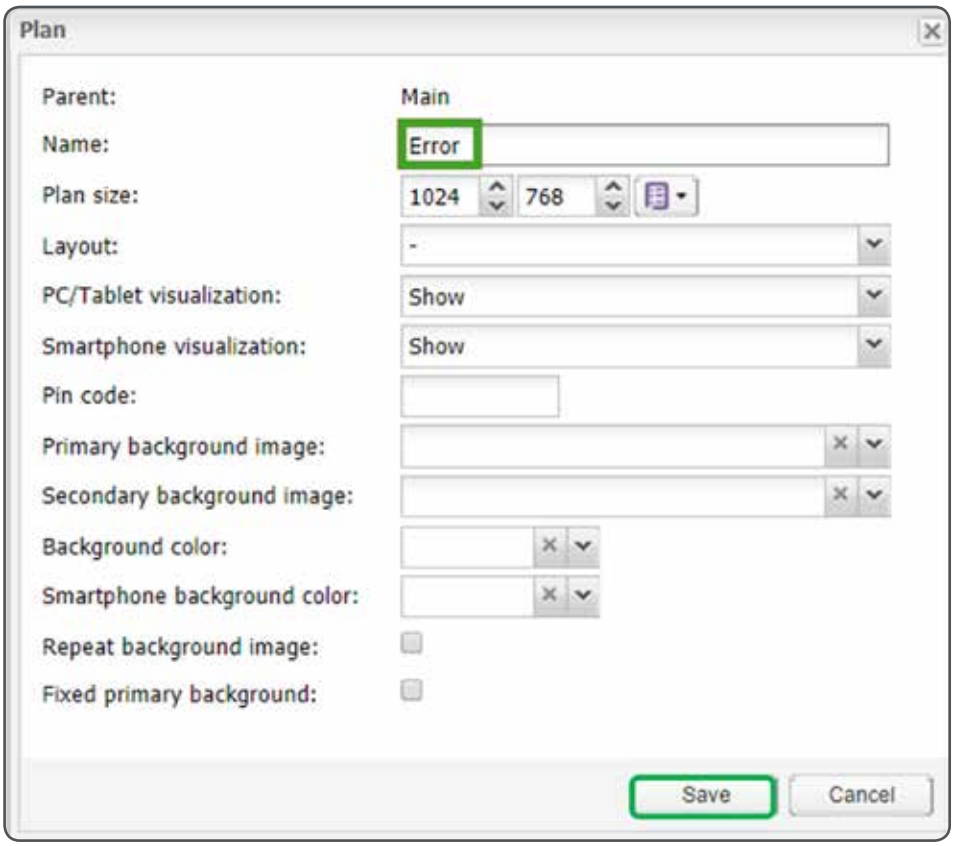
Add an error user interface/page plan

Path: ***Configurator*** → ***Vis. Structure*** tab → Add/Import .

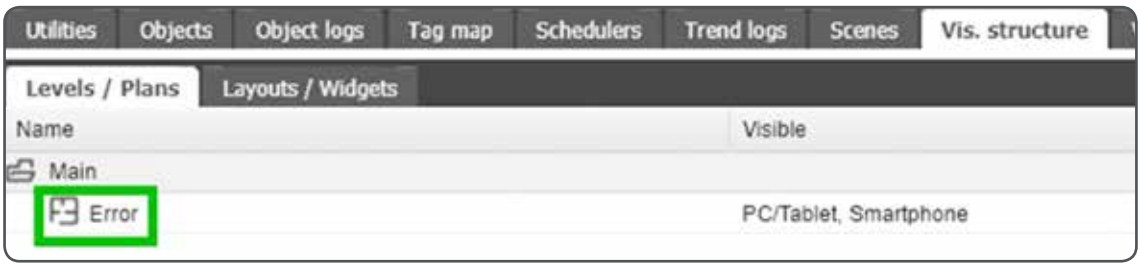
When the ***Select an action*** box appears, select ***Add plan***.



When the ***Plan*** window appears, enter *Error* in the ***Name*** field and then click ***Save***.



The Error page will now be visible in the ***Vis. structure*** tab.



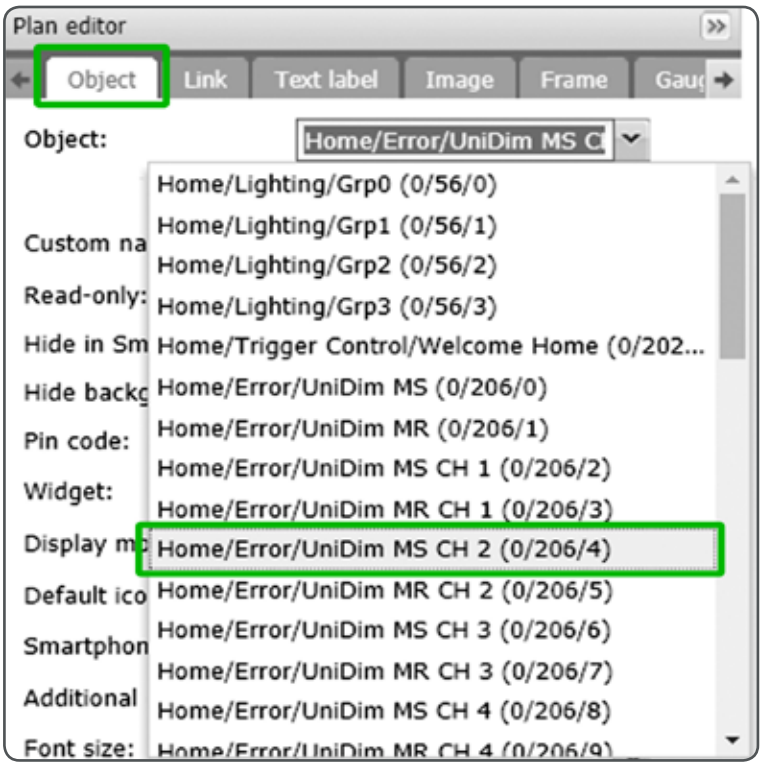
Creating visualization elements for status and error messages on error page/plan

Path: ***Configurator*** → ***Visualization*** tab → ***Error page*** from the structure view.

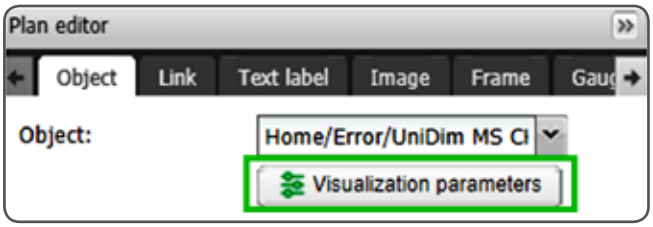
Click ***Unlock current plan for editing***.

Click the ***Object*** tab in the right-hand panel of the Plan Editor.

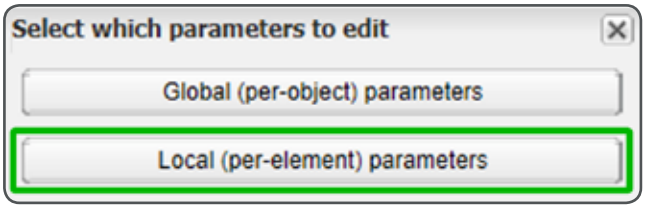
At ***Object***, select *Home/Error/UniDim MS CH 1 (0/206/2)* from the drop-down list.



Click the ***Visualization parameters*** button.



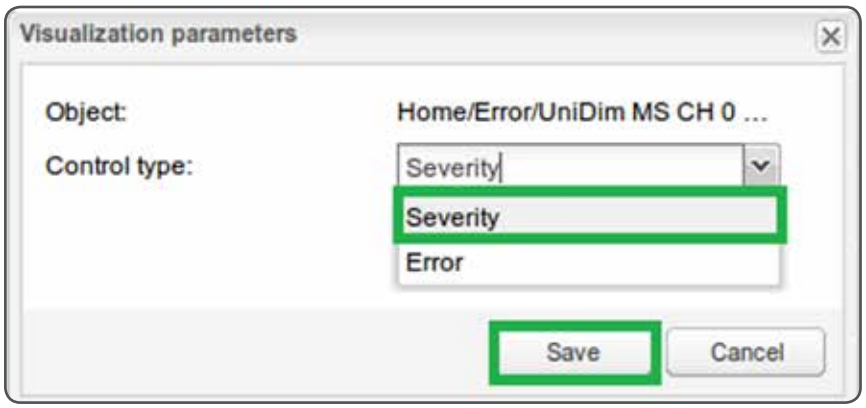
At ***Select which parameters to edit*** window click ***Local (per-element) parameters***.



Each universal dimmer channel object has error and severity information (level values and text messages).

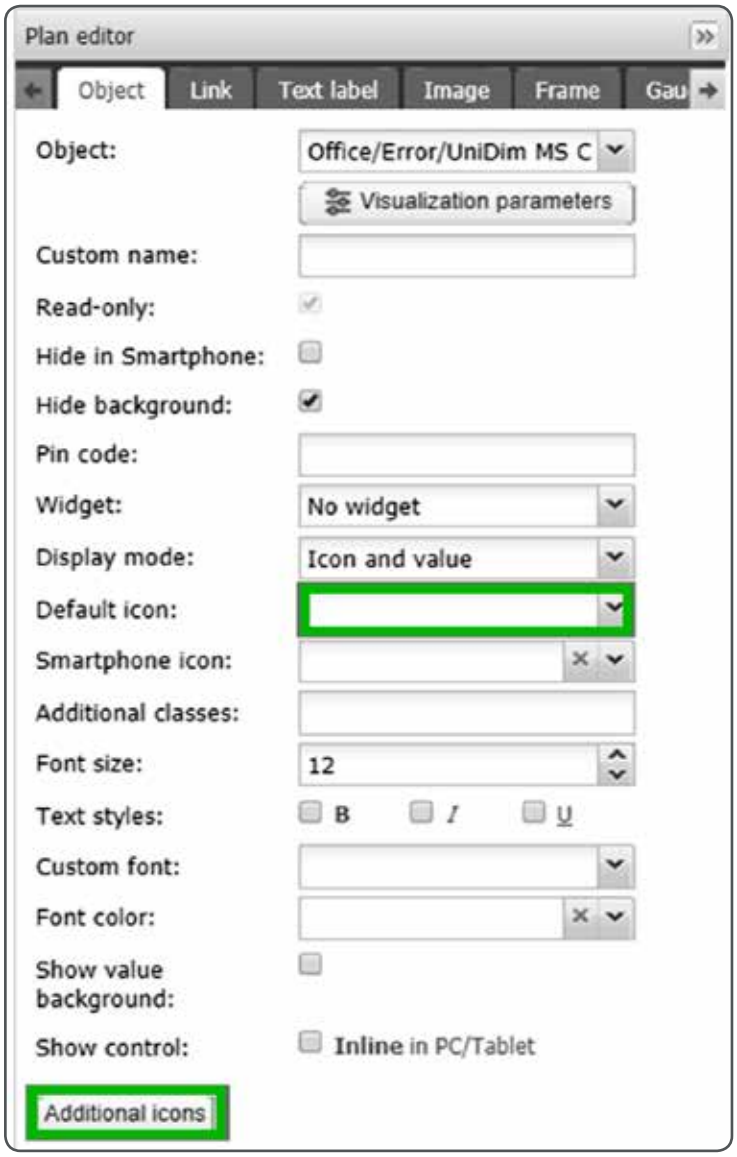
Set ***Control type*** as *Severity*.

Click the ***Save*** button.



In the ***Object*** tab of the Plan Editor, select a ***Default icon*** from the dropdown selection.

Click ***Additional icons***.

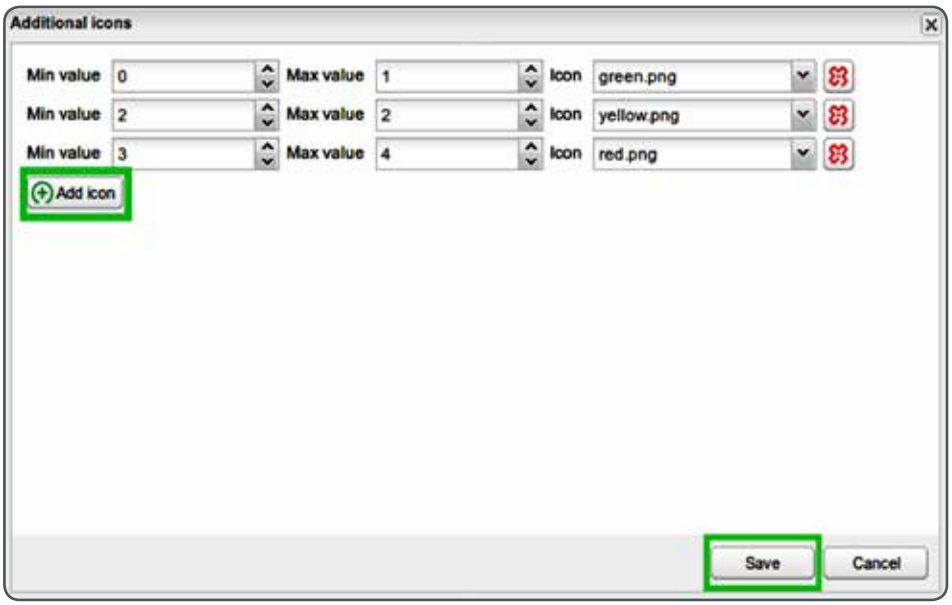


From the ***Additional Icons*** window, select *Add icon*.

Set the ***Min Value*** and ***Max Value***.

Select the required icon from the ***Icon*** drop down, and repeat for all icons as required.

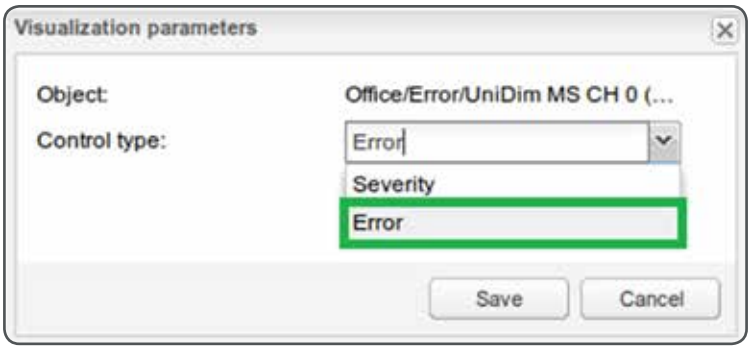
When all objects have been assigned an icon and values, click ***Save***.



In the Plan Editor, select the ***Object*** tab, then click ***Unlock current plan for editing***.

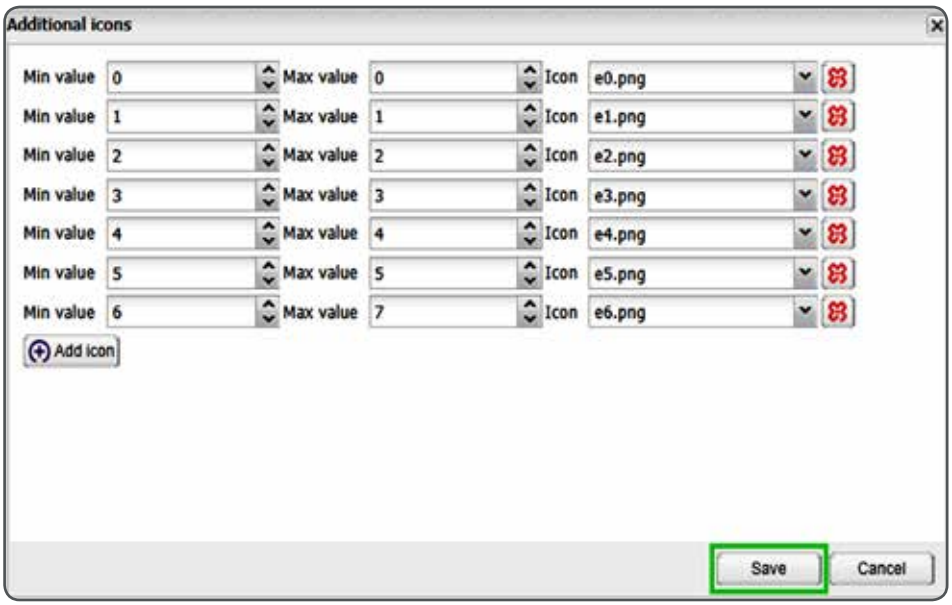
Click ***Add to plan*** to add new visual elements to the page.

Repeats steps to add ***Error*** type to *Office/Error/UniDim MS CH 1 (0/206/2)*. Select ***Control Type*** as *Error* in the ***Visualization parameters > Local (per-element) parameters*** window.



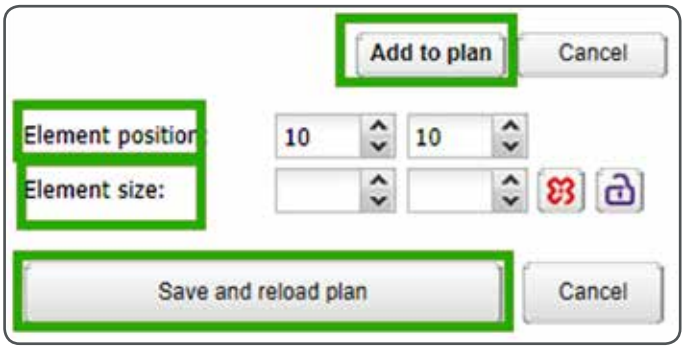
Change the ***Default*** icon and modify the Additional icons as well.

Click ***Save*** when complete.



Drag the elements and resize if desired.

Click ***Apply*** and then ***Save and reload plan***.



Add a floor plan as the background

Path: ***Configurator*** → ***Visualization*** tab → ***Error page*** from the structure view.

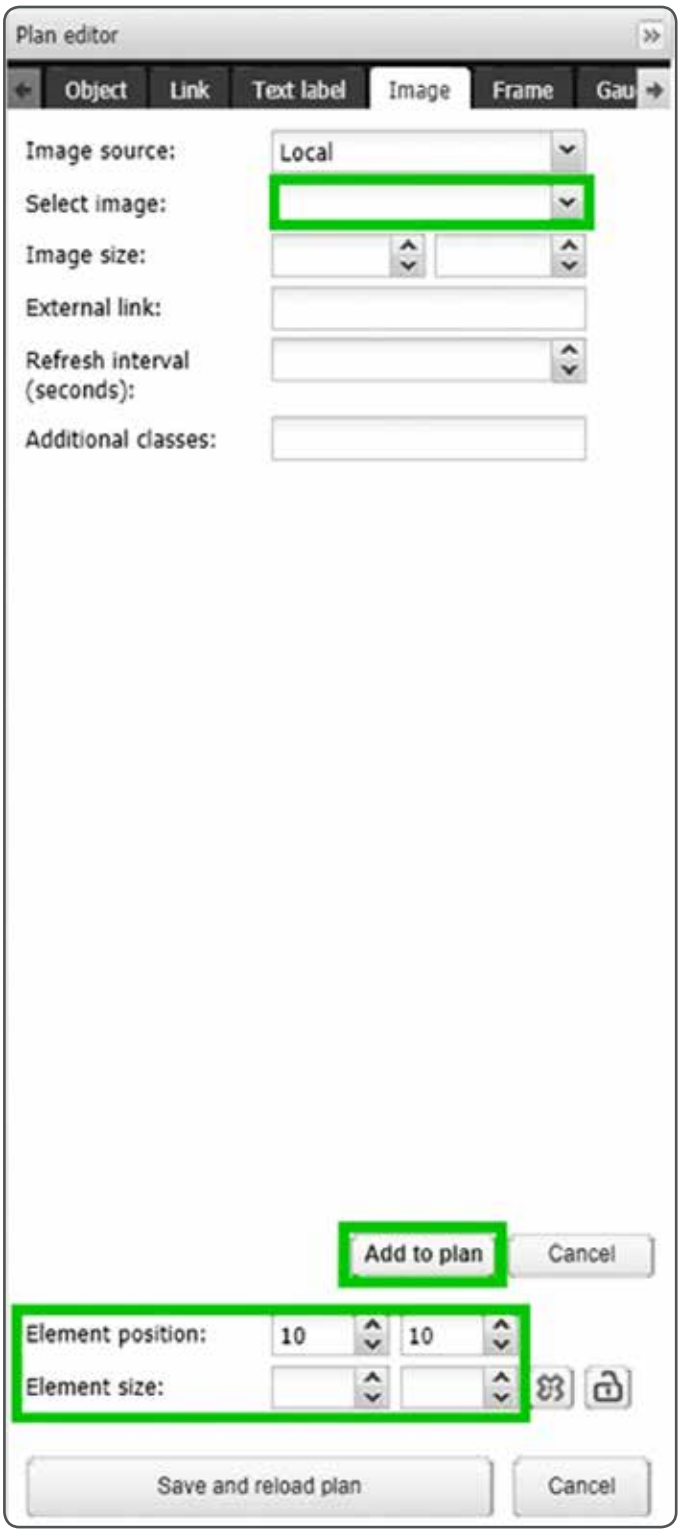
Click the ***Unlock current plan for editing***. Ensure no visual elements are selected in the page/plan.

In the Plan editor, select the ***Image*** tab.

At the ***Select image*** dropdown, select the required floor plan.

Change the ***Element position*** and ***Element size*** if required.

Click ***Add to plan***.



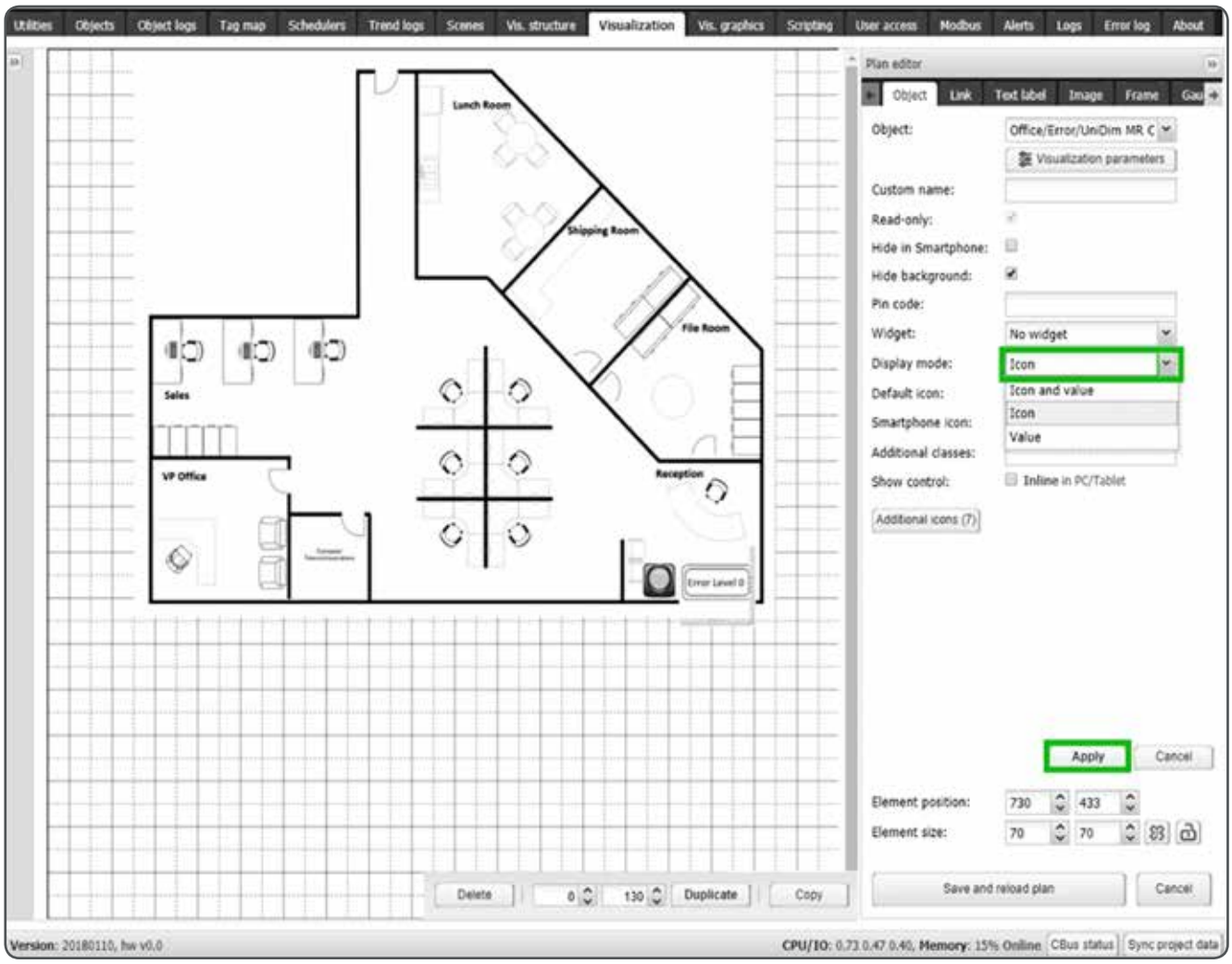
Rearrange the elements by dragging them to Reception in the floor plan.

Change their ***Element*** size if necessary.

To only display the icon on the floor plan, select the visual element and choose ***Display Mode*** = *Icon* for both elements.

Click ***Apply*** after finalising the changes.

Click ***Save and reload plan***.



Adding object elements

Add four visual elements (error level and severity level) for both the ***Most severe*** object *Office/Error/UniDim MS CH 1* and ***Most recent*** object *Office/Error/UniDim MR CH 1* for Universal dimmer channel 1 on top of the Reception area. In the example, *Table 24* shows that we have four visual elements for Reception and two for the restrooms.

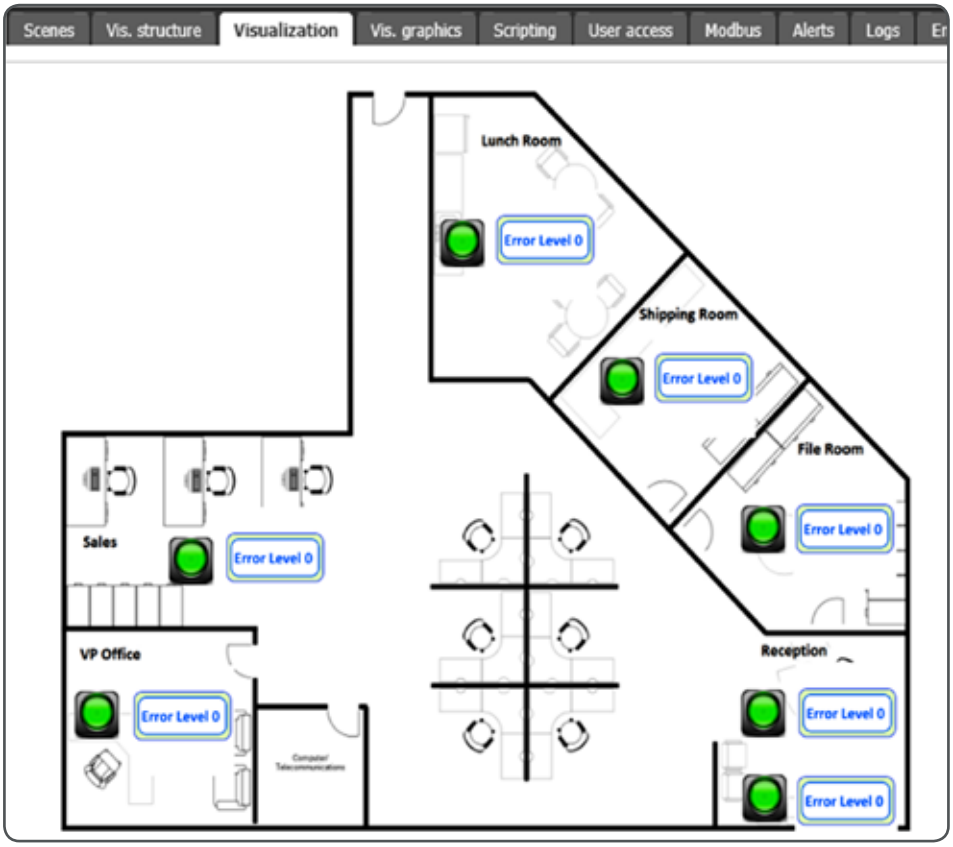
Table 24: *Visual elements for the reception and restroom areas*

Installed Location	C-Bus Device	Object Used	Most Severe / Most Recent
Reception	universal dimmer channel 1	Office/Error/UniDim MS CH 1	Most Severe
		Office/Error/UniDim MR CH 1	Most Recent
File Room	universal dimmer channel 2	Office/Error/UniDim MR CH 2	Most Recent
Shipping Room	universal dimmer channel 3	Office/Error/UniDim MS CH 3	Most Severe
Lunch Room	universal dimmer channel 4	Office/Error/UniDim MR CH 4	Most Recent
VP Office	universal dimmer2 channel 1	Office/Error/UniDim 2 MS CH 1	Most Severe
Sales	universal dimmer2 channel 2	Office/Error/UniDim 2 MS CH 1	Most Severe

Add the remaining Object elements in *Table 24*. Each object will have two visual elements, resulting in 14 visual elements for the seven objects.

Use the ***Duplicate*** button and then modify the copied elements to greatly speed up your task.

Arrange the icons based on *Table 24* on the floor plan image.



Adding virtual objects to the plan page

Add the virtual objects shown in *Table 25* to the plan/page, including the virtual device objects of: virtual unit; virtual network; and virtual project.

Table 25: *Virtual objects for virtual unit, virtual network and virtual project*

C-Bus Device	Object Used	Most Severe/Most Recent
Virtual Unit	Office/Error/UniDim MR	Most Recent
	Office/Error/UniDim MS	Most Severe
Virtual Network	Office/Error/WholeNet MR	Most Recent
	Office/Error/WholeNet MS	Most Severe

Based on *Table 25*, the following visualization shows all visual elements for this example.

Add text labels on the virtual device objects

To add a text label, click the ***Text label*** tab.

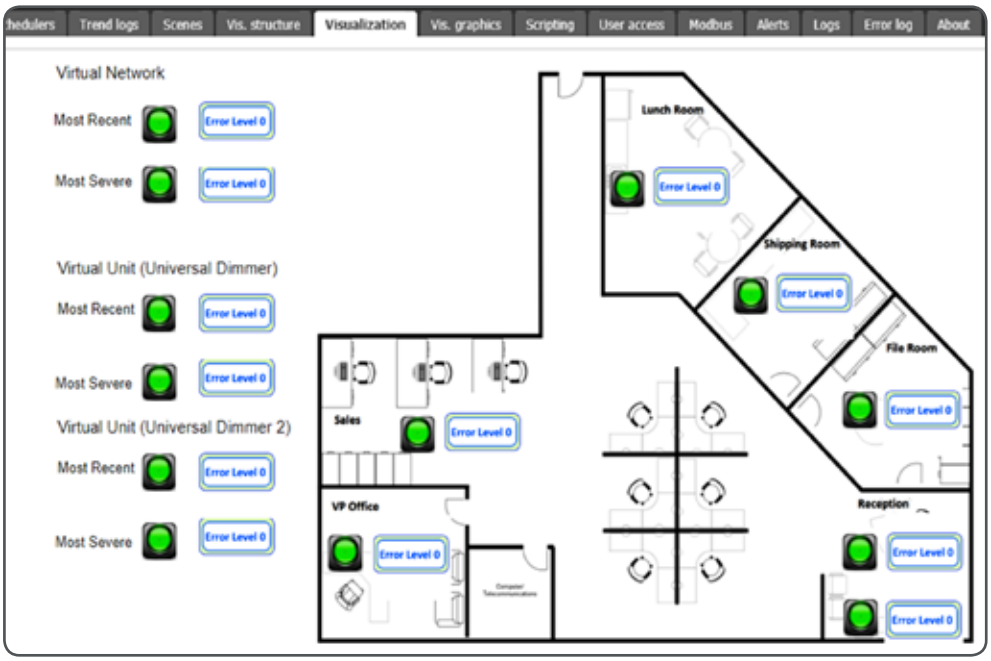
Enter text in the ***Text*** field.

Click the ***Add to plan*** button.

Click ***Save and reload plan***.

Drag the text to the proper location and adjust the size using the ***Font size*** field.

The finished page might look like this visualization.



Add error message text for error objects

Now that the icons to indicate the status of ***Most severe*** and ***Most recent*** have been entered, add the error objects to the page, including the Error text message shown in *Table 26*.

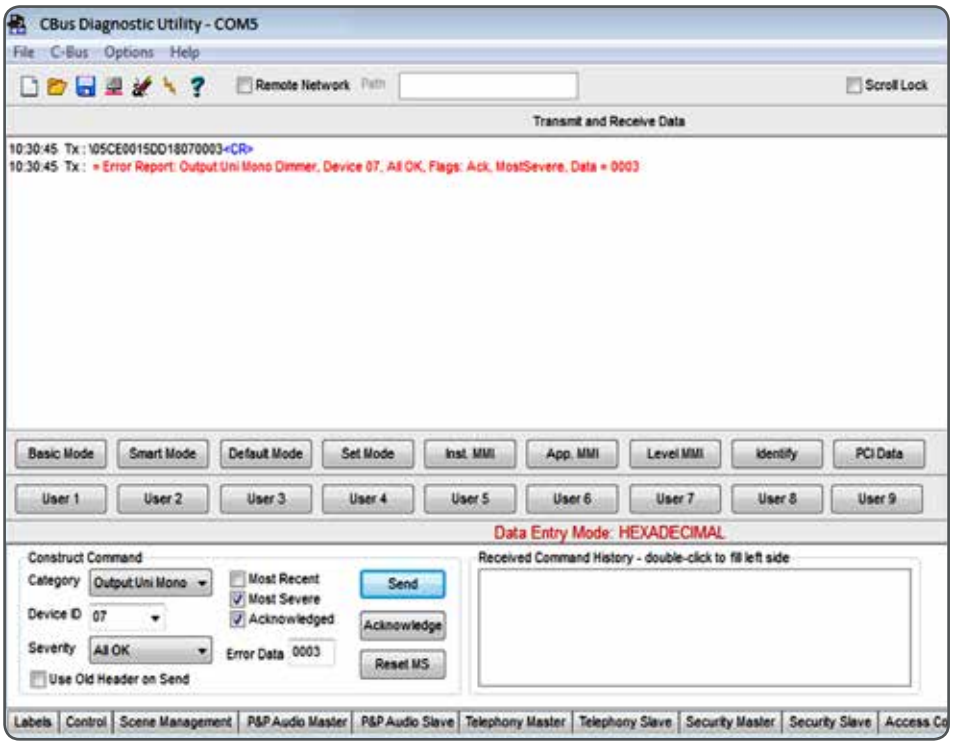
Table 26: *Error objects for locations used in the example*

Installed Location	C-Bus Device	Object Used	Most Severe / Most Recent
Reception	Universal dimmer channel 1	Office/Error/UniDim MS CH 1	Most Severe
		Office/Error/UniDim MR CH 1	Most Recent
File Room	Universal dimmer channel 2	Office/Error/UniDim MR CH 2	Most Recent
		Office/Error/UniDim MS CH 2	Most Severe
Lunch Room Shipping Room File Room Reception VP Office	Virtual Unit (universal dimmer)	Office/Error/UniDim MR	Most Recent
		Office/Error/UniDim MS	Most Severe
All areas	Virtual Network	Office/Error/WholeNet MR	Most Recent
		Office/Error/WholeNet MS	Most Severe

If continuing from the previous step, exit edit page/plan mode by clicking ***Save and reload plan***.

If there is no connection to the physical devices, connect them to get values for the objects.

An alternative is to use the C-Bus Diagnostic Utility to send an error message for each of the physical objects. The next screenshot shows how to use the C-Bus Diagnostic Utility to send an error message for *Office/Error/UniDim MS CH 1*.



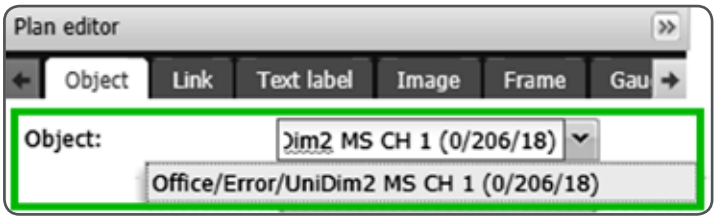
Add error message for severity of objects

Path: ***Configurator*** → ***Visualization*** tab → ***Error page*** on structure view.

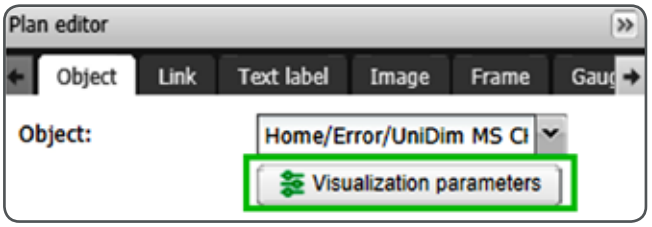
Select ***Unlock current plan for editing*** in the Plan Editor.

Make sure no visual elements on the page are selected.

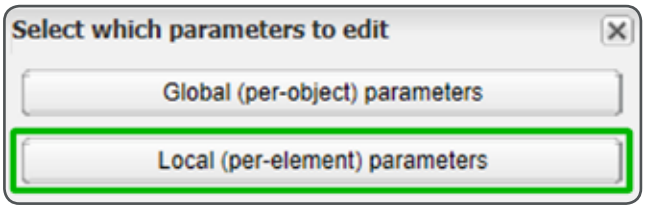
Click the ***Object*** Tab, and select ***Object*** = *Office/Error/UniDim MS CH 1*.



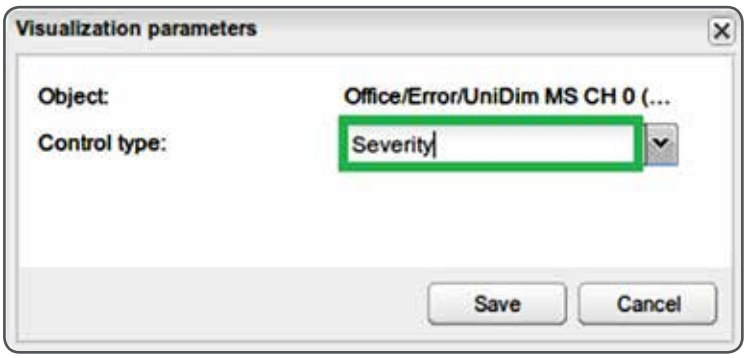
Click the ***Visualization parameters*** button.



At ***Select which parameters to edit*** window click ***Local (per-element) parameters***.



To add severity for object *Office/Error/UniDim MS CH 1*, in the ***Visualization parameters*** window, select ***Control type*** = *Severity*.



Click ***Save***.

In Plan Editor – ***Object*** tab, select ***Display mode*** = *Value* (from the drop down list).

Click ***Add to plan***.

Drag to a desired location and click ***Apply***.

In the Plan Editor, Text Label, enter ***Text*** = *All OK* to identify the icon.

Click ***Add to Plan***, then ***Save and reload plan***.

Repeat steps to add all objects in *Table 26* on page 96.

The final plan with the finished visual elements might look like this:

Create visualization elements for commands on error page/plan

Add two user parameters to clear ***Most severe*** errors and to send ***acknowledge all error objects*** with an Lua script.

Path: ***Configurator*** → ***Objects*** tab → ***Add new object***.

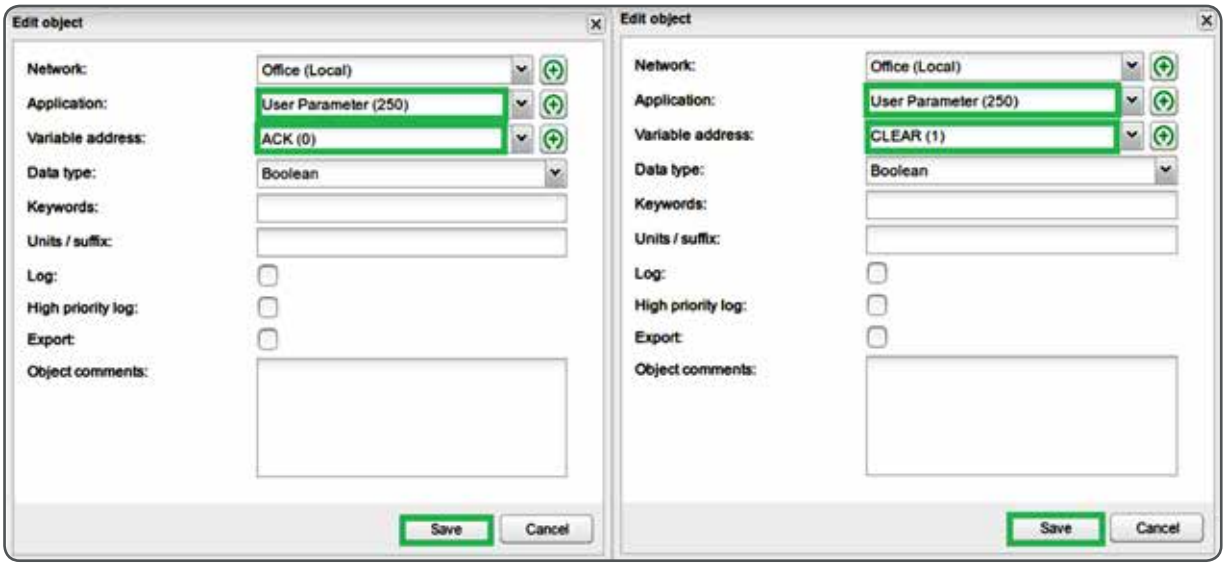
At the ***Edit object*** window, select ***Application*** = *User Parameter (250)*.

Click the plus sign next to ***Variable address*** to enter Variable address of *0*.

Enter the name *ACK*.

Click ***Save***.

Repeat the steps to add the second user parameter object with ***Variable address*** as *1* and name it *CLEAR*.



Click the icon in the Event script column to enter the event script editor.

Enter following Lua script:

```
if event.getvalue() == true then
   -- for all error object in the network 0
   for _, varAddr in ipairs(GetCBusErrorVarAddr(0)) do
     local errobj = GetCBusErrorByVarAddr(0, varAddr)
     if (errobj ~= nil and errobj.ackd == 0 and errobj.devtype == 0)
then
        CBusAckErrorByVarAddr(0, varAddr)
     end
   end
   SetUserParam(0, 0, false)
end
```

Click ***Save and close*** to exit.

Repeat these steps to add the following Lua script on the second user parameter via the event script editor:

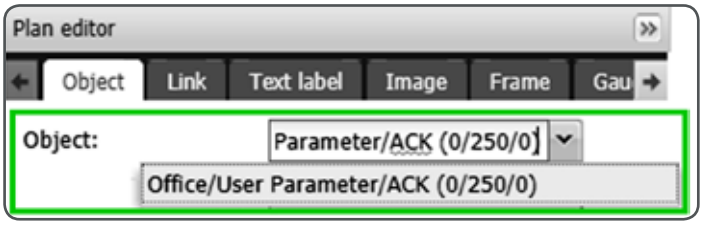
```
if event.getvalue() == true then
  -- for all error object in the network 0
  for _, varAddr in ipairs(GetCBusErrorVarAddr(0)) do
    local errobj = GetCBusErrorByVarAddr(0, varAddr)
      if (errobj ~= nil and errobj.most_recent == 0 and errobj.
devtype == 0) then
      CBusClearMostSevereErrorByVarAddr(0, varAddr)
   end
  end
  SetUserParam(0, 1, false)
end
```

Path: ***Configurator*** → ***Visualization*** tab → ***Error page*** on structure view.

Click ***Unlock current plan for editing***, making sure that no visual elements are selected.

In the Plan Editor, select the ***Object*** tab.

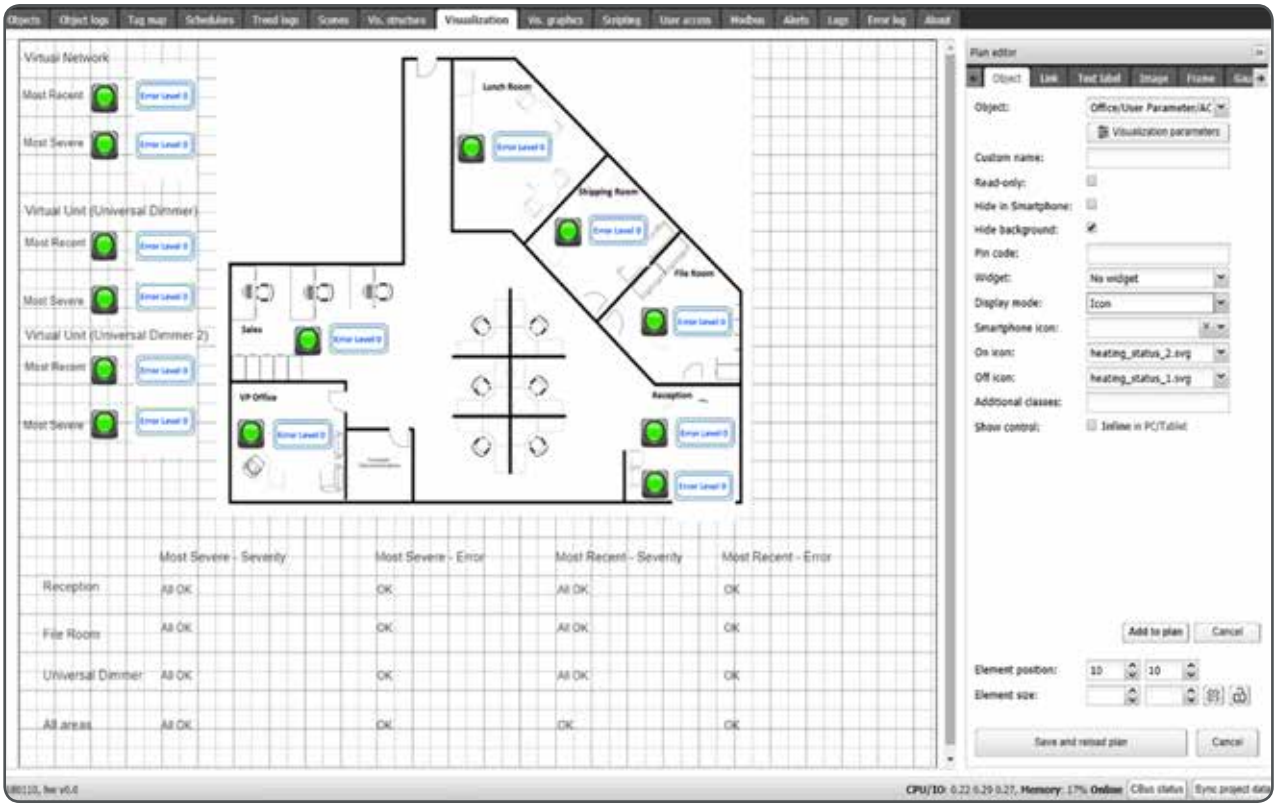
Set ***Object*** = *Office/User Parameter/ACK (0/250/0)*.



In ***Display*** mode, from the drop list select ***Icon***.

Add icons for *On* and *Off*.

Click ***Add to plan***.



Drag icons to the required location on the page, then click ***Apply***.

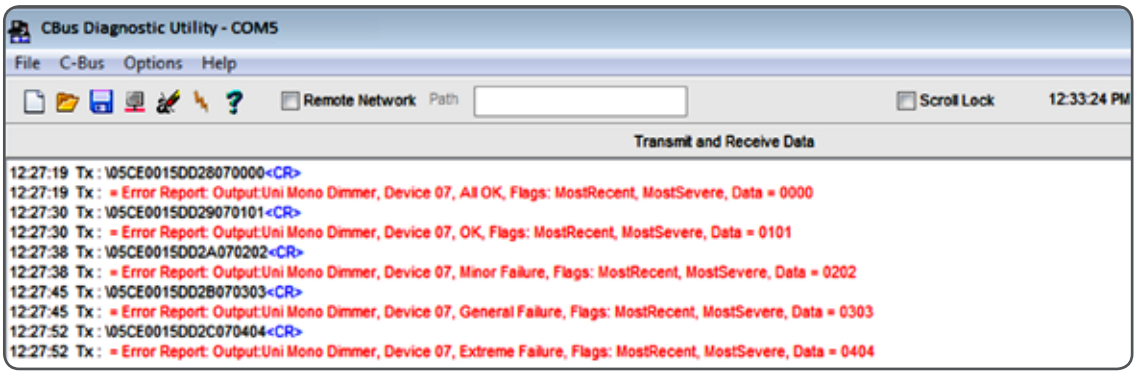
Repeat these steps to add the object *Office/User Parameter/CLEAR (0/250/1)*.

Add two text labels *ACK all* and *Clear all* in the front of the 2 buttons. See *Add text labels on the virtual device objects* on page 94.

The CBus Diagnostic Utility or a configured physical C-Bus universal dimmer/ key unit can now send an error report. When a report is received, the status icon changes and reports in the error page.

The C-Bus Diagnostic Utility sends these messages:

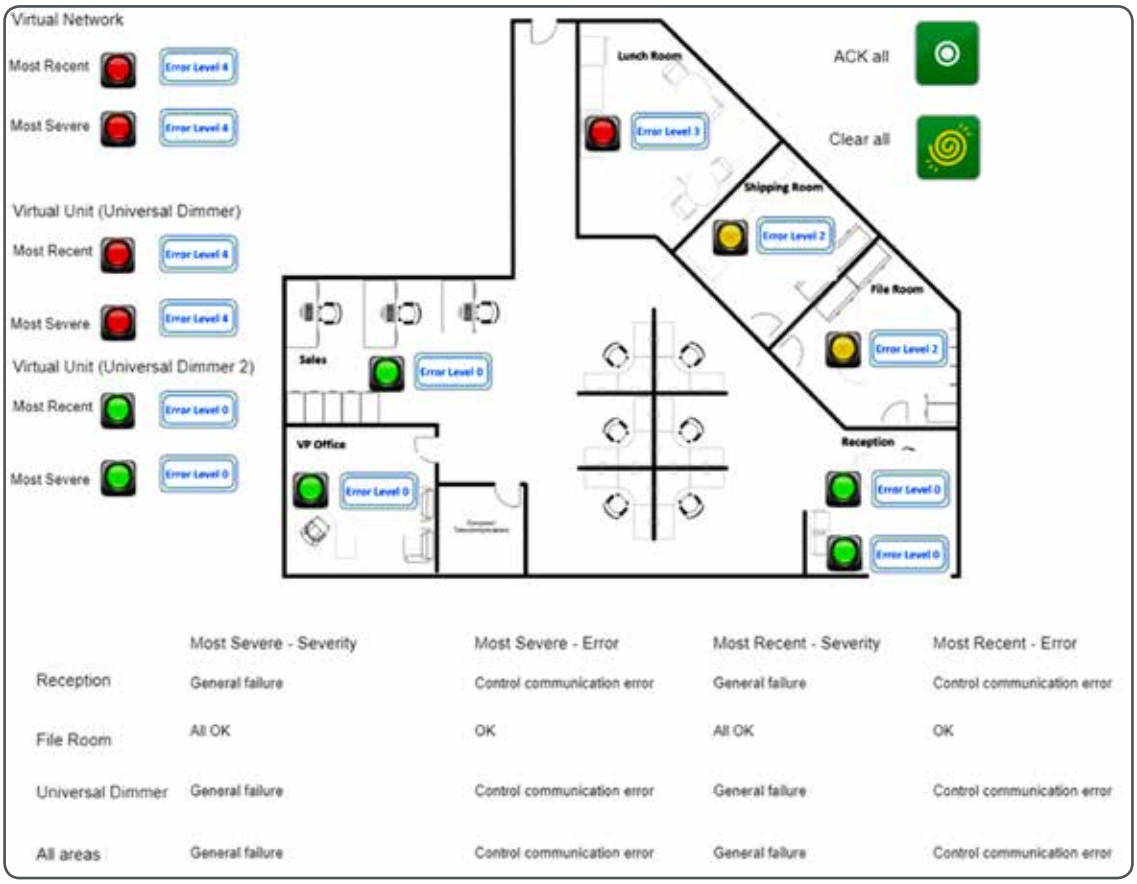
- universal dimmer channel 1 with severity level 1 and error level 1 for most severe and most recent without ACK.
- universal dimmer channel 2 with severity level 2 and error level 2 for most severe and most recent without ACK.
- universal dimmer channel 3 with severity level 3 and error level 3 for most severe and most recent without ACK.
- universal dimmer channel 4 with severity level 4 and error level 4 for most severe and most recent without ACK.



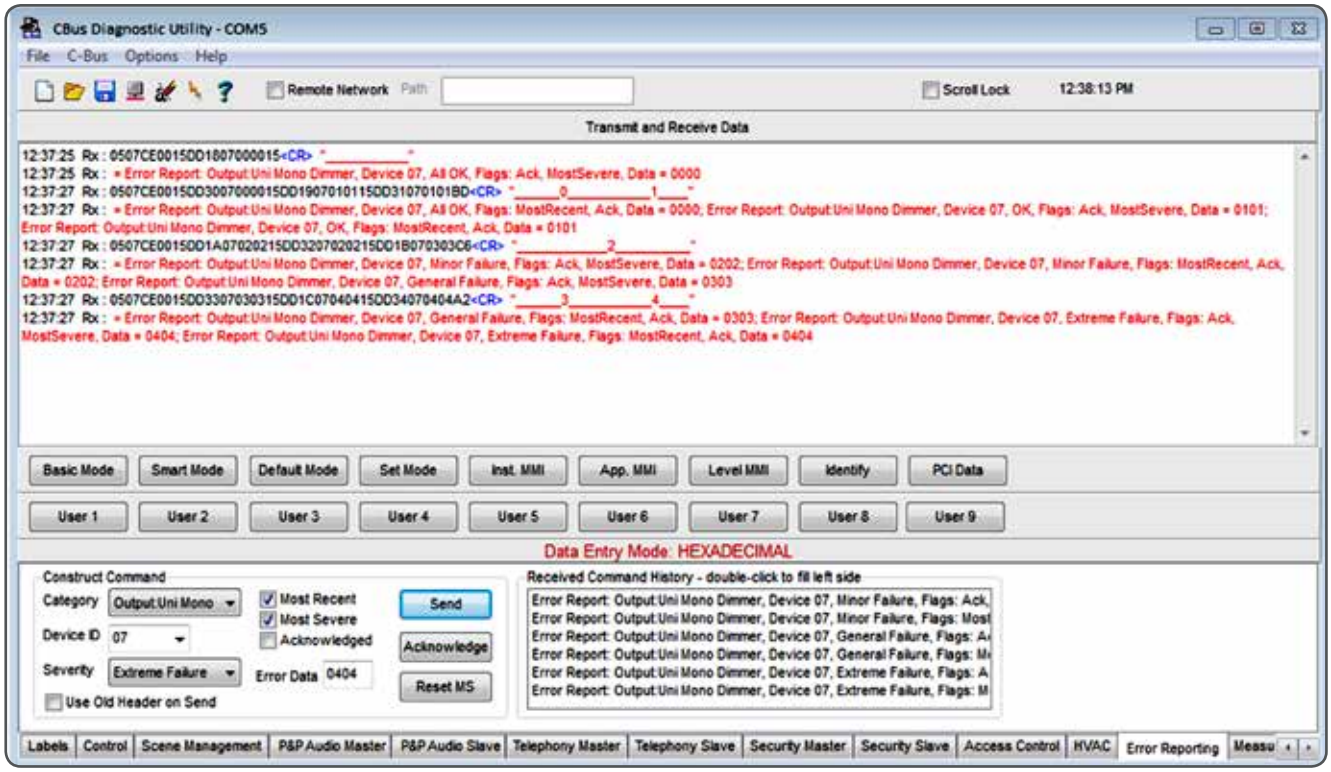
Error codes: The first two digits represent the channel and the last two digits represent the error value.

Example: *Data = 0404* is channel 4, error value 4.

After changes, the error page displays as follows:



Click the ***ACK all*** button to get ACK messages sent by the C-Bus Diagnostic Utility.



Click the ***Clear all*** button.

The Error Reporting tab can be enabled by right-clicking on the tab bar.

Example of automated email sent on error condition using event script

In this example, when a **Most Recent** error occurs at the project level, the Automation Controller will send an email to a nominated email address.

Path = ***Objects*** tab → ***Event script*** edit icon.

Enter this Lua script modifying *user@example.com* with a preferred email address.

```
errobj = event.getvalue()
if (errobj.severity <=2 or errobj.ackd == 1) then -- If severity is
low then 2 or is has not been acknowleged
  return
else

                subject = CBusFormatError(errobj)
  if (errobj.severity == 0) then
    StrSeverity = "All OK"
  elseif (errobj.severity == 1) then
    StrSeverity = "OK"
  elseif (errobj.severity == 2) then
    StrSeverity = "Minor"
  elseif (errobj.severity == 3) then
    StrSeverity = "General"
  elseif (errobj.severity == 4) then
    StrSeverity = "Extreme"
  else
    StrSeverity = "Unknown"
  end

  message = 'There is an ' .. StrSeverity .. ' ' .. errobj.error
.. ' Error has occurred on ' .. errobj.sysidstr .. ' Channel '..
errobj.channel

  mail('user@example.com', subject, message)

end
```

Click ***Save and close*** to exit.

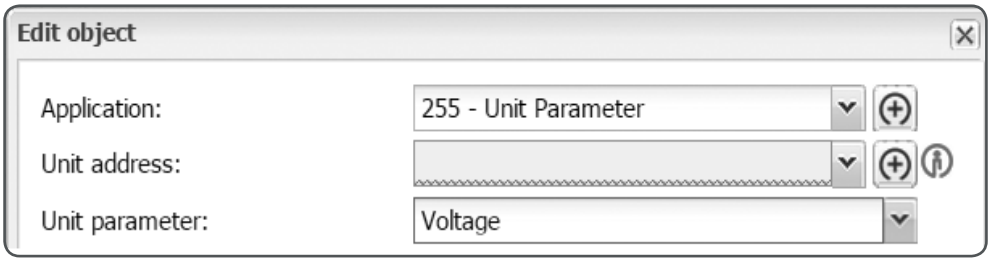
4.3.8 Add unit parameters

The *Unit parameter* application is used to get pre-set information such as the C-Bus voltage. The Automation Controller polls C-Bus units to get these values. The measured value is transferred with a 24 bit floating point number and the unit is coded with 8 bit.

Click on the ***Add new object*** button at the bottom of the ***Objects*** tab to open an *Edit object* window.

Select the unit parameter application

In the ***Application*** list, select: 255 - Unit Parameter.



Add unit address

Click the Unit address ***Add*** button to create a new *Unit address* (0–255).

Only select units that can deliver the requested value.

ID Select: 0–255.

Name Max 32 characters, except "/". The name is linked with the ID. Change the name in the ***Tag map*** tab.

Unit parameters

- Voltage (C-Bus voltage measured by a C-Bus device with a unit address)
- Light level
- Temperature.

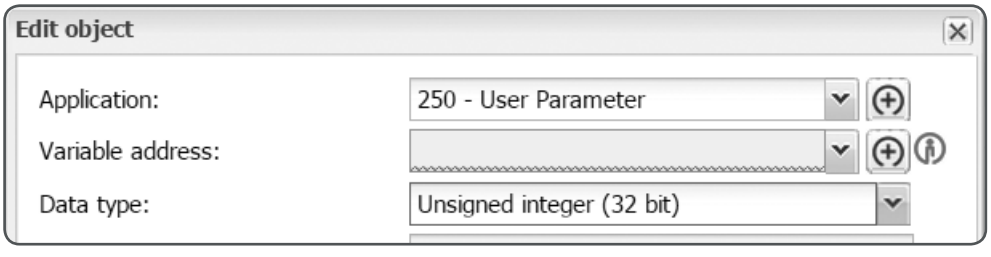
4.3.9 Add user parameters

User parameters are variables of different data types (e.g. signed integer or boolean). They can be used for visualization or for scripting. They are also used for mapping with values of Modbus Registers. They can also be exported to BACnet IP. User parameters are not exported to C-Bus Toolkit projects.

Click on the ***Add new object*** button at the bottom of the ***objects*** tab to open an *Edit object* window.

Select the user parameter application

In the Application list, select: *250 - User Parameter*.



Add a new device ID

Click the Variable address ***Add*** button to create a new *Device ID* (0–65535).

ID Select: 0–65535.

Name Max 32 characters, except "/". The name is linked with the ID. Change the name in the ***Tag map*** tab.

Select a data type

- Boolean
- Unsigned integer (32 bit)
- Signed integer (32 bit)
- Floating point (32 bit)
- RGB colour
- Time/day
- Date
- String (255 Byte).

4.3.10 Additional parameters for all applications

The following parameters can be set for all applications when adding a new object or editing an existing one.

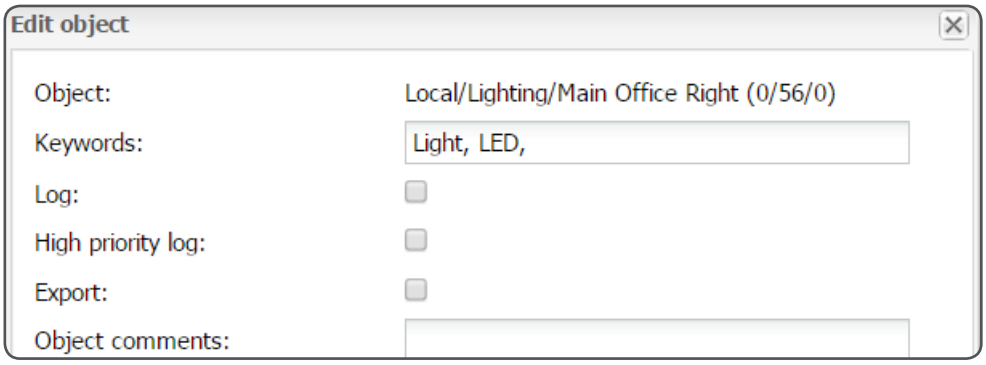
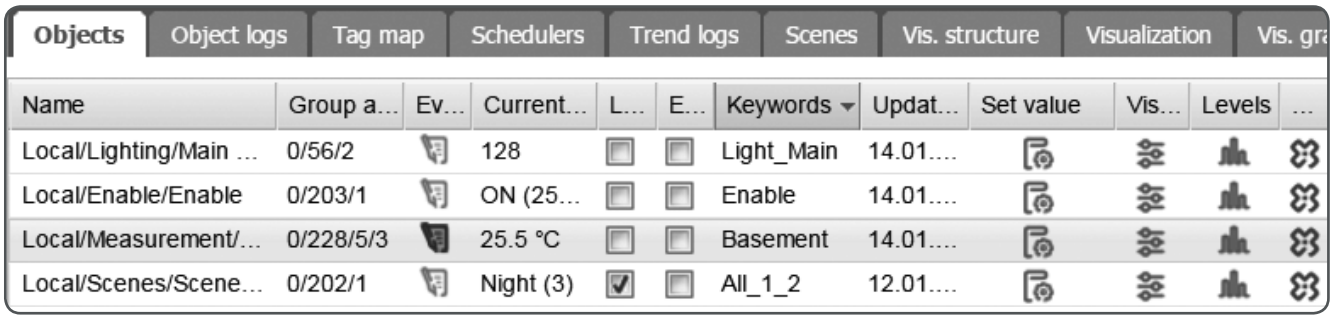


Table 27: *Additional parameters for applications*

Keywords	Keywords are assigned to the object. Keywords can be used in scripts. For example, the Keyword "Light" can be used for a central function and the keyword "LED" for some LED lights.
Log	Activates logging for this object. Logs appear in the ***Object logs*** tab. This parameter can also be selected in the *objects* list.
High priority log	This option shifts high priority logs towards the top of the list in the ***Object logs*** tab. If the defined limit of logs is exceeded, low priority logs at the end of the list are deleted first.
Export	The *Export* parameter makes the object visible to remote XML requests, for example to make the object available in BACnet. This parameter can also be selected in the *objects* list.
Object comment	Optional additional information.

4.4 Edit and test objects

Edit all objects in the ***Objects*** list.



Click an object in the list to display the object properties. Edit the parameters *Keywords*, *Logs*, *High priority Log* and *Export*. See *Additional parameters for all applications* on page 105.

- Change the names (tags) linked to the composed addresses in the ***Tag map*** tab.
- The new composed addresses, which are visible in the *Group Address* column cannot be changed. If necessary, delete the object and then create a new one.

4.4.1 Preparation for visualization

Edit visualization parameters in the ***Vis. Parameters*** and via the ***Levels*** button in the ***Objects*** list.

Table 28: *Vis. Parameters and Levels*

Vis. Parameter	Click a slider icon under *Vis. Parameter* in the ***Objects*** list and then select a control type. For more information, see *Visualization parameters and control type* on page 147.
Levels	Link levels with names (e.g. 0 = Off and 255 = On). Use levels and names in the visualization (see *Overview of control types* on page 145). Edit levels and names in the ***Tag map*** tab.

4.4.2 Control values

Table 29: *Control values*

Set values	When C-Bus is connected, the information *C-Bus Online* appears at the bottom of the window. With this button open a window and select values for sending. In the case of a lighting application select a level and a ramp rate. If there is a prepared a temperature alarm, send the value that triggers the alarm. Set and save values. The new value is then visible in the ***Current value*** column. When the value is send, the background colour of row turns for same seconds to green and then to grey.
Current value	This column displays the last send or received value.
Auto update enabled	Object values will automatically be updated when changed. They will highlight green for a few seconds. With a click on the ***Auto update enabled button*** this function can be disabled.
Refresh button	Click the ***Refresh*** button  at the bottom of the list to manually refresh the list.
Bus sniffer	By default, the Bus sniffer function is enabled. When the sniffer function is disabled or C-Bus is offline, new objects will not be displayed. See *Sniffer function* on page 107.

4.4.3 Additional options for editing

Table 30: *Additional options for editing*

Delete	Click a delete button in the list to delete the associated object.
Clear	Click the *Clear* button at the bottom of the list to delete all currently visible objects in the list. This is useful when used with the filter function. A warning is displayed when this button is clicked.
Mass delete	Click the *Mass Delete* button at the bottom of the list to delete objects from the current filter. A warning is displayed when this button is clicked.
Object filter	• **Name:** Any part of the object name. • **Address:** The wild cards "*" and "," can be used, e.g. 0/2/* • **Application:** Select from a list. • **Keywords:** Separate multiple keywords with "," and select whether the filter will find matches for all keywords entered, or just one or more keywords. The object filter is located on the left side of the list and can be opened and closed with a button.

4.5 Sniffer function

When the sniffer function is enabled and C-Bus is online, new objects are automatically added to the ***objects*** list. Existing objects with the same composed address are not overwritten.

New objects are not automatically added to the ***Tag map*** list, they can be added manually and optionally change the names. See *Tag map* on page 110.

4.5.1 Communication with C-Bus

C-Bus must be physically connected via one of the C-Bus RJ45 connectors on the Automation Controller. If C-Bus is online, the status is shown in the bottom bar of the ***Configurator*** page *See "3.5.4 Bus Status" on page 36*.

Set and send values

Set and save values. The value appears in the ***Current value*** column of the ***Objects*** list (click on the ***Objects*** tab of the configurator page). When the value is saved, the background colour of the row turns green for a few seconds.

4.5.2 Discover new objects

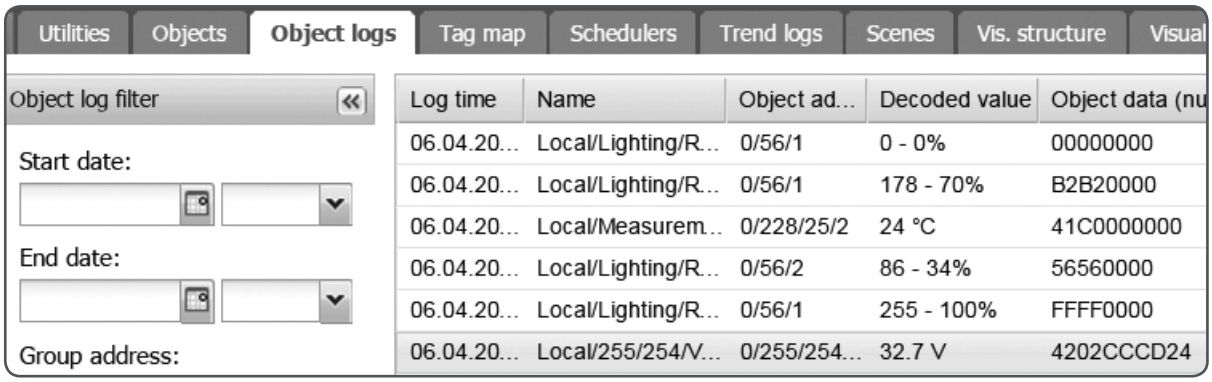
By default, the sniffer function is disabled. Enable the sniffer function in the ***General configuration*** tab.

Path: ***Utilities*** tab → ***General configuration*** tab.

When a new object is discovered, a new row appears and the background colour of the row is green for a few seconds. The row background colour also turns green for a few seconds when a new value for an object is received.

4.6 Object logs

An object's event history is displayed in ***Object logs*** tab. Logging must be enabled for the object, after which all events are logged.



4.6.1 Filter functions

Table 31: *Filter functions*

Start date	Enter date and time
End date	Enter date and time
Group address	Enter composed address (e.g. 0/56/3)
Network	Select network from list (e.g. 0-local)
Application	Select application from list.
Keywords	Enter one or more keywords. Keywords must be separated with a comma (e.g. floor1,left).
Value	Enter a value (e.g. 255).

4.6.2 Clear

Click the ***Clear*** button at the bottom of the ***Object logs*** tab to clear all logs.

4.6.3 High priority log

This option moves high priority logs closer to the top of the list. If the defined limit of logs is exceeded, low priority logs at the end of the list are deleted first.

4.6.4 Log size

By default, the log size is set to 200. The log size can be changed in the general configuration (path: ***Utilities*** tab → ***General configuration*** button).

Excessive object logging degrades the performance of the Automation Controller.

4.7 Export from Application Controller to Toolkit

When adding new applications to the Application Controller, export all applications to your original Toolkit project.

Use the most recent version of C-Bus Toolkit (1.15 or higher). Install the full package including USB drivers. The C-Gate software is required to export and import to the Automation Controller. C-Gate can also be used as part of a C-Bus control system.

The export of the *applications* of the Automation Controller is done in the ***Utilities*** tab.

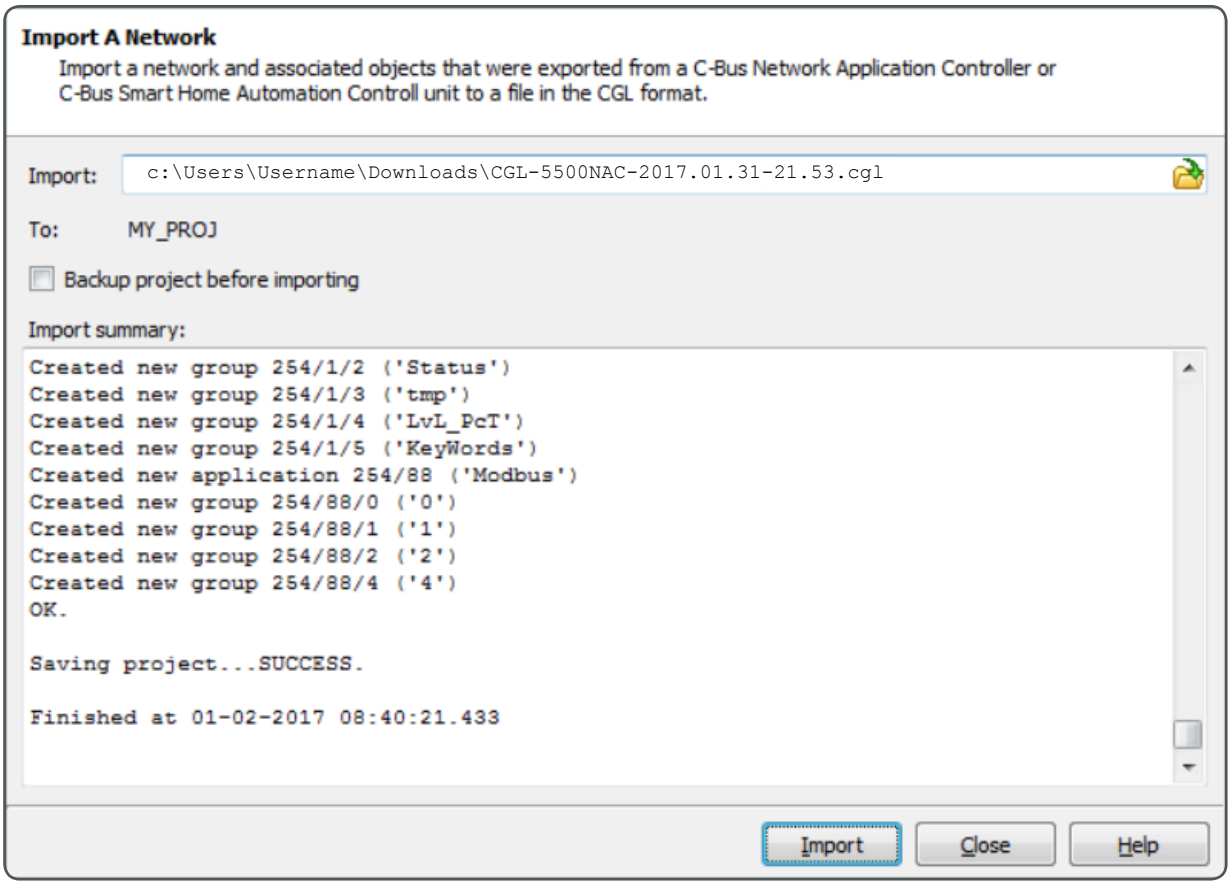
Path: ***Configurator*** → ***Utilities*** tab → ***CGL export*** button.

Steps in the *Configurator* of the Automation Controller

① Click the ***CGL export*** button.
② The file will be downloaded to the computer, where your browser runs (e.g. `CGL-AutomationController-2017.01.31-10.16.cgl`).

Steps in the Toolkit

① Open **C-Bus Toolkit**.
② Create a new C-bus project or select an existing project to import the CGL file. If more than one Automation Controllers is in a site, they will each have their own unique CGL file.
③ Click the ***Import CGL*** button at the button of the project window.
④ Select file to import (e.g. `CGL-AutomationController-2017.01.31-10.16.cgl`).
⑤ By default, a backup project will be created before import.
⑥ Click the ***Import*** button.
⑦ Save the backup file (e.g. `MY_PROJ_31_Jan_2017_1029_1.15.0.cbz`).
⑧ A message about the import appears (see next screen shot).



5 Tag map

The ***Tag map*** tab provides a detailed view about all objects. Once new objects are imported or added, access them in the tag map. Use the tag map to change all names (tags) and add new applications, groups and levels. The tag map view is hierarchical, and can be expanded and collapsed in sections to focus on the required objects.

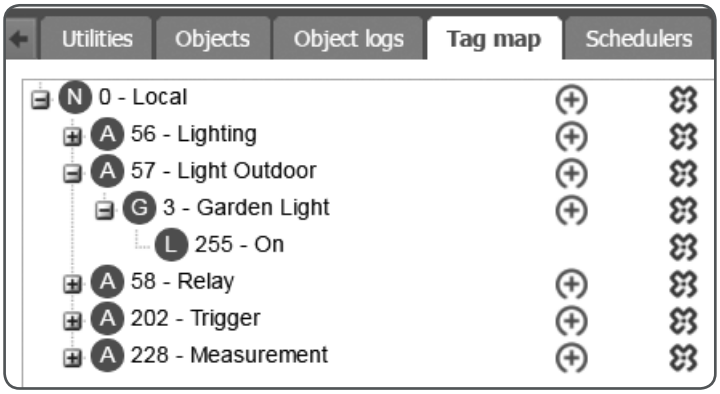
Path: ***Configurator*** → ***Tag map*** tab.

5.1 Structure

The tag map is a tree structure using the following hierarchy:

- Network
- Application
- Group
- Level.

5.1.1 Example of a tag map



Ⓝ	Network (e.g. 0 - Ground floor)
Ⓐ	Application (e.g. 56 - Lighting)
Ⓖ	Group (e.g. 1 -Room 1)
Ⓛ	Level (e.g. 255 - On)
+	Expand and show subordinate step
–	Collapse and show superordinate step
⊕	Add: • Network • Application • Group • Level
	Delete step and all subordinated steps

5.2 Edit tag names

On each level, the number and the name (tag) are shown. In the ***Objects*** list, the numbers of network, application and group are composed and separated with a slash.

e.g. `0/58/1 = Local/Lighting Upper Floor/Room 1.`

5.2.1 What cannot be edited

In the case of a measurement application, editing is possible for the tag names of network, application and device ID (group address), but **not** the channel number.

e.g. `0/228/1`**`/1`** `= Local/Measurement/Temperature`**`/1`**.

5.2.2 Editing a name

To edit a name, click the row in the map. Use the + and – buttons to expand and collapse the map.

5.3 Add new tags

Add new tags using the tag map, but it is recommended to add new applications and groups in the objects list or to import a CGL file from Toolkit. The new applications and groups are then displayed in the tag map.

An exception to this is for objects detected by the sniffer function. These objects appear in the ***Objects*** list but are not displayed in the tag map. In this case, use the tag map to add these applications and group addresses and optionally change the tag names.

5.3.1 Add application group level

Levels are used for the visualization (see *Overview of control types* on page 145).

Names can be defined for levels (e.g. 255 = On and 0 = Off). The levels defined in the tag map are also available in the ***Objects*** list.

Add a new tag via the ***Add new tag*** button or via an ***Add*** button in the ***Tag map*** list.

6 Schedulers

The ***Scheduler*** provides control of different building functions using date and time. Typical functions include lighting, shutter control, heating and cooling control, and scene control for multiple functions. Use the application *250 - User Parameter*, to trigger an event-based script.

Use the configurator to add objects and edit their schedule times.

The end user can edit scheduled times but cannot add new objects.

6.1 Access to the Scheduler

The end user can access the scheduler from either the main menu or the visualization view.

6.1.1 From the main menu

Click the ***Scheduler*** button on the main menu.

6.1.2 From the visualization view

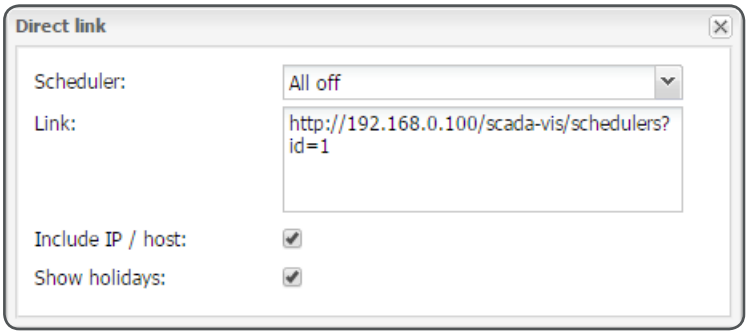
Link to all schedulers

- To open the main scheduler page from the visualization, configure the *Schedulers* link in the plan editor of the visualization. See *Link* on page 140.
- To display the scheduler as a frame within the visualization, configure a *Frame* in the plan editor of the visualization. See *Frame* on page 153.

Schedulers are not supported on a Smartphone Visualization page.

Link to a specific scheduler

① In the configurator ***Schedulers*** tab (see page 113), click the ***Direct link*** button to display the Direct link window.



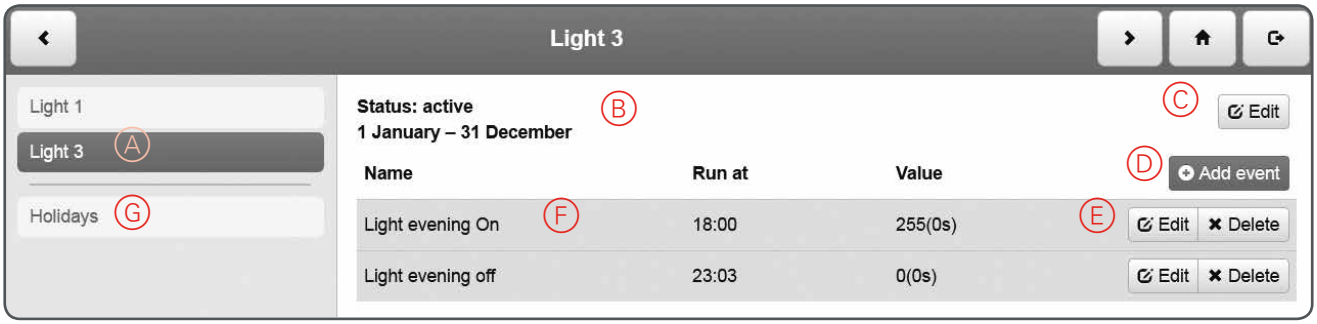
② In the ***Direct link*** window, choose a scheduler. The link to the scheduler is displayed in the Link field.

To link a scheduler in the visualization of another Automation Controller, complete IP address. Tick *Include IP / host* to display the complete IP address in the Link field.

③ Copy the scheduler link displayed in the Link field.
④ Open the configurator ***Visualization*** tab. See *Link* on page 140.
⑤ Configure a Link in the Plan Editor, using the following settings:

Link to:	Select “External link”
External Link:	Paste the link copied from the Link field (see step ③) e.g. /scada-vis/schedulers?id=1

Configuring from the visualization view of the scheduler

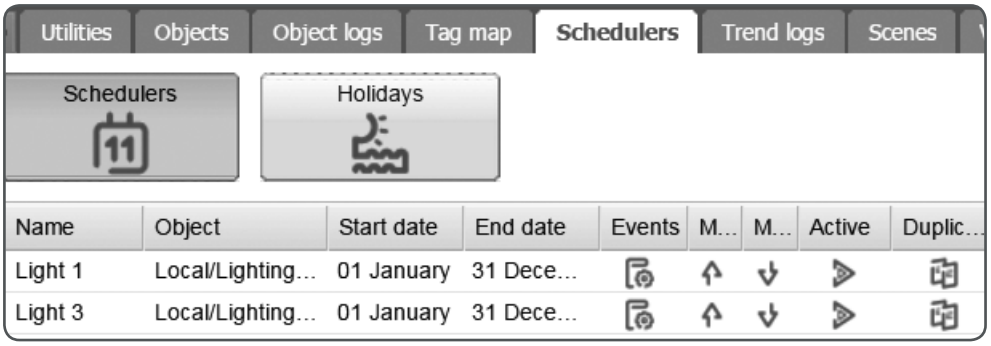


Ⓐ *Scheduler*: Schedulers are created in the configurator.
Ⓑ Status: Displays as **active** or **inactive**.
Ⓒ *Edit Scheduler button*: Set scheduler status with start and end dates, holiday behaviour.
Ⓓ *Add event button*: Set name, run time (e.g. switching) and value.
Ⓔ *Edit event button*: Change name, run time (e.g. switching) and value.
Ⓕ *Event*.
Ⓖ *Holidays*: Set special time periods for all schedulers.

When an *Object* of an application is linked to a scheduler the user can add, edit and delete *Events*. In addition the user can edit different Holidays.

6.2 Schedulers and events

Configure all scheduling functions using the configurator ***Schedulers*** tab. Each scheduler must be linked with an object.



6.2.1 Add a scheduler

Path: ***Configurator*** → ***Schedulers*** tab → ***Schedulers*** button → ***Add Scheduler*** button.

Table 32: *Scheduler settings*

Object	Select an object to control.
Active	Set the scheduler to the **active** or **inactive** status. Can also be edited in a visualization—see Ⓑ on page 113.
Name	Select a name. Can also be edited in a visualization—see Ⓓ Ⓔ on page 113.
Start/End date	Defines when the Scheduler will work. The default dates are: January 1 to December 31. Can also be edited from a visualization—see Ⓒ on page 113.

6.2.2 Add an event

Add events in a visualization or in the configurator.

- In a **visualization,** click the ***Add event*** button—see Ⓓ on page 113.
- In the **configurator,** click the ***Add event*** button in the ***Events*** list of the specific scheduler.

  Path: ***Scheduler*** tab → ***Schedulers*** button → ***Scheduler*** List → ***Events*** icon.

Table 33: *Event settings*

Active	Set the event to the active or inactive status
Name	Select a name
Run at	• Sunrise • Sunset Sunrise and Sunset times are dependent on the time zone or the specific longitude and latitude of the location. See *Set date and time* on page 32. • Specific time
Start time offset (Sunrise or Sunset)	Select a time offset: –11 h 59 min to +11 h 59 min
Start time (Specific time)	Time when the event is triggered (switching time)
Day of the week (Specific time)	None ... All (Mo, Tu, ... Su) e.g. Tu-We, Fr Default: All
Weekday in month (Specific time)	None to All (1st, 2nd, 3rd, 4th, 5th, last) e.g. 1st, 3rd Default: All
Days of the month (Specific time)	None ... All (1, 2, ... 31) e.g. 1, 14, Default: All
Months (Specific time)	None ... All (Jan, Feb, ... Dec) e.g. Jan-Mar
Year	No entry, Year e.g. 2017 = only in 2017 Default: No entry = recurring every year
Holidays	Holiday periods can by defined can be applied for all schedulers. For each event select one of the following options: • No effect • Do not run on holidays • Run only on holidays Default: No effect

Value	Select values specific to the application of the object e.g. Lighting: *Target level* and *Ramp rate*

When all settings are selected, click the ***Save*** button.

Example with day of week

Every Friday, switch the light off at 10 pm.

Apply the following settings:

Start time	22:00
Day of the week	Fr
Weekday in month	All
Days of the month	All
Months	All
Year	
Holidays	No effect
Value	Target level: 0 Ramp rate: 0 s

6.2.3 Add holidays

Define holiday periods and then apply them to any scheduler. Holidays can be applied differently to each event:

- Event is not affected.
- Event does not run on holidays.
- Event runs only on holidays.

Add holidays in the visualization view or in the configurator.

- In the **visualization**, click ***Holidays*** and then the ***Add holiday*** button.
- In the **configurator**, click the ***Add holiday*** button in the ***Holidays*** list.

  Path: ***Schedulers*** tab → ***Holidays*** button → ***Holidays*** List → ***Events*** icon.

Settings for day of week (visualization view)

Table 34: *Holiday settings (day of week)*

Holiday type	Day of the week
Day of the week	Select a week of the month: 1st, 2nd, 3rd, 4th, 5th, last
	Select a day: Monday, Tuesday, ... Sunday
Month	Select a month: January, February, ... December
Duration (days)	Select 1 ... 90 days.
Recurring every year	Checked = yes Unchecked = no

Settings for specific date (visualization view)

Table 35: *Holiday settings (specific date)*

Holiday type	Specific date
From date	Select in calendar view: • Year • Month • Day
To date	Select in calendar view: • Year • Month • Day
Month	Select a month: January, February, ... December
Recurring every year	Check to activate Uncheck to deactivate

7 Trend logs

Trend logs, or *data logging*, allow the end user to store selected data and compare that data over time. Trends can be accessed via the main menu or via the visualization. Trend logs are stored on the Automation Controller's internal microSD card.

Use the ***Configurator*** to add and edit objects.

The end user can select different trend views but cannot create new objects.

7.1 Access to trend logs

The end user can access the trend logs from either the main menu or the visualization view.

7.1.1 From the main menu

Click the ***Trends*** button on the main menu.

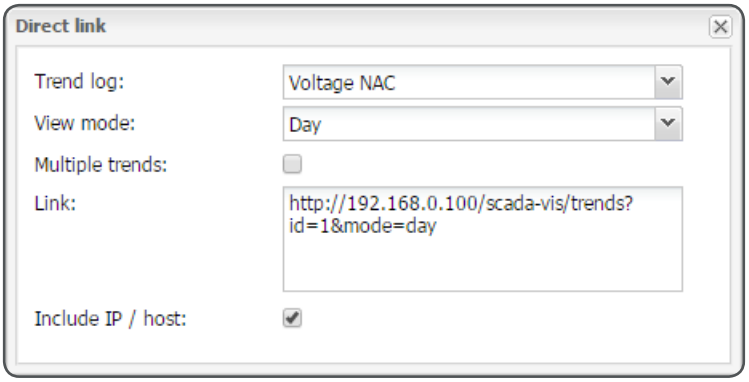
7.1.2 From the visualization view

Link to all trend logs

- To open the main trends page from the visualization, configure the Trend link in the plan editor of the visualization (see *Link* on page 140).
- To display the trends as a frame within the visualization, configure the Frame in the plan editor of the visualization (see *Frame* on page 153). Trends are not supported on a ***Smartphone Visualization*** page.

Link to a specific trend log

① In the configurator ***Trend logs*** tab (see page 120), click the ***Direct link*** button to display the ***Direct link*** window.



② In the Direct link window, choose a *Trend* log. The link to the trend log is displayed in the *Link* field.

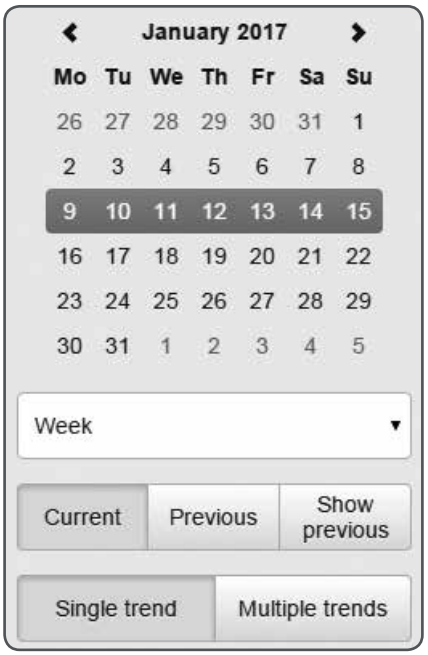
To link a trend log in the visualization of another Automation Controller, complete the IP address. Tick *Include IP/host* to display the complete IP address in the Link field.

③ Copy the *Link* field to use it in the plan editor of the visualization.
④ Open the configurator ***Visualization*** tab. See *Link* on page 140.
⑤ Configure a link in the plan editor, using the following settings:

Link to:	Select “External link”
External Link:	Paste the link copied from the Link field (see step ③) e.g. /scada-vis/trends?id=1&mode=day

7.2 Views of trend logs

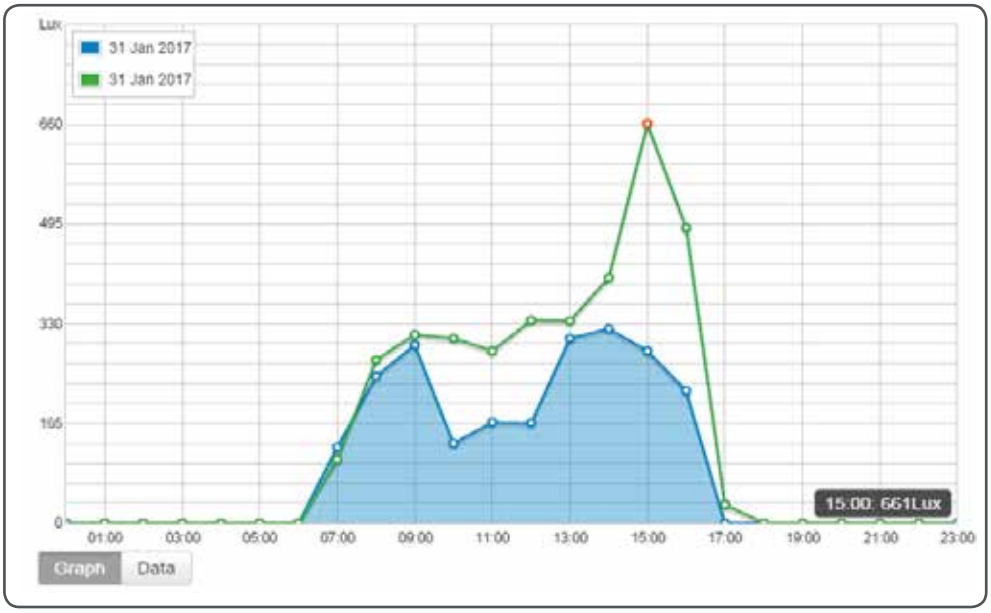
On the ***Trends*** page, select from different views.



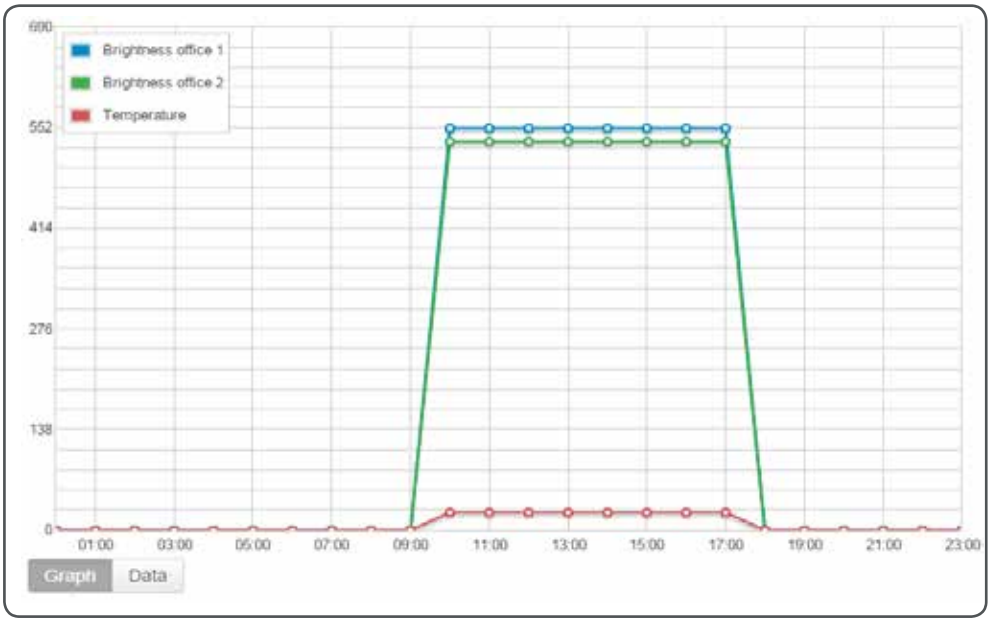
Available trend log views

- View period of trend:
  - *Day*
  - *Week*
  - *Month*
  - *Year*.
- View current and previous trend:
  - **Current:** Click the ***Current*** button and select the *day/week/month/year* in the calendar that will be the *Current* view. The curve *Current* is always shown.
  - **Previous:** click the ***Previous*** band and select the *day/week/month/year* in the calendar that will be the *Previous* view. The curve *Previous* is not shown until also the ***Show previous*** button is clicked.
- View single or multiple trends:
  - ***Single trend*** button
  - ***Multiple trend*** button.
- View graph or data:
  - ***Graph*** button
  - ***Data*** button.

Example with current and previous trend and period day

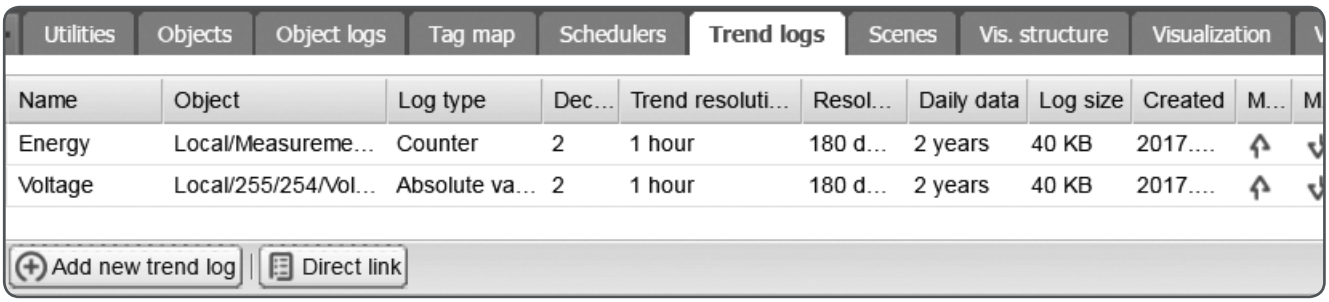


Example with multiple trend and period day



7.3 Configure trend logs

Configure trend logs in the ***Trend logs*** tab.



Add new trend log

To add a new trend log, click the ***Add new trend log*** button at the bottom of the trend log tab (see previous screen shot). Configure the following trend log parameters.

Table 36: *Trend log parameters*

Object	Select object to log.
Name	Name of the trend log.
Log type	• **Counter:** Sums up the delta between the received value and the last received value (negative delta is not regarded). • **Counter with negative delta** (negative delta is also taken into account). • **Absolute:** Saves the actual values.
Trend resolution	Average value of the counted samples for the specific time interval to be displayed on the trend (5, 10, 15, 20, 30 min, 1 hour).
Decimal places	If the object is a floating point type, it is mandatory to enter the number of decimal places.
Resolution data	Storage time for short time data (max. 5 years).
Daily data	Storage time for long term data (max. 10 years).
Always show zero	If checked, the Y-axis begins with 0. If unchecked, the Y-axis begins at lowest real value. Unchecking this option improves trend resolution if values are always far from 0.

8 Scenes

8.1 Overview

The ***Scenes*** tab, allows easily configured scenes.

Path: ***Configurator*** → ***Scenes*** tab → ***Scenes*** list.

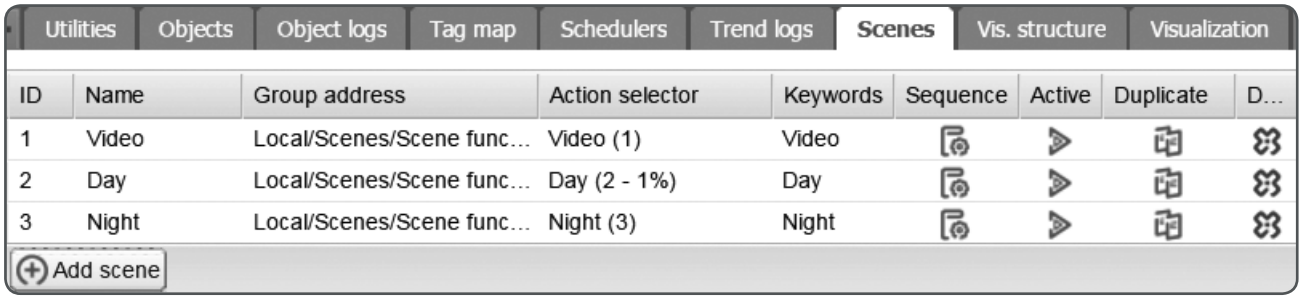
Use a scene to change multiple room functions at the touch of a button. Setting a scene allows dimming of the room lighting to a specific value, move the blinds into the desired position and switch on the power supply to the socket outlets in a room. As an example of a pure lighting scene, control three channels of a dim actuator using the scenes *day*, *night* and *video*.

For a standard scene, the following is needed:

- A ***Trigger group*** to set different scenes. Use the application *202 - Trigger Control*.
- One ***Action selector*** for each scene. The action selector is associated with the ***Trigger Control***.
- Components for each scene. These components will be controlled via applications (e.g. 56 Lighting).
- Values (e.g. levels and ramp rates) of the components for each scene. The values can by edited or saved online via C-Bus.
- Buttons or functions to set scenes. These buttons can be physical C-Bus push buttons or elements on a screen (see *Visualization* on page 124). Alternatively, a scene can be set by a physical timer or a software function (see *Schedulers* on page 112).

8.2 Configuration

For each scene, configure a *Trigger group* in the ***Scenes*** list.



Add scene

Click the ***Add scene*** button to open the scene parameters.

Table 37: *Scene parameters*

Name	Name of the scene
Scene is active	Active: scene is ready for usage.
Group address	Select a trigger group (Application 202 Trigger Control).
Action Selector	For each value of the Action Selector, define the display text. The display text appears in a visualization (see *Local and remote scene control* on page 122). For example, if a night scene is triggered with value 3, edit a level with the object value 3 and the display text “Night”. Click the ***Levels*** button to configure levels.
Keywords	Keywords can be used for scripts.

Click ***Save*** button when parameters are set.

Click a row in the ***Scene*** list to open the scene parameters for editing.

Add components

A list of objects for one scene is called *Sequence*. Click the symbol in the ***Sequence*** column (see page 121) to add an object for each component and select values. The following actions are available.

Add objects to a scene

Click the ***Add objects*** button to open the list of objects. Select the objects required to be controlled by the scene and then click the ***Save*** button.

Set values to the objects of a scene

Select one or more objects and click the ***Set value*** button. Select a value and click the ***Save*** button.

Run scene

When controlling a scene online via C-Bus, click the ***Run scene*** button.

Save live values

With this function, it allows changing of the values of the components independent of the Automation Controller. For example, prepare a lighting scene by physically setting each light to the brightness required. Then, select the corresponding objects and click the ***Save live values*** button. The actual values are displayed in the list.

Delete

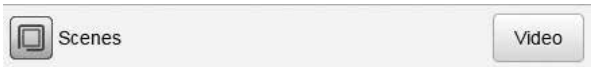
Click the ***Delete*** button to remove objects from a scene, e.g. to delete unwanted objects from a scene that is duplicated as a starting point.

Duplicate

In many cases, the same trigger groups and components are used in scenes. Click the symbol in the ***Duplicate*** column (see page 121) to create a copy of a scene.

8.2.1 Local and remote scene control

Scenes can be controlled via a PC/Tablet or a Smartphone visualization.

PC/Tablet	Smartphone
	

Remotely control local scenes configured in the Automation Controller by using a C-Bus push-button configured with a scene function:

① Configure Automation Controller keys as scene keys.
② Use the trigger group of the local scenes.
③ Use the values of the action selectors linked with local scenes.

Scenes configured in another C-Bus device can be controlled from the PC/Tablet or Smartphone visualization. The same trigger group can be used in the Automation Controller and the other C-Bus device.

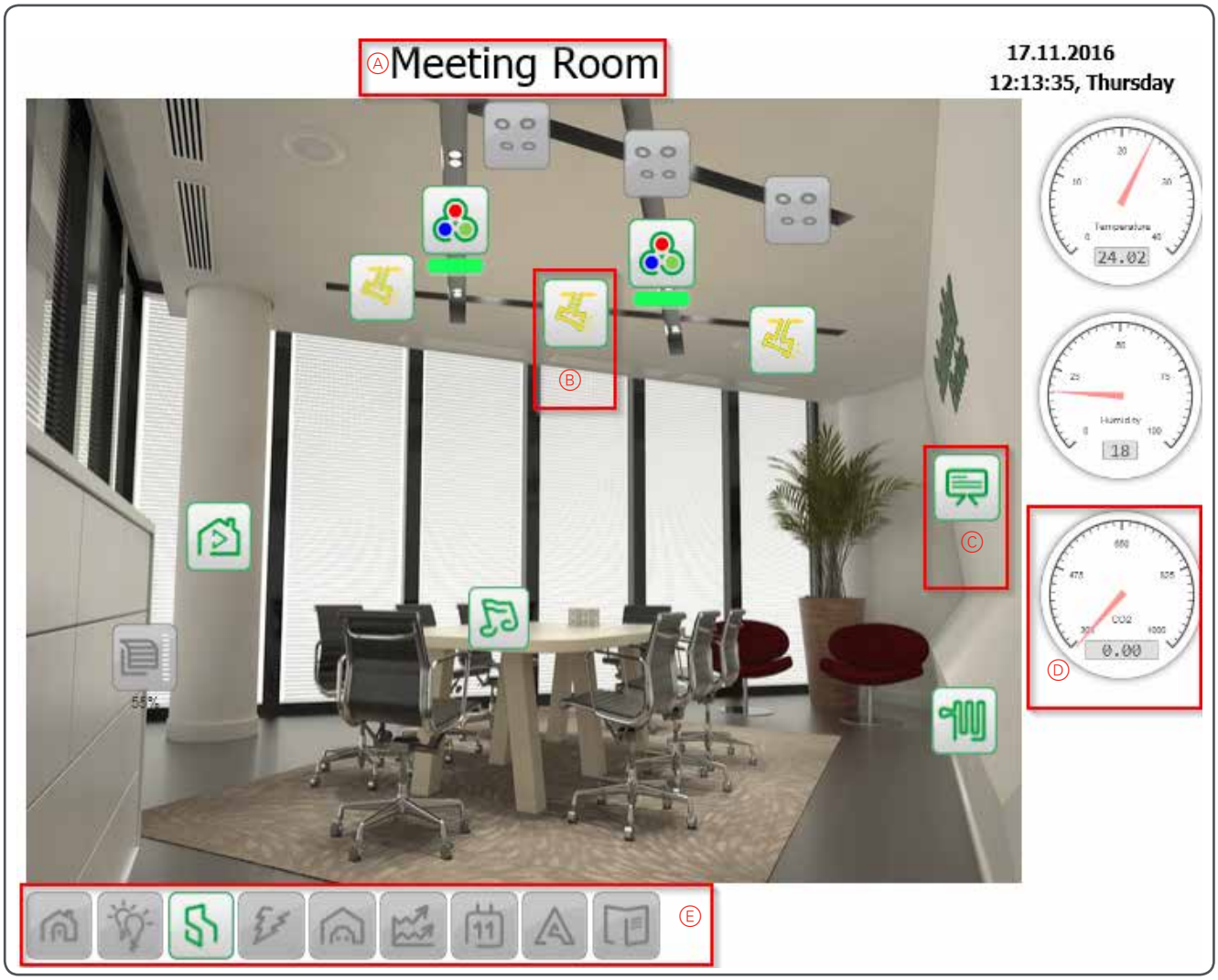
9 Visualization

9.1 Overview

A visualization is a structured collection of pages called *Plans*. Each plan is linked with a *Level,* which normally represents one floor of a building. A plan can contain graphics, text and control elements. Communication with active elements is based on C-Bus *Objects*.

Each plan can be used as *PC/Tablet visualization*, *Smartphone visualization* or both.

Example of a visualization page for PC/Tablet



Ⓐ Text label: The name of the plan.

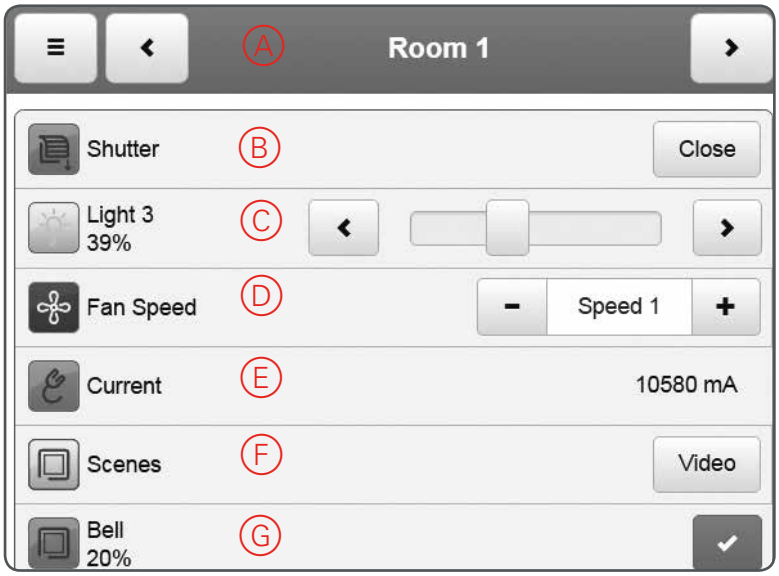
Ⓑ Object (switch): Touch the object icon for direct control and status indication.

Ⓒ Object (control): Touch the object to display a slider control to adjust a setting.

Ⓓ Gauge: Touch the selected control element to display a value, e.g. temperature.

Ⓔ Links: Provides access to other visualization pages in the plan. Touch a link to display that visualization page.

Example of a visualization page for Smartphones



Ⓐ Navigation: Access to all pages (plans), next page, previous page.
Ⓑ Shutter/blind - Display: Tags.
Ⓒ Slider - Display: Percentage (0-100%).
Ⓓ Cycle/Fan - Display: Tags.
Ⓔ Read only - Display: Icon and value.
Ⓕ Trigger select - Tags.
Ⓖ Bell press - Display: Tags.

The *Smartphone Visualization* displays a list with the configured object and link elements. Text label, image, gauge and graph elements are only displayed in the *PC/Tablet Visualization*. Each object, link and frame element can optionally be hidden in the *Smartphone Visualization*.

The *Smartphone Visualization* adapts automatically to the screen size. For *PC/Tablet Visualization*, however, defining the plan size is mandatory. Templates are available for various resolutions and screen types (e.g. landscape WSVGA or portrait HD).

By default, start the ***PC/Tablet Visualization*** or the ***Smartphone Visualization*** via the ***Start page***.

To create a visualization, click the ***Configurator*** button.

Configure user access to visualization pages (see *User access* on page 156). For each user, different levels of access can be given for visualization pages (plans), schedulers and trends.

The Network Automation Controller is designed for a maximum of 50 users for visualization. The Wiser for C-Bus Automation Controller is limited to a maximum of 8 users.

9.1.1 Steps of a visualization

① (Optional) Start with a general configuration of the visualization pages in the ***Vis. configuration*** tab.
② Upload any background and Plan images (e.g. icons) via the ***Vis. graphics*** tab.
③ Create at least 1 *Level* and 1 *Plan* in the ***Vis. structure*** tab (each visualization page is linked to a *Level*).
④ (Optional) Create other structure elements such as *Layouts* and *Widgets*.
⑤ Add the content to the *Layouts*, *Widgets* and *Plans* in the ***Visualization*** tab.
⑥ When adding objects to the plan, the objects must be available (see *C-Bus objects* on page 38). (An *object* is a C-Bus object in a network with an application and a group.)

Table 38: *Visualization steps*

Chapter	Path	Steps
Visualization configuration on page 127	***Configurator*** → ***Utilities*** → ***Vis. configuration*** or ***Configurator*** → ***Visualization*** → ***Vis. configuration*** button	(Optional) **Select general settings for the visualization.** Default Settings: • PC/Tablet sidebar: Shows as overlay. • PC/Tablet view: Centers plans, enables auto-sizing. • PC/Tablet page transition: No transition.
Visualization graphics on page 129	***Configurator*** → ***Vis. graphics***	(Optional) **Add graphical elements to include later via the *Visualization* → *Image* tab.** • Icons: Add icons for new functions. • Images/Backgrounds: Add project-related images. • Fonts: Add fonts for *Text labels.* • Edit Custom CSS: If available, upload and edit new Cascading Style Sheets.
Layouts and Widgets on page 131	***Configurator*** → ***Vis. structure*** → ***Layout/Widgets***	(Optional) **Add layouts and widgets to the structure.** A *Layout* can be used as a template for one or more plans (visualization page). A *Widget* is a small visualization page which pops up when pressing the associated button (icon) on the plan. A widget is linked with an object. The elements for the layouts and widgets must be added in the ***Visualization*** tab.
Levels and Plans on page 133	***Configurator*** → ***Vis. structure*** → ***Levels/Plans***	**Mandatory** **Add Levels and Plans to the structure.** **(Minimum = 1 level with 1 plan)** A level is usually a part of a building (e.g. ground floor). A plan is a room or a functional overview. Default plan settings: • Plan size: **1024 x 768** • Layout: PC/Tablet visualization: **show** Smartphone visualization: **show.** The plan elements must be added in the ***Visualization*** tab.

Chapter	Path	Steps
Create visualization content on page 136 *Link* on page 140 *Text label* on page 142 *Image* on page 143 *Objects* on page 144 *Gauge* on page 152 *Frame* on page 153 *Graph* on page 153 *Camera* on page 155	***Configurator*** → ***Visualization***	**Mandatory** **Create content on a plan, layout or widget.** • Link: Navigate to other plans. • Text label: Provides textual information. • Image: Image on the plan, background image or URL. • Object: Controls functions and shows status. • Frame: Shows *Scheduler* or *Trend logs*. • Gauge: Displays values. • Graph: Shows a real time graph on a plan. • Camera: Displays the stream of an IP camera.

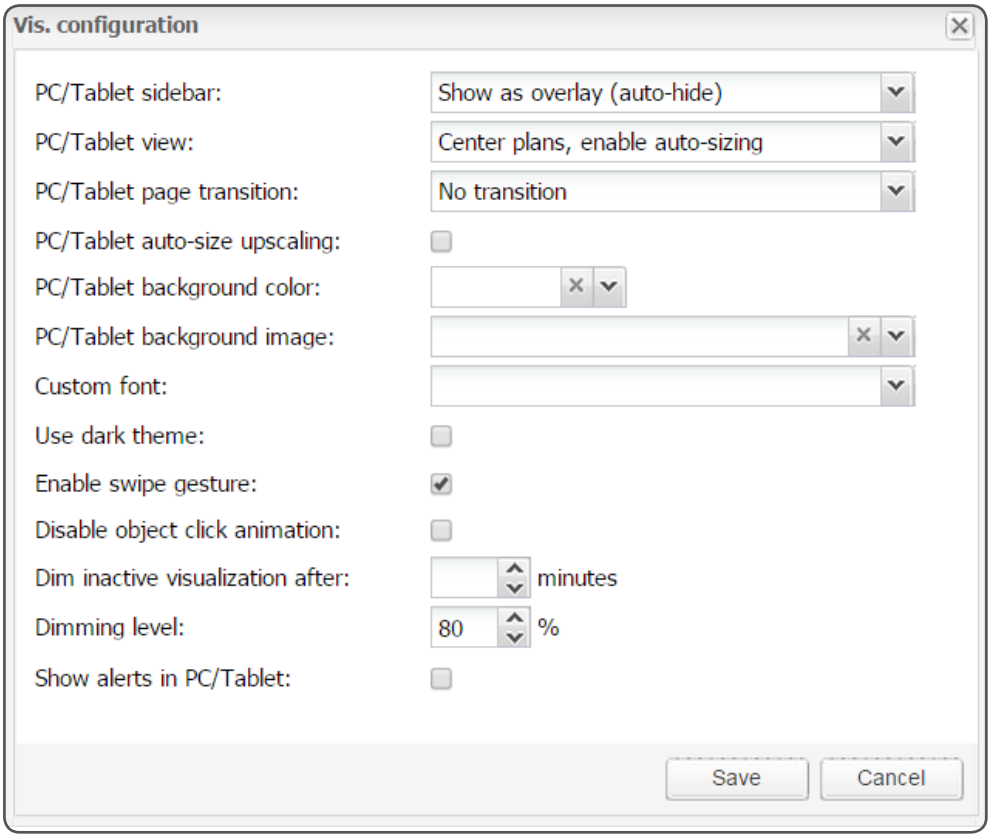
9.2 Visualization configuration

Configure all pages of the visualization via the ***Vis. Configuration*** button:

Open the visualization configuration via one of the following paths:

- ***Configurator*** → ***Utilities*** tab → ***Vis. configuration*** button or
- ***Configurator*** → ***Visualization*** tab → ***Vis. Configuration*** button.

9.2.1 Visualization configuration parameters



PC/Tablet sidebar

Enables a sidebar containing a list of plans in the visualization. The sidebar is used to navigate between visualization pages (plans). Sidebar display options include:

- Show as overlay (auto-hide)
- Show docked
- Hide (full-screen mode)
- Docked/with auto-hide option/hidden.

PC/Tablet view

Defines how plans will be displayed. Options include:

- Align plans to top left, no size limits
- Center plans, limit size
- Center plans, enable auto-sizing
- Center horizontally, auto size width.

Information about auto-sizing

If using Google Chrome or Mozilla Firefox, auto-sizing is supported.

PC/Tablet page transition

Enables different transition effects for page changes in the visualization.

PC/Tablet auto/size upscaling

Enables automatic rescaling for multiple screen resolutions.

PC/Tablet background colour

Applies a common background colour for all visualization pages. Alternatively, select a background colour per *Layout* or per *Plan*.

PC/Tablet background image

Applies a common background image for all visualization pages. Alternatively, select a background image per *Layout* or per *Plan*.

Custom font

Applies a common font for the visualization.

Use dark theme

Inverts colours, fonts, graphs and controls to match a dark styled visualization.

Enable swipe gesture

Enables swiping between plans on tablets and smartphones.

Disable object click animation

By default, objects change their size slightly when clicked on. Select this option to prevent this effect.

Dim inactive visualization after

An energy-saving feature for battery powered devices. Select the number of minutes before an inactive visualization is dimmed.

Dimming level

Adjusts the brightness level of dimmed screens as a percentage of full brightness.

Show alerts in PC/Tablet

By default, alerts are displayed in the ***Alerts*** tab within the configurator. When this parameter is enabled, alerts are also displayed as pop-up messages in the *PC/Tablet visualization*:

9.3 Visualization graphics

Graphics must first be uploaded before they can be used in the visualization. In the ***Vis. graphics*** tab, the following sub-tabs are available:

- ***Icons***
- ***Images/Backgrounds***
- ***Fonts***
- ***Edit custom CSS.***

List of Icons

9.3.1 Icons and images

Icons

A basic package of icons is preinstalled. Additional icons can be downloaded.

Image formats

JPEG, GIF, PNG and SVG images are supported. The file name can contain letters, numbers, underscore and minus characters.

Name (optional)

Add a name for an image to appear in the list when adding a new object. An image name can contain letters, numbers, underscore and minus characters.

Add/Delete new images

- Use the ***Add/Delete*** buttons on the bottom of the ***Icons*** tab and ***Images/ Backgrounds*** tabs to add or delete images.
- Upload individual files as-is, or multiple files in a ZIP archive.
- Max size of an individual graphic: 2 MB
- Max size of a whole ZIP archive: 32 MB.

9.3.2 Fonts

True Type (TTF) and Open Type (OTF) fonts are supported. In order to access new fonts in the visualization editor, firstly install the font(s) and then click the ***Save and reload plan*** button.

9.3.3 Edit custom CSS (Cascading Style Sheets)

Advanced function: The visualization style and design is controlled by a Cascading Style Sheet (CSS). The CSS determines the style of all control buttons, *Smartphone visualization*, *Scheduler* and *Trend*.

Install a modified CSS containing different style definitions for the visualization elements.

Any replacement CSS must contain all of the classes and definitions of the original CSS to avoid degrading the appearance and functionality of the visualization pages.

9.4 Visualization structure

Use the ***Vis. Structure*** tab to create all building *Levels* and *Plans*.

Additionally, it allows creation of *Layouts* and *Widgets.*

A *Plan* is a visualization page, which can contain graphics, text and control elements.

A *Layout* is an advanced background layer for plans. The layout can:

- contain the same type of elements as a plan, and
- be associated with one or many plans.

All elements of a layout are visible in the background of the plan, while elements of the plan are displayed in the foreground. Typically, a layout contains common elements for the plans, such as menu buttons or time and date.

The dimensions (pixel size) of the layout and plan can be different or identical. The center point of both layout and plan is the same when displayed in the visualization.

A *Widget* is a small visualization page which can pop-up on top of the plan when a button is clicked or tapped.

The widget size must always be smaller than the plan size.

Example of a Plan with a Layout and a Widget

Ⓐ Layout: The area in the blue frame.

Ⓑ Plan: The area in the red frame.

Ⓒ Widgets: The area in the orange frame.

Content cannot be added to *Layouts*, *Widgets* and *Plans* using the ***Vis. Structure*** tab. Once the structure is created using the ***Vis. Structure*** tab, add the content via the ***Visualization*** tab.

9.4.1 Layouts and Widgets

Layouts / *Widgets* are optional design elements that can be used on plans.

A layout can be added to a plan (see *Levels and Plans* on page 133).

A widget can be added to an object (see *Objects* on page 144).

Add new Layout/Widget

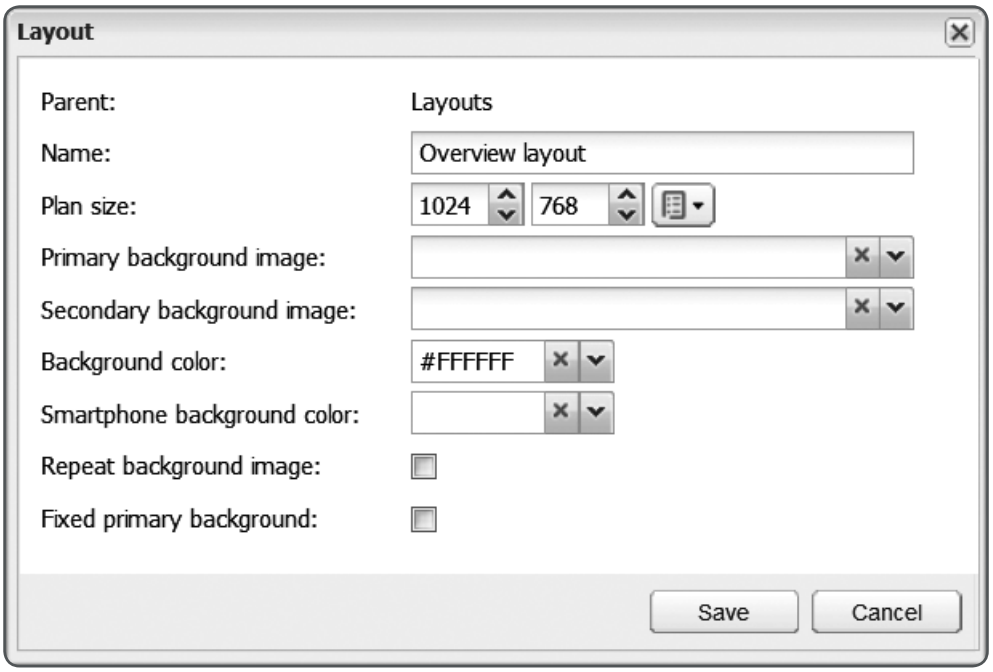
Add a new *Layout* or *Widget* with these controls:

- The **+** button (layout or widget)
- The ***Add new layout*** button
- The ***Add new widget*** button.

Actions with levels including all plans or individual plans

- Export as file (*.tar) with the ***Export*** button
- Import a file (*.tar) with the ***Add/Import*** button
- Duplicate with the ***Duplicate*** button
- Delete with the ***Delete*** button

Example of Layout parameters



Plan Size

The size of the *Layout* and the *Plan* can be different. When displayed, the center point of both layout and plan is the same. The *Widget* size must be smaller than the plan on which it is placed.

Background images

First import the image via the ***Vis. Graphics*** tab. First import the image via the ***Vis. Graphics*** tab.

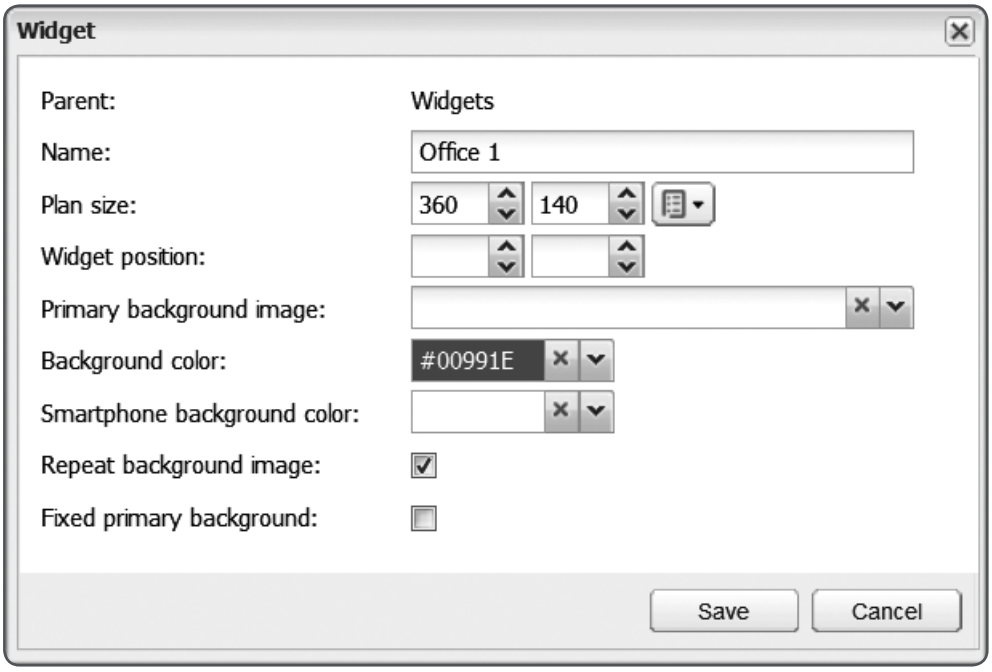
A background image is aligned with the top left corner of the plan and is displayed at its original size.

In the ***Visualization*** Tab, images can be placed as image elements and positioned freely when the page is designed.

Repeat background image

A small image can be repeated (tiled) across the *Layout*.

Example of Widget parameters



Widget position

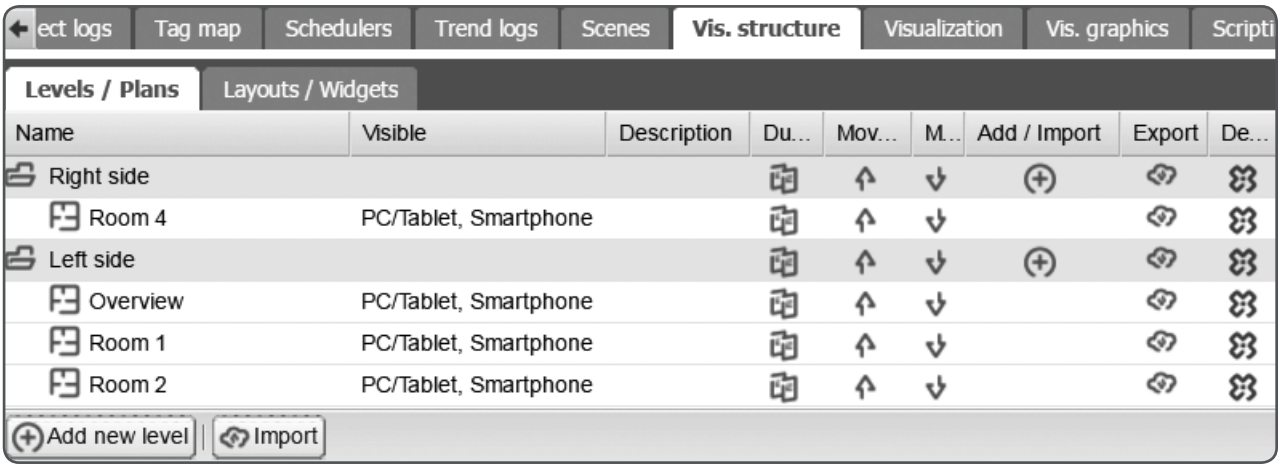
By default, *Widgets* are displayed next to the icon which calls the widget. Define another position on the plan for the widget if desired.

9.4.2 Levels and Plans

A *Plan* is visualization page which can contain graphics, text and control elements. In the ***Vis. structure*** tab, *Plans* are linked to the *Levels*. Define the size and background colours and images in the ***Vis. structure*** tab, however the content of a plan has to be added in the ***Visualization*** tab.

Levels usually represent a part of a building, such as a floor. Levels can also be used for different functions such as lighting, shutter control and heating. Use levels as structure elements for plans in the navigation/sidebar of the visualization.

The ***Levels/plans*** sub-tab In the ***Vis. Structure*** tab contains the function icons and buttons for creating and editing levels and plans.



Add new level

Add a new main *Level* with the ***Add new level*** button (several main levels are possible). Protect access to the level with a Pin Code (3–8 digits).

Add a second level or plan

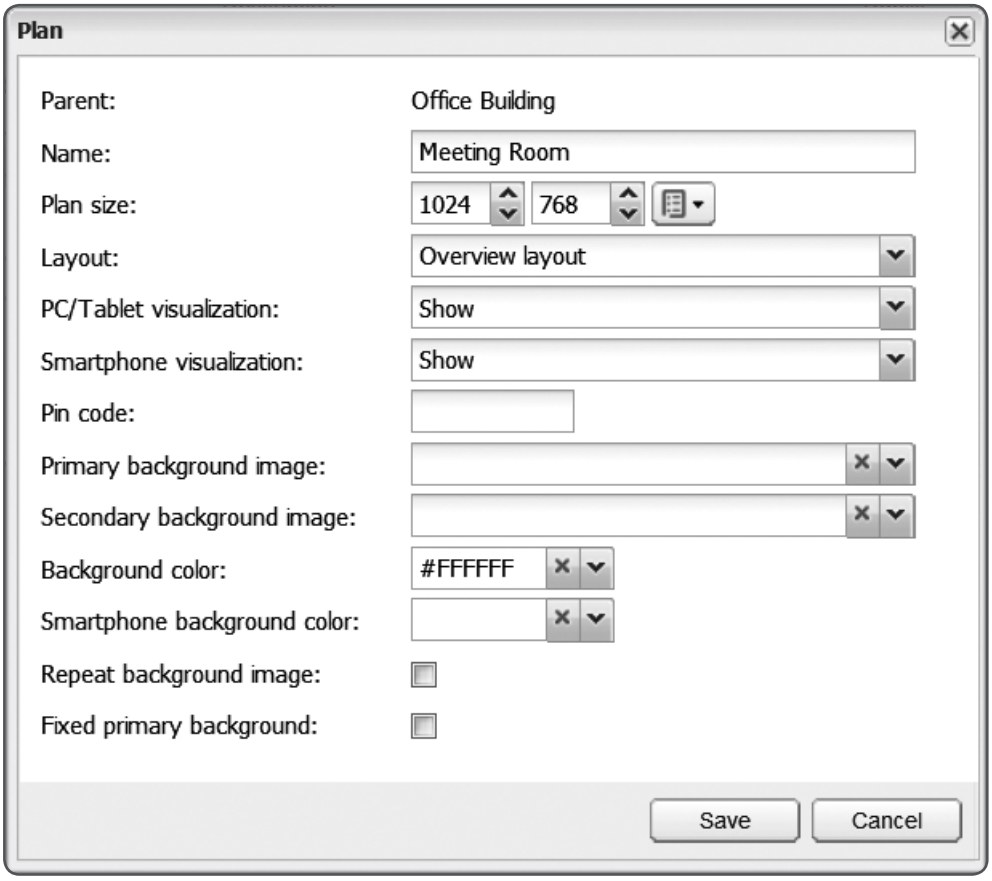
Add a second *Level* or a new *Plan* by clicking the main level ***Add/Import*** button:

Plans can be added to a second level, but not a third level. Protect access to the level with a Pin Code (3–8 digits).

Actions with levels including all plans or individual plans

- Re-order within a Level using the ***Move up/down*** buttons
- Export as file (*.tar) with the ***Export*** button
- Import a file (*.tar) with the ***Add/Import*** button 
- Duplicate with the ***Duplicate*** button
- Delete with the ***Delete*** button  .

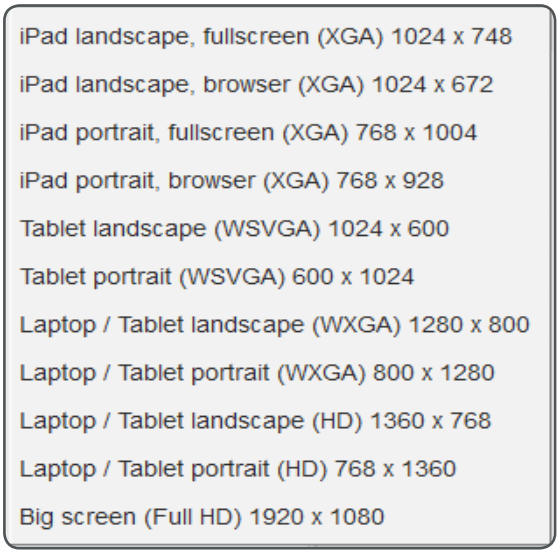
Example of plan parameters



Plan size in pixels

Click the plan size button to set the plan size:

Enter specific values or select a pre-set size:



Layout (optional)

Assign a layout to the plan. (First create a layout—see *Layouts and Widgets* on page 131).

PC/Tablet visualization and Smartphone visualization

- Show
- Show make default (= start page)
- Hide.

Background images (optional)

Firstly import the image via the ***Vis. Graphics*** tab.

A background image is aligned with the top left corner of the plan and displayed at its original size.

In the ***Visualization*** Tab, images can be placed as elements and positioned freely when the page is designed.

Repeat background image

An small image can be repeated (tiled) across the *Layout*.

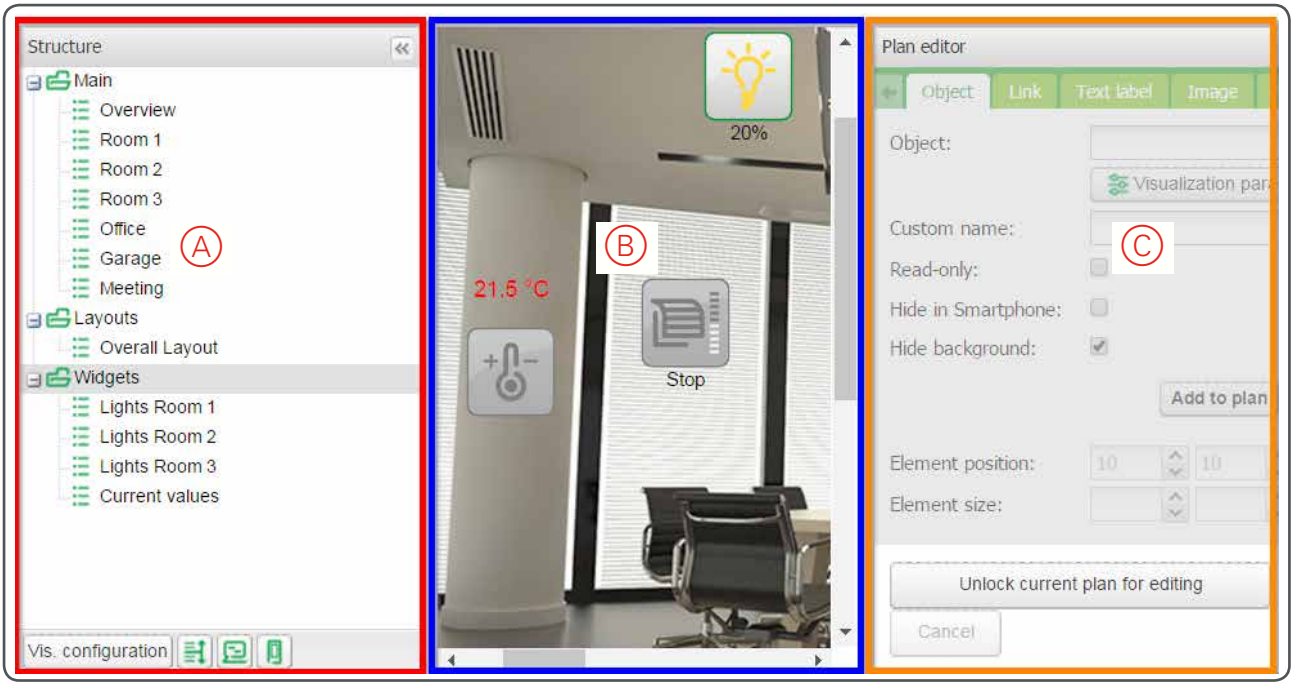
9.5 Create visualization content

9.5.1 Overview

In the ***Visualization*** tab, add or modify the content to the plans, layouts and widgets. Toggle between editing and visualization mode.

- In visualization mode select a plan, layout or widget in the structure and test functions on the visualization map.
- In editing mode add/modify content on the visualization map using plan editor.

Areas of the Visualization tab in the visualization mode



Ⓐ	Structure panel	List of *Plans*, *Layouts* and *Widgets* created in *Vis. structure*.
Ⓑ	Visualization map	Content of the *Plan/Layout/Widget* currently selected in the structure.
Ⓒ	Plan editor	Tool to add and configure elements which are visible in the visualization map.

Main steps of editing

To add or modify content, follow these steps:

① Select a *Plan*, *Layout* or *Widget* in the ***Structure panel***.
② Click the ***Unlock current plan for editing*** button to activate editing mode.
③ Add content using the ***Plan editor.***
④ Select elements on the ***Visualization map*** and modify them as needed.
⑤ Click on the ***Save and reload plan*** button to finish editing.
⑥ Perform a final check by opening the ***PC/Tablet visualization*** and the ***Smartphone visualization***.

9.5.2 Structure panel

Use the structure panel to select a *Plan*, *Layout* or *Widget* for configuration.

9.5.3 Additional tools and functions

Reorder Smart phone objects

In a newly created *Smartphone visualization*, the objects are listed in a default order. Click the  button at the bottom of the structure panel to reorder the objects.

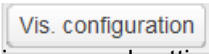
Access to PC/Tablet visualization

Click the  button at the bottom of the structure panel to open the final ***PC/Tablet visualization*** in the default browser.

Access to Smartphone Visualization

Click the  button at the bottom of the structure panel to open the final ***Smartphone visualization*** in the default browser.

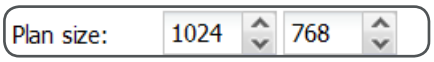
Visualization configuration

Click the 

button at the bottom of the structure panel to open the visualization general settings. See *Visualization configuration* on page 127.

Size of a Plan/Layout/Widget

In editing mode, modify the size of the selected visualization map using selectors at the bottom of the structure panel. See *Levels and Plans* on page 133.

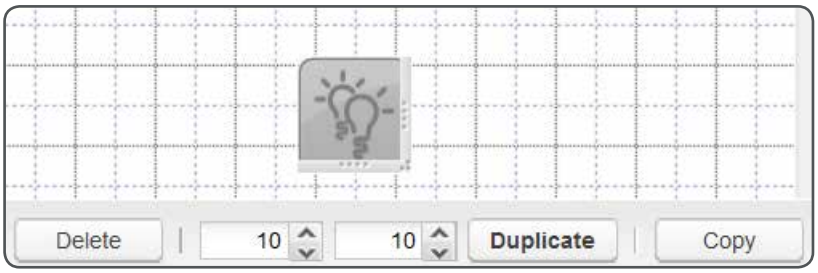


9.5.4 Visualization map

Use the visualization map to test the visualization and to edit while in editing mode. Toggle between visualization and editing modes using the buttons on the bottom of the plan editor.

Editing functions on the visualization map

In editing mode, the visualization map is shown with a 20 × 20 pixel grid.



When an object is selected, the buttons on the bottom of the visualization map can be used for the following functions.

Table 39: *Visualization map function buttons*

Delete	Delete the object.
Duplicate	Make a copy of the object. The duplicate is placed relative to the original according to the coordinates entered in the selectors (0, 0 places the duplicate on top of the original).
Copy	Copies the object to the clipboard, to paste into another plan/layout/widget.
Paste	Pastes the object from the clipboard. This button is hidden when the clipboard is empty.

9.5.5 Plan editor

Use the Plan Editor to add new elements to the visualization map and to modify existing elements.

Available elements

Table 40: *Plan editor elements*

General	Add, modify and safe elements, size and position of Elements. See *General functions in the plan editor* on page 139.
Object	Display/control the value of an *Object* in the ***Object*** list, i.e. a C-Bus object in a network with an application and a group. See *Objects* on page 144.
Link	Open another plan. See *Link* on page 140.
Text label*	Static text string, e.g. the name of the room. See *Text label* on page 142.
Image*	*Local static image stored in the **Vis. graphics** tab or external link to a web page, e.g. URL with dynamic content.* See *Image* on page 143.
Frame	Show internal/external web pages on the visualization page. See *Frame* on page 153.
Gauge*	An analogue graphic element to display values. See *Gauge* on page 152 .
Camera	Used to show a stream (MJPEG) from an IP camera. See *Camera* on page 155.
Graph*	Is used to show real time graphs of logged values. See *Graph* on page 153.

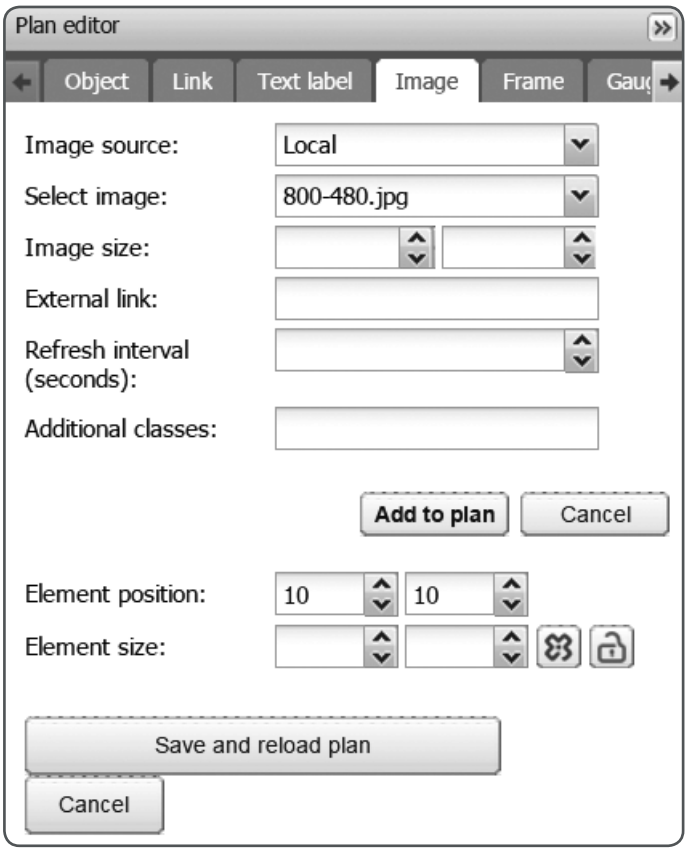
* These elements are not displayed in the Smartphone Visualization.

9.6 General functions in the plan editor

9.6.1 Actions in the plan editor

The following actions can be applied for all elements in the plan editor.

Plan editor (example)



Add a new element

① Click the ***Unlock current plan for editing*** button (if visualization mode is active).
② Select and edit parameter values.
③ Click the ***Add to plan*** button.

Modify an existing element

① Click the ***Unlock current plan for editing*** button (if visualization mode is active).
② Select element on the visualization.
③ Select and edit parameter values.
④ Click the ***Apply*** button.

Save or cancel changes before changing to visualization mode

- Click the ***Save and reload plan*** button to save your changes.
- Click the ***Cancel*** button at the bottom of the plan to cancel your changes.

Element position

The element will be placed on the visualization map in the position entered here (default 10,10 = top left corner).

Element size

- Optionally change the size of elements such as icons, images or frames here.
- Once the element is added to the plan, directly resize the element on the visualization map by dragging the nodes on the borders (stretch) or the corners (keep ratio).
- Click the ***Restore*** button to restore the element's original size.

9.6.2 Main steps of editing

Follow these steps to add or modify content:

① Select a *Plan*, *Layout* or *Widget* in the ***Structure panel***.
② Click the ***Unlock current plan for editing*** button to activate editing mode.
③ Add content using the ***Plan editor***.
④ To modify an element, select the element on the ***Visualization map***.
Position, copy, duplicate or delete elements on the visualization map. It is also possible to send values. Many object elements have additional functions in the PC/Tablet visualization. See *Objects* on page 144.
⑤ Click the ***Save and reload plan*** button to finish editing.
⑥ Perform a final check by opening the ***PC/Tablet Visualization*** and the ***Smartphone Visualization***. Text label, image, gauge and graph elements are only displayed in the ***PC/Tablet Visualization***. Each object, link and frame element can optionally be hidden in the ***Smartphone Visualization***.

After changing settings, refresh the browser to view the changes in the visualization. Refresh via browser menu or use the keyboard short-cuts [Crtl] + [N] or [Crtl] + [F5].

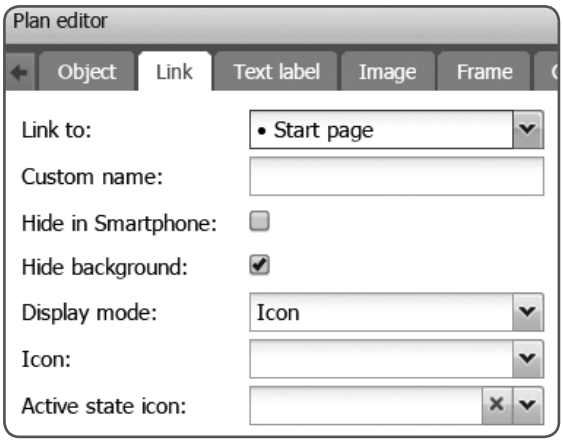
9.7 Link

Navigation links between the visualization pages are automatically created and can be used without any additional configuration work. This is done in the ***Vis. structure*** tab, where plans must always be linked to a level or a sub-level. All pages can be accessed via the sidebar on the left of the visualization page.

In addition, *Links* can be added to the visualization map. Use an icon, text or transparent surface as a link to other pages. To configure a link, open the ***Link*** tab of the ***Plan editor***.

9.7.1 Parameter

Plan editor with activated ***Link*** tab

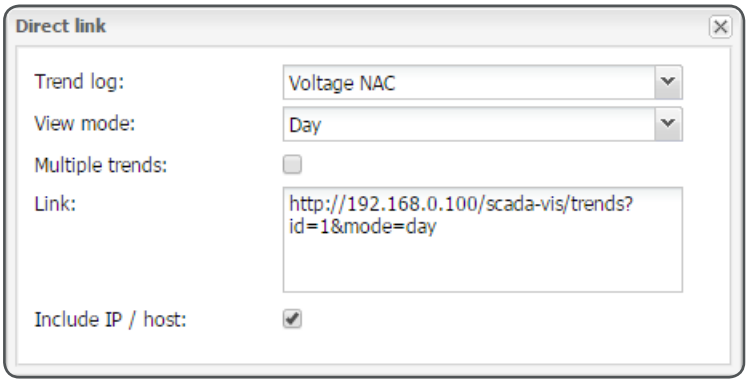


'Link to:' options

Table 41: *'Link to' options*

Start page	The start page of the visualization.
Name of plan	A plan created in the ***Vis Structure*** tab.
Trend logs	Default visualization pages (has to be configured).
Scheduler	Default visualization pages (has to be configured).
External link	Opens a web page (http://...). Can be used to view one trend. Copy the trend path via the ***Direct link*** button on the ***Trend log*** tab. See the next screen shot (Direct link).
Next plan	Navigation link to the next plan.
Previous plan	Navigation link to the previous plan.

When configuring an external link to a trend log, it is possible to link to a trend log in the same Automation Controller or another one. When linking to another Automation Controller, tick the *Include IP/host* option in the Direct link window to display the complete IP address. See *Trend logs* on page 117.



Custom name

If setting Display mode to *Value* (see *Display mode* below), the custom name is displayed as the link.

Hide background

The background of the icon is not shown.

To create a transparent area for a link, select an empty SVG file for the icon and then hide the icon background.

Display mode

Icon	An icon is used as the link.
Value	The text of the *Custom name* parameter is used as the link.

Active state icon

Set another image to display in the icon when the linked page is opened. For example, create both grey (standard) and yellow (active) versions of an icon image and then assign the grey image to *Icon* and the yellow image to *Active state icon*.

9.7.2 Examples

Link with icons

In this example, when a plan is opened either by the link icon or by the sidebar menu, the active state icon for the link is shown in green colour and the other icons are shown in grey colour. Three link icons are placed on a layout associated with the three plans.

Display mode	Icon
Icon	Icon with grey colour
Active state icon	Icon with green colour

Link with text

In this example, a user can click on some text to open a plan. Three link texts are placed on a plan with three rooms.

Custom name	Text (e.g. Main Office)
Display mode	Value (= Custom name)
Font size/color	As required

Link as transparent area

In this example, the user can click transparent links that are positioned over rooms on part of a building plan. Empty SVG files can be used as icons (see *Hide background* on page 141).

Display mode	Icon
Icon	Empty SVG file
Hide background	Activated (makes icon with empty.svg transparent)

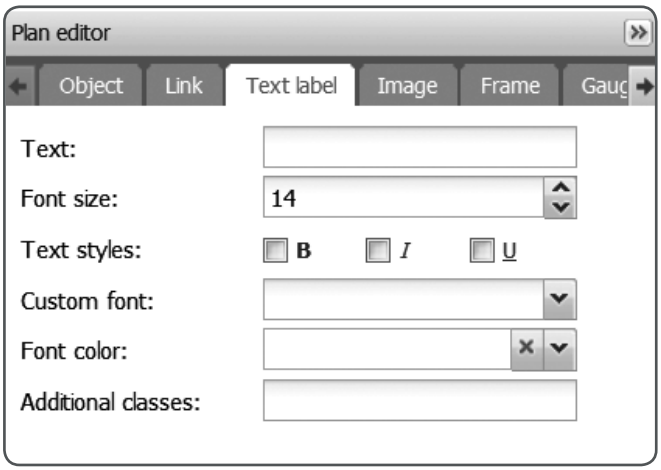
9.8 Text label

A *Text label* is a static element. Add text labels and position them on the visualization map. Open the ***Link*** tab of the ***Plan editor*** to edit a text label.

Text labels are only visible in the ***PC/Tablet Visualization***.

9.8.1 Parameters

Plan editor with activated *Text label* tab



Set the *Font size*, *Text Style*, *Font* and the *Font Color*.

9.9 Image

Images can be positioned and resized on the visualization map. Use l*ocal static images stored in the **Vis. graphics** tab, or remote web pages (e.g. a URL with dynamic content)*. Open the ***Image*** tab of the ***Plan editor*** to configure an image.

Images are only viable in the ***PC/Tablet Visualization*** view.

9.9.1 Parameters

Plan editor with activated *Image* tab

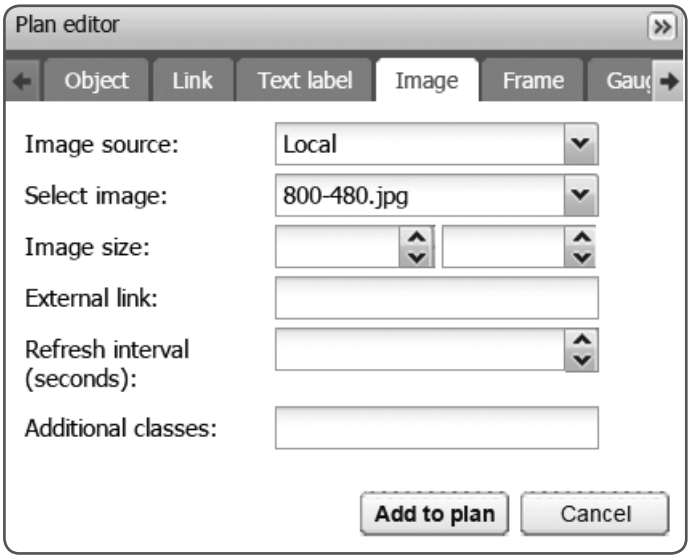


Image source

Local Select an image previously added to ***Vis graphics → Images / Backgrounds***.

Remote Select the source URL of the image (http://....) This option is useful for example to grab dynamic weather forecast images.

Image size

- After the image has been added to the plan, resize it directly on the visualization map by dragging either the nodes on the borders (stretch) or the nodes on the corners (keep ratio).
- It is possible to specify the image size by entering numbers.
- Restore the image to its original size by clicking the ***Restore*** button on the bottom of the ***Image*** tab.

External link (optional)

Enter a URL to open an external web page (e.g. http://www.mywebpage.com) when the image is clicked/pressed.

9.10 Objects

In a visualization, *Objects* are used to control or monitor different functions. Depending on the applications and the function, different *Control types* and parameters can be set.

To configure a control type, open the ***Object*** tab of the ***Plan editor***.

Path: ***Visualization*** tab → ***Structure*** → ***Plan editor*** → ***Object*** tab.

Steps

① Select an object in the ***Object*** tab of the plan editor.
② Click the ***Visualization parameters*** button:
  – Select the control type, additional parameters and then save.
  – These parameters can also be edited in the ***Objects*** list (see *Edit and test objects* on page 106).
③ Select other parameters in the ***Objects*** tab of the plan editor.
④ Add the configured object to the plan with the ***Add to plan*** button.
⑤ Check the function:
  – On the plan it allows sending of values.
  – Complex control types like the *Circular slider* are not visible until the plan is reloaded. Click the ***Save and reload plan*** button on the plan editor.
  – On the ***PC/Tablet Visualization*** and the ***Smartphone Visualization***, check the final functions of the element. Open the visualizations from the start page.

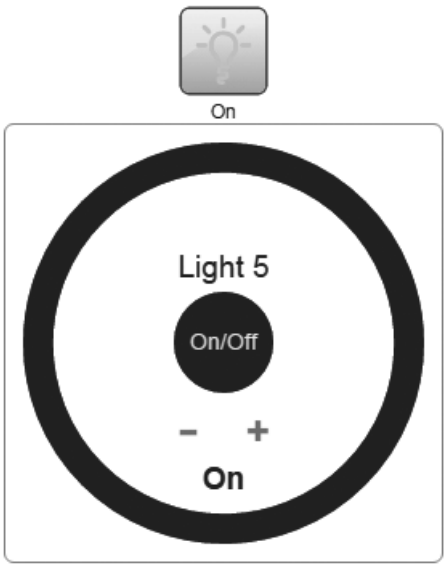
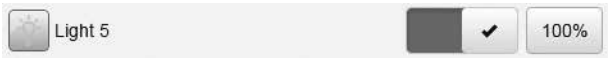
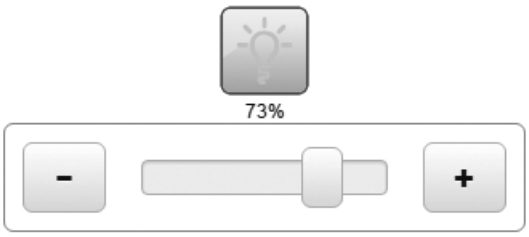
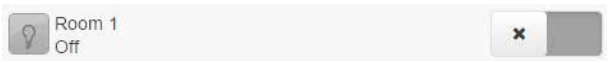
After changing the settings, refresh the browser to see the changes in the visualization. Refresh via the browser menu or use the keyboard short cuts [Crtl] + [N] or [Crtl] + [F5].

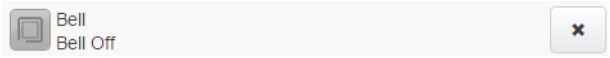
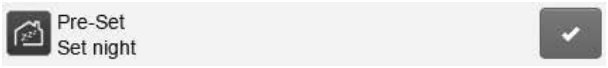
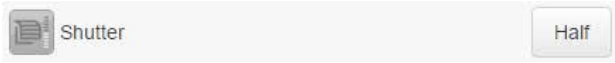
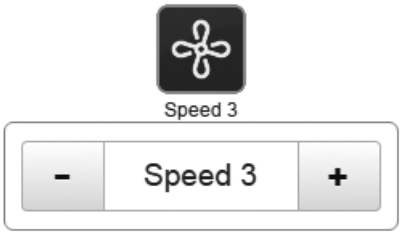
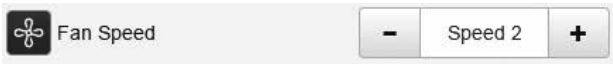
To use names (tags) instead of values, add and edit levels in the ***Objects*** list (see *Edit and test objects* on page 106). In the case of the shutter/blind, for example, add levels for additional control functions and optionally use the *Tag map* (see *Add new tags* on page 111).

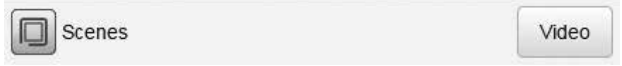
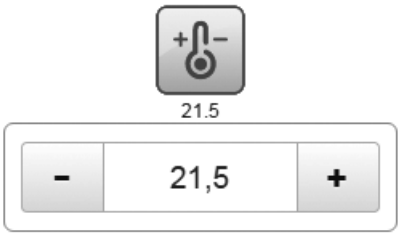
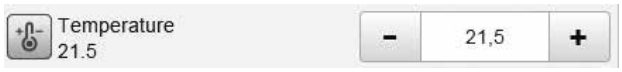
9.10.1 Overview of control types

Different control types are available, depending on the application.

Table 42: *Control types*

Application/Description	Control type
48-127 Lighting and related **228 Measurement** **250 User parameter values** • Custom name • On/Off button • Nudge/Steps: –, + • Value or name (tag) for levels is displayed	**Circular slider (PC/Tablet):** 
• Circular slider opens via click on icon	**Circular slider (Smartphone):** 
48-127 Lighting and related **228 Measurement** **250 User parameter values** • Vertical or horizontal slider • Nudge/Steps: – and +	**Slider (PC/Tablet):** 
• Custom name • Horizontal slider • Nudge/Steps: – and +	**Slider (Smartphone):** 
48-127 Lighting and related **250 User parameter boolean** • Value or name (tag) for level is displayed	**Toggle (PC/Tablet):** 
• Custom name • Value or name (tag) for level is displayed	**Toggle (Smartphone):** 
48-127 Lighting and related • Sends a value when button is pressed and 0 when released • Value or name (tag) for level is displayed	**Doorbell (PC/Tablet):** 

Application/Description	Control type
• Custom name • Sends a value when button is pressed and 0 when released • Value or name (tag) for level is displayed	**Doorbell (Smartphone):** 
48-127 Lighting and related • Sends a value when button is pressed • Value or name (tag) for level is displayed	**Pre-Set (PC/Tablet):** 
• Custom name • Sends a value when button is pressed • Value or name (tag) for level is displayed	**Pre-Set (Smartphone):** 
48-127 Lighting and related • Sends a value when button is pressed (level) • 3 levels are pre-set (level translation mode): -Close (0) -Stop (5) -Open (255) • Additional levels can be added to reach positions (e.g. 128 = Half)	**Shutter/Blind (PC/Tablet):** 
• Custom name • List with levels opens when button is pressed	**Shutter/Blind (Smartphone):** 
48-127 Lighting and related • Sends next value when + or – button is pressed • Cycle: -max. to min. level -min. to max. level • Levels: names and values must be set • Actual level is displayed	**Cycle/Fan (PC/Tablet):** 
• Custom name • Sends next value when + or – button is pressed • Actual level is displayed	**Cycle/Fan (Smartphone):** 

Application/Description	Control type
202 Trigger Control (Set scenes) • Sends a value when button is pressed (action selector) • Levels: can be edited in the ***Visualization*** tab or ***Scenes*** tab • Actual level is displayed	***Trigger select*** **(PC/Tablet):** 
• Custom name • Sends a value when button is pressed • Actual level is displayed	***Trigger select*** **(Smartphone):** 
228 Measurement **250 User parameter values** • Sends next value when + or – button is pressed • Direct input • Decimal places • Nudge/steps: step width • Min. and max. value	**Direct input/Step +/– (PC/Tablet):** 
• Custom name • Sends next value, when + or – button is pressed • Direct input	**Direct input/Step +/– (Smartphone):** 

9.10.2 Visualization parameters and control type

Control types are described in *Overview of control types* on page 145.

Access the visualization parameters in two ways:

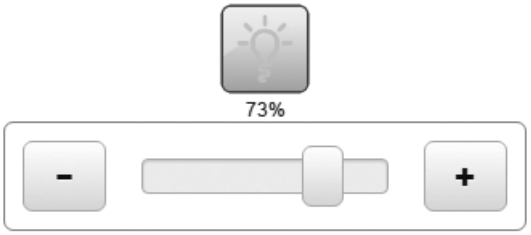
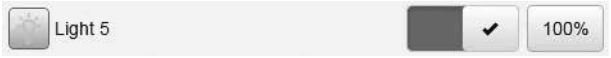
- Click the ***Visualization parameters*** button in the ***Object*** tab of the plan editor.
- Click the ***Vis. parameters*** button in the ***Objects*** list (see *Edit and test objects* on page 106).

When starting in the ***Object*** tab of the plan editor, select an object there first and then go to *Visualization parameters* and select the *Control type*.

Circular slider and *Shutter/Blind* control types are explained below.

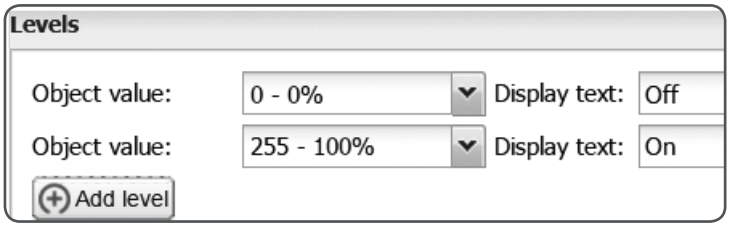
9.10.3 Circular slider

Example

Application/Description	Control type
48-127 Lighting/Lighting like • Vertical or horizontal slider • Nudge/Steps: – and +	**Slider (PC/Tablet):** 
• Circular slider opens via click on icon	**Circular slider (Smartphone):** 

Visualization parameters

Table 43: *Visulaization parmaters for circular slider*

Control type	Circular slider
Value display	• Percentage (0-100) • Level (0-255) • **Tags** In the example, *Tags* is selected. The levels are set in the ***Objects*** list of the ***Levels*** column. Path: ***Configurator*** → ***Objects*** tab → ***Objects*** list. 
Ramp rate	Set the time to reach the value (from 0 to 255): • Minimum: 0 s (instantaneous) • Maximum: 15 min.
Minimum value	0 Example: 25 = Dimming starts at 10%.
Maximum value	255 Example: 230 = Dimming stops at 90%.
Nudge/step value	Step width for control: 1-255 (e.g. 25 = steps in 10%).
Slider colour	Select a colour for the slider.
Background colour	Select (pre-set: no colour).
Round line cap	Use a rounded shape at the beginning and end of the circle line.
Hide title	Do not display the default or custom name.
Hide nudge buttons	Disable the step + and – buttons.
Line thickness	Select the line thickness.
Size	Select the control size.
Custom On/Off label	Change the text displayed for the On/Off button in the center of the controller.

When all parameters are selected, press the ***Save*** button. Now set the parameters in the ***Object*** tab of the plan editor (see *Object element parameters* on page 151).

9.10.4 Shutter/Blind

Example

Application/Description

48-95 Lighting/Lighting like

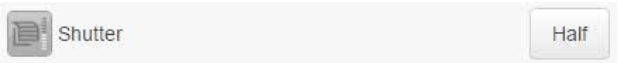
- Sends a value when button is pressed (level)
- 3 levels are pre-set (level translation mode):
  -Close (0)
  -Stop (5)
  -Open (255)
- Additional levels can be added to reach positions (e.g. 128 = Half)

Control type

Shutter/Blind (PC/Tablet):

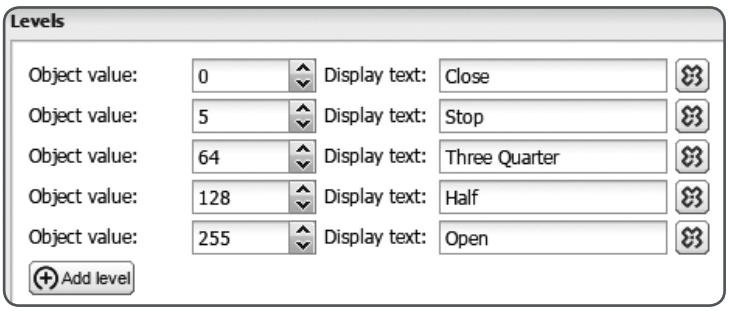
Shutter/Blind (Smartphone):

- Custom name
- List with levels opens when button is pressed



Visualization parameters

Table 44: *Visualization parameters for shutter/blind*

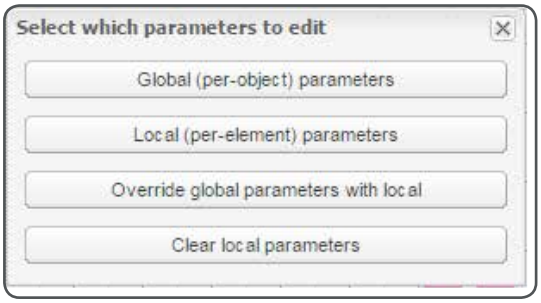
Control type	Shutter/blind
Value display	• Percentage (0-100) • Level (0-255) • **Tags** *Tags* must be selected. Set the levels in the ***Objects*** list of the ***Levels*** column. The levels *Close* (0), *Stop* (5) and *Open* (255) are pre-set. These values are used in the level translation mode. Each level triggers a command. For added convenience, set additional position values (levels) in the range 3% to 97% (7-247). In the example, the pre-set values and 2 additional levels are set in the ***Objects*** list of the ***Levels*** column. Path: ***Configurator*** → ***Objects*** tab → ***Objects*** list. 
Show icons in PC/ Tablet	Icons can be displayed instead of names. Add icons in the ***Object*** tab of the plan editor, using the ***Additional icons*** button. Path: ***Configurator*** → ***Visualization*** tab → ***Plan editor*** → ***Object*** tab.

9.10.5 Global and local parameters

By default, use global visualization parameters for each object. If using several elements to control an object, use local visualization parameters.

Initially, global and local parameters are the same. Change the local parameters, for example, to use one button for dimming with a slider and another to switch on and off with a toggle button. In this case, select different control types in the local parameters of each object element.

Access global and local parameters via the ***Visualization parameters*** button in the plan editor ***Object*** tab.



Local parameters cannot be set in the ***Objects*** list.

9.10.6 Object element parameters

Configure object elements in the ***Object*** tab of the plan editor.

After selecting an object, open the *Visualization parameters* and select the *Control type*. Now select the object's element parameters.

Table 45: *Object element parameters*

Object	Select object from the drop down list.
Visualization parameters	Click this button and select the *Control type* and other parameters, then save. Then continue setting the following parameters .
Custom name	A unique name for the object. *Custom name* is used for some control types and for the ***Smartphone Visualization***.
Read only	Select this option to only monitor, not control, values.
Hide in smartphone	Select object is not visible in ***Smartphone Visualization***.
Hide background	Show icon without background.
Pin code	Add a PIN to protect the object element value. Once set, the PIN is required whenever an attempt is made to change the value.
Widget	Assign a *Widget* that opens when the element is pressed. The widget must first be created so that it appears in the list. A widget cannot be tested in the editor mode, only in ***PC/Tablet Visualization***.
Display mode	Display mode refers to the first level of visualization. The options include: • Icon and value • Icon only • Value only. Additional elements can be opened for the control. See *Overview of control types* on page 145.
Default icon	The icon which is displayed when no other icon is assigned to the actual value (See **Additional icons** below). For same control types, set an *On icon* and an *Off icon*.
Smartphone icon	(Optional) If different icons are required between the PC/Tablet and Smartphone visualizations, use this parameter to set a different icon for the Smartphone visualization.

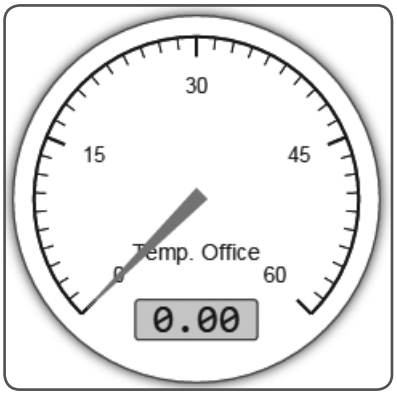
Font size	Affects the font size of the value display
Text style	Affects the text style (e.g. bold, italic) of the value display
Show value background	A solid background is displayed behind the value.
Show control (Inline in PC/Tablet)	Shows the control element instead of the icon button. (PC/Tablet only.)
Additional icons	Shows different icons, depending on the current value. For each new icon, define a minimum and a maximum value. In operation, if the object value does not fall within the range of any of the additional icons, the default icon is displayed.

9.11 Gauge

Use a gauge to display values. The gauge is typically used to display measured values such as temperature, brightness or energy.

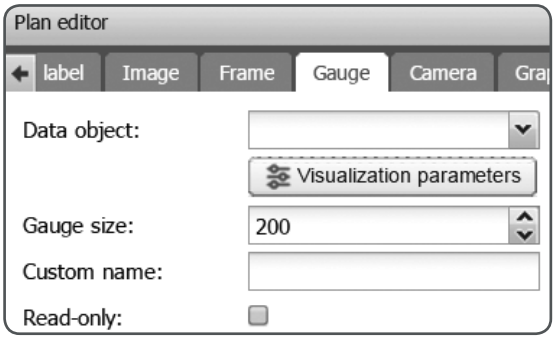
A gauge is only visible in *PC/Tablet Visualization* view.

To configure a gauge, open the ***Gauge*** tab of the ***Plan editor***.



9.11.1 Parameters

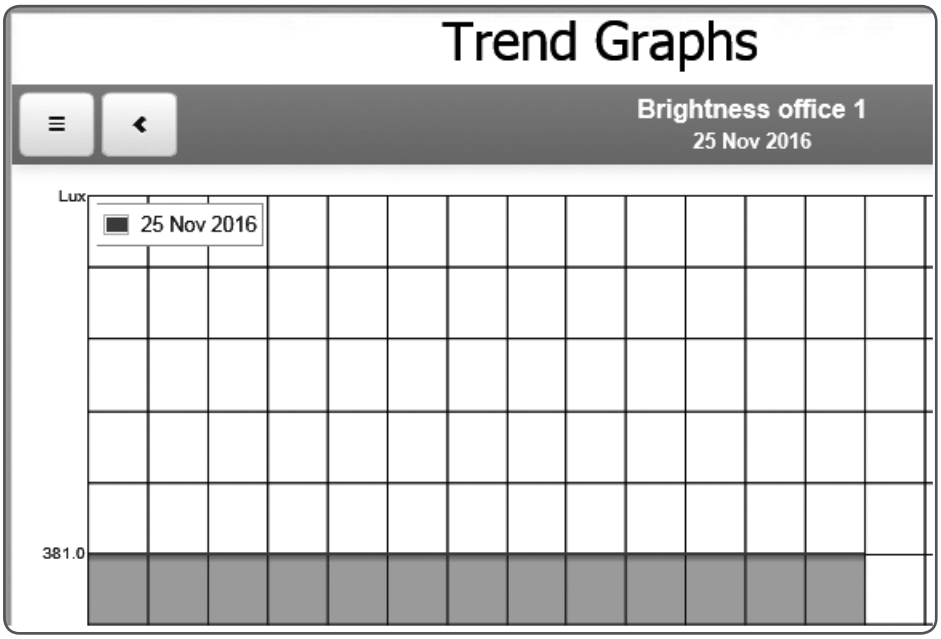
Plan editor with activated *Gauge* tab



Data object	Select a C-Bus object. A typical application is measurement (228).
Gauge size	Minimum size is 100, maximum size is 500.
Custom name	The name to display in the gauge (e.g. *Temp. office*).
Read only	**Enabled:** The value is displayed but cannot be changed. **Disabled:** The value can be changed by clicking/pressing the gauge.
Visualization parameters	Click the ***Visualization parameter*** button to open the visualization parameters (see *Objects* on page 144). Define *Global Parameters* for an object or *Local parameters* for an element. (The local parameters are only applied to the configured gauge.)

9.12 Frame

Use a frame to display internal or external web pages in the visualization. *Schedulers* and *Trend logs* can be shown in a frame on a visualization page. To configure a frame, open the ***Frame*** tab of the ***Plan editor***.



9.12.1 Parameters

Plan editor with activated *Frame* tab

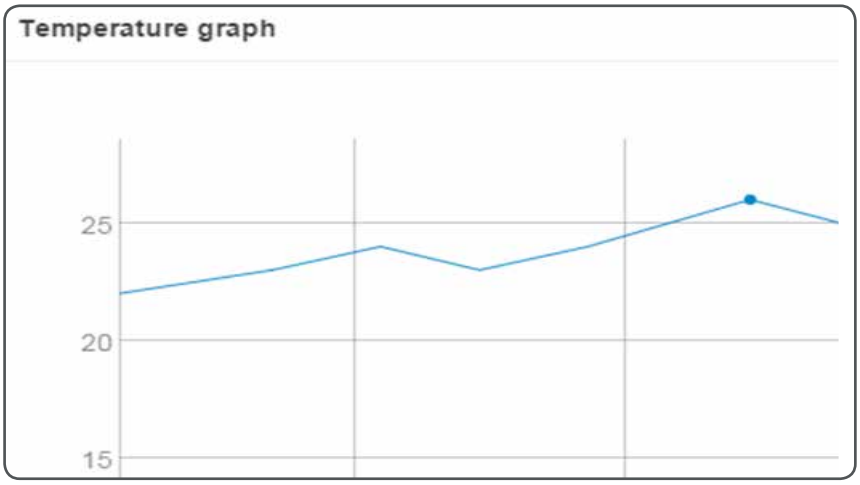
Source	• URL = http://... • Scheduler • Trend logs
Frame size	The width and height of the frame
Refresh interval	The rate at which element values are updated. The refresh interval options are 0, or a setting between 1 s and 3600 s.

- Frames do not display well in *Smartphone Visualization* view and will be hidden in this visualization.
- Stretch the frame to maximum width if *Scheduler* or *Trend* is used. The recommended minimum width is 1024.
- Not all web pages work correctly in frames. Test external web pages within frames using the *PC/Tablet Visualization* view.

9.13 Graph

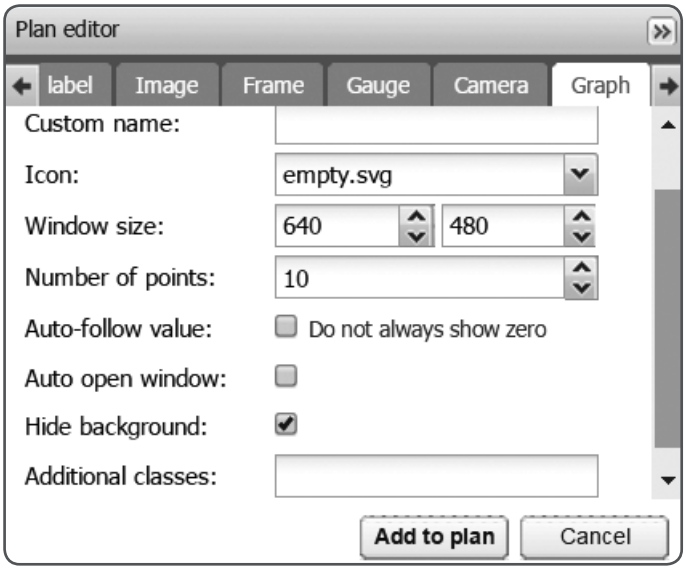
A graph shows current and previous monitored values in the *PC/Tablet Visualization* view, e.g. logged temperature values. To configure a graph, open the ***Graph*** tab of the ***Plan editor***.

Logging must be enabled on the ***Objects*** tab for the application being used.



9.13.1 Parameters

Plan editor with activated *Graph* tab



Data object	Select an object. A typical application is measurement (228).
Custom name	The name which is displayed as the graph title (e.g. *Temperature graph*).
Icon	Select an icon for the button that opens the graph.
Window size	The width and height of the graph.
Number of points	Number of data points to show in the graph (5-200).
Auto open window	Select this option to open the graph window automatically when the plan opens.
Auto-follow value	Select this option to improve graph resolution for objects which never reach zero value (e.g. CO_2 level).
Hide background	Select this option to make the background of the icon transparent.

- Graphs are not visible in the *Smartphone Visualization* view.

- Use a gauge to show an actual measured value (see *Gauge* on page 152 and *Link* on page 140) and then configure the gauge to open the graph when clicked. To set this up, do the following:
  ① Create and save an empty SVG file.
  ② Add the empty SVG file to the ***Images/Backgrounds*** tab in the ***Vis. graphics*** tabs.
  ③ Select the empty SVG file as *Icon* (Parameter of the Graph).
  ④ Activate the check box *Hide background* (Parameter of the Graph).
  ⑤ Click the ***Add to plan*** button to get the icon on the *Visualization map* (Parameter of the Graph).
  ⑥ Position the icon (empty SVG) over the gauge and adapt the size of the icon to the gauge (*Visualization map*).

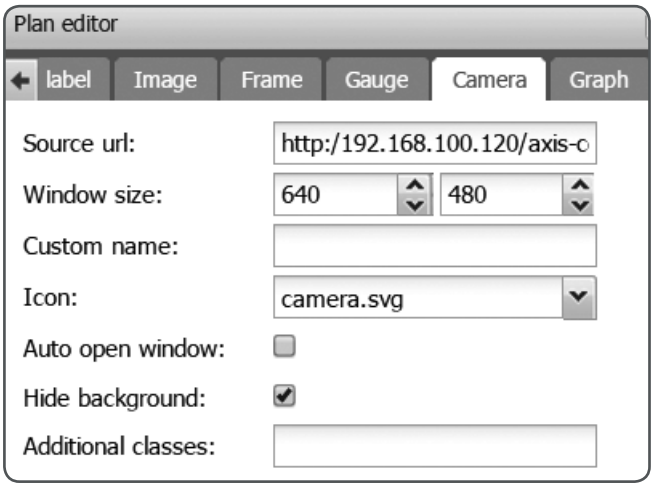
9.14 Camera

The video stream of an IP camera can be displayed on a visualization page.

- Only cameras which support HTTP MJPEG streaming in a web browser can be visualized. The Automation Controller redirects the stream from the camera to the browser.
- If the camera is external to the Automation Controller network, the IP of the camera needs to be port-forwarded through the router. When adding the external camera, use the IP with the correct port (IP:port).
- If using ***Smartphone Visualization***, check the plan in this view and adapt the window size.

9.14.1 Parameters

Plan editor with activated *Camera* tab



Source URL	The source address of the video stream.
Window size	The width and height of the camera view window.
Custom name	A unique name for the camera view.
Auto open window	Select this option to open the camera view automatically when the *Plan* opens.
Hide background	Make the background of the icon transparent.

10 User access

Add users with individual access rights. By default, users can access the following modules from the start page:

- PC/Tablet Visualization
- Smartphone Visualization
- Scheduler
- Trends.

Access can be partially or fully given, or denied, for each module. Partial access allows individual selection of pages. Access can be given or denied for each visualization page, scheduler and trend.

The Network Automation Controller is designed for a maximum of 50 users for visualization. This limit is not physical but dependent on maximum CPU load.

The Wiser for C-Bus Automation Controller is limited to a maximum of 8 users.

User access settings provide different levels of protection:

- **Unrestricted access:** All users are given access to all pages without restrictions.
- **Common restricted access:** All users are given access to all pages via a common PIN.
- **Individual restricted access:** Access to visualization, scheduler and trend logs is controlled via individual user login and password so that access can be defined for each user.

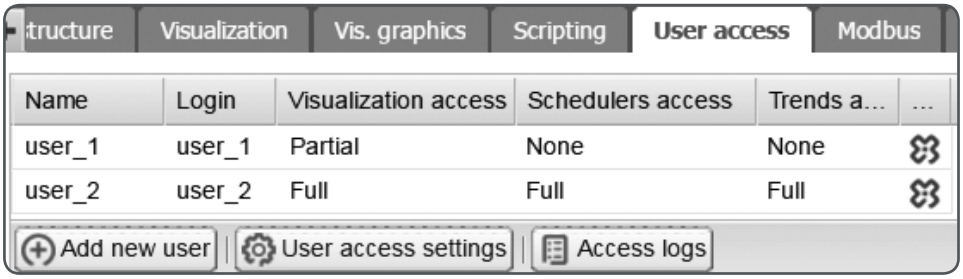
Access can be given to the start page or directly to one of the modules:

- PC/Tablet Visualization
- Smartphone Visualization
- Scheduler
- Trends.

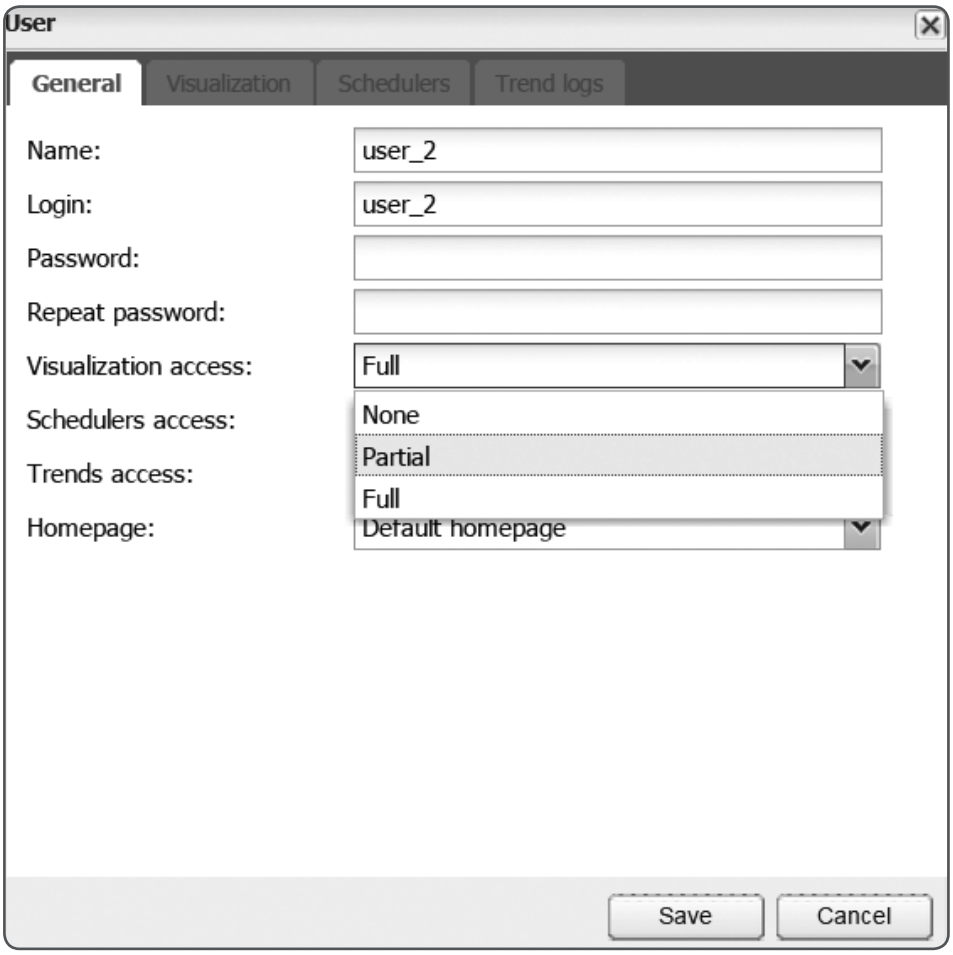
Access to the configuration is defined under *Admin access* (See *Access to the Controller* on page 27). The default address is: http://192.168.0.10/scada-main.

10.1 Add and edit user

To add a user, click the ***Add new user*** button in the ***User access*** tab.



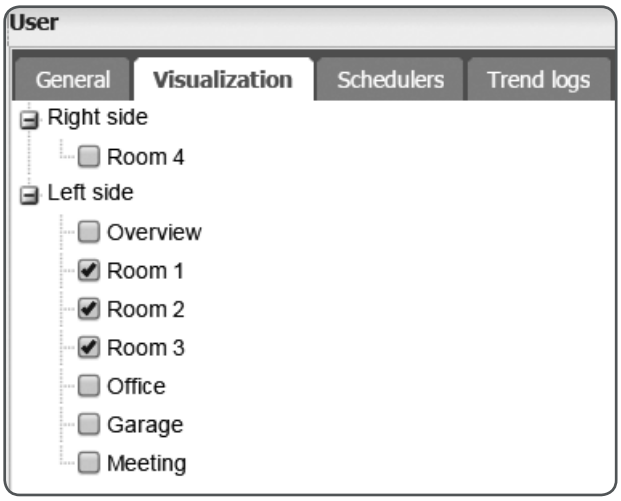
The user window is displayed:



10.1.1 User parameters

Table 46: *User parameters*

Name	User name
Login	Length 2 to 20 characters, accepted characters: -, _, a-z, A-Z, 0-9.
Password	Length: 8 to 20 characters. Any character accepted.

Visualization/ Schedulers/Trend access	For each module, access can be partially or fully given, or denied: • None • Partial • Full. If partial is selected, access to visualization, scheduler and trend pages can be individually configured. The following example shows a configuration with access to 3 pages of the visualization for the specific user: 
Homepage	This parameter depends on selected Default homepage in the User access Settings (see *User access settings* on page 158). If the Start page is selected in the user access settings, the start page is the homepage for all users. From the start page, the users get access to the other modules as configured. If another page is selected in the user access settings, an individual homepage can be selected for each user. If a user gets access to different modules, select the *Start page*: • Start page • PC/Tablet visualization • Smartphone • Schedulers • Trend logs.

10.2 User access settings

User access settings are applied for all users.

Click the ***User access settings*** button in the ***User access*** tab.

Table 47: *User access settings*

Disable password for visualization	• If the password is disabled, full access to visualization, schedulers and trends is given to everyone. A logout is not required. The user settings are not applied (see *Add and edit user* on page 157). If the *Visualization PIN code* is set, this common code is required to access visualization, schedulers and trends. • If the password is enabled, a login is required and the user settings are applied (see *Add and edit user* on page 157).
Default homepage	• If the *Default homepage* is set to *Start page*, this page is the home page for all users. • If the *Default homepage* is set to another page and a password for visualization is required, the homepage depends on the user settings (see *Add and edit user* on page 157).
Visualization PIN code	Access can be protected with a common PIN code (3 to 8 digits). The PIN code remains valid until the browser is closed.

Direct access via browser address

Table 48: *Direct access browser addresses*

Page/Module	Address (with default IP address)
PC/Tablet Visualization	http://192.168.0.10/scada-vis
Smartphone/Visualization	http://192.168.0.10/scada-vis/touch
Scheduler	http://192.168.0.10/scada-vis/schedulers
Trends	http://192.168.0.10/scada-vis/trends
Start page	http://192.168.0.10/home
Configurator	http://192.168.0.10/scada-main

Direct access is also protected with the login and password.

11 Scripting

A script is a small, non-compiled program, written in the scripting language LUA. LUA is a general scripting language used in many products/computers. A reference manual is available for purchase at www.lua.org

Scripting can be used for wide ranging of functions. Examples of general commands are:

- If-elseif-else-then
- While
- Repeat and for loops
- Math functions
- Compare functions
- Logic functions
- I/O functions.

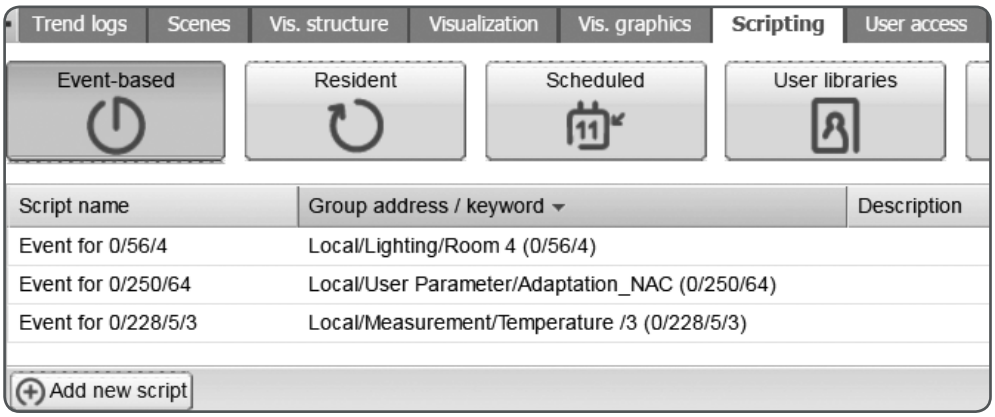
LUA can also be used for special functions, such as:

- Calculate sunrise/sunset
- Send e-mail
- Control of RS-232
- Control of Modbus.

Logic functions that can an be created with C-Bus products can also be created with LUA scripts, and much more.

11.1 Prepare a script

When creating a new script, first decide how the script will be triggered (started). Then choose a script type in the ***Scripting*** tab.



The following script types are available:

- Event-based scripts:
  - Scripts are triggered when the associated *Objects* are updated. The composed *Group Address* of an application or a *Keyword* can be used. With a keyword, a script can be updated by all objects assigned to the keyword.
  - Event-based scripts are the most commonly used.
- Resident scripts are triggered according to specified cycle time in seconds (0-60 s).
- Scheduled scripts:
  - Scripts are triggered according to specified time and/or date.
  - Scripts can be triggered by the specific minute, hour, day of the week, day of the month and month of the year.

  - Scripts can be triggered more frequently, such as every x minutes/hours/days.
- Start up scripts are triggered every time the system starts (power up, reboot or hardware reset).

11.1.1 Steps to prepare a script

① Open the ***Scripting*** tab. (Path: ***Configurator*** → ***Scripting***)
② Select the type of script with one of the following buttons:
  - ***Event-based***
  - ***Resident***
  - ***Scheduled***
  - ***Start-up (init) script***.
③ Press the ***Add new script*** button at the bottom of the ***Scripting*** list, edit the parameters and save. A new line in the in scripting list is displayed.
④ Start editing the script. Click the icon in the ***Editor*** column of the scripting list.

If the script editor is already open, event-based scripts can also be opened from the ***Objects*** list.

11.1.2 Parameters for event-based scripts

Event-based scripts are triggered (started) when the associated *Objects* are updated. The composed *Group Address* of an application (see *Add new objects* on page 42) or a *Keyword* can be used. See *Additional parameters for all applications* on page 105.

Click the ***Event-based*** button and set the following parameters.

Table 49: *Event-based script parameters*

Script name	The name to display in the scripting list.
Group address/ keyword	*Group addresses* or *Keywords* can be entered manually or selected from the drop-down list. All composed *Group addresses* from the ***Objects*** list are displayed. *Keywords* already assigned to the objects are also displayed.
Active	When *Active* is selected, the script is enabled. Otherwise, the script is disabled. This parameter can also be set from the scripting list.
Category	An optional name that can be used to group scripts in the ***Scripting*** list and the *Print script listing* (***Scripting*** tab → ***Tools*** button).
Description	Optional.

To edit the new script, click the corresponding icon in the ***Editor*** column of the scripting list.

11.1.3 Parameters for resident scripts

Resident scripts are triggered according to a specified cycle time in seconds (0–60 s). Click the ***Resident*** button and set the following parameters.

Table 50: *Resident script parameters*

Script name	The name to display in the scripting list.
Sleep interval (seconds)*	Interval (0–60 s) after which the script will be executed.
Active	When *Active* is selected, the script is enabled. Otherwise, the script is disabled. This parameter can also be set from the scripting list.

Category	An optional name that can be used to group scripts in the ***Scripting*** list and the *Print script listing* (***Scripting*** tab → ***Tools*** button).
Description	Optional.

To edit the new script in the list press the script icon in the ***Editor*** column.

* Avoid using a short sleep interval, especially 0. Resident scripts with sleep interval 0 have a high impact on the CPU load.

11.1.4 Parameters for scheduled scripts

Click the ***Scheduled*** button and set the following parameters.

Table 51: *Scheduled script parameters*

Script name	The name to display in the scripting list.
Minute/Hour/Day of the month	Uses the **cron** format for date/time parameters. Lookup *Help* for more information about the date/time format. Example values for minutes: **Value** — **Execute** * — Every minute */20 — Every 20 minutes 20 — When the minute is 20 1-10 — Every minute from 1 to 10 (inclusive) 15,50-51 — When the minute is 15, 50 and 51.
Month of the year	(Optional) Select the check box if required.
Day of the week	(Optional) Select the check box if required.
Active	When *Active* is selected, the script is enabled. Otherwise, the script is disabled. This parameter can also be set from the scripting list.
Category	An optional name that can be used to group scripts in the ***Scripting*** list and the *Print script listing* (***Scripting*** tab → ***Tools*** button).
Description	Optional.

To edit the new script in the list press the script icon in the ***Editor*** column.

11.1.5 Start up script

The start up script runs each time the system starts (e.g. power up, reboot or hardware reset). There are no parameters to edit for the start up script.

Click the ***Start up (init) script*** button to open the script editor.

If the script editor is already open, open the start up script for editing via the ***Scripts*** tab on the left side of the editor.

Example

In this example, the start up script is configured to wait for C-Bus to come online and then switch a floor light on at full brightness (level 255) immediately after the system starts (or is restarted). The composed group address of the light is: (0/56/3).

```
1 -- init script (comment)
2 WaitForCBusStatus (nil)
3 SetCBusLevel(0, 56, 3, 255, 0)
```

Backup and restore

Backup and restore the start up script (see *Backup and restore libraries* on page 170).

Path: ***Configurator*** → ***Scripting*** tab → ***User libraries*** button.

11.2 Script editor

Existing event-based, resident and scheduled scripts are located in the corresponding ***Script*** list (see *Prepare a script* on page 160). Click the corresponding *Script icon* in the list to open the script editor. The editor opens directly when the *Start-up (init)* script button is pressed.

Path: ***Configurator*** → ***Scripting*** tab → ***Event-based***, ***Resident*** or ***Scheduled*** key → ***Script*** list.

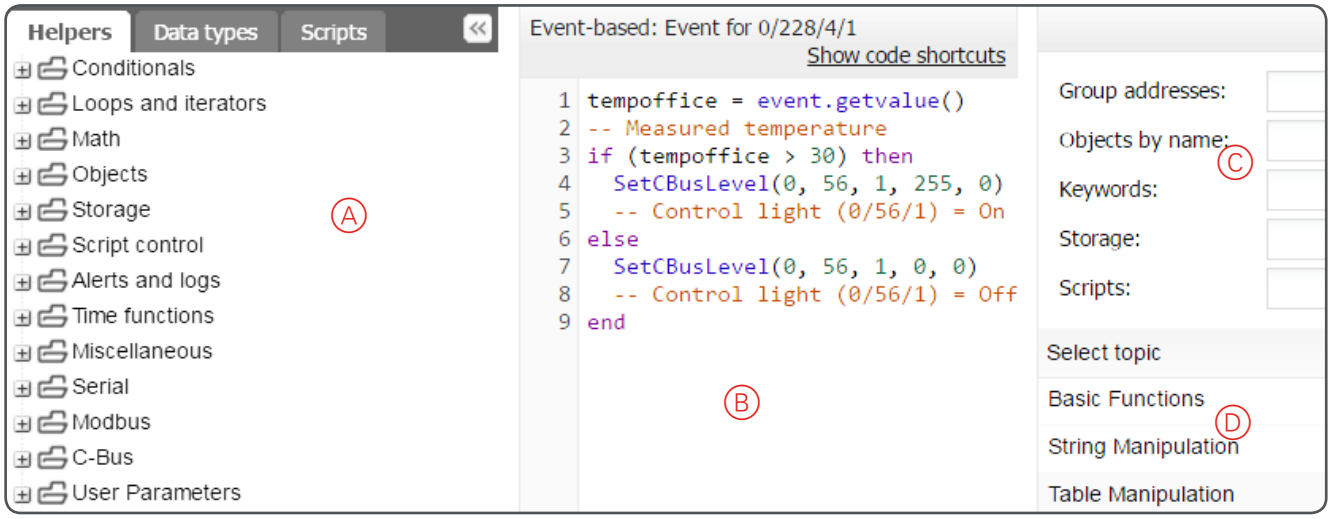
Event-based scripts also open when the *script icon* is clicked in the ***Event script*** column of the ***Objects*** list.

11.2.1 Editor

The editor panel is displayed in the center of the script editor (see Ⓑ in the next screen shot). Here, type script commands in line by line. The left and right panels of the script editor display features to support script editing.

Example

The following screen shot shows the editor with the short event-based script.



Ⓐ Helpers: code snippets (including comments for help, copying to editor is possible)

Ⓑ Editor

Ⓒ Lists: objects, storage values and scripts (copying to editor is possible)

Ⓓ Select topic: code and help information (copying of code to editor is possible).

This event-based script is triggered when a temperature value is sent. The script causes a control light to be switched on when the measured temperature is higher than 30 °C. In this script, four different functions are used.

Table 52: *Event-based script functions*

Line Ⓑ	Functions	Description	Helpers Ⓐ
1	`event.getvalue()`	The variable gets the temperature value of a C-Bus application, in this case a measurement application with the composed group address (0/228/4/1).	*Objects →* *Get event value*
3, 6, 9	`if-then-else-end`	Conditional function that includes actions when the condition is true and other actions when the condition is not true.	*Conditionals →* *If - Then - Else*
4, 7	`setCBus(net, app,group, value, ramp rate)`	Sends a level command at a given ramp rate for the network, application and group. In this example, the values 0 and 255 are sent with the composed group address (0/56/1) found on the right side of the editor Ⓒ. This composed group address (0/56/1) must be adapted to the syntax of the *SetCBusLevel* command: `0, 56, 1, x, 0`	*C-Bus →* *Set C-bus level*
2,5,8	`-- (2x dash)`	Comment line	

In this example, the ***Helpers*** tab Ⓐ can be used to click a function and add it to the editor. The variable parts of the script can then be replaced. For example, in line 3 of the script, `(condition)` is replaced with `(tempoffice > 30)`.

Helpers

The ***Helpers*** tab contains predefined code snippets. Comment lines are included in the snippets to help edit the code. Click a Helper snippet to add the code to the Editor.

Table 53: *Helpers tab code snippets*

Helper	Subfunction
Conditionals	If -Then, If (AND) -Then, If (OR) - Then If - Then - Else If - Else If
Loops and Iteration	Hashtable iteration, Array iteration Numeric for loop, While loop, Repeat .. Until loop
Maths	Absolute value Ceiling, Floor Round to integer, Random value
Objects	Get event value (used for event-based scripts)
Storage	Get data from storage Save data to storage
Script Control	Enable, Disable script Get script status
Alerts and Logs	Alert, Formatted Alert, (appear in ***Alerts*** tab) Log variables (appear in ***Log*** tab)
Time functions	Delay script execution
Miscellaneous	Send an email Sunrise/sunset, Convert sunrise/sunset to readable
Serial	Communication via RS-232 or RS-485 Open connection, Close connection Write to port, Blocking read, Timeout read
Modbus	Create Modbus TCP object, Create Modbus RTU object Open Modbus TCP connection, Open Modbus RTU connection Close connection Set slave address Read ..., (single/multiple coils, discrete input, registers, input registers) Write ..., (single bit, multiple bits, single registers, multiple registers

Helper	Subfunction
C-Bus	Convert ... tag to address (network, application, group, level) Convert ... address to tag (network, application, group, level) Convert a set of network, application, group to tags Convert a set of network, application, group to addresses Convert. ... to ... (percentage, level) Get C-Bus ..., Set C-Bus ..., (level, state, ramp rate, target level) Get C-Bus ..., Set C-Bus ..., (object by keyword, measurement value) Get lighting ..., Set lighting ..., (state, level) Get enable ..., Set enable ..., (state, level) Get trigger level, Set trigger level Set state of (C-Bus remote on, C-Bus remote on) Get C-Bus (language, unit address, status) Set ... label (C-Bus label, Unicode C-Bus label) Get scene ..., (ID, name, level, setting) Get unit parameter ..., (age, status) Wait for C-Bus status
User parameter	Get user parameter, Set user parameter
IO	Read digital ... (input, input range, input text representation) Set relay state, Get relay state Set LED mode, Set LED state, Toggle LED state

11.2.2 Lists

The right panel of the editor displays list boxes with preconfigured objects, storage values and scripts.

Table 54: *Examples of preconfigured list items*

List	Examples of copy	Examples of usage
Group addresses	'0/56/1'	`value = GetCBusState(0, 56, 1)`
Objects by name	'Local/Light/Kitchen'	`grp_number = GetCBusGroupAddress(0, 56, 'Kitchen')`
Keywords	'lights_office'	`value = GetCBusByKW({'lights_office'}, 'or')`
Storage	'light1_On'	`data = storage.get('light1_On')`
Scripts	'Office temp'	`script.enable('Office temp')`

11.2.3 Find and replace

To search code in a script, replace code and enter code from the topic. For each of these functions, use keyboard shortcuts. Before starting, click a line of the script in the editor to place the text cursor.

Table 55: *Keyboard shortcuts for find and replace*

Keys	Result	Next steps
[Crtl] + [F]	Box for search opens Code is highlighted in yellow	Enter code and press [Enter]
[Crtl] + [G]	Shows next highlighted code	
[Shift] + [Crtl] + [G]	Shows previous highlighted code	
[Shift] + [Crtl] + [F]	Box for replace opens	Enter code "replace" and press [Enter] Enter code "with" and press [Enter] Select: *Yes*, *No*, *All*, *Stop*

Keys	Result	Next steps
[Crtl] + [Space]	List with commands and functions opens	Select in the list and press [Enter] or Enter first letter, select in list and press [Enter]

For more help, use functions and commands from the ***Helpers*** tab or the ***Select Topic*** list (see *Editor* on page 163).

11.2.4 Logs for testing

Use logs for testing and debugging a script. The ***log*** command converts variables to human readable form and stores them. In the ***Helpers*** list, access log variables via ***Alerts and logs*** → ***Log variables***.

Function

```
log(var1, var2, var3, ...)
```

The function accepts variables of the following data types.

Table 56: *Function variable data types*

Type of variable	Variable	Message
Boolean	var1 =true	boolean: true
Number	var2 =255	number: 255
String	var3 = 'text'	string: text
Not found/valid	var4 not defined	nil

A table can be defined with up to 5 nested levels.

```
var1 ={f1 =var4, t2= var5, ...}
```

Example of table

```
var1 ={f1 =var4, t2= var5}
```

Example of message

```
table: [f1]: number: 25 [f2]: boolean: false
```

The logging messages are listed in the ***Logs*** tab of the ***Configurator*** page.

When editing a script, open the current logs immediately with the ***Logs*** button.

Example of a script with logging

Event-based: Office Temp. (Local/Measurement/Temp/1 (0/228/4/1))

```
1  tempoffice = event.getvalue()
2  -- Get measured temperature
3  if (tempoffice > 30) then
4    SetCBusLevel(0, 56, 1, 255, 0)
5    -- Control light (0/56/1) = On
6  else
7    SetCBusLevel(0, 56, 1, 0, 0)
8    -- Control light (0/56/1) = Off
9  end
10 -- Logging
11     -- Get level of Control light
12     Ctrl_light = GetCBusLevel('Local','Lighting','Control light')
13 log(tempoffice, Ctrl_light)
```

Example of logs opened in the script editor

Options

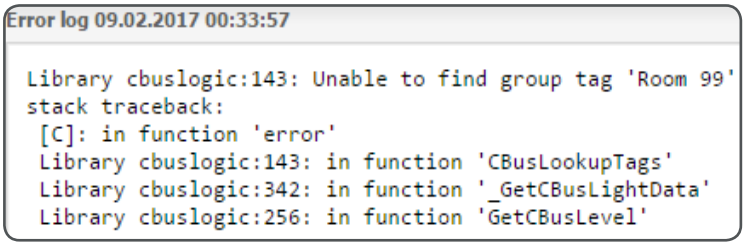
- Automatically scroll content when new logs appear
- Show logs only for current script
- Clear logs (button).

11.2.5 Error logs

Error logs are displayed in the ***Error log*** tab of the ***Configurator*** page. Error messages are generated by the system. Errors can be detected in scripts or in the configuration (see *Error logs* on page 37).

Example script

A lighting channel (local network/Lights 1/room 99) does not respond, e.g. the address *Room 99* does not exist in ***Objects*** list or ***Tag map***.



When editing a script, open error logs by clicking the ***Error log*** button.

11.2.6 Finish the script

Once finished editing the script, use the following buttons at the bottom bar of the script editor.

Table 57: *Script editor buttons*

Button	Usage
Enabled/Disabled	Enabled: script is active and can be tested Disabled: script is not active

Button	Usage
Run	When enabled, it allows event-based scripts to run. Depending on the application, send different values to trigger the script. **Lighting** • Target level • Ramp rate **Measurement** • Value • Unit.
Logs	Opens current logs (see *Logs for testing* on page 166)
Error logs	Opens error logs. Error messages from scripts are displayed in the ***Error logs*** tab (see *Error logs* on page 37).
Save	Save the script.
Save and close	Save the script and close the ***Editor*** window.
Close	Close the ***Editor*** window.

11.3 Common functions

Common functions is a library of globally used functions. They can be called from any script, any time, without special inclusions. The functions *Send an e-mail* and *Sunrise/sunset calculation* are included by default.

Path: ***Configurator*** → ***Scripting*** tab → ***Common functions*** button.

When the script editor is open, use the ***Scripts*** tab in the left panel.

11.3.1 E-mail

The function *Send an e-mail* is preconfigured with this email address:

example@gmail.com.

Use the e-mail function to send information related to an event. Examples include:

- Send an e-mail when the measured temperature is too high.
- For control, use a scheduled script.

To use the *Send an e-mail* function, perform the following checks and configuration:

① The Automation Controller must be in a network with access to the Internet, e.g. access via a network router.
  – The basic network settings of the Automation Controller are explained in the chapter (see *Change IP settings of the Ethernet interface* on page 29).
  – Check the network connection with *Network Utilities* (see page 206).

② An active e-mail account is mandatory together with information to connect to the SMTP server.

③ Insert the active e-mail account data into the ***Common functions*** script.

④ Prepare and edit an event-based script with the *Send an e-mail* function. Use a function in the ***Helpers*** tab: ***Miscellaneous*** → ***Send an e-mail***.

Access via email has been tested. Be aware, however, that access configuration may be changed by email providers in the future (e.g. to enhance security standards).

Insert individual e-mail data

Edit the email function strings to include the settings for your email provider.

Common functions

```
1  -- user function library
2
3  -- send an e-mail
4  function mail(to, subject, message)
5    -- make sure these settings are correct
6    local settings = {
7      -- "from" field, only e-mail must be specified here
8      from = 'example@gmail.com',
9      -- smtp username
10     user = 'example@gmail.com',
11     -- smtp password
12     password = 'mypassword',
13     -- smtp server
14     server = 'smtp.gmail.com',
15     -- smtp server port
16     port = 465,
17     -- enable ssl, required for gmail smtp
18     secure = 'sslv23',
19   }
```

Parameter	Description	Default setting
from	Specify the sender e-mail	'example@gmail.com'
user	User name for the e-mail account used for this function	'example@gmail.com'
password	Password for the e-mail account used for this function	'mypassword'
server	SMTP server address	'smtp.gmail.com'
port	SMTP server port	465
secure	Type of secure connection	sslv23

Click the ***Save*** button at the bottom bar of the editor. The edited email data will be included whenever the *Send an e-mail* function in a script is used.

11.3.2 Sending an e-mail using a script

Use a function in the ***Helpers*** tab to include the *Send an e-mail* function in a script.

Path: ***Miscellaneous*** → ***Send an e-mail***.

In the script, edit the following three strings (see lines 2, 3 and 4 in the following screen shot).

Event-based: Event for 0/56/4 (Local/Lighting/Room 4 (0/56/4))

```
1  -- make sure mail settings are set in user function library
2  subject = 'E-mail test'
3  message = 'Testing e-mail'
4  mail('user@example.com', subject, message)
```

Parameter	Description	Default setting
subject	Subject of message	'E-Mail text'
message	Text of message	'Testing e-mail'
Destination	Email address of the receiver	'user@example.com'

11.3.3 Backup and restore common functions

Backup and restore common functions (see *Backup and restore libraries* on page 170).

Path: ***Configurator*** → ***Scripting*** tab → ***User libraries*** button.

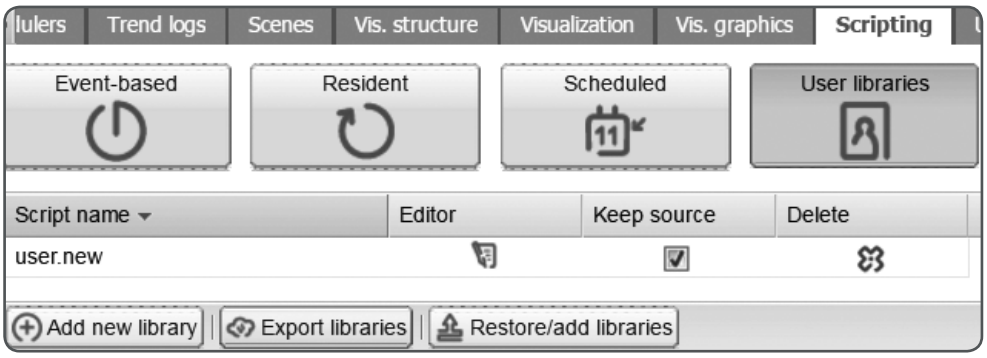
11.4 User libraries

User libraries usually contain user defined functions that can be called from other scripts. To use functions defined in a user library, they must be included at the start of the script. For example, a user library with the name 'test' will be included as follows:

```
require('user.test')
```

Path: ***Configurator*** → ***Scripting*** tab → *User libraries* button.

11.4.1 Add and edit a library



Click the *Add new library* button in the bottom bar to prepare a new library.

The following parameters are available:

Auto load library	When selected, the script is loaded when the Automation Controller starts.
Keep source	By default, this parameter is enabled. Once disabled, the code is compiled and cannot be used for further editing.

Click the ***Editor*** icon in the list to open the script editor. Now edit the library script in the same way as other scripts.

From the script editor, access existing user libraries via the ***Scripts*** tab.

11.4.2 Backup and restore libraries

Click the ***Export libraries*** button to backup all *User libraries*, the *Start-up (init) script* and the *Common functions*. The backup file has the following name:

```
Libraries-Hostname-jjjj.mm.dd-hh.mm.tar
```

Click the ***Restore/add libraries*** button to load a backup file or a file containing new libraries.

- Existing libraries with the same name as a restored or added library are overwritten.
- It is also possible to exchange files from other Automation Controllers.

11.5 Tools

The tools offer various options for the export and import of scripts.

11.5.1 Backup and restore scripts

Backup

① Click the ***Tools*** button and select ***Backup scripts***.

② (Optional) Include *Common functions* and the *Start-up (init)* script.
The backup file has the following name:

```
Scripting-Hostname-jjjj.mm.dd-hh.mm.tar
```

Restore

① Click the ***Tools*** button and select ***Restore scripts***.
② (Optional) Remove or keep the existing scripts before a backup.

11.5.2 Print script listing

① Click the ***Tools*** button and select ***Print script listings***.
A text page appears. The list is structured by categories (optional set).
② Click with right mouse button
③ (Optional) Select from the following options:
– Print
– Save as
– Show source code.

11.5.3 Edit custom Java script

This advanced function allows insertion of Java script code.

11.5.4 Show logs window

All log data are listed here (the list is a duplicate of the ***Logs*** list in the ***Logs*** tab). Use this tool to check the logged data when testing and debugging a script. Alternatively open this list from the script editor.

11.6 Alerts

System messages or user alert information from user edited scripts are displayed in the ***Alerts*** tab of the ***Configurator*** page.

In the following example, system starts and a user edited temperature alert are shown.

Alert time	Script name	Message
09.02.2017 01:00:10	Event for 0/228/5/1	Temperature too high, 36.0 °C
08.02.2017 08:11:35	system	System start
08.02.2017 05:16:29	system	System start

11.6.1 Alert command in a script

The following command stores an alert message and the current system time in the main database:

```
alert('message', var1, var2, ...)
```

Example with alert message

Event-based: Event for 0/228/5/1 (Local/Measurement/Temp_Office1/1 (0/228/5/1))
Show code shortcuts

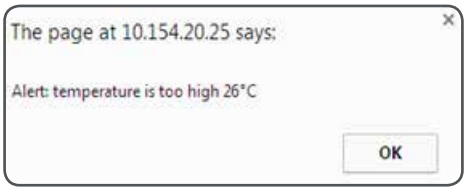
```
1  tempval = event.getvalue()
2  --Temperature of measurement application
3  if (tempval > 30) then
4    -- Control Light on
5    SetCBusLevel(0, 56, 1, 255, 0)
6    alert('Temperature too high, %.1f °C', tempval)
7  else
8    -- Control Light off
9    SetCBusLevel(0, 56, 1, 0, 0)
10 end
```

In this example, when a measured temperature is higher than 30 °C (e.g. 36 °C), the following message appears in the ***Alert*** list:

Temperature too high, 36 °C

Show alerts in PC/Tablet

By default, alerts appear in the ***Alerts*** tab within the configuration. They are displayed in the *PC/Tablet visualization* when enabled in the settings of the Visualization configuration (see page 127).



Show an alert as a message in PC/Tablet

The alert message can optionally be shown on a visualization page. Text and numeric values can be set to a string variable and set to *User parameter* application with data type *String (255 byte)*:

```
-- Set to variable message: string, digits of variable tempval and
°C
message =
  'Temperature too high,'..string.format("%d",tempval)..string.
format("°C")
-- Set message to user parameter (variable address: Temp_info)
SetUserParam('Local', 'Temp_info', message).
```

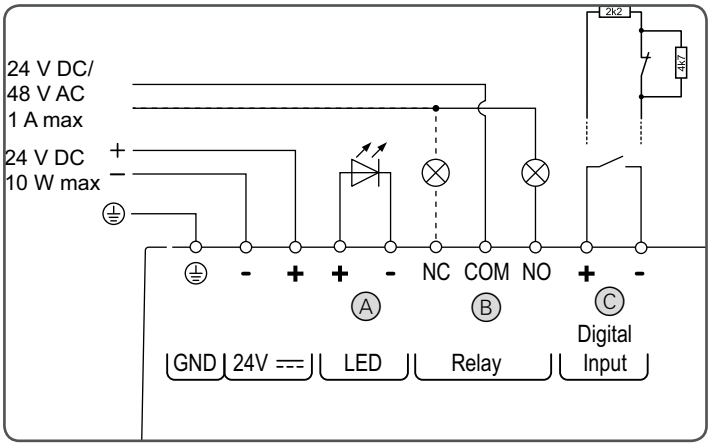
The user parameter can be configured for a visualization page (see *Objects* on page 144).

12 Inputs and Outputs

12.1 Overview

This chapter describes the configuration of the relay output, the LED driver output and the digital input. These external interfaces can be controlled via scripts. Script commands with comments are listed in the ***Helpers*** tab of the script editor (see *Script editor* on page 163). The following sections include examples of possible configurations.

12.1.1 Technical data



Ⓐ	LED Output driver	40 mA current limited
Ⓑ	Relay output	NO, NC, Common 48 V AC/24 V DC 1 A max.
Ⓒ	Digital input	Potential-free contact (switch/relay) Monitored input: • Impedance 2.2 kΩ (closed) • Impedance 6.9 kΩ (open)

12.2 Configuration of relay output

The relay output can be switched on and off via C-Bus applications. A standard use case is to switch the output via a lighting application. It is also possible to configure a *250 - User Parameter*. With this application, the output can be switched via values sent from Modbus slaves or visualization pages.

Example

In this example, a *Lighting application* is described. The load is connected to the NO (normally open) contact. The sequence is as follows:

- A sensor in the local network (0) sends the levels 0 or 255 with the ramp rate 0 s.
- If the state of the relay is set to *true*, the relay is switched on:
  - The NO contact is closed
  - The NC contact is open
  - The relay LED is green.

The following script sets the relay state:

Event-based: Event for relay (Local/Relay/Output NO (0/58/1))

```
-- Value from the sensor
switchoutput = event.getvalue()
if (switchoutput > 0) then
  -- Set state of the relay true: NO contact = closed
  SetRelayState(true)
else
  -- Set state of the relay false: NO contact = open
  SetRelayState(false)
end
```

All commands are listed in the ***Helpers*** tab of the script editor:

- Conditionals: If - Then - Else
- I/O: Set relay state.

Optionally, the relay can be switched from a visualization page. This can be configured via an *Object* element using the group address *Local/Relay/Output NO* (see *Objects* on page 144).

12.2.1 Commands in the Helpers tab

In the ***Helpers*** tab of the script editor, find an ***I/O*** command.

Path: ***Configurator*** → ***Scripting*** tab → ***Event-based*** button → ***Editor*** icon in the ***Script*** list.

Set relay state

```
-- Set state of the relay to on
SetRelayState(true)
```

If the value of the variable is set to *true*, the NO contact is closed.

Get relay state

```
-- Get state of the relay
value = GetRelayState()
```

If the NO contact is closed, the variable returns *true*.

The returned value of the relay state can also be used to switch an LED on or off.

12.3 Configuration of LED output

The LED output can be switched on and off via C-Bus applications. A common use case is to switch the LED via a lighting application. It is also possible to configure a *250 - User Parameter*. With this application, a LED can be switched via values sent from Modbus slaves or visualization pages.

12.3.1 Example with on and off

The LED output can be configured in a similar way to the relay output (see *Configuration of relay output* on page 173). Using an event-based script, the LED can be switched on and off.

12.3.2 On command

```
SetLEDState(true)
```

12.3.3 Off command

```
SetLEDState(false)
```

12.3.4 Example with LED flashing on/off

In this example, the LED flashes on/off when the temperature is higher than 30 °C. The minimal cycle (flashing) time is 1 s. The sequence is as follows:

- The temperature can be measured by a temperature sensor of a C-Bus device and sent by the application *228 - Measurement*.
- A resident script can be used to evaluate the measured temperature.
- If the temperature is above a specified value, the LED state toggles (on/off). For example, the *Toggle LED state* command can be triggered with a sleep interval (e.g. 1 s).
- If the temperature is lower than the specified value, the LED switches off.

The following script sets the LED to on/off flashing or to permanent off state:

Resident: Temp room_2 > 30 = flashing LED

```
1  -- Get temperature from measurement:  temp_room_2 (float)
2  -- Tag name of device ID: 'Sensor_room 2' Channel: 1
3  -- Tag name of network: 'Local'
4  temp_room2  =  GetCBusMeasurement('Local', 'Sensor room 2', 1)
5  -- LED flashes, when temperature is > 30 °C (1 s on / 1 s off)
6  if temp_room2 > 30 then
7    -- Toggle LED state every second (time of resident script)
8    ToggleLEDState()
9  else
10   -- Switch LED Off
11   SetLEDState(false)
12 end
```

All commands are listed in the ***Helpers*** tab of the script editor:

- C-Bus: Get C-Bus measurement value
- Conditionals: If - Then - Else
- I/O: Toggle LED state
- I/O: Set LED state.

A high temperature event can also be shown in the alerts list (see *Alerts* on page 171).

12.3.5 Commands in the Helpers tab

In the ***Helpers*** tab of the script editor, find an ***I/O*** command.

Path: ***Configurator*** → ***Scripting*** tab → ***Event-based*** button → ***Editor*** icon in the ***Script*** list.

Set LED state

```
-- Set state of LED on
SetLEDState(true)
```

If the value of the variable is set to *true*, the LED is switched on.

Get LED state

```
-- Get state of LED
value = GetLEDState()
```

If the LED is switched on, the variable returns *true*.

Toggle LED state

```
-- Toggle state of LED
ToggleLEDState()
```

If the command is executed, the LED output toggles from the current state to the opposite state (e.g. from *off* to *on*).

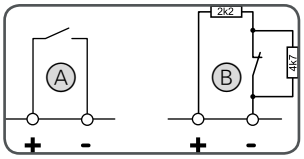
Set LED mode

```
-- Set mode of LED to 'normal'
SetLEDMode('normal')
```

No other modes can be set.

12.4 Reading of digital input

The digital input of the Automation Controller is compatible with either a potential-free contact or a monitored cable using End of Line Resistance.



Ⓐ Digital input with potential free contact

Ⓑ Digital input with monitored cable

12.4.1 States of potential free contact

Table 58: *States of potential free contact*

LED Controller	Potential free contact (switch/relay)	Text representation script
Yellow	Input is open circuit	open
Red	Input is closed circuit	short

12.4.2 States of monitored input

Table 59: *States of monitored input*

LED Controller	Monitored input	Text representation script
Green	Input in high resistance (6.9 kΩ) Switch open state	alarm
Yellow	Open circuit (> 12 kΩ)	open
Red	Short circuit (< 1 kΩ)	short
Off	Input in low resistance (2.2 kΩ) Switch closed state	ok

12.4.3 Example

In this example, text messages are used to show if a potential free window contact is open or closed. The sequence is as follows:

- The state of the digital input is repeatedly read and evaluated in a resident script.
- A variable is set to “Window open” or “Window closed”, and then set to a *250 - User parameter* application with the data type *string*.

- The characters of the user parameter can be shown on a visualization page. The user parameter can also be configured as an *Object* element (see *Objects* on page 144).

The following script can be used to set a text message as “Window open” or “Window closed” depending on the contact state.

Resident: Digital input open closed

```
-- Read state of digital input (resistance below 3.5 kΩ = true)
digital_input = ReadDigitalInput()
-- It is controlled, if the contact is closed or open
if digital_input == true then
  -- Set a string to the visualisation_message
  visualisation_message = 'Window is closed'
else
  visualisation_message = 'Window is open'
  -- Set a string to the visualisation_message
end
-- Set the visualisation message to User Parameter 250
-- User parameter: 'Status window' (Data type: String (255 bytes))
-- Tag name of network: 'Local'
SetUserParam('Local', 'Status window', visualisation_message
```

All commands are listed in the ***Helpers*** tab of the script editor:

- I/O: Read digital input
- Conditionals: If - Then - Else
- C-Bus: Set user parameter
- I/O: Set LED state.

Commands in the Helpers tab

In the ***Helpers*** tab of the script editor, find an ***I/O*** command.

Path: ***Configurator*** → ***Scripting*** tab → ***Event-based*** button → ***Editor*** icon in the ***Script*** list.

Read digital input

```
-- Read state of digital input
value = ReadDigitalInput()
```

If the resistance is below 3.5 kΩ, the variable returns *true*.

Read digital input range

```
-- Read range of digital input
value = ReadDigitalInputRng()
```

If the resistance is between 1 kΩ, and 12 kΩ, the variable returns *true*.

Read digital input text representation

```
-- Read text representation of digital input state
value = ReadDigitalInputTxt()
```

The following strings are returned.

Table 60: *Digital input strings and LED indications*

Value	Resistance	LED Controller
alarm	Input in high resistance (6.9 kΩ) Switch open state	Green
open	Open circuit (> 12 kΩ)	Yellow
short	Short circuit (< 1 kΩ)	Red
ok	Input in low resistance (2.2 kΩ) Switch closed state	Off

13 USB-A

13.1 Overview

A USB type A connector on the Automation Controller provides connection to USB expansion devices.

- USB 1.1 and 2.0 devices are supported.
- FAT and FAT32 file system formatted flash drives can be attached.
- USB 2.0 provides a bandwidth of 480 Mbit/s, corresponding to an effective image data rate of 40 MB/s.
- Devices complying with the USB specification may consume a total of 500 mA from the bus. Devices with a power rating of up to 2.5 W can therefore be supplied via the bus.
- Data transmission is possible in both directions.

13.2 USB flash drive configuration commands

Commands are listened in the ***Input and Output Facilities*** topic list of the script editor.

Path: ***Configurator*** → ***Scripting*** tab → ***Event-based*** button → ***Editor*** icon in the ***Script*** list.

The LUA reference manual available at www.lua.org provides further information.

13.2.1 Mounting

Before a drive is read from or written to it must be mounted and when your script is done it will unmount it, doing so reduces the chance of data loss if the power fails.

The following script functions will perform this task:

```
function mount_usb(check)
  devs = io.ls('/sys/class/block/')
  table.sort(devs)
  for _, dev in ipairs(devs) do
    if dev:match('^sd%a%d$') then
      part = dev
      break
    elseif not devn and dev:match('^sd%a$') then
      devn = dev
    end
  end

  part = part or devn
  if part then
    os.execute('umount -f /mnt 2>&-')
    if check then
      io.readproc('fsck.fat -a /dev/' .. part)
    end
    res, stat = io.readproc('mount /dev/' .. part .. ' /mnt 2>&1')
    if stat == 0 then
      return true
    else
      return false, 'USB mount failed: ' .. tostring(res)
    end
```

```
  else
    return false, 'No valid USB devices found'
  end
end

function unmount_usb()
  os.execute('umount -f /mnt 2>&-')
end
```

Incorporate this into your script (or have it as a user library) when accessing the drive call 'mount_usb' from your script. When the function is successful it will mount the drive under the '/mnt' directory. It takes an optional boolean argument, if it is true then it will check the file system before mounting, it defaults to off as it takes several seconds to run.

- Note that only the first partition on the drive will be mounted.
- Also note that while it is possible to connect multiple USB drives via a hub the above script will only mount the first it detects. It is possible to extend it to properly handle more but that is beyond the scope of this document

Additionally other file systems are supported such as EXT2/3/4, these have fewer intrinsic limits than FAT but require extra software to read on Microsoft Windows or Mac OS X.

Read

```
io.readfile(file)
```

Function to read the entire contents of 'file' and return it as a string. Returns nil if there is an error.

Write

```
io.writefile(file, data)
```

Function that saves 'data' to 'file' and overwrites any existing content. 'data' is a value converted to a string. The function returns a true when the file can be opened for writing, or nil when the file cannot be accessed.

Open

```
io.open(file, mode)
```

The open command is a lower level function, which opens 'file' for IO and returns a file handle. The 'mode' can be one of the following:

"r" Open the file read-only.
"w" Open the file write-only and truncate any existing data.
"a" Open the file write-only and append to the end of the file.
"r+" Open the file read/write, fails if the file doesn't exist.
"w+" Open the file read/write and truncate any existing data.
"a+" Open the file read/write and append to the end of the file.

Returns the file handle on success, returns nil and an error on failure.

The file will be closed when no longer required. Further information can be found in the Lua reference manual.

Example 1

```
-- Update the file '/mnt/data' with a value,
-- replacing any contents with the string 'abcdef' and a new line.
-- Emits an alert if the flash drive is not accessible.
res, err = mount_usb()

if res then
 io.writefile('/mnt/data', 'abcdef\n')
 unmount_usb()
else
 alert(err)
end
```

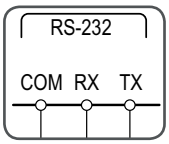
Example 2

```
-- Append 'abcdef' and a new line to the file '/mnt/data'
-- Emits an alert if the flash drive is not accessible.
-- Note that the maximum size of a file in FAT/FAT32 is 4Gbyte.
res, err = mount_usb()
if res then
 f, err = io.open('/mnt/data', 'a')
 if f ~= nil then
   f:write('abcdef\n')
   f:close()
 else
   alert('Unable to write to file: ' .. err)  end
 unmount_usb()
else
 alert(err)
end
```

14 RS-232

14.1 Overview

Interaction with other equipment is possible via an isolated RS-232 interface. The RS-232 serial interface is one of the most widely used communication standards for data transmission between two devices over short distances.



TX = Transmit
RX = Receive
COM = Common

Typical not guaranteed data transmission with different cable lengths:

Table 61: *Typical RS-232 baud rates for different cable lengths*

Baud rate (bit/s)	Max. cable length (m)
2400	60
4800	30
9600	15
19200	7.6
38400	3.7
57600	2.6
11500	1.5
230400	1.0

The achievable error free baud rate depends on the specific hardware, cable length, cable type and quality, electromagnetic noise and other factors.

Note the wiring and configuration instructions in the manuals of the connected RS-232 equipment.

14.2 Configuration commands

Commands for the data transfer between the Automation Controller and another device using RS-232 are described in the following script.

Serial commands are listed in the ***Helpers*** tab of the script editor.

Path: ***Configurator*** → ***Scripting*** tab → ***Event-based*** button → ***Editor*** icon in the ***Script*** list.

Open connection

```
require('serial')
-- communication example with 38400 bit/s
port = serial.open('/dev/RS232', {
  baudrate = 38400,
  databits = 8,
  stopbits = 1,
  parity = 'none',
  duplex = 'half'
})
```

Table 62: *RS-232 open connection settings*

Settings	Values
Baud rate	300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400
Data bits	5, 6, 7, 8
Stop bits	1, 2
Parity	´none', 'even', 'odd'
Duplex	'half', 'full'

Note the configuration instructions in the manuals of the connected RS-232 equipment. The Automation Controller and the other device must use the same settings and values.

Write to port

```
port:write('test data')
```

Blocking read

```
-- script will block until 10 characters are read
data = port:read(10) Blocking read
```

Timeout read

```
-- script will wait for 10 characters for 20 seconds
data = port:read(10, 20)
```

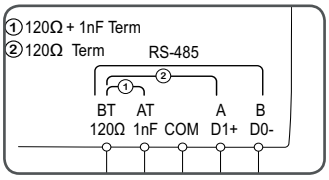
Close serial port

```
port:close()
```

15 RS-485

15.1 Overview

The Automation Controller has an isolated RS-485 interface, which can be used for a serial data transmission between two or more devices.



Modbus

A D1+ = Data transfer

B D0- = Data transfer

COM = Common

① = Optional in-build low power terminator of 120 Ω + 1 nF via link AT–BT

② = Optional in-build legacy terminator of 120 Ω = link BT-A

- D1+ and D0– = twisted wires of RS-485 cable.
- COM = common wire.
- Shield must be connected to earth at end of the line.
- The line must be terminated at each end:
  - The Automation Controller has an optional inbuilt low power terminator of **120 Ω + 1 nF** via link AT–BT. This will be used when the Automation Controller is at one end of the line.
  - In large installations, install the master in the middle of the line. In this case, the inbuilt line termination will not be used.
- Topology:
  - Daisy chain or multi-drop with short stubs (40 m/number of stubs/derivations, max. 20 m).
  - The line must be terminated at each end.
  - Cable: e.g. Belden 9842 (2 twisted pairs, shielded, imp. 120 Ω).
- Baud rate and cable length:
  - The achievable error-free baud rate depends on length of a line, number of devices, cable type and quality, correct terminations, electromagnetic noise and other factors.
  - The Automation Controller supports up to 230400 bit/s. When the data rate is below 100 kbps, data can typically not guaranteed be transmitted at a distance up to 1200 m. At higher data rates, the cable length must be reduced:

Table 63: *Typical RS-485 baud rates for different cable lengths*

Baud rate (bit/s)	Max. cable length (m)
300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600	1200 m
115200	750 m
230400	600 m

  - All devices on a RS-485 line must use the same baud rate. The max. cable length for a given baud rate may be reduced.

The Modbus protocol can be used on the RS-485 port. The configuration of a Modbus line is described in the Modbus chapter on page 185:

- Settings using profiles see *Modbus RTU Settings* on page 188

- Settings using scripts see *Modbus RTU configuration commands* on page 196.

15.2 Configuration commands

In the following script commands for the data transfer between the Automation Controller and another device using RS-485 are described. ***Serial*** commands are listed in the ***Helpers*** tab of the script editor.

Path: ***Configurator*** → ***Scripting*** tab → ***Event-based*** button → ***Editor*** icon in the ***Script*** list.

Note the wiring and configuration instructions in the manuals of the connected RS-485 equipment.

15.2.1 Open connection

If using the scripts in the ***Helpers*** tab of the script command edit `'/dev/RS485'` instead of `'/dev/RS232'`.

```
require('serial')
-- communication example with 38400 bit/s
port = serial.open('/dev/RS485', {
  baudrate = 38400,
  databits = 8,
  stopbits = 1,
  parity = 'none',
  duplex = 'half'
})
```

Table 64: *RS-485 open connection settings*

Settings	Values
Baud rate	300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400
Data bits	5, 6, 7, 8
Stop bits	1, 2
Parity	´none', 'even', 'odd'
Duplex	'half'

Write to port

```
port:write('test data')
```

Blocking read

```
-- script will block until 10 characters are read
data = port:read(10) Blocking read
```

Timeout read

```
-- script will wait for 10 characters for 20 seconds
data = port:read(10, 20)
```

Close serial port

```
port:close()
```

16 Modbus overview

Modbus is an open standard for client/server communication. The client sends a request message and the server sends a response message. The values of a server are saved in registers which can be accessed by the client.

The Automation Controller supports Modbus RTU serial communication via the RS-485 interface and Modbus TCP communication via the RJ45 Ethernet interface. In Modbus RTU mode, the **client** is usually referred to as the **master** and the **server** as the **slave**.

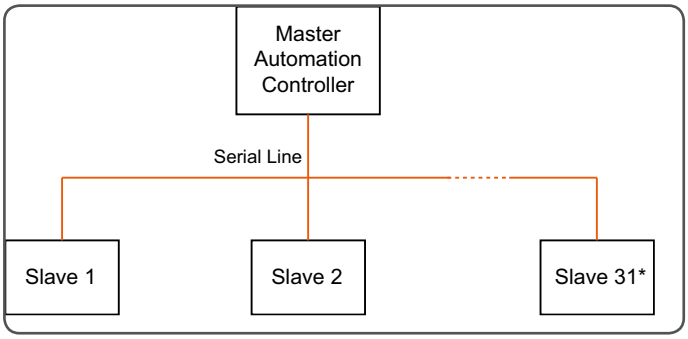
In general, the Automation Controller is the Modbus master/client. The supported registers of a slave/server can be mapped with objects of the application *250 - User Parameter*. Thus, inputs, outputs, measured values and states of a server can be monitored and controlled. Use these objects for the basic functions of the Automation Controller such as visualization, trend logging and scheduling, as well as for advanced functions using LUA scripting.

Tables of Modbus registers are available in Modbus device manuals. For general information about the Modbus protocol and data model, refer to Modbus.org.

As the Automation Controller supports both Modbus RTU and Modbus TCP, buildings can be analysed and controlled in depth. With Modbus TCP, it is also possible to communicate remotely with Modbus devices installed in different buildings.

16.1 Access via RS-485

The most common method of Modbus communication is Modbus RTU serial communication via the RS-485 interface. The Automation Controller has RS-485 connectors for Modbus and acts as a master.



* By Modbus specification a serial line is limited to 31 slaves.

16.1.1 Slaves

By Modbus specification a serial line is limited to 31 slaves. The Network Automation Controller is not limited but designed for 31 slaves. The Wiser™ for C-Bus® Automation Controller is limited to a maximum of 6 Modbus slaves.

Schneider Electric offers a wide range of Modbus RTU devices which can be easily integrated with the Automation Controller:

- PM: Power Meter range (e.g current, voltage, power, power factor, frequency, energy)
- iEM: Watt-hour meter range (e.g. current, voltage, power, power factor, frequency, energy)
- SIM10M: Smart Interface module (Pulse counter e.g. for water, gas, watt hours)
- Masterpact: Circuit Breaker Manager
- Compact: Circuit breakers for high loads

- Vigilohm: Insulation monitoring
- Smartlink RTU: smart communication I/O module
- SE 8000: Room Controllers
- TC 303: Digital Fan Coil Thermostat.

A complete building can be analysed and controlled in depth via a Modbus RTU line. Moreover, access Modbus slaves in another RTU line by using Modbus TCP over Modbus gateways.

16.1.2 Easy configuration with profiles

The slaves in a RTU line can be easily configured. Preinstalled profiles (*.json files) are available for the slaves. A profile provides a list of usable registers of the slave. Select the registers required and map these with objects of the application *250 - User Parameter*. Step by step configuration is described in *Configure preinstalled profiles* on page 187. There is also a list of available profiles.

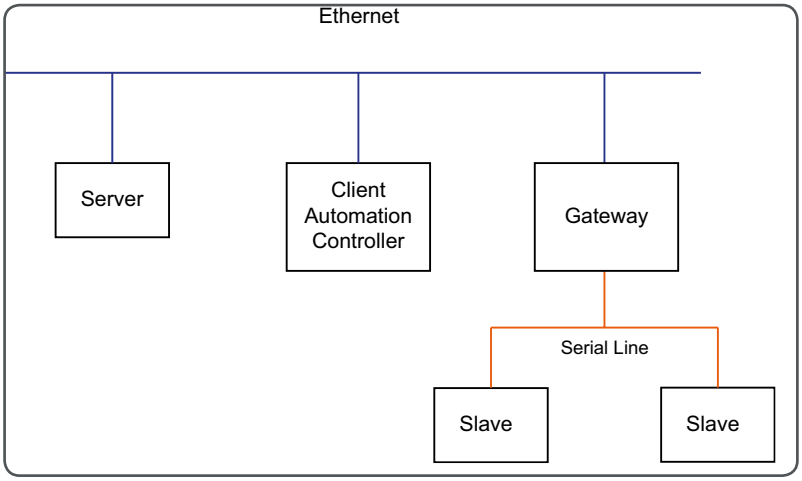
If there is no preinstalled profile for the device to be integrated, download a profile and adapt it for your product. See *Add and edit profiles* on page 191.

16.1.3 Configuration with scripts

If familiar with programming in LUA, use scripts (see *Modbus settings using scripts* on page 193). If using the Automation Controller as a slave, it it mandatory to use scripts instead of a profile. A combination of profiles and scripts cannot be used.

16.2 Access via Ethernet

The Automation Controller can communicate as a client with servers via TCP/IP. Servers or slaves can either be directly accessed via Ethernet or via a Modbus/IP gateway.



An example of direct connection via Ethernet is *Smartlink IP*, where the Automation Controller acts as a client and Smartlink IP acts as a server. The configuration of IP-communication, channels, inputs and mapping is done via the profile *Smartlink TCP*.

A gateway connects Modbus via Ethernet (RJ45) with a Modbus serial line (RS-485 interface). Up to 31 Modbus RTU slaves can be connected with a gateway. The configuration can be performed via the profiles of the RTU slaves. The connection type for each slave in the serial Modbus line is TCP/IP. All slaves must be linked with the IP address of the gateway.

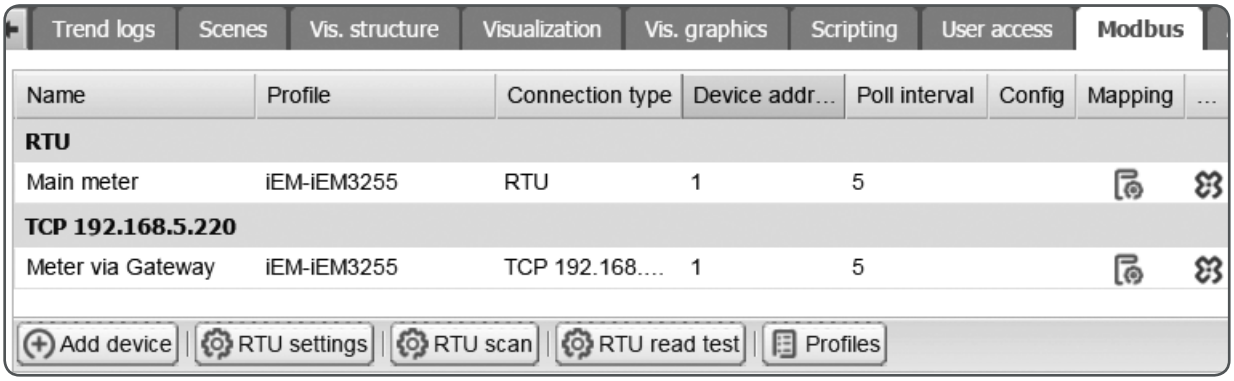
Step by step instructions for configuring RTU slaves are described in *Configure preinstalled profiles* on page 187. Therein is a list of available profiles.

The Modbus TCP Interface allows up to 100 open TCP connections (e.g. servers/ slaves).

16.3 Configure preinstalled profiles

Configuration is done in the ***Modbus*** tab. Modbus slaves (servers) can be added via either the ***Add device*** button or the ***RTU scan*** button.

Path: ***Configurator*** → ***Modbus*** tab.



16.3.1 Steps

Perform the following steps to configure Modbus slaves (servers) using profiles:

① Enable Modbus RTU communication (click the ***RTU Settings*** button).
② (Optional) Scan for Modbus RTU devices. The *List of preinstalled Modbus profiles* on page 187 shows the Modbus devices which can be scanned via the ***RTU scan*** button. If RTU is not enabled, scanning does not work.
③ If the RTU scan function is not possible, click the ***Add device*** button and assign the correct profile.
④ Edit the Modbus device (***Modbus*** list) to set the polling period.
⑤ Select registers and check the data type.
⑥ If necessary, create new objects of the *250 User Parameter* application.
⑦ Map the objects of the application to the selected registers.

16.3.2 List of preinstalled Modbus profiles

Table 65: *Preinstalled Modbus profiles*

Profile	Description	RTU scan
Compact_NSX-Compact_NSX_E	Distribution Application Type E for NSX	Yes
Masterpact_NT_NW-Masterpact_A	Circuit Breaker Manager for Masterpact	Yes
Masterpact_NT_NW-Masterpact_H	Circuit Breaker Manager for Masterpact	Yes
Masterpact_NT_NW-Masterpact_P	Circuit Breaker Manager for Masterpact	Yes
PM-PM1200	Power Meter PM1200	No
PM-PM210	Power Meter PM210	No
PM-PM3250	Power Meter PM3250	Yes
PM-PM3255	Power Meter PM3255	Yes
PM-PM5110	Power Meter PM5110	No
PM-PM5111	Power Meter PM5111	No
PM-PM5310	Power Meter PM5310	No

Profile	Description	RTU scan
PM-PM5330	Power Meter PM5330	No
PM-PM5350	Power Meter PM5350	No
PM-PM710	Power Meter PM710	No
PM-PM750	Power Meter PM750	No
PM-PM810	Power Meter PM810	No
PM-PM820	Power Meter PM820	No
PM-PM850	Power Meter PM850	No
PM-PM870	Power Meter PM870	No
PM-PM9C	Power Meter PM9C	No
iEM-iEM3150	Watt-hour Meter iEM3150	Yes
iEM-iEM3155	Watt-hour Meter iEM3155	Yes
iEM-iEM3250	Watt-hour Meter iEM3250	Yes
iEM-iEM3255	Watt-hour Meter iEM3255	Yes
iEM-iEM3350	Watt-hour Meter iEM3350	Yes
iEM-iEM3355	Watt-hour Meter iEM3355	Yes
iEM-iEM2150	Watt-hour Meter iEM2150	Yes
iEM-iEM2150	Watt-hour Meter iEM2150	Yes
SIM10M	SIM10M	No
TC303	Digital Fan Coil Thermostat	No
SE8300	Low voltage FCU Controller	No
SE8600	RTU, Heat pump & IAQ Controller	No
SER8300	Line Voltage FCU Controller	No
Smartlink-RTU	Acti9 Smartlink-RTU	Yes
Smartlink-TCP	Acti9 Smartlink-TCP	No
Vigilohm IM20	Vigilohm IM20 (Insulation monitoring)	Yes
Vigilohm IM400	Vigilohm IM400 (Insulation monitoring)	Yes

16.3.3 Modbus RTU Settings

In order to communicate with Modbus slaves connected over Modbus RTU (serial), it is mandatory to enable the communication and set the connection details.

Path: ***Modbus*** tab → ***RTU-Settings*** button.

RTU (serial) enabled

This option enables Modbus RTU communication.

Port

The default setting of the serial port is */dev/RS485*. If the Port field is empty, */dev/RS485* is automatically inserted when the ***Save*** button is pressed.

Baud rate

All Modbus devices must use the same setting.

All metering devices can run either on 9600 bit/s (max. cable distance 1200 m) or 19200 bit/s (max. cable distance 900 m). 19200 bit/s is default for most devices.

Some devices can also work with other speeds.

Parity

All Modbus devices must use the same setting.

Select either *None*, *Even*, or *Odd*. *Even* with *One stop bit* is the default for most devices.

Duplex

Must be *Half-duplex* for RS-485.

Reset to defaults

This button resets all RTU setting parameters to their default.

16.3.4 Modbus RTU scan

Devices with the *RTU scan* feature can be detected and added using the *RTU scan* function (see *List of preinstalled Modbus profiles* on page 187).

Path: ***Modbus*** tab → ***RTU scan*** button.

Steps

① Ensure that Modbus slaves are connected to the Automation Controller via the RS-485 interface.
② Click the ***RTU scan*** button.
③ Select the start and end device addresses of the slaves.
④ Click the ***Save*** button.
⑤ The Controller recognises Modbus devices that can be detected via the RTU scan function. Modbus devices that are not capable of being detected must firstly be assigned a correct profile and then edit the names and the poll intervals.
⑥ Choose which devices to be inserted in the ***Modbus device*** list.

16.3.5 Add Modbus device

Manually add Modbus devices.

Path: ***Modbus*** tab → ***Add device*** button.

Steps

① Click the ***Add device*** button.
② Edit and save the following parameters:

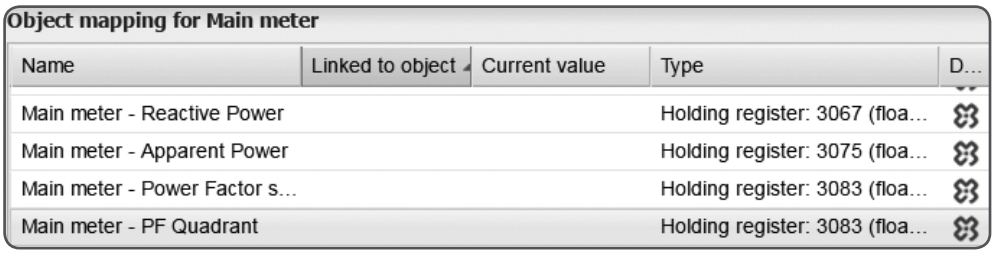
Table 66: *Modbus device parameters*

Name	Name of Modbus device
Connection type	RTU (RS-485) TCP/IP
Profile	Select an installed profile
Device address	Select the Modbus device address. **Note:** It is mandatory to select this address on the Modbus device (e.g. via the display menu or the configuration software of the device).
Poll interval	Select an interval. Value 5 is the default (new values are read every 5 seconds).
IP address	If the *TCP/IP* connection type is selected, set the IP address. If the Modbus device (server) is directly connected to the Automation Controller (client), set the IP address of the Modbus device. If the Modbus device (slave) is connected via a RTU/TCP/IP gateway, set the IP address of the gateway (e.g. EGX300, Smartlink IP).
Port	(Optional) Set the Port for Modbus TCP communication. The standard Modbus default Port is 502.

16.3.6 Modbus Mapping

When a Modbus device is added, bindings between Modbus registers and objects have to be created. This is done by setting the register mapping. The registers of the Modbus device can be mapped with objects of the application *250 - User Parameter*. For each device in the Modbus device list, open a mapping list.

Path: ***Modbus*** tab → ***Modbus device*** list → ***Mapping*** column → ***Object mapping*** list.



Each row of the Mapping table represents one of the Modbus registers (defined in the device profile). Decide which registers to map, then check the data type of the selected registers. Link with objects of the C-Bus application *250 - User Parameter.*

For the mapping, only select existing C-Bus objects with the correct application and data type.

Examples

- The register type of the *Reactive Power* is *float32*. In this case, link with the data type *Floating point (32 bit*). The application is *250 - User parameter*.
- The register type of the *PF Quadrant* is internally converted from *float32* to *uint32 (float32=>uint32)*. In this case, link with data type *Unsigned Integer (32 Bit)*. The application is *250 - User parameter*.

16.3.7 Add Objects

It is recommended to prepare objects before the mapping procedure. New objects can be added in the ***Objects*** tab (see *Add user parameters* on page 104.

For each register in the ***Object mapping*** list, edit the following parameters:

Table 67: *Object mapping parameters*

Name	Name of the register in the object mapping list.
Link to object	Select the object where the value read from the Modbus register will be saved. Only select existing objects from the drop down menu.
Value send delta	Set the value of delta. If the change of value read from Modbus register is bigger than this delta, the value is sent to the object.
Unit/suffix	If the *User parameter* application is selected, a unit is used from the profile or can be modified.
Keywords	Keywords are assigned to the object. Keywords can be used in scripts (optional).
Description	(Optional).

16.3.8 RTU read test

RTU read test allows quick reading of values stored in Modbus registers with different parameters.

Path: ***Modbus*** tab → ***RTU read test*** button.

For testing, select a register address from the ***Object mapping*** list. This functionality is also very useful when creating and testing your own Modbus profiles.

16.4 Add and edit profiles

If your Modbus device profile is not in the list of preinstalled profiles, define your own profile.

16.4.1 Structure of a profile

Modbus device profiles are distributed in *.json files. Use any common text editor to create and edit your profile.

Consider using a text editor with enhanced support for .json files. These editors display .json file content with syntax colouring/formatting and this makes the file easier to edit and save.

The structure of a profile is shown below:

```
{
"manufacturer": "Schneider Electric",
"description": "Example device",
"mapping": [
{ "name": "Output 1", "bus_datatype": "bool", "type": "coil",
  "address": 0,"writable": 1 },
{ "name": "Input 1", "bus_datatype": "float16",
  "type": "inputregister", "address": 0,
"value_multiplier": 0.001, "units": "V" }
]
}
```

16.4.2 Mapping parameters

Each “mapping” line in the .json file contains mapping information for one Modbus register or coil. All the possible mapping parameters are listed in the table below.

Table 68: *Profile mapping parameters*

Parameter	Description	Type	Required
Name	Object name, e.g. Output 2.	String	Yes
bus_datatype	C-Bus object **data** type, key from dt table, e.g. float32.	String/ Number	Yes
type	Modbus register type. Possible values: coil, discreteinput, register, inputregister.	String	Yes
address	Register address (0-based).	Number	Yes
writable	Set to true to enable writing to the register if type is either coil or register.	Boolean	No
write_only	Set to true to disable reading coil or register value when “writable” is enabled.	Boolean	No
datatype	Modbus value data type. If set, conversion will be done automatically. Possible values: bool, uint16, int16, float16, uint32, int32, float32, uint64, int64, quad10k, s10k.	String	No
value_delta	New value is sent when the difference between previously sent value and current value is larger than delta. Defaults to 0 (send after each read).	Number	No
value_base	Add specified number to the resulting value.	Number	No
value_multiplier	Multiply resulting value by the specified number,	Number	No

Parameter	Description	Type	Required
value_bitmask	Bit mask to apply. Shifting is done automatically based on the least significant '1' found in the mask.	Number	No
value_nan	Array of 16-bit integers. If specified and the read operation returns the same array, no further processing of the value is done.	Array	No
value_conv	Apply one of the inbuilt conversion functions.	String (Int)	No
value_custom	Name of an inbuilt enumeration or a list of key → value mapping. Resulting value will be 0 if key is not found.	String/ Object	No
internal	Not visible to the user when set to **true**. Must be used for scale registers.	Boolean	No
units	Object units/suffix.	String	No
address_scale	Address of the register containing value scale. Value = value * 10 ^ scale	Number	No
read_count	Number of registers to read at once (for devices that only support reading of a specific block of registers).	Number	No
read_swap	Swap register order during conversion (endianness).	Boolean	No
read_offset	Position of first register of data from the block of registers (0-based).	Number	No
timeout	Specify device timeout in seconds. If the slave device does not reply within specified time, it is treated as a timeout error. Default values: 0.5 s for Modbus RTU, 3 s for Modbus TCP	Number	No
write_multiple	This parameter sets the multiple writing function (function 15 or 16 is used instead of function 5 or 6). If "Type" is set to "register" and "Write_multiple" is set to "true", Modbus function 16 is used for writing to the register. If "Type" is set to "coil" and "Write_multiple" is set to "true", Modbus function 15 is used for writing to the coil. Default value is "false", which means that Modbus function 5 or 6 (depending on register type) is used for writing.	String	No

When the Modbus devices are physically connected, use the RTU read test. This allows quick reading of values stored in Modbus registers (See *RTU read test* on page 190).

16.4.3 Export profiles

When creating a new profile, use an existing profile as an example. If a preinstalled profile must be modified, e.g. to add registers, it can be exported, modified in Notepad and imported again. If required to keep both versions, rename the new json file.

Path: ***Modbus*** tab → ***Profiles*** button → ***Profiles*** list → ***Export*** icon.

Steps

① Click the ***Profiles*** button.
② Click the ***Export*** icon in the profiles list and select where to save the .json file.

Editing of the file with a text editor is now possible.

16.4.4 Import profiles

Import new or changed .json files in the ***Profiles*** list.

Path: ***Modbus*** tab → ***Profiles*** button → ***Profiles*** list → ***Add profile*** button.

Steps

① Click the ***Add Profiles*** button and select the .json file.

If something is wrong, e.g. the characters *", [ ] { }* are missing or in the wrong place, this error message appears: *Invalid profile selected*. If a parameter is not recognised, it will just be skipped.

② After successful import, the profile becomes available in the ***Profiles*** list.

16.4.5 Delete profiles

Path: ***Modbus*** tab → ***Profiles*** button → ***Profiles*** list → ***Delete*** icon.

Click the ***Delete*** icon to delete a profile.

16.5 Modbus settings using scripts

16.5.1 Function Codes and corresponding Master Functions

All the possible Modbus function codes that can be used in the Automation Controller are listed below. There is an LUA function in the Automation Controller for each function code.

All of these functions can be used for both Modbus TCP and Modbus RTU.

FC#01 Read Coils

Name	Read single coil
Command	`coil = mb:readcoils(address)`
Arguments	[address]: address of the coils
Returned values	1: ON, 0: OFF
Exception codes:	01 or 02 or 03 or 04

Name	Read multiple coil
Command	`coil = mb:readcoils(start, count)`
Arguments	[start]: address of the first coil to read [count]: number of coils to read (max. 2000)
Returned values	1: ON, 0: OFF
Exception codes	01 or 02 or 03 or 04
Script example	`coil1,coil2,coil3 = mb:readcoils(1000, 3)` Value read from coil address 1000 is returned into variable coil1. Value read from coil address 1001 is returned into variable coil2. Value read from coil address 1002 is returned into variable coil3.

FC#02 Read Discrete Inputs

Name	Read discrete input
Command	`value = mb:readdiscreteinputs(address)`
Arguments	[address]: address of the input
Returned values	1: ON, 0: OFF
Exception codes:	01 or 02 or 03 or 04

Name	Read discrete inputs
Command	`value = mb:readdiscreteinputs(address, count)`
Arguments	[address]: address of first input to read [count]: number of inputs to read (max. 2000)
Returned values	1: ON, 0: OFF
Exception codes:	01 or 02 or 03 or 04
Script example	`bool1, bool2= mb:readdiscreteinputs(10,2)` Value read from discrete input address 10 is returned into variable bool1. Value read from discrete input address 11 is returned into variable bool2.

FC#03 Read Holding Registers

Name	Read registers
Command	`value = mb:readregisters(address, count)`
Arguments	[address]: address of the first register to read [count]: number of registers to read (max 125)
Returned values	2 byte values
Exception codes:	01 or 02 or 03 or 04
Script example	`value1,value2 = mb:readregisters(1100, 2)` Value read from register address 1100 is returned into variable value1. Value read from register address 1101 is returned into variable value2.

FC#04 Read Input Registers

Name	Read input registers
Command	`value = mb:readinputregisters(address, count)`
Arguments	[address]: address of the first input register to read [count]: number of input registers to read (max. 125)
Returned values	2 byte values
Exception codes:	01 or 02 or 03 or 04
Script example	`value1, value2 = mb:readinputregisters(1015, 2)` Value read from input register address 1015 is returned into variable value1. Value read from input register address 1016 is returned into variable value2.

FC#05 Write Single Coil

Name	Write single bit
Command	`mb:writebits(address, value)`
Arguments	[address]: address of the coil [value]: true or false

FC#06 Write Single Register

Name	Write single register
Command	`mb:writeregisters(addresss, value)`
Arguments	[address]: address of the register [value]: value of the register

FC#0F Write Multiple Coils

Name	Write multiple bits
Command	`mb:writebits(address, value1, value2, value3,…)`
Arguments	[address]: start address of the coils [value1]: true or false , [value2]: true or false, ... (max. 1968 bits)
Script example	`mb:writebits(1000, true, false)` Write bit at address 1000 to true (on) Write bit at address 1001 to false (off)

FC#0F Write Multiple Registers

Name	Write multiple registers
Command	`mb:writeregisters(address, value1, value2, value3, ...)`
Arguments	[address]: start address of the registers [value1]: number, [value2]: number, ... (max. 123 registers)

Exception codes

```
mb:readcoils(start, count)
```

```
mb:readdiscreteinputs(start, count)
```

```
mb:readregisters(start, count)
```

```
mb:readinputregisters(start, count)
```

These commands read one or more registers/coils from the start address and return all values when successful. When an error occurs, three variables are sent back:

- Nil
- Exception code description
- Exception code (see *Table 69*).

Table 69: *Modbus function exception codes*

Code	Name	Meaning
01	Illegal Function	The Function Code received in the query is not an allowable action for the server (or slave). This may be because the function code is only applicable to newer devices, and was not implemented in the unit selected. It may also indicate that the server (or slave) is in the wrong state to process a request of this type, for example because it is not configured and is being asked to return register values.
02	Illegal Data Address	The data address received in the query is not an allowable address for the server (or slave). More specifically, the combination of reference number and transfer length is invalid. For a controller with 100 registers, a request of offset 96 and a length of 5 will generate exception 02.
03	Illegal Data Value	The value contained in the query data field is not an allowable value for the server (or slave). This indicates a fault in the structure of the remainder of a complex request, such as the implied length is incorrect. It specifically does NOT mean that a data item submitted for storage in a register has a value outside the expectation of the application program, since the Modbus protocol is unaware of the significance of any value of any register.
04	Failure in Associated Device	An unrecoverable error occurred while the server (or slave) was attempting to perform the requested action.

Code	Name	Meaning
05	Acknowledge	Specialized in conjunction with programming commands. The server (or slave) has accepted the request and is processing it, but a long duration of time will be required. This response is returned to prevent a timeout error from occurring in the client (or master). The client (or master) can next issue a poll program complete message to determine if processing is completed.
06	Busy, Rejected Message	Specialized use in conjunction with programming commands. The server (or slave) is engaged in processing a long-duration program command. The client (or master) will retransmit the message later when the server (or slave) is free.
07	NAK – Negative Acknowledgement	Specialized use in conjunction with programming commands. The server (or slave) is engaged in processing a long-duration program command. The client (or master) will retransmit the message later when the server (or slave) is free.
08	Memory Parity Error	Specialized use in conjunction with function codes 20 and 21 and reference type 6, indicates the extended file area failed to pass a consistency check. The server (or slave) attempted to read a record file, but detected a parity error in the memory. The client (or master) can retry the request, but service may be required on the server (or slave) device.
0A	Gateway Path Unavailable	Specialized use in conjunction with gateways. Indicates that the gateway was unable to allocate an internal communication path from the input port to the output port for processing the request.
0B	Gateway Target Device Failed to respond	Specialized use in conjunction with gateways. Indicates that no response was obtained from the target device. Usually means that the device is not present on the network.

For more information, see: http://modbus.org.

16.5.2 Modbus RTU configuration commands

Create Modbus RTU object

```
require('luamodbus')
mb = luamodbus.rtu()
```

Open Modbus RTU connection

```
-- 19200 baud rate, even parity, 8 data bits, 1 stop bit,
-- half duplex
mb:open('/dev/RS485', 19200, 'E', 8, 1, 'H')
mb:connect()
```

Terminal name

`'/dev/RS485'`

Supported Baud rates

300 bit/s
600 bit/s
1200 bit/s
2400 bit/s
4800 bit/s
9600 bit/s
19200 bit/s
38400 bit/s
57600 bit/s
115200 bit/s
230400 bit/s

All Modbus devices in a line must use the same setting.

The achievable error-free baud rate depends on length of a line, number of Modbus devices, cable type and quality, correct terminations, electromagnetic noise and other factors (see *Access via RS-485* on page 185).

Parity

“N“	None	choose between one and two stop bits
“E“	Even	one stop bit is set
“O“	Odd	one stop bit is set

The default parity mode of Modbus is “even” parity.

Data bits and stop bits

Data bits: [Number of data bits = 5, 6, 7, 8]
Stop bits: [Number of stop bits 1, 2]

Duplex

“H“	Half duplex
“F”	Full duplex (not supported in RS-485)

Delay Between Frames

```
-- Wait for 1.5 seconds
os.sleep(1.5)
```

Some devices require considerable time after the end of response until they are ready to receive the following request from the master. If the behaviour of the device is not known, select (1.5).

Set slave address

```
-- Set slave address to 123
mb:setslave(123)
```

Address range is [1..247].

Read registers

```
-- Read from address 1000 and write it to value
value = mb:readregisters(1000)
```

Close modbus connection

```
mb:close()
```

Example 1

```
-- init modbus on first script execution
       if not mb then
       require('Luamodbus')
       mb = Luamodbus.rtu()
       mb:open('/dev/RS485', 38400, 'E', 8, 1, 'H')
       mb:connect()
       end
       mb:setslave(30)
       mb:flush()
```

Example 2

```
-- Timeout interval between two consecutive bytes of the same
-- message
       mb:getbytetimeout()
       mb:setbytetimeout(timeout)
```

Example 3

```
-- Timeout interval used to wait for a response:
       mb:getresponsetimeout()
       mb:setresponsetimeout(timeout)
```

Example 4

```
-- Timeout interval used for an incoming indication from
-- master (Slave mode only)
       mb:getreceivetimeout()
       mb:setreceivetimeout(timeout)
```

16.5.3 Modbus master functions

`mb:setslave(slaveid)`

Sets slave id to read/write data from/to.

`mb:readcoils(start, count)`

start – address of first coil to read.

count – number of coils to read.

`mb:readdiscreteinputs(start, count)`

start – address of first discrete input to read.

count – number of discrete inputs to read

`mb:readregisters(start, count)`

start – address of first holding register to read.

count – number of holding registers to read.

`mb:readinputregisters(start, count`

start – address of input register to read.

count – number of input registers to read.

Returns all values on success and nil, error description on error.

`mb:writebits(start, v1, [v2, [v3, ...]])`

Writes values to coils from start address.

`mb:writeregisters(start, v1, [v2, [v3, ...]])`

Writes values to registers/coils from the start address.

Single write will be used when only one value is supplied, multiple write otherwise returns all of values written on success and nil, error description on error.

```
mb:reportslaveid()
```

Reads slave internal data.

Returns values on success.

Returns nil, error description on error.

16.5.4 Modbus slave functions

Receive data from master

```
mb:receive()
```

Receives data from master with 1 minute timeout.

Returns data as a binary string on success.

Returns nil, error description on error.

Set Modbus mapping of slave device

```
mb:setmapping(coils, inputs, holding_regs, input_regs)
```

Creates memory mapping for the registers with size specified for each type.

Handle slave

```
mb:handleslave()
```

Waits for an incoming indication from master and sends a reply when necessary.

Get functions

```
mb:getcoils(start, count)
mb:getdiscreteinputs(start, count)
mb:getinputregisters(start, count)
mb:getregisters(start, count)
```

Gets one or many register/coil/input values from mapping from the start address.

Returns all values on success.

Returns nil, error description on error, exception code if applicable.

Set functions

```
mb:setcoils(start, v1, [v2, [v3, ...]])
mb:setdiscreteinputs(start, v1, [v2, [v3, ...]])
mb:setinputregisters(start, v1, [v2, [v3, ...]])
mb:setregisters(start, v1, [v2, [v3, ...]])
```

Sets values to register/coil mapping from the start address.

Returns true on success.

Returns nil, error description on error, exception code if applicable.

Callback functions

`mb:setwritecoilcb(fn)`
`mb:setwriteregistercb(fn)`

Sets a callback function for coil/register write event.

Callback will accept two parameters: coil/register address and value (boolean or number).

For multiple writes callback is executed for each coil/register separately.

Use nil to remove a callback.

17 BACnet

17.1 Overview

BACnet is designed to allow communication of building automation and control systems for applications such as heating, ventilation, air conditioning control, lighting control, access control, fire detection systems and their associated equipment. The BACnet protocol provides exchange information for building automation devices, regardless of the particular building service they perform.

There are two formats for BACnet: BACnet Ethernet and BACnet IP. Both formats use Ethernet media for communications. The Automation Controller only supports BACnet IP format via the RJ45 Ethernet connector.

The Automation Controller can act as a BACnet server only (not as a client). The Automation Controller serves data which can be read by BACnet client devices (like Building Management Systems). BACnet client devices can write data to the server. This allows for a wide functional integration. For example, the Automation Controller gets Modbus master measurement data from a Modbus RTU and delivers the data via BACnet to a BMS (Building Management System).

BACnet IP is a protocol that complies with the following standards:

ASHRAE	American Society of Heating, Refrigerating and Air-Conditioning Engineers
ANSI	American National Standards Institute
ISO	International Organization for Standardization

The Automation Controller has been certified by BACnet Testing Laboratories (BTL) as a BACnet Application Specific Controller (B – ASC).

The Automation Controller complies with all the necessary interoperability requirements.

17.2 List of all BACnet Interoperability Building Blocks (BIBBs) supported

Table 70: *BACnet supported interoperability building blocks*

Data Sharing	ReadProperty-B	DS-RP-B
	ReadPropertyMultiple-B	DS-RPM-B
	WriteProperty-B	DS-WP-B
	COV-B	DS-COV-B
Device and Network Management	Dynamic Device Binding-B	DM-DDB-B
	Dynamic Object Binding-B	DM-DOB-B
	DeviceCommunicationsControl-B	DM-DCC-B
	TimeSynchronization-B	DM-TS-B
	UTCTimeSynchronization-B	DM-UTC-B
	ReinitializeDevice-B	DM-RD-B

17.2.1 BACnet Object Types Supported

Device ID

Analogue Value

Binary Value

17.2.2 Data Link Layer Options

Media: BACnet IP

Option: Register as a Foreign Device

17.3 Schneider Electric Building Management System

Schneider Electric offers *StruxureWare* as a BACnet certified Building Management System. *Building Operation WorkStation* is software used to configure and commission *Enterprise Server* (software) and the *Automation Server* (hardware) which can retrieve and send data to the C-Bus Automation Controller.

The following image shows a list of BACnet data points in the *StruxureWare Building Operation WorkStation* software (SBO).

Name	Description	Value
Local%2FEnable%2FEnable_value_1	Local/Enable/Enable_value_1 (0/203/0)	1,00
Local%2FLighting%2FLighting_value_1	Local/Lighting/Lighting_value_1 (0/56/0)	158,00
Local%2FMeasurement%2FMeasurement_value_1%2F0	Local/Measurement/Measurement_value_1/0 (0/228/0/0)	15,00
Local%2FUnit Parameter%2FUnit_parameter_1%2FTemperature	Local/Unit Parameter/Unit_parameter_1/Temperature (0/255...	2,00
Local%2FUser Parameter%2FCurrent A	Local/User Parameter/Current A (0/250/0)	0,43
Local%2FUser Parameter%2FCurrent B	Local/User Parameter/Current B (0/250/1)	0,43
Local%2FUser Parameter%2FCurrent C	Local/User Parameter/Current C (0/250/2)	0,42
Local%2FUser Parameter%2FDevice status	Local/User Parameter/Device status (0/250/18)	Inactive
Local%2FUser Parameter%2FDevice tripped	Local/User Parameter/Device tripped (0/250/19)	Active
Local%2FUser Parameter%2FSet output	Local/User Parameter/Set output (0/250/20)	Inactive

The configuration of the C-Bus Automation Controller is done in three steps:

① Select objects in the objects list and activate the *Export* function.
② Configure BACnet communication (BACnet Settings).
③ Check the view of BACnet objects and optionally save as a .csv file that can be used for documentation.

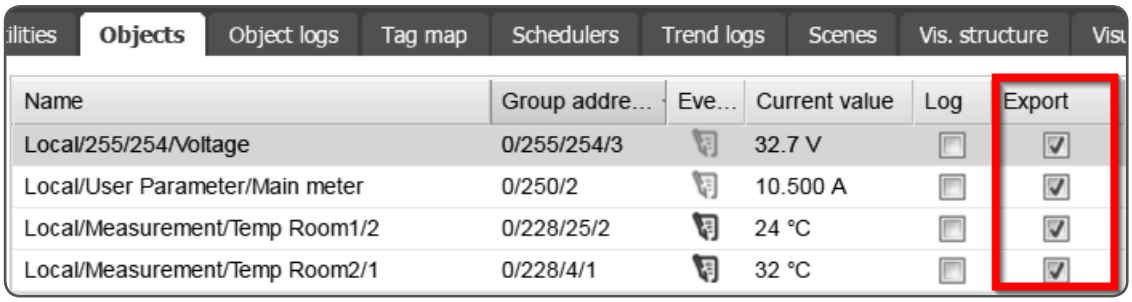
17.4 Object export

Use the following applications for exporting to BACnet:

Number	Application
56 (48-127)	Lighting
203	Enable
202	Trigger control
228	Measurement
250	User Parameter
255	Unit Parameter

Path: ***Configurator*** → ***Objects*** tab → ***Export*** column.

Select the objects which are to be made available for BACnet in the Export column.



Most C-Bus applications deliver, or are controlled by (AV) analogue numeric values 0-255 as well as status indication. These can be used as on/off/dim commands or scene triggers to C-Bus units, depending upon the C-Bus application being used.

The application *250 - User Parameter* with the *Boolean* data type will appear as binary values (BV) with a range of 0-1. On C-Bus, this translates to 0=Off and 1=On.

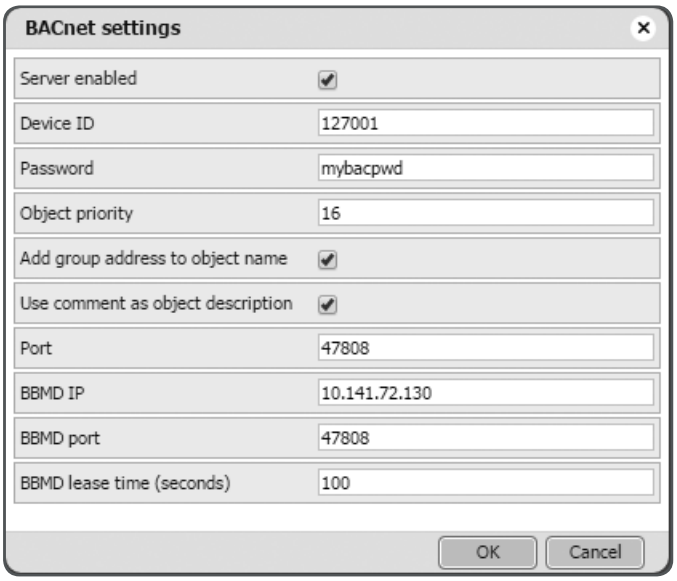
17.5 BACnet configuration

The Automation Controller can be configured as a BACnet server in the *BACnet Settings* of the Automation Controller. The BACnet *Building Management System* on the client site will then discover the selected data.

17.5.1 BACnet settings

Configure the Automation Controller as a BACnet server in the *BACnet Settings*.

Path: ***Configurator*** → ***Utilities*** button → ***System*** button → ***Network*** tab → ***BACnet Settings***.



Server enabled	Enable/disable Automation Controller as a BACnet server.
Device ID	BACnet device ID which must be unique on the BACnet network.
Password	BACnet password.

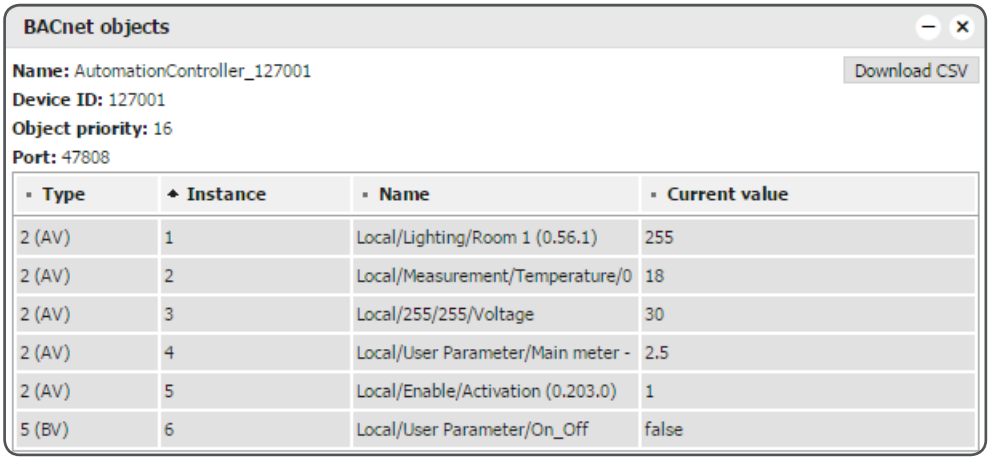
Objects priority	Define the priority array to which the Automation Controller writes. The default value of the object property is 16. The Automation Controller writes to the *Relinquish Default* (RD) property by first reading (Upload from BMS) and takes the current value of the object. It is not possible to change the value of the *Relinquish Default* property afterwards. If the object read from the Automation Controller has a higher value than the RD property, it raises the Overwritten flag.
Add group address to object name	Names of BACnet objects contain information about group address (see *BACnet objects* on page 204).
Use comment as object description	Comments in objects are visible in the SBO (*StruxureWare Building Operation WorkStation* software) as a description.
Port	BACnet port, default 47808.
BBMD* IP	Router IP.
BBMD* port	Router port.
BBMD* lease time (seconds)	Registration resend interval.

* BBMD is for port forwarding. Typically, this feature is not used. BBMD is an optional setup parameter and does not require configuration unless using BBMD onsite.

17.5.2 BACnet objects

Open a list of BACnet objects with the selected parameters.

Path: ***Configurator*** → ***Utilities*** button → ***System*** button → ***Network*** tab → ***BACnet objects***.



In the *Type* column of the above screen shot, there are five analogue (AV) and one binary value (BV).

BACnet objects can optionally be downloaded to a .csv file via the ***Download CSV*** button. This list can be used for documentation.

17.5.3 BACnet COV setting

The first 256 objects exported to BACnet can use the change of value (COV) subscription on the client side. This parameter defines the minimum change of value (delta) which implies a change of the value on the client side.

The default value is 1.

Path: ***Configurator*** → ***Utilities*** button → ***System*** button → ***Network*** tab → ***BACnet COV settings***.

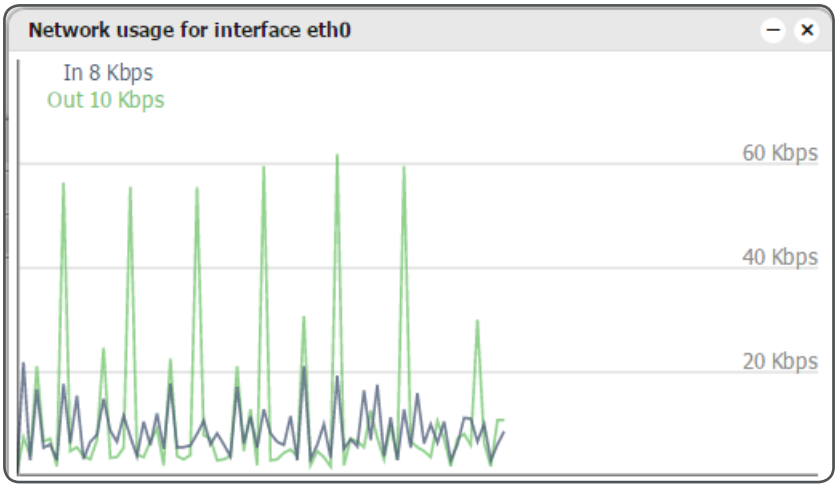
18 Advanced network functions

18.1 Network Settings

Path: ***Configurator*** → ***Utilities*** button → ***System*** button → ***Network*** tab.

18.1.1 Interfaces

Click ***Interfaces*** in the ***Network*** tab and then click ***eth0*** to configure the IP address (see *Change IP settings of the Ethernet interface* on page 29). Click the *Show network usage* button to check the network traffic.



18.1.2 Routes

The routing table shows the network routes associated with the Automation Controller.

Click ***Routes*** in the ***Network*** tab.

Routes

Interface	Destination	Gateway	Network mask	Flags
eth0	0.0.0.0	192.168.5.5	0.0.0.0	UG
eth0	192.168.5.0	0.0.0.0	255.255.255.0	U
usb0	192.168.254.0	0.0.0.0	255.255.255.0	U

eth0 network adapter

196.168.5.5 is the address of the gateway (router), i.e. access to Internet.

192.168.5.0 is the address of the local network (default is 192.168.0.0).

usb0 network adapter

192.168.254.0 is the address of the USB-B network connection. The IP address of the Automation Controller on this network is 192.168.254.10.

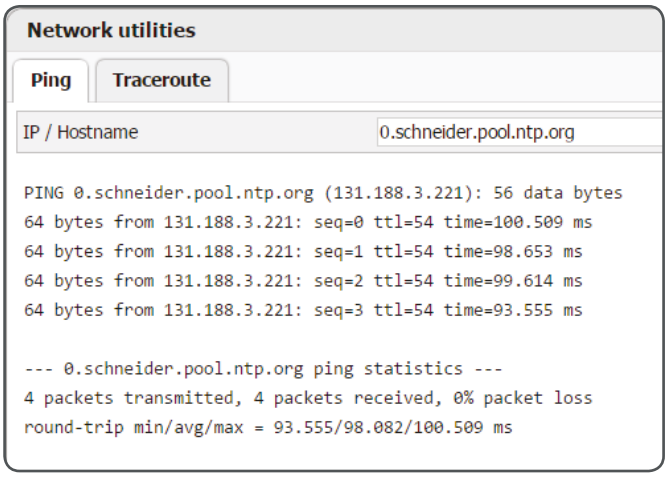
18.2 Network utilities

Use network utilities to check the connection to other network devices via ***Ping*** and ***Trace route***. For example, check the connection between the Automation Controller and a Modbus IP gateway or a IP camera, locally or via the public address.

Path: ***Configurator*** → ***Utility*** tab → ***System*** button → ***Status*** tab → ***Network utilities***.

Depending on your network settings, the response to a ping command may be blocked.

Example of pinging a network time protocol server (NTP)



Traceroute displays the path and delay times to a destination on the Internet.

18.3 Access via HTTP, HTTPS or VPN

Port forwarding is used to get remote access to an IP device in a local network. Settings must be made in the network router. See the manual for the particular router for instructions on how to set and use port forwarding.

If there is no static address on the public side of your router, a dynamic DNS service is needed to reach the your router. DNS service providers can supply information about configuring a dynamic DNS service.

HTTP or HTTPS

NOTICE

HTTP IS NOT A SECURE MEANS OF COMMUNICATION

- Information transmitted via HTTP is not encrypted and therefore insecure.
- Only use **HTTPS** for port forwarding and remote access.

Failure to follow these instructions may result in network security breaches.

VPN

Many network solution providers offer a way to set up a virtual private network (VPN) connection to get direct secure access via Internet to a local network. Usually a VPN solution requires VPN server and VPN client software.

Depending on the security standards used, VPN access can provide a high level of security.

18.3.1 HTTP Server

On the *System* page, set an additional HTTP and HTTPS port. By default, HTTP port 80 and HTTP port 443 are already set.

Path: ***Configurator*** → ***Utilities*** button → ***Services*** button → ***Http Server***.

18.4 Remote services

Use remote services (XML/JSON) to remotely activate functions or control objects.

Path: ***Configurator*** → ***Utilities*** button → ***System*** button → ***Services*** tab → ***Remote services***.

18.4.1 Parameters

Service stats

Enable/disable the function.

Username

The username is *remote.*

Password

The default password is *remote*.

Allow only exported objects

Only control objects with the activated export function can be checked. This function is set in the ***Objects*** tab (see *Additional parameters for all applications* on page 105).

18.4.2 Examples

Show alerts

Alert time	Script name	Message
05.02.2017 22:52:28	system	System start
04.02.2017 16:19:22	system	System start

Alerts are displayed in the *Alerts* tab and can be shown in XML format in the browser.

Example of browser command:

```
http://remote:remote@192.168.0.10/scada-remote/?m=rss&r=alerts
```

Example of browser response:

```
...
  <item>
    <title>System start</title>
    <pubDate>Mon, 06 Feb 2017 09:22:35 +1030</pubDate>
...
```

Set value

In this example, the value of an object (light) with the composed address 0/56/1 is set to 255.

Example of browser command:

```
http://remote:remote@192.168.0.10/scada-remote/?m=json&r=grp&fn=write&alias=0/56/1&value=255
```

Example of browser response:

```
true
```

The new value of 0/56/1 is 255 and the light is switched on remotely.

18.5 FTP server

The Automation Controller includes an FTP server. By default, the FTP server is disabled.

Path: ***Configurator*** → ***Utilities*** button → ***System*** button → ***Services*** tab → ***FTP server***.

NOTICE

USE OF THE FTP SERVER CAN AFFECT AUTOMATION CONTROLLER OPERATING FILES.

The FTP server can expose files that are necessary for the Automation Controller to work properly.

- Exercise care when working with files via the FTP server.
- Do not move or delete any files that are necessary for Automation Controller operation.

Failure to follow these instructions may result in the Automation Controller becoming inoperative.

18.5.1 Parameters

Server status

Enable/disable the function.

Port

The default port is 22.

Username

The username is *ftp*.

Passive mode min port

Minimum port for passive mode.

Passive mode max port

Maximum port for passive mode.

18.6 Remote diagnostic

Remote diagnostic allows access to the controller via SSH (Secure Shell), a network protocol for secure services. By default, remote diagnostic is disabled.

Path: ***Configurator*** → ***Utilities*** button → ***System*** button → ***Services*** tab → ***Remote diagnostic***.

NOTICE

USE OF THE REMOTE DIAGNOSTIC FUNCTION CAN AFFECT AUTOMATION CONTROLLER OPERATING FILES

Remote diagnostic can expose files that are necessary for the Automation Controller to work properly.

- **Only use the Remote diagnostic function when recommended by Schneider technical support.**
- Exercise care when using the *Remote diagnostic* function.
- Do not move or delete any files that are necessary for Automation Controller operation.

Failure to follow these instructions may result in the Automation Controller becoming inoperative.

By default, *remote diagnostic* is disabled. When enabled, Port 22 must be forwarded on the router.

19 Open source license information

This Schneider Electric product is provided with certain open source software components (collectively, “OSS”) developed by third parties. The OSS is licensed in accordance with the OSS license terms set forth below.

19.1 GNU General Public License v.2

The following open source software components are licensed in accordance with the GNU General Public License v.2 reprinted below.

a. Linux Kernel

NOTE: This copyright does “not” cover user programs that use kernel services by normal system calls - this is merely considered normal use of the kernel, and does “not” fall under the heading of “derived work”. Also note that the GPL below is copyrighted by the Free Software Foundation, but the instance of code that it refers to (the linux kernel) is copyrighted by me and others who actually wrote it.

Copyright (c) 1991-1992 Linus Torvalds

b. LEDE Build System

Copyright (c) 2016 LEDE Project

c. Base-files

Copyright (c) 2007-2016 OpenWrt.org

Copyright (c) 2010 Vertical Communications

d. Eibd

Copyright (c) 2005-2011 Martin Koegler

e. Luarrd

Copyright (c) 1997-2015 Tobias Oetiker

f. Opkg

Copyright (c) 2011 Mikhail Gusarov

g. Rrdtool

Copyright (c) 1997-2015 Tobias Oetiker

h. Pcsk

Copyright (c) 2004 Norbert Buchmuller

i. Bacnet

Copyright (C) 2012 Steve Karg

Bacnet special exception to GNU General Public License v.2:

As a special exception, if other files instantiate templates or use macros or inline functions from this file, or you compile this file and link it with other works to produce a work based on this file, this file does not by itself cause the resulting work to be covered by the GNU General Public License. However the source code for this file must still be made available in accordance with section (3) of the GNU General Public License.

This exception does not invalidate any other reasons why a work based on this file might be covered by the GNU General Public License.

j. Vsftpd

Copyright (c) Chris Evans

As copyright holder, I give permission for vsftpd to be linked to the OpenSSL libraries. This includes permission for vsftpd binaries to be distributed linked against the OpenSSL libraries. All other obligations under the GPL v2 remain intact.

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.

675 Mass Ave, Cambridge, MA 02139, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

19.1.1 Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

19.1.2 GNU General Public License

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0 This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The “Program”, below, refers to any such program or work, and a “work based on the Program” means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term “modification”). Each licensee is addressed as “you”.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1 You may copy and distribute verbatim copies of the Program’s source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2 You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3 You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:
   a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
   b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
   c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for non-commercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

   The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4 You may not copy, modify, sub-license, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sub-license or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5 You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6 Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7 If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/ donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8 If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9 The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10 If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

19.1.3 No Warranty

11 BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12 IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH

HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

19.1.4 End of Terms and Conditions

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

<one line to give the program’s name and a brief idea of what it does.>

Copyright (C) 19yy <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) 19yy name of author

Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w’.

This is free software, and you are welcome to redistribute it under certain conditions; type `show c’ for details.

The hypothetical commands `show w’ and `show c’ should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w’ and `show c’; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program

`Gnomovision’ (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989

Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is

what you want to do, use the GNU Library General Public License instead of this License.

19.2 GNU Lesser General Public License v.2.1

The following open source software components are licensed in accordance with the GNU Lesser General Public License v.2.1 reprinted below.

a. Libmodbus

Copyright (c) 2008-2014 Stephane Raimbault

b. Px5g

Copyright (c) 2009 Steven Barth

Copyright (c) 2014 Felix Fietkau

c. Uci

Copyright (c) 2008-2014 OpenWrt.org

Copyright (c) 2016 LEDE Project

GNU LESSER GENERAL PUBLIC LICENSE

Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc.

51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

19.2.1 Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library,

you must provide complete object files to the recipients, so that they can re-link them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

19.2.2 Terms and Conditions for copying and distribution and modification

0 This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called “this License”). Each licensee is addressed as “you”.

1 A “Library” means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

2 The “Library”, below, refers to any such software library or work which has been distributed under these terms. A “work based on the Library” means either the library or any derivative work under copyright law: that is to say, a work containing the library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term “modification”.)

“Source code” for a work means the preferred form of the work for making modifications to it. For a Library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

3 You may copy and distribute verbatim copies of the Library’s complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

4 You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a. The modified work must itself be a software library.
b. You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
c. You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
d. If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful. (For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License,

whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

5 You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

6 You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

7 A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a “work that uses the Library”. Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a “work that uses the Library” with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a “work that uses the library”. The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a “work that uses the Library” uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

8 As an exception to the Sections above, you may also combine or link a “work that uses the Library” with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer’s own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

- Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
- Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

9 You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

10 You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

11 You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

12 Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

13 If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

14 It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

15 If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

16 The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

17 If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of

all derivatives of our free software and of promoting the sharing and reuse of software generally.

19.2.3 No Warranty

1 BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

2 IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

19.2.4 End of Terms and Conditions

How to apply these terms to your new libraries:

- If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).
- To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

  one line to give the library’s name and an idea of what it does.

  Copyright (C) *year name of author*

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU

Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library `Frob' (a library for tweaking knobs) written by James Random Hacker.

signature of Ty Coon, 1 April 1990, Ty Coon, President of Vice.
That's all there is to it!

19.3 MIT license

The following open source software components are licensed in accordance with the MIT license reprinted below.

a. Libexpat

Copyright (c) 1998-2000 Thai Open Source Software Center Ltd and Clark Cooper

Copyright (c) 2001-2016 Expat maintainers

b. Luacjson

Copyright (c) 2010-2012 Mark Pulford

c. Luaexpat

Copyright (c) 2003-2007 The Kepler Project

d. Luajit

Copyright (c) 2005-2017 Mike Pall

e. Luasocket

Copyright (c) 2004-2013 Diego Nehab

f. Luasql

Copyright (c) 2003-2016 the Kepler Project

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

19.4 MIT-like license

The following open source software components are licensed in accordance with the MIT-like license reprinted below.

a. Ncurses

Copyright (c) 1998-2007, 2008 Free Software Foundation, Inc.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, distribute with modifications, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE ABOVE COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name(s) of the above copyright holders shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization.

19.5 BSD-3 license

The following open source software components are licensed in accordance with the BSD-3 license reprinted below.

a. Blsh

Copyright (c) 2013-2014 Freescale Semiconductor, Inc.

b. Pcre

Copyright (c) 1997-2017 University of Cambridge

c. Redis

Copyright (c) 2006-2015 Salvatore Sanfilippo

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1 Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2 Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3 Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR

OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

19.6 BSD-2 license

The following open source software components are licensed in accordance with the BSD-2 license reprinted below.

a. File

Copyright (c) 1986, 1987, 1989, 1990, 1991, 1992, 1994, 1995 Ian F. Darwin

Maintained 1994 - by Christos Zoulas

This software is not subject to any export provision of the United States

Department of Commerce, and may be exported to any country or planet.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1 Redistributions of source code must retain the above copyright notice immediately at the beginning of the file, without modification, this list of conditions, and the following disclaimer.
2 Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

19.7 BSD-2-like license

The following open source software components are licensed in accordance with the BSD-2-like license reprinted below.

a. Nginx

Copyright (c) 2002-2017 Igor Sysoev

Copyright (c) 2011-2017 Nginx, Inc.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1 Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2 Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

19.8 Open source software components (1)

The following open source software components are licensed in accordance with the license reprinted below.

a. Dropbear

Copyright (c) 2002-2015 Matt Johnston

Portions Copyright (c) 2004 Mihnea Stoenescu

All rights reserved.

19.8.1 License Terms

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

19.9 LIBPNG license

The following open source software components are licensed in accordance with the LIBPNG license reprinted below.

a. Libpng

COPYRIGHT NOTICE, DISCLAIMER, and LICENSE:

If you modify libpng you may insert additional notices immediately following this sentence.

This code is released under the libpng license.

libpng versions 1.0.7, July 1, 2000 through 1.6.29, March 16, 2017 are Copyright (c) 2000-2002, 2004, 2006-2017 Glenn Randers-Pehrson, are derived from libpng-1.0.6, and are distributed according to the same disclaimer and license as libpng-1.0.6 with the following individuals added to the list of Contributing Authors:

Simon-Pierre Cadieux

Eric S. Raymond

Mans Rullgard

Cosmin Truta

Gilles Vollant

James Yu

Mandar Sahastrabuddhe

Google Inc.

Vadim Barkov

and with the following additions to the disclaimer:

There is no warranty against interference with your enjoyment of the library or against infringement. There is no warranty that our efforts or the library will fulfill any of your particular purposes or needs. This library is provided with all faults, and the entire risk of satisfactory quality, performance, accuracy, and effort is with the user.

Some files in the “contrib” directory and some configure-generated files that are distributed with libpng have other copyright owners and are released under other open source licenses.

libpng versions 0.97, January 1998, through 1.0.6, March 20, 2000, are Copyright (c) 1998-2000 Glenn Randers-Pehrson, are derived from libpng-0.96, and are distributed according to the same disclaimer and license as libpng-0.96, with the following individuals added to the list of Contributing Authors:

Tom Lane

Glenn Randers-Pehrson

Willem van Schaik

libpng versions 0.89, June 1996, through 0.96, May 1997, are Copyright (c) 1996-1997 Andreas Dilger, are derived from libpng-0.88, and are distributed according to the same disclaimer and license as libpng-0.88, with the following individuals added to the list of Contributing Authors:

John Bowler

Kevin Bracey

Sam Bushell

Magnus Holmgren

Greg Roelofs

Tom Tanner

Some files in the “scripts” directory have other copyright owners but are released under this license.

libpng versions 0.5, May 1995, through 0.88, January 1996, are Copyright (c) 1995-1996 Guy Eric Schalnat, Group 42, Inc.

For the purposes of this copyright and license, “Contributing Authors” is defined as the following set of individuals:

Andreas Dilger

Dave Martindale

Guy Eric Schalnat

Paul Schmidt

Tim Wegner

The PNG Reference Library is supplied “AS IS”. The Contributing Authors and Group 42, Inc. disclaim all warranties, expressed or implied, including, without limitation, the warranties of merchantability and of fitness for any purpose. The Contributing Authors and Group 42, Inc. assume no liability for direct, indirect, incidental, special, exemplary, or consequential damages, which may result from the use of the PNG Reference Library, even if advised of the possibility of such damage.

Permission is hereby granted to use, copy, modify, and distribute this source code, or portions hereof, for any purpose, without fee, subject to the following restrictions:

1 The origin of this source code must not be misrepresented.
2 Altered versions must be plainly marked as such and must not be misrepresented as being the original source.
3 This Copyright notice may not be removed or altered from any source or altered source distribution.

The Contributing Authors and Group 42, Inc. specifically permit, without fee, and encourage the use of this source code as a component to supporting the PNG file format in commercial products. If you use this source code in a product, acknowledgment is not required but would be appreciated.

END OF COPYRIGHT NOTICE, DISCLAIMER, and LICENSE.

19.10 Open source software components (2)

The following open source software components are licensed in accordance with the license reprinted below.

a. Zlib

Copyright (C) 1995-2017 Jean-loup Gailly and Mark Adler

19.10.1 License Terms

This software is provided ‘as-is’, without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1 The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required. 2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
2 This notice may not be removed or altered from any source distribution.

Jean-loup Gailly Mark Adler

jloup@gzip.org madler@alumni.caltech.edu

The following open source software components are licensed in accordance with the OpenSSL license and the original SSLeay license reprinted below.

a. Openssl

Copyright (c) 1998-2017 The OpenSSL Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1 Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2 Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3 All advertising materials mentioning features or use of this software must display the following acknowledgment: “This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org/)”
4 The names “OpenSSL Toolkit” and “OpenSSL Project” must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org.
5 Products derived from this software may not be called “OpenSSL” nor may “OpenSSL” appear in their names without prior written permission of the OpenSSL Project.
6 Redistributions of any form whatsoever must retain the following acknowledgment “This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)”

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

==========

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

Original SSLeay License

Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)

All rights reserved.

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com).

The implementation was written so as to conform with Netscapes SSL.

This library is free for commercial and non-commercial use as long as the following conditions are aheared to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com).

Copyright remains Eric Young's, and as such any Copyright notices in the code are not to be removed.

If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used.

This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1 Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
2 Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3 All advertising materials mentioning features or use of this software must display the following acknowledgement: "This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)" The word 'cryptographic' can be left out if the rouines from the library being used are not cryptographic related :-).
4 If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement:

"This product includes software written by Tim Hudson (tjh@cryptsoft.com)"

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The licence and distribution terms for any publicly available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution license [including the GNU Public License.]

19.11 Open source software components (3)

The following open source software components is dedicated to the public domain by its authors in accordance with the terms below.

a. Sqlite3

All of the code and documentation in SQLite has been dedicated to the public domain by the authors. All code authors, and representatives of the companies they work for, have signed affidavits dedicating their contributions to the public domain and originals of those signed affidavits are stored in a fire-safe at the main offices of Hwaci. Anyone is free to copy, modify, publish, use, compile, sell, or distribute the original SQLite code, either in source code form or as a compiled binary, for any purpose, commercial or non-commercial, and by any means.

The previous paragraph applies to the deliverable code and documentation in SQLite - those parts of the SQLite library that you actually bundle and ship with a larger application. Some scripts used as part of the build process (for example the "configure" scripts generated by autoconf) might fall under other open-source licenses. Nothing from these build scripts ever reaches the final deliverable SQLite

library, however, and so the licenses associated with those scripts should not be a factor in assessing your rights to copy and use the SQLite library.

All of the deliverable code in SQLite has been written from scratch. No code has been taken from other projects or from the open internet. Every line of code can be traced back to its original author, and all of those authors have public domain dedications on file. So the SQLite code base is clean and is uncontaminated with licensed code from other projects.

19.12 BSD license

The following open source software components are licensed in accordance with the modified BSD license reprinted below.

a. Unzip

Copyright (c) 1990-2009 Info-ZIP. All rights reserved.

For the purposes of this copyright and license, “Info-ZIP” is defined as the following set of individuals:

Mark Adler, John Bush, Karl Davis, Harald Denker, Jean-Michel Dubois, Jean-loup Gailly, Hunter Goatley, Ed Gordon, Ian Gorman, Chris Herborth, Dirk Haase, Greg Hartwig, Robert Heath, Jonathan Hudson, Paul Kienitz, David Kirschbaum, Johnny Lee, Onno van der Linden, Igor Mandrichenko, Steve P. Miller, Sergio Monesi, Keith Owens, George Petrov, Greg Roelofs, Kai Uwe Rommel, Steve Salisbury, Dave Smith, Steven M. Schweda, Christian Spieler, Cosmin Truta, Antoine Verheijen, Paul von Behren, Rich Wales, Mike White.

19.12.1 License Terms

This software is provided “as is,” without warranty of any kind, express or implied. In no event shall Info-ZIP or its contributors be held liable for any direct, indirect, incidental, special or consequential damages arising out of the use of or inability to use this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the above disclaimer and the following restrictions:

1 Redistributions of source code (in whole or in part) must retain the above copyright notice, definition, disclaimer, and this list of conditions.

2 Redistributions in binary form (compiled executables and libraries) must reproduce the above copyright notice, definition, disclaimer, and this list of conditions in documentation and/or other materials provided with the distribution. The sole exception to this condition is redistribution of a standard UnZipSFX binary (including SFXWiz) as part of a self-extracting archive; that is permitted without inclusion of this license, as long as the normal SFX banner has not been removed from the binary or disabled.

3 Altered versions--including, but not limited to, ports to new operating systems, existing ports with new graphical interfaces, versions with modified or added functionality, and dynamic, shared, or static library versions not from Info-ZIP--must be plainly marked as such and must not be misrepresented as being the original source or, if binaries, compiled from the original source. Such altered versions also must not be misrepresented as being Info-ZIP releases--including, but noT limited to, labelling of the altered versions with the names “Info-ZIP” (or any variation thereof, including, but not limited to, different capitalizations), “Pocket UnZip,” “WiZ” or “MacZip” without the explicit permission of Info-ZIP. Such altered versions are further prohibited from mis-representative use of the Zip-Bugs or Info-ZIP e-mail addresses or the Info-ZIP URL(s), such as to imply Info-ZIP will provide support for the altered versions.

4 Info-ZIP retains the right to use the names “Info-ZIP,” “Zip,” “UnZip,” “UnZipSFX,” “WiZ,” “Pocket UnZip,” “Pocket Zip,” and “MacZip” for its own source and binary releases.

All of the code and documentation in SQLite has been dedicated to the public domain by the authors. All code authors, and representatives of the companies they work for, have signed affidavits dedicating their contributions to the public domain and originals of those signed affidavits are stored in a fire-safe at the main offices of Hwaci. Anyone is free to copy, modify, publish, use, compile, sell, or distribute the original SQLite code, either in source code form or as a compiled binary, for any purpose, commercial or non-commercial, and by any means.

The previous paragraph applies to the deliverable code and documentation in SQLite - those parts of the SQLite library that you actually bundle and ship with a larger application. Some scripts used as part of the build process (for example the “configure” scripts generated by autoconf) might fall under other open-source licenses. Nothing from these build scripts ever reaches the final deliverable SQLite library, however, and so the licenses associated with those scripts should not be a factor in assessing your rights to copy and use the SQLite library.

All of the deliverable code in SQLite has been written from scratch. No code has been taken from other projects or from the open internet. Every line of code can be traced back to its original author, and all of those authors have public domain dedications on file. So the SQLite code base is clean and is uncontaminated with licensed code from other projects.

19.13 BSD license

The following open source software components are licensed in accordance with the modified BSD license reprinted below.

a. Unzip

Copyright (c) 1990-2009 Info-ZIP. All rights reserved.

For the purposes of this copyright and license, “Info-ZIP” is defined as the following set of individuals:

Mark Adler, John Bush, Karl Davis, Harald Denker, Jean-Michel Dubois, Jean-loup Gailly, Hunter Goatley, Ed Gordon, Ian Gorman, Chris Herborth, Dirk Haase, Greg Hartwig, Robert Heath, Jonathan Hudson, Paul Kienitz, David Kirschbaum, Johnny Lee, Onno van der Linden, Igor Mandrichenko, Steve P. Miller, Sergio Monesi, Keith Owens, George Petrov, Greg Roelofs, Kai Uwe Rommel, Steve Salisbury, Dave Smith, Steven M. Schweda, Christian Spieler, Cosmin Truta, Antoine Verheijen, Paul von Behren, Rich Wales, Mike White.

19.13.1 License Terms

This software is provided “as is,” without warranty of any kind, express or implied. In no event shall Info-ZIP or its contributors be held liable for any direct, indirect, incidental, special or consequential damages arising out of the use of or inability to use this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the above disclaimer and the following restrictions:

1 Redistributions of source code (in whole or in part) must retain the above copyright notice, definition, disclaimer, and this list of conditions.
2 Redistributions in binary form (compiled executables and libraries) must reproduce the above copyright notice, definition, disclaimer, and this list of conditions in documentation and/or other materials provided with the distribution. The sole

exception to this condition is redistribution of a standard UnZipSFX binary (including SFXWiz) as part of a self-extracting archive; that is permitted without inclusion of this license, as long as the normal SFX banner has not been removed from the binary or disabled.

3 Altered versions--including, but not limited to, ports to new operating systems, existing ports with new graphical interfaces, versions with modified or added functionality, and dynamic, shared, or static library versions not from Info-ZIP--must be plainly marked as such and must not be misrepresented as being the original source or, if binaries, compiled from the original source. Such altered versions also must not be misrepresented as being Info-ZIP releases--including, but noT limited to, labelling of the altered versions with the names “Info-ZIP” (or any variation thereof, including, but not limited to, different capitalizations), “Pocket UnZip,” “WiZ” or “MacZip” without the explicit permission of Info-ZIP. Such altered versions are further prohibited from mis-representative use of the Zip-Bugs or Info-ZIP e-mail addresses or the Info-ZIP URL(s), such as to imply Info-ZIP will provide support for the altered versions.

4 Info-ZIP retains the right to use the names “Info-ZIP,” “Zip,” “UnZip,” “UnZipSFX,” “WiZ,” “Pocket UnZip,” “Pocket Zip,” and “MacZip” for its own source and binary releases.

List of tables

Schneider Electric (Australia) Pty Ltd

Customer Care Australia:
Phone: 1300 369 233
Email: customercare.au@schneider-electric.com
www.clipsal.com
www.schneider-electric.com.au

Schneider Electric (New Zealand) Pty Ltd
Schneider Electric (NZ) Ltd
38 Business Parade South
East Tamaki 2013
Auckland
New Zealand

Customer Care New Zealand:
Phone: 0800 652 999
Email: sales@nz-schneider-electric.com

5500NAC_5500SHAC_FW_2018_1.1_EN